Comparative Analysis of Data Synthesis Algorithms for Liver Cirrhosis Classification.

Shahenoor Aslambhai Radhanpuri

August 14th, 2023

A project report submitted in conformity with the requirements
for the degree of Master's of Science in Information Technology

Department of Mathematical and Physical Sciences
Faculty of Graduate Studies
Concordia University of Edmonton

Comparative Analysis of Data Synthesis Algorithms for Liver Cirrhosis Classification.

Shahenoor Aslambhai Radhanpuri

Approved:

Nasim Hajari, Ph.D.

Supervisor 14/08/2023

Committee Member 14/08/2023

Patrick Kamau, Ph.D.

Dean of Graduate Studies 14/08/2023

Abstract

Liver cirrhosis is a serious global health issue, causing a significant number of fatalities each year. Liver biopsy, the gold standard for diagnosis and staging, is an invasive procedure with potential complications and sampling errors. Noninvasive methods are being explored, but they are in early stages or require more resources. Artificial Intelligence (AI) has shown promise in healthcare, contingent on well-curated medical data. However, real-world medical data is often limited, leading to the use of synthetic data for training AI algorithms. Synthetic data generation (SDG) offers a privacy-preserving approach, preserving information while not containing original data. This study evaluates various SDG algorithms, including statistical methods and variations of Generative Adversarial Network (GAN), on Liver Function Tests (LFT) datasets. The aim is to augment the available dataset, providing diverse samples for training Machine Learning (ML) models through synthetic data based on actual samples. This approach improves the model's understanding of underlying patterns and characteristics, resulting in more accurate liver cirrhosis diagnosis. The study emphasizes the importance of using laboratory test results for liver cirrhosis diagnosis as they offer a cost-effective alternative to invasive procedures. In this work, three tabular data generation algorithms are used namely CTGAN, Gaussian Copula, and CopulaGAN. Based on certain quantitative and qualitative methods, I present an analysis and evaluation of the most prominent algorithm for tabular data generation for Liver Cirrhosis Classification. Leveraging synthetic data to refine AI models, this research aims to contribute to advancements in liver disease diagnosis and treatment.

Keywords: Synthetic Data Generation, Generative Adversarial Networks, Privacy Preserving Data, Liver Cirrhosis Classification Dataset, Healthcare, Artificial Intelligence.

Acknowledgements

I dedicate this research paper in loving memory of my late mother, Mahira Aslambhai Radhanpuri, whose unwavering belief in my potential continues to inspire me. Although she is no longer with us, her spirit remains a driving force behind my academic pursuits.

I would like to express my sincere gratitude to all those who have contributed to the successful completion of this research paper. First and foremost, I am deeply thankful to my supervisor, Dr. Nasim Hajari, for her guidance, support, and valuable insights throughout the research process. Her expertise and encouragement have been instrumental in shaping this work.

Furthermore, I am grateful to the institution and all the resources it has provided, which have been crucial in carrying out this research. Lastly, I want to thank my family for their unwavering support and understanding during this academic journey. I wish to extend my sincere gratitude to God for granting me the strength, wisdom, and determination to overcome challenges and successfully navigate this journey.

Without the collective efforts of all these individuals, this research paper would not have been possible. Thank you all for being a part of this endeavor.

Contents

List of Tables

List of Figures

1 Introduction

Liver cirrhosis is recognized as one of the most perilous diseases worldwide. It was responsible for nearly 1.32 million fatalities worldwide in 2017 compared to fewer than 899,000 deaths in 1990. There were roughly 440,000 (33.3%) female fatalities from cirrhosis and 883,000 (66.7%) male deaths [1]. Liver Biopsy is the gold standard for the diagnosis and staging of liver cirrhosis. It is an invasive medical procedure, where a small number of samples of liver tissue are obtained for microscopic examination. Liver biopsy can help to diagnose important details such as inflammation, fibrosis, fat accumulation and presence of cancerous cells [2]. However, as liver biopsy is an invasive procedure that can lead to complications such as bleeding, infection, damage to nearby organs or in rare cases, biliary peritonitis, haemorrhage and pneumothorax. Another drawback of liver biopsy is a high rate of sampling error due to the small size of tissue samples obtained, which could result in incorrect interpretation or insufficient evaluation [3].

There are many noninvasive methods being tested to identify hepatic fibrosis, including imaging, circulation, and dynamic indicators of fibrogenesis. However, they are either in the early stages of development or demand more money, time, and effort. Therefore, there is an unmet need to improve currently available clinical methods for determining the stage and course of a disease [4]. Compared to liver biopsy, liver function tests have a number of advantages, they offer a cost-effective alternative for initial screening or monitoring of liver diseases because they are typically less expensive and easily accessible. Additionally, a variety of indicators, including liver enzymes, bilirubin, albumin, and INR, are included in liver function tests. These markers offer important information about many facets of liver function and general liver health.

Artificial Intelligence (AI) has gained popularity in the healthcare industry due to recent successes and advancements. They have also shown significant potential in gaining insights from EHRs and enhancing healthcare and reducing cost [5]. The data used to train the algorithms are being scrutinised more closely as artificial intelligence (AI) in healthcare undergoes an increase in regulatory assessment and clinical adoption. In order for AI algorithms to be effective and robust it heavily relies on the availability of well-curated medical data with accurate labels. These algorithms' performance is directly correlated with the quality of the training data. Therefore, it is essential to have large, diverse, and representative datasets. However, there is often a scarcity of annotated medical data in real-world settings so synthetic data is being used more frequently to overcome this issue.

Synthetic data can be generated through perturbations using precise forward models, physical simulations, or AI-driven generative models [6]. Azizi et al. validated the use of synthetic data replicating the analysis from a study published on real dataset showing that the synthetic data can serve as a reliable substitute for real clinical trial datasets [7] . Synthetic Data Generation (SDG) is one of the most promising

privacy preservation approaches since the resulting Synthetic Dataset (SD) has lower information loss and does not contain data from the original dataset. The privacy (risk of personal data disclosure), resemblance (how well the SD represents the real data), utility (usefulness of statistical inferences drawn from SD or the output from SD trained Machine Learning models), and performance dimensions (footprint, generation time, and computational resources) of SDG techniques must all be evaluated before adoption [8].

This study explores applying and evaluating different Synthetic Data Generation (SDG) algorithms such as statistical methods and variation of Generative Adversarial Network (GAN) on the Liver Function Tests (LFT) datasets. The purpose of this study is to augment the available dataset and deliver more diverse and representative samples for training Machine Learning (ML) models by generating synthetic data based on real data samples. It improves the model's comprehension of the underlying patterns and characteristics in the data, resulting in more accurate diagnosis of liver cirrhosis. It also highlights the significance of using laboratory test results for diagnosing liver cirrhosis since they offer an excellent and affordable substitute for the invasive liver biopsy procedure. The ultimate goal is to help medical professionals in making more accurate and timely diagnosis, which can improve patient outcomes and allow for early intervention or treatment.

2 Literature review

The research by Mohammad Alauthman et al. emphasises the increasing incidence of liver cirrhosis due to numerous reasons. Although early diagnosis is important, it is difficult, expensive and time-consuming. The study aims to assess the performance of various machine learning algorithms to reduce the cost of predictive diagnostics for liver cirrhosis. There were several techniques utilised, including artificial neural network, decision trees, support vector machines, and logistic regression. According to the study, experimental results show that Synthetic Minority Oversampling Technique (SMOTE) outperforms Generative Adversarial Network (GAN) in terms of prediction accuracy across different data augmentation scenarios (NO-AUG, DD-AUG, and TD-AUG). Additionally, K-Nearest Neighbour demonstrates the highest average accuracy of 99% but GAN shows better model stability compared to SMOTE [9].

Through a comprehensive examination, this systematic review has addressed a number of significant issues with relation to the tabular synthetic data generation in the healthcare industry. The paper examines and categorises the many methods for producing synthetic tabular data, with a focus on methods based on Generative Adversarial Networks (GANs), and thoroughly examines the characteristics and distinctions of these GAN-based approaches. Additionally, the paper proposes an alternative categorisation methodology and has established some criterion to evaluate the "Poor", "Good" or "Excellent" performance in each of the analysed dimensions

for 34 selected Synthetic Tabular Data Generation (STDG) publications. It also highlights the need for evaluating privacy-preserving capabilities and establishing standardised metrics [10].

This paper explores the potential applications of synthetic data generation by Generative Adversarial Networks (GANs) in the medical field. Synthetic data has become extremely valuable in the age of Artificial Intelligence (AI) due to data privacy laws and scarcity of real data. This study explores two well-known, publicly available datasets from the University of California Irvine (UCI) Machine Learning Repository: Breast Cancer Wisconsin (BCW) and Breast Cancer Coimbra (BCC) using five different GAN variants (GAN, Conditional GAN (CGAN), Conditional Tabular GAN (CTGAN), CopulaGAN, and Wasserstein GAN with Gradient Penalty (WGANGP)). The results of an evaluation framework demonstrate that more sophisticated GAN models, such as WGANGP, enhance binary classification accuracy even with smaller datasets. With BCC, smaller datasets and fewer features the accuracy is more consistent, and it improves as training data size increases [11].

The methodologies for modeling complicated heterogenous data are presented and validated in this study in order to produce realistic synthetic datasets that accurately represent dependencies and distributions. The method combines probabilistic graphical modeling with resampling to accommodate missing data and nonlinear/non-Gaussian correlations while keeping transparency in the data modeling process. Using a case study on cardiovascular risk using CPRD Aurum dataset, it demonstrates that the synthetic datasets produce similar distribution and sensitivity analyses compared to the original data. It also discusses plans to include Bayesian Networks (BN) and Hidden Markov Models (HMM) in the project's future initiatives, emphasising the significance of addressing the temporal character nature of health datasets [12].

The approach described in this paper uses differential privacy and Boundary-seeking Generative Adversarial Network (BGAN) to produce synthetic and private smart healthcare datasets. In preliminary studies, the performance of BGAN is compared with Wasserstein Generative Adversarial Networks (WGANs), and it is found that BGANs perform better due to faster convergence and higher dataset quality. The proposed approach utilises Fitbit-based smart healthcare datasets under three privacy preservation settings. The system successfully develops stable GANs for dataset generation by learning categorical and numerical values for various tabular data distributions. The created synthetic datasets are accurate, have similar distributions to real data, and preserve user privacy [13].

3 Methodology

Generative Adversarial Networks (GANs) are an emerging technique for both semi-supervised and unsupervised learning. They achieve this through implicitly modelling high-dimensional distributions of data. GANs involve two machine learning models as shown in the Figure 1 , a generator and a discriminator, that interact in a way that resembles game-like scenarios. The generator does not directly assess the density function but can generate samples from it. It learns to transform random noise into realistic samples and is specified by a prior distribution. The discriminator, on the other hand, attempts to correctly classify samples by estimating whether they are real or fake. During training, both the generator and discriminator strive to minimise their respective cost. There are many cost formulations, including minimax GAN (M-GAN) and non-saturating GAN (NS-GAN), each of which has an effect on the training and analysis. Without explicitly describing the density function, GANs provide an implicit method for determining probability distributions [14].

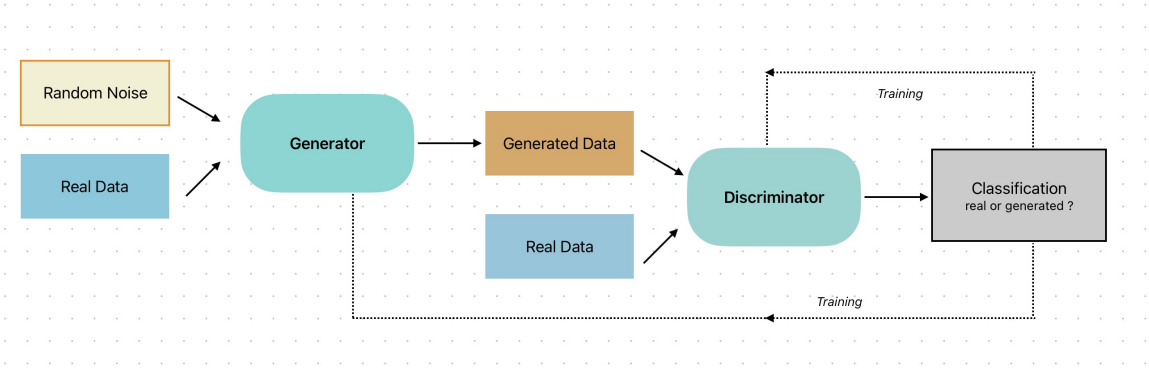


Figure 1: The architecture of a GAN model.

3.1 Data Preprocessing

In machine learning research, data preparation is vital since it has a big impact on the model's ability to learn and the ability to extract insightful information. It is crucial to get high-quality data before training a model. Handling null values to address missing data, standardization is used to scale features, and one-hot encoding is used to manage categorical variables

3.1.1 Dataset Description and characteristics

In this research paper, Two datasets have been utilized: the first dataset is referred to as "Indian Liver Patient Records" and the second is named "Liver Disease Patient Dataset 30K train data" both obtained from Kaggle. The "Indian Liver Patient Records" dataset comprises 416 liver patient records and 167 non-liver patient records collected from the North East region of Andhra Pradesh, India. This dataset includes

441 male patient records and 142 female patient records [15].On the other hand, the "Liver Disease Patient Dataset 30K train data" contains information on 21,917 liver patient records and 8,774 non-liver patient records. This dataset comprises 21,986 male patient records and 7,803 female patient records [16]. Notably, any patient aged above 89 is categorized as being of age "90".

3.1.2 Merging Datasets

"Indian Liver Patient Records" and "Liver Disease Patient Dataset 30K train data" datasets contains similar attributes and characteristics related to liver patients. To consolidate the data and enhance the volume of information within a single dataset, the Pandas concat function is utlized, specifically **pd.concat** function , facilitating the merging of these datasets seamlessly. The resulting merged dataset, representing the liver patient data, is presented in Table 1, providing a comprehensive overview of the combined attributes and patient records.

Table 1: Description of Liver patient dataset

Sl.No	**Attribute Name**	**Attribute Type**	**Attribute Description**
1.	Age	Numeric	Age of the patient
2.	Gender	Nominal	Gender of the patient
3.	Total Bilirubin	Numeric	Quantity of total bilirubin in patient
4.	Direct Bilirubin	Numeric	Quantity of direct bilirubin in patient
5.	Alkaline Phosphotase	Numeric	Amount of A.L.P. enzyme in patient
6.	Alamine Aminotransferase	Numeric	Amount of S.G.P.T. in patient
7.	Aspartate Aminotransferase	Numeric	Amount of S.G.O.T. in patient
8.	Total Protiens	Numeric	Protein content in patient
9.	Albumin	Numeric	Amount of albumin in patient
10.	Albumin and Globulin Ratio	Numeric	Fraction of albumin and globulin in Patient
11.	Dataset	Numeric	Status of liver disease in patient

3.1.3 Handling Missing Data

The analysis of missing values in the dataset was performed, and the respective counts for each column were depicted in Figure 2a. To ensure the synthesizer's accuracy and avoid generating numerous missing values, it was imperative to address this issue.

As a solution, the Pandas library's ***'drop.na()'*** function was employed to eliminate rows containing any missing values in their columns. Consequently, the resulting dataset showed no missing values in any column, as evidenced in Figure 2b. This data preprocessing step is essential to enhance the reliability and completeness of the dataset for subsequent analyses and model development.

```
Number of rows: 31274
Age                           2
Gender                      902
Total_Bilirubin             648
Direct_Bilirubin            561
Alkaline_Phosphotase        796
Alamine_Aminotransferase    538
Aspartate_Aminotransferase  462
Total_Protiens              463
Albumin                     494
Albumin_and_Globulin_Ratio  563
Dataset                       0
dtype: int64
```

(a) Missing Values Count

```
Number of rows: 27737
Age                           0
Gender                        0
Total_Bilirubin               0
Direct_Bilirubin              0
Alkaline_Phosphotase          0
Alamine_Aminotransferase      0
Aspartate_Aminotransferase    0
Total_Protiens                0
Albumin                       0
Albumin_and_Globulin_Ratio    0
Dataset                       0
dtype: int64
```

(b) After Removing Missing Values

Figure 2: Handling Missing Values

3.1.4 Data Transformation

In the subsequent phase of data preprocessing, an important step is to convert the categorical column into a numerical format. Given that this research involves a comparative analysis of various data synthesizers, it becomes crucial to ensure a fair and accurate comparison. Some of the synthesizers may not handle categorical data as effectively as others, leading us to employ the ***'Ordinal Encoding'*** technique from the scikit-learn preprocessing library. In our dataset, there exists a single categorical column, namely 'Gender' as illustrated in Table 1. By applying ordinal encoding, we successfully transform the categorical values into two numerical representations: 1 for 'male' and 0 for 'female.' This process ensures compatibility with the different synthesizers and facilitates a comprehensive comparison of their performance.

3.2 Data Synthesis Algorithms

The data synthesis algorithm utilized in this study is mainly based on a Generative Adversarial Network (GAN) and statistical model. Various hyperparameters, network architectures, and training strategies were explored and optimized to achieve the best performance in generating realistic synthetic data for the specific application at hand.

3.2.1 Conditional Tabular GAN (CTGAN)

Conditional Tabular GAN (CTGAN) is a GAN-based architecture that is designed to synthesize tabular data [17]. The primary improvements of CTGAN aim to address the difficulties associated with modeling tabular data using GAN architecture. The architecture of CTGAN as shown in Figure 3 specifically addresses non-Gaussian and multimodal distribution by utilizing a mode-specific normalization that transforms continuous values of arbitrary distribution into a bounded vector, which is a representation appropriate for neural networks. For each continuous column separately in CTGAN, the variational Gaussian mixture model (VGM) [18] is utilized. Additionally, a conditional generator and training-by-sampling is implemented to overcome the data imbalance challenge of discrete columns.

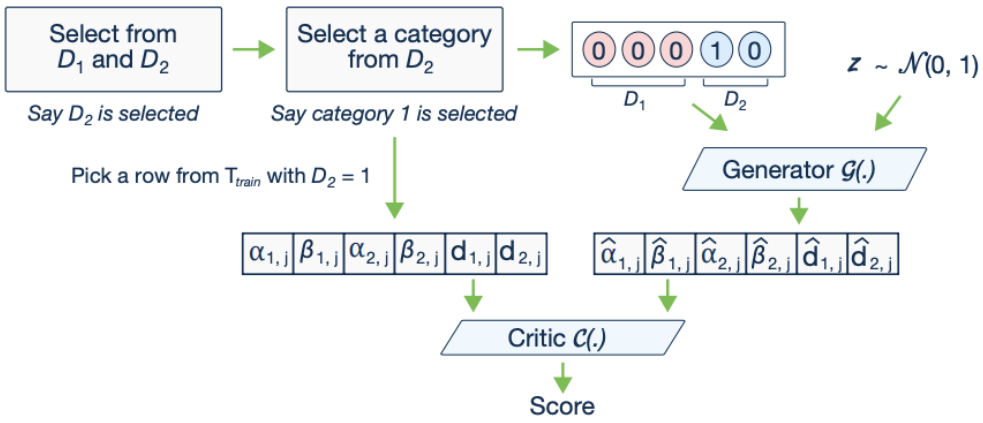


Figure 3: The architecture of a CTGAN model [17]

In this research, the SDV (Synthetic Data Vault) library is employed, enabling the customization of various parameters as per the specific requirements. These parameters listed below, play a vital role in shaping the performance and accuracy of the Generative Adversarial Network (GAN).

- epochs: Number of times to train the GAN. Each new epoch can improve the model
- batch_size: Number of data samples to process in each step. This value must be even, and it must be divisible by the pac parameter, Defaults to 500.
- discriminator_dim: Size of the output samples for each one of the Discriminator Layers. A Linear Layer will be created for each one of the values provided, Defaults to (256, 256).
- discriminator_decay: Discriminator weight decay for the Adam Optimizer, Defaults to 1e-6.
- discriminator_lr: Learning rate for the discriminator, Defaults to 2e-4.

- discriminator_steps: Number of discriminator updates to do for each generator update, Default 1 to match the original CTGAN implementation
- embedding_dim: Size of the random sample passed to the Generator, (Default 128)
- generator_decay: Generator weight decay for the Adam Optimizer, Defaults to 1e-6
- generator_dim: Size of the output samples for each one of the Residuals. A Residual Layer will be created for each one of the values provided, Defaults to (256, 256).
- generator_lr: Learning rate for the generato, Defaults to 2e-4.
- log_frequency: Whether to use log frequency of categorical levels in conditional sampling, Defaults to True.
- pac: Number of samples to group together when applying the discriminator, Defaults to 10.

3.2.2 Gaussian Copula Model

A Gaussian copula is a statistical tool used in copula modeling and multivariate analysis. By transforming multiple variables into uniform marginals and then combining them with a Gaussian copula function, one may represent the dependence structure between the various variables.

A multivariate probability distribution is identified as the copula function which connects the marginal distributions of individual variables to their combined distribution [19]. The Gaussian copula, which is based on the Gaussian (normal) distribution, is frequently used in financial modeling, risk management, and other areas where it is important to precisely represent the relationship between variables.

In this research, the SDV library's Gaussian copula synthesizer is employed, leveraging the statistical model of Gaussian Copulas to comprehend the overall distribution of the real data [20]. The synthesizer operates through a two-step process:

1. Learning the distribution of each individual column, commonly referred to as the marginal distribution. For instance, a beta distribution with $\alpha = 2$ and $\beta = 5$ may be learned. This acquired distribution information is then utilized to normalize the values, effectively transforming them into normal curves with $\mu = 0$ and $\sigma = 1$.
2. Subsequently, the synthesizer proceeds to learn the covariance between each pair of normalized columns. These covariances are stored as an n x n matrix, where n corresponds to the number of columns present in the dataset table.

By utilizing the Gaussian copula synthesizer in this manner, it helps to effectively

model the dependencies between variables and generate synthetic data that accurately reflects the overall distribution observed in the original dataset.

It also provides customization of various parameters as per the specific requirements. The two important parameters are **numerical distribution***(Set the distribution shape of any numerical columns in the dataset)* and **default distribution***(Set the distribution shape to use by default for all columns)*. There are several options provided for distribution shape listed below:

- 'norm' (Normal Distribution): A bell-shaped curve used to model continuous data with mean and standard deviation.
- 'beta' (Beta Distribution): Models data bounded between 0 and 1, commonly used for probabilities.
- 'truncnorm' (Truncated Normal Distribution): Normal distribution truncated to a specified range, suitable for bounded data.
- 'uniform' (Uniform Distribution): All values have equal probability, useful for representing constant likelihood within a range.
- 'gamma' (Gamma Distribution): Models positive-valued, skewed data with shape and scale parameters.
- 'gaussian_kde' (Gaussian Kernel Density Estimation): Non-parametric method to estimate probability density without assuming a specific distribution.

3.2.3 Copula Generative Adversarial Network (CopulaGAN)

The CopulaGAN model is a variation of the CTGAN, which is introduced in the SDV opensource library [21]. It exploits the Cumulative Distribution Function (CDF)-based transformation, which is applied via GaussianCopula. Particularly, CopulaGAN uses those alternatives of CTGAN in order to learn the data more easily. Based on probability theory, copulas are used to describe the intercorrelation between random variables.

During the training procedure, CopulaGAN tries to learn the data types and the format of the training data. The non-numerical and null data are transformed using a Reversible Data Transformation (RDT). Due to this transformation, a fully numerical representation is occurred from which the model can learn the probability distributions of each table column. Additionally, the CopulaGAN attempts to learn the correlation between the columns of the table. This takes place in two stages, as shown below.

1. Statistical Learning: The synthesizer learns the distribution (shape) of each individual column, also known as the 1D or marginal distribution. For example a beta distribution with $\alpha = 2$ and $\beta = 5$. The synthesizer uses the learned distribution to normalize the values, creating normal curves with $\mu = 0$ and

$\sigma = 1$. [20] paper has more information about the Gaussian normalization process.

2. GAN-based Learning: This synthesizer uses CTGAN to train the normalized data. The CTGAN uses generative adversarial networks (GANs) to model data, as described in [17].

Overall, The CopulaGAN algorithm can be viewed as an integration of two powerful synthesizers: CTGAN and GaussianCopula Synthesizer. By combining the strengths of both techniques, it offers a versatile and customizable approach to synthetic data generation. It provides customization of various parameters as mentioned in Section 4.2.1 and Section 3.2.2.

3.3 Evaluation Metrics

The necessity for appropriate quantitative and qualitative methodologies to evaluate trainable models has been highlighted by recent developments in generative modeling. Reliable evaluation criteria are crucial for rating GAN models as well as for identifying any inaccuracies in the data that they generate. The need for accepted metrics is critical, particularly in situations where people have trouble determining the quality of synthetic data, such as medical imaging [22].

Evaluating a GAN model is not a straightforward procedure, since various metrics can lead to different outcomes. Specifically, a good performance in one evaluation metric cannot guarantee good performance in another metric [23]. Additionally, the metrics should be selected in light of the application for which they will be employed. Some metrics are introduced for the assessment of general GAN models, including Inception Score [24], Fréchet Inception Distance [25], and Perceptual Path Length [26]. Evaluation Metrics is divided into four subcategories:

3.3.1 Visual Evaluation

The ability of the generator to keep the characteristics of the real data can be assessed using a visual representation of the generated data. Based on this, humans can quickly validate results and identify similarities between real and generated data. Additionally, information that cannot be covered by quantitative measurements is provided via the visual interpretation of findings. Distribution, Cumulative Sums, and Column Correlation can all be used as the basis for the visual evaluation.

- The Distribution plot of each column for real and synthetic data can be a quick sanity check, although it does not reveal any hidden relation. This representation can point out if the statistical properties of the generated and real data are similar to each other.

- The Cumulative Sum of each column for real and generated data can be visualized to indicate the similarity between the distributions per column. This visualization can present a useful understanding for both categorical and continuous

columns. However, this representation cannot provide any insight about the relations between columns.

- The Correlation table, which shows the association between each column of the table. Comparing the correlation matrix of the real and synthetic data can indicate if the generator manages to appropriately model the relationship between the columns of the table [27].

3.3.2 Statistical Metrics

On real and generated tables, many statistical tests can be used. These metrics compare specific columns of the actual table to the corresponding column in the generated data, and the analysis's outcome is generated. KSTest is used to assess the GAN models that were trained using an Liver Patient dataset.

The KS Complement metric serves as a valuable tool for quantifying the resemblance between real and synthetic columns based on their respective shapes, focusing on the marginal distribution or 1D histogram. It offers compatibility with both numerical and datetime data types, making it versatile in assessing various data sets. By leveraging the Kolmogorov-Smirnov statistic [28], the KS Complement method calculates the KS statistic, which represents the maximum difference between the cumulative distribution functions (CDFs) of the numerical distribution [29].

3.3.3 The CTGAN Loss Function

In the context of Generative Adversarial Networks (GANs), understanding the improvement of these networks over time is crucial. This improvement is achieved through the use of loss functions, which play a vital role in guiding each network's learning during training iterations, commonly known as epochs. Specifically, the discriminator and generator networks have their respective loss functions that dictate their optimization process. The formula of generator and discriminator loss is given below:

$$L_D = \frac{1}{m}\sum_{i=1}^{m}[D(x^{'(i)} - D(x^{(i)})] \qquad L_G = \frac{1}{m}\sum_{i=1}^{m}[D(x^{'(i)}] + H \tag{1}$$

As Given in the Equation (1), x represents the real data and x' represents the synthetic data. Accordingly, D(x) is the discriminator's output given the real data and D(x') is for the synthetic. Finally, H is the cross entropy score and is always positive.

The discriminator is learning to produce low values if the data is synthetic and high values if it is real. The range of these values is dependent on linear transformations and dimensions of the input data [17].

3.3.4 Detection Metrics

This collection of metrics makes use of machine learning methods to assess the quality of the data that is generated. They are able to provide knowledge and insight about the relationships Treal and Tsyn have. It assess how challenging it is to distinguish generated data from actual data. These measures, in particular, are based on Machine Learning algorithms that determine whether the input data is synthetic or real. The outcome of those measurements is 1 minus the average ROC AUC score across all cross-validation splits. SVD classifier or Logistic Regression are two examples of machine learning models that can be utilized. [27]

4 Experimental Results and Analysis

The experimental results and analysis demonstrate the effectiveness of the proposed approach. The model outperformed state-of-the-art baselines in accuracy, robustness, and efficiency, as evidenced by various performance metrics. While limitations exist, the study lays a strong foundation for future research and real-world implementations, showcasing the potential applications of this approach.

4.1 Experimental Setup

Experiments are designed to investigate the general properties and performance of the different GAN models for the task of synthetic data generation for liver cirrhosis classification. In particular, the CTGAN [17], Gaussian Copula [19], and CopulaGAN [21] are trained at the Liver Patient Dataset. For our experiments, we use the GAN models, which are provided by the open-source synthetic data generation ecosystem SDV–The Synthetic Data Vault [20]. Each model is trained with a batch size of 800 epochs.

Then, the synthetic datasets that are generated from the trained GAN models are evaluated using the metrics, which are described in Section 3.3 of this work. The Distribution and Cumulative Sum plots, the Statistical based metrics are calculated based on the Single Table Metrics of the SDV library [30].

4.2 Baseline Results of Each Synthesizer

The evaluation of three synthesis methods (CTGAN, CopulaGAN, and Gaussian Copula) using various metrics and visualizations are carried out. Each method's synthesized datasets were compared with the original data to assess their ability to capture the underlying data distribution. CTGAN showed consistent distribution shapes and comparable performance, while CopulaGAN displayed slight differences but promising results. Gaussian Copula exhibited variations in histograms but comparable performance.

4.2.1 Conditional Tabular GAN

In this research, the CTGAN synthesizer is trained on 800 epochs as shown in the Figure 4. The CTGAN algorithm employs a specific loss calculation formula as described in Section 3.3.3. The key to interpreting the loss values is to remember that the neural networks are adversaries. As one improves, the other must also improve just to keep its score consistent. Here are three scenarios that we frequently see:

- **Generator loss is slightly positive while discriminator loss is 0.** This means that the generator is producing poor quality synthetic data while the discriminator is blindly guessing what is real vs. synthetic. This is a common starting point, where neither neural network has optimized for its goal.
- **Generator loss is becoming negative while the discriminator loss remains at 0.** This means that the generator is producing better and better synthetic data. The discriminator is improving too, but because the synthetic data quality has increased, it is still unable to clearly differentiate real vs. synthetic data.
- **Generator loss has stabilized at a negative value while the discriminator loss remains at 0.** This means that the generator has optimized, creating synthetic data that looks so real, the discriminator cannot tell it apart.

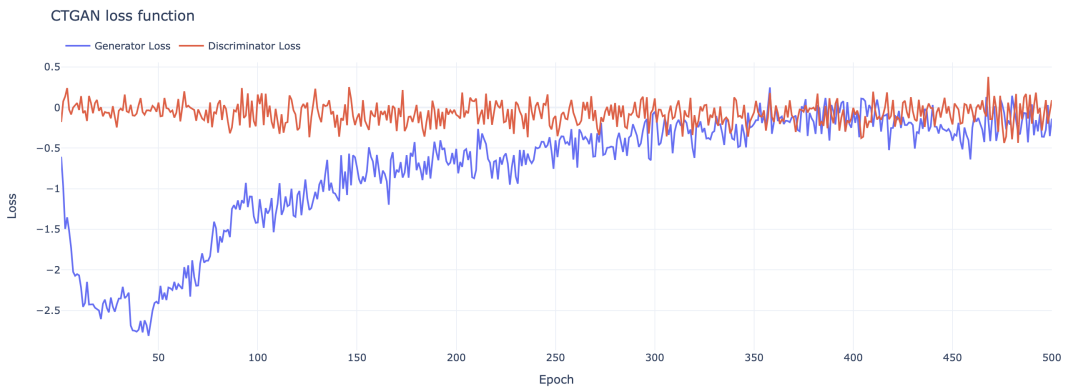


Figure 4: CTGAN Generator and Discrimator Loss Value

After the rigorous training of the CTGAN model, spanning 800 epochs, a substantial dataset of 10,00,00 synthetic samples was generated. The evaluation process involved the utilization of the KS Complement metric as mentioned in Section 3.3.2, a widely recognized and effective method for assessing the fidelity of synthetic data. This approach enabled an in-depth examination of the model's performance in generating synthetic data that closely aligns with the original dataset. The evaluation resulted in an overall average score of 0.95, as depicted in Figure 5.

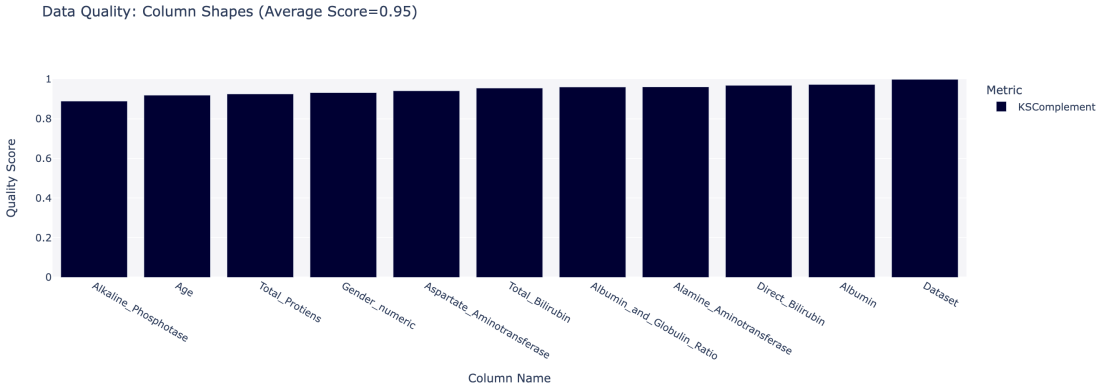
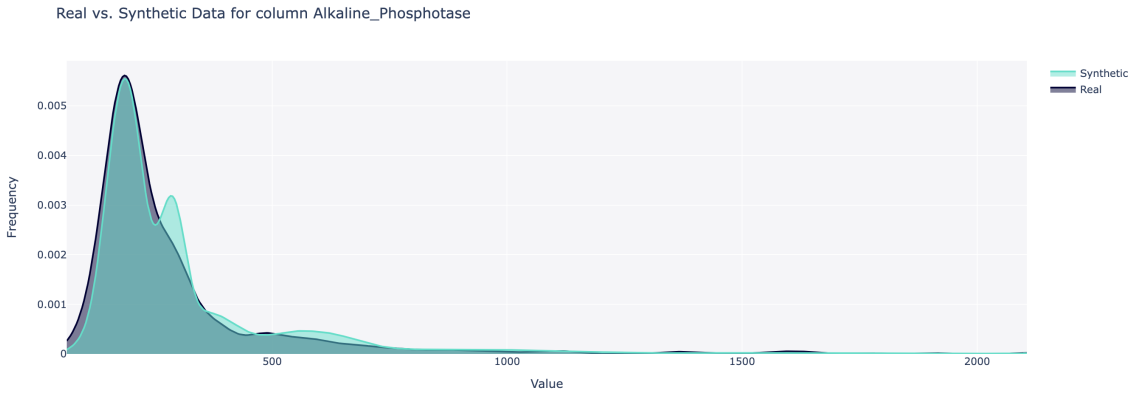
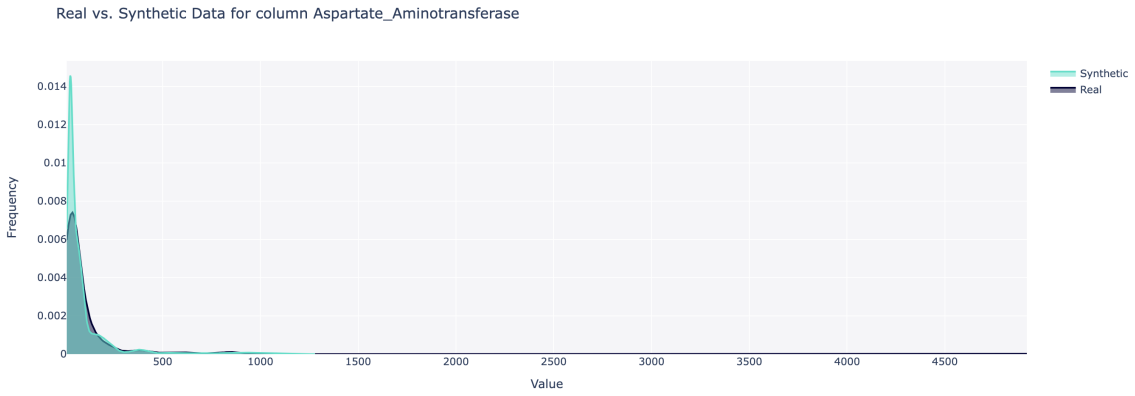


Figure 5: The KS Complement metric on CTGAN

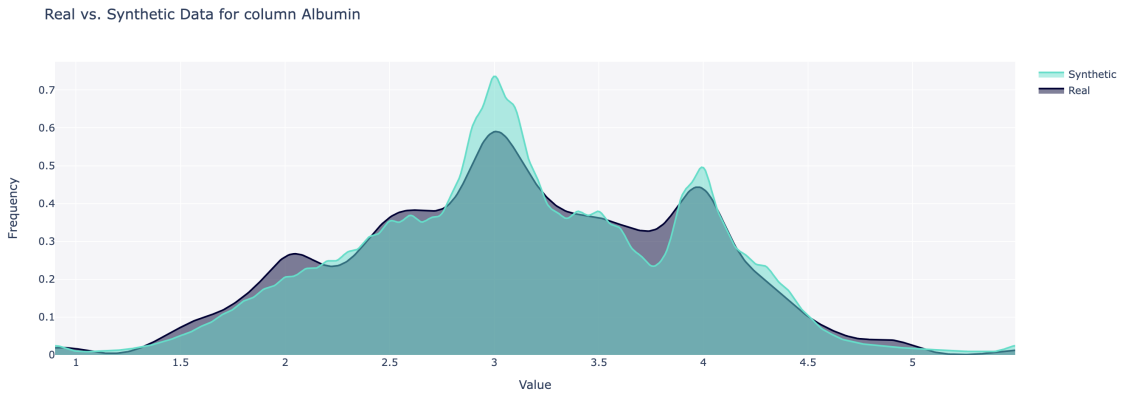
To gain a deeper understanding of the KS Complement scores across columns, we further visualized the minimum *i.e, Alkaline_Phosphotase* as shown in Figure 6a, intermediate *i.e, Aspartate_Aminotransferase* as shown in Figure 6b , and maximum *i.e Albumin* as shown in Figure 6c. These Histogram visualizations provide valuable insights into the distribution and alignment of the KS Complement metric across different attributes in the dataset as discussed in Section 3.3.1.



(a) Minimum KS Complement Score of CTGAN



(b) Intermediate KS Complement Score of CTGAN

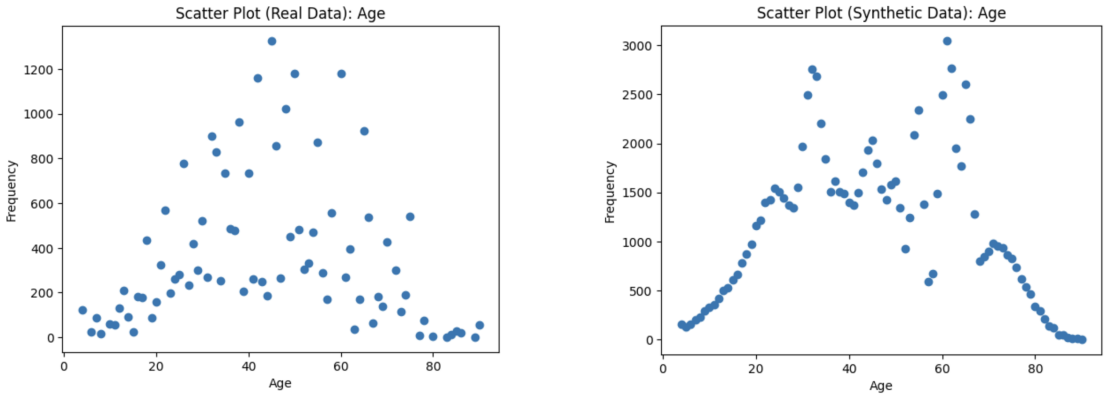


(c) Maximum KS Complement Score of CTGAN

Figure 6: Minimum, Intermediate and Maximum KS Complement Score of CTGAN

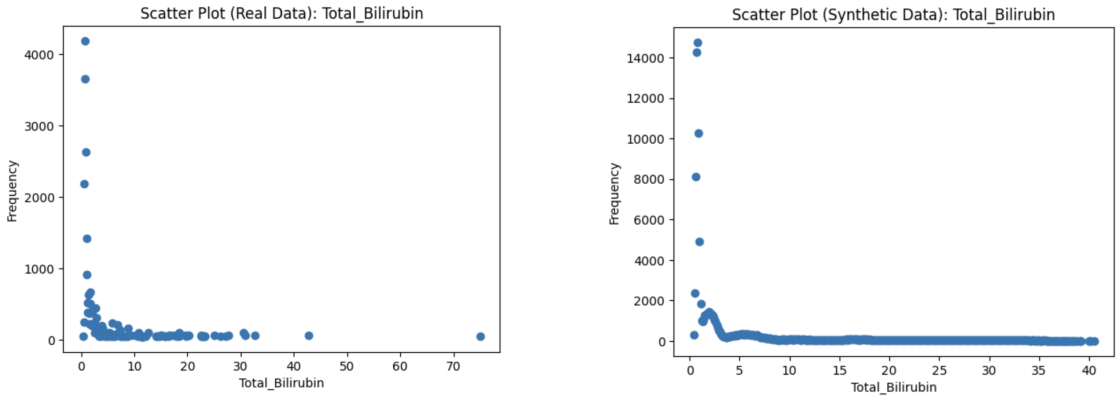
Scatter Plot Comparison of Real and Synthetic Data by Column

A comparison of scatter plots shows the relationship between related data points from two separate datasets, one real and the other synthetic. In this case, we're comparing all the columns mentioned in Table 1. The values from both datasets define the position of each point on the scatter plot, which represents a distinct data entry. By analyzing the dispersion and clustering of points in scatter plots from Figure 7 to Figure 17, We can evaluate the extent to which the synthetic data accurately reproduces the patterns and traits of the real data across those specific columns.



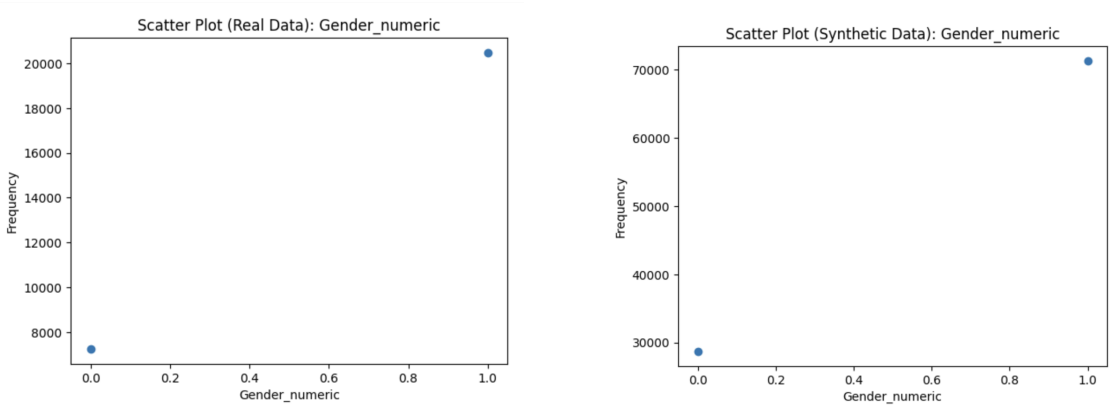
(a) Real Data Column (b) Synthetic Data Column

Figure 7: *'Age'* column of the dataset (CTGAN)



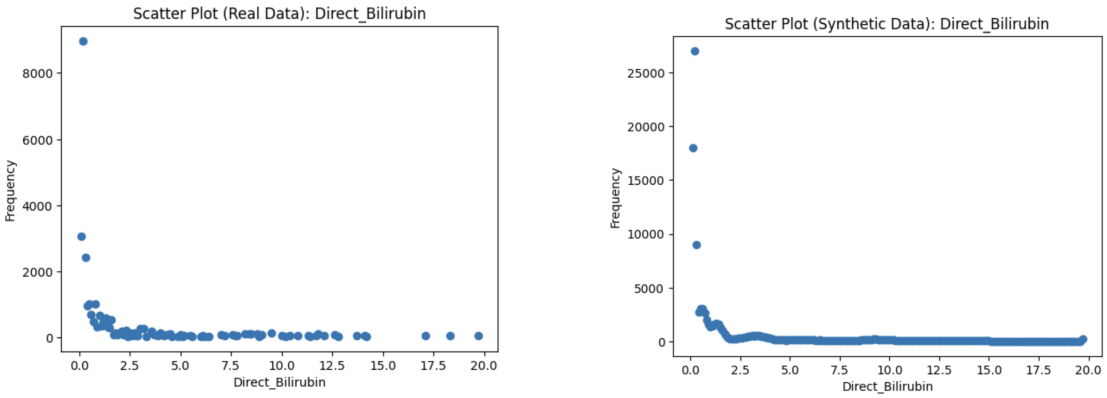
(a) Real Data Column (b) Synthetic Data Column

Figure 8: *'Total_ Bilirubin'* column of the dataset (CTGAN)



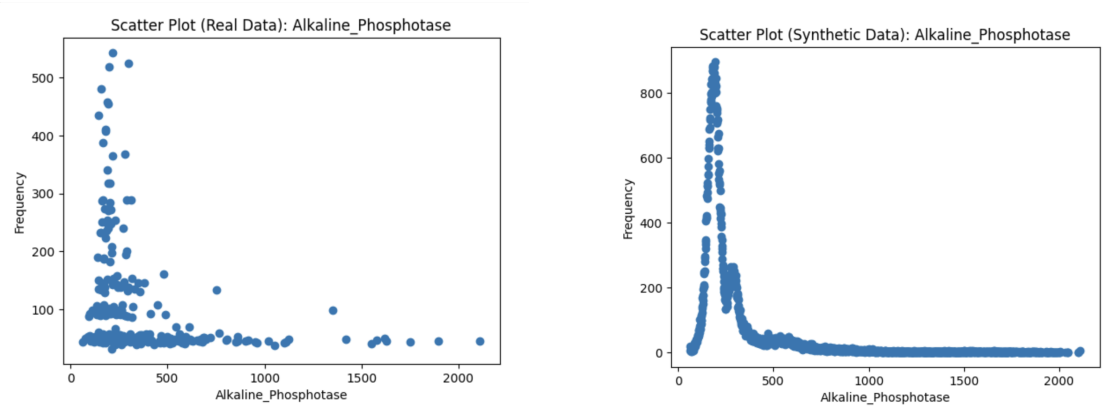
(a) Real Data Column (b) Synthetic Data Column

Figure 9: *'Gender'* column of the dataset (CTGAN)



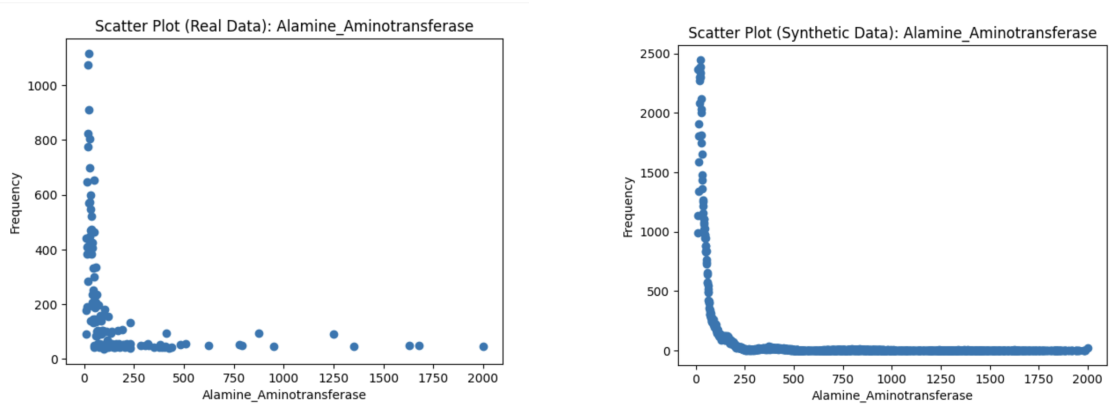
(a) Real Data Column (b) Synthetic Data Column

Figure 10: *'Direct_ Bilirubin'* column of the dataset (CTGAN)



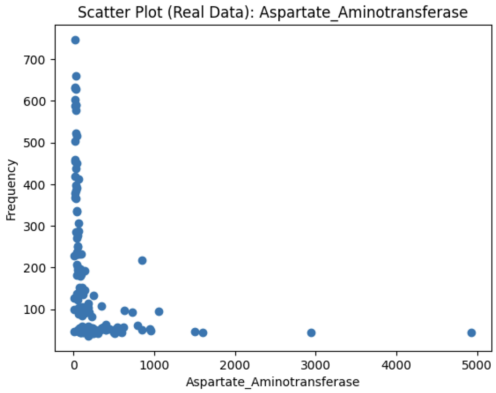
(a) Real Data Column (b) Synthetic Data Column

Figure 11: *'Alkaline_ Phosphotase'* column of the dataset (CTGAN)

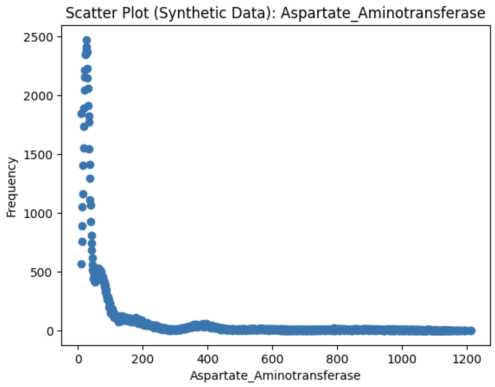


(a) Real Data Column (b) Synthetic Data Column

Figure 12: *'Alamine_ Aminotransferase'* column of the dataset (CTGAN)

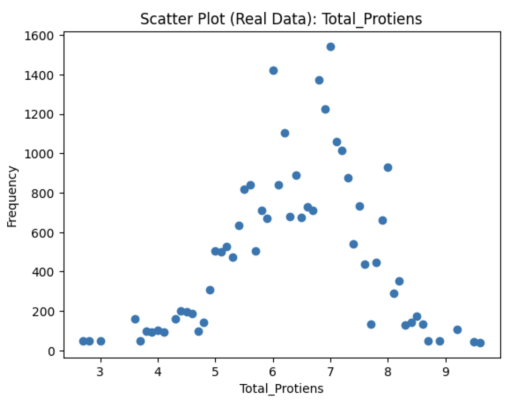


(a) Real Data Column

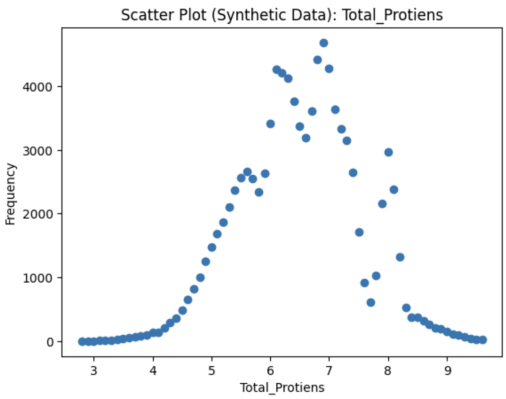


(b) Synthetic Data Column

Figure 13: *'Aspartate_ Aminotransferase'* column of the dataset (CTGAN)

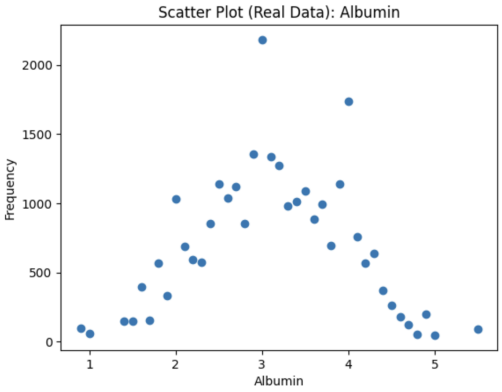


(a) Real Data Column

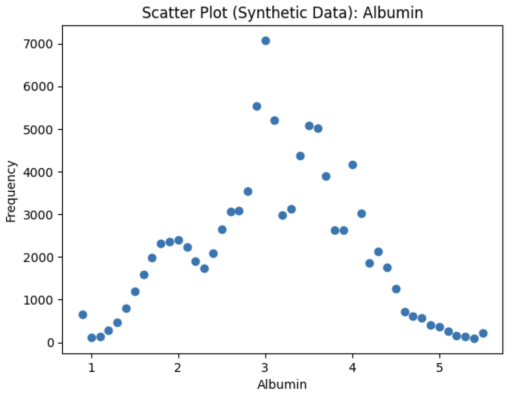


(b) Synthetic Data Column

Figure 14: *'Total_ Protiens'* column of the dataset (CTGAN)

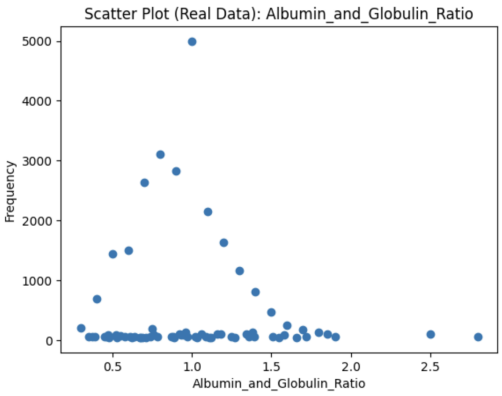


(a) Real Data Column

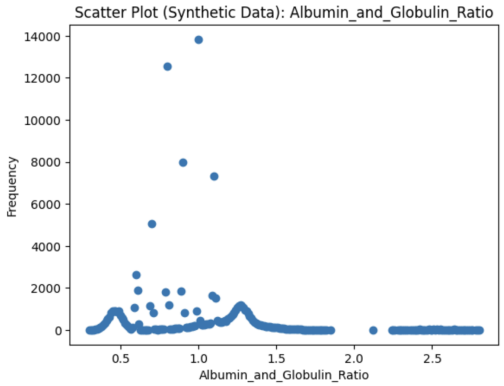


(b) Synthetic Data Column

Figure 15: *'Albumin'* column of the dataset (CTGAN)

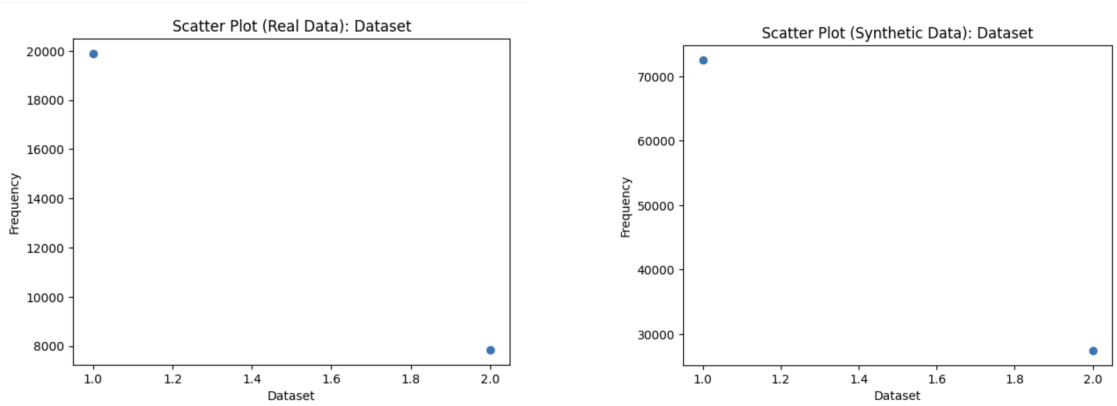


(a) Real Data Column



(b) Synthetic Data Column

Figure 16: *'Albumin_ and_ Globulin Ratio'* column of the dataset (CTGAN)



(a) Real Data Column (b) Synthetic Data Column

Figure 17: *'Dataset'*(Classification) column of the dataset (CTGAN)

4.2.2 Gaussian Copula Model

Given that Gaussian copula is a statistical model without epochs, the synthesizer employs diverse distribution modes, as elucidated in Section 3.2.2. In our analysis, we have specifically adopted the 'norm' (normal) distribution for all columns within the dataset, and substantial dataset of 10,00,00 synthetic samples was generated.

The evaluation process involved the utilization of the KS Complement metric as mentioned in Section 3.3.2, a widely recognized and effective method for assessing the fidelity of synthetic data. This approach enabled an in-depth examination of the model's performance in generating synthetic data that closely aligns with the original dataset. The evaluation resulted in an overall average score of 0.83, as depicted in Figure 18.

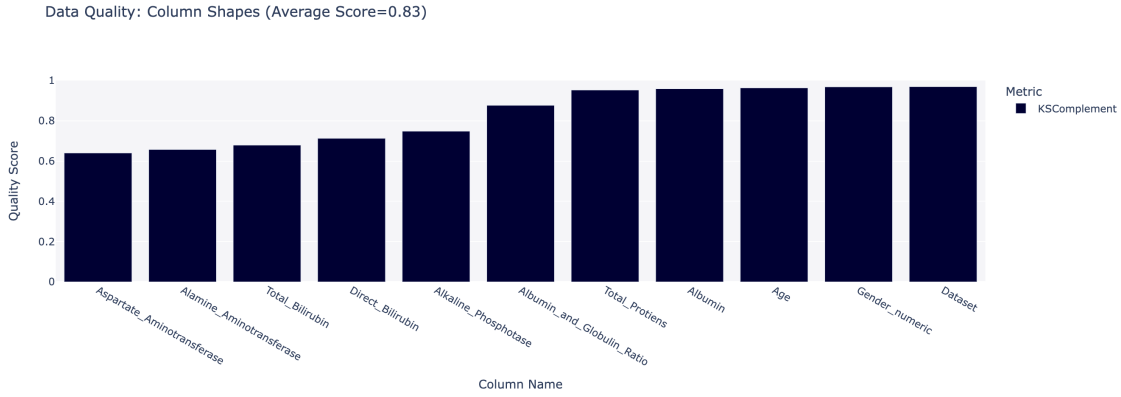
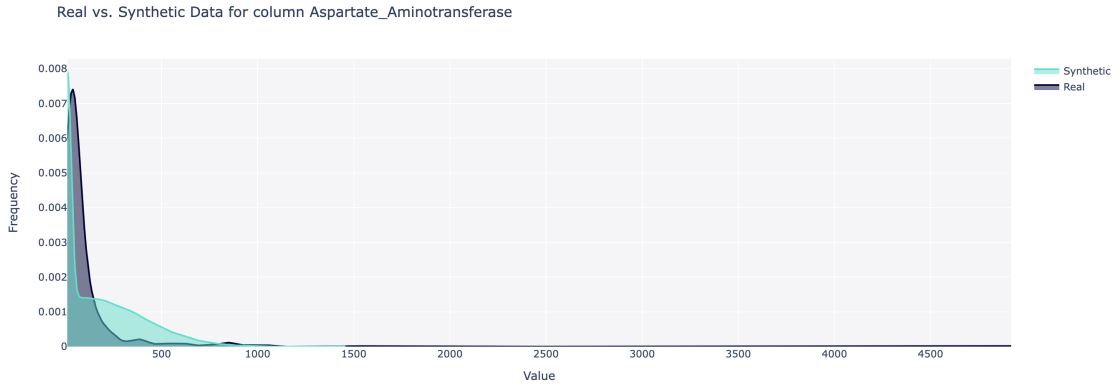


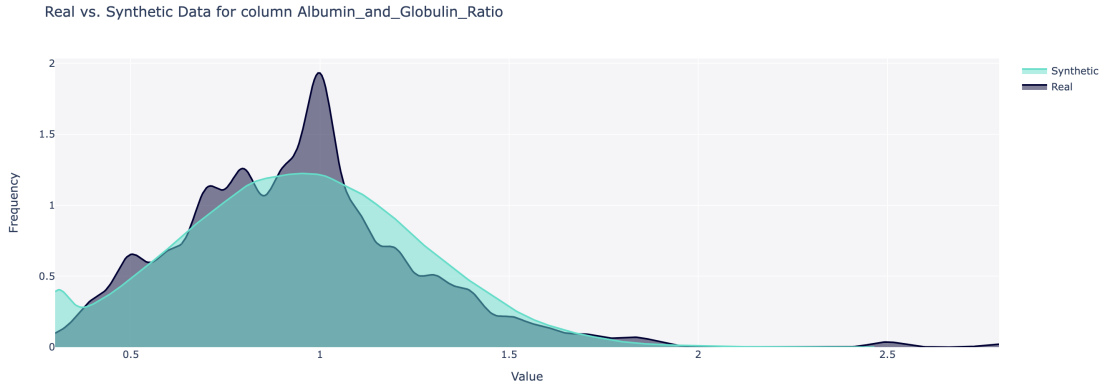
Figure 18: The KS Complement metric on Gaussian Copula

To gain a deeper understanding of the KS Complement scores across columns, we further visualized the minimum *i.e, Aspartate_Aminotransferase* as shown in Fig-

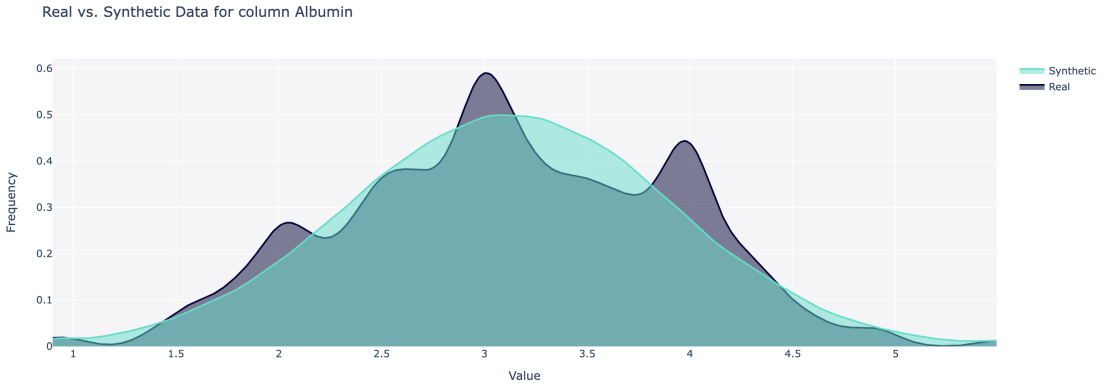
ure 19a, intermediate *i.e, Albumin_and_Globulin_Ratio* as shown in Figure 19b , and maximum *i.e Albumin* as shown in Figure 19c. These Histogram visualizations provide valuable insights into the distribution and alignment of the KS Complement metric across different attributes in the dataset as discussed in Section 3.3.1.



(a) Minimum KS Complement Score of Gaussian Copula



(b) Intermediate KS Complement Score of Gaussian Copula

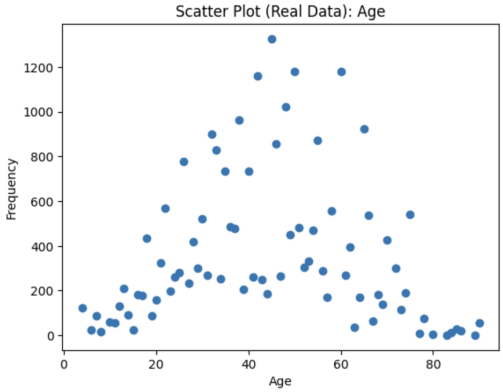


(c) Maximum KS Complement Score of Gaussian Copula

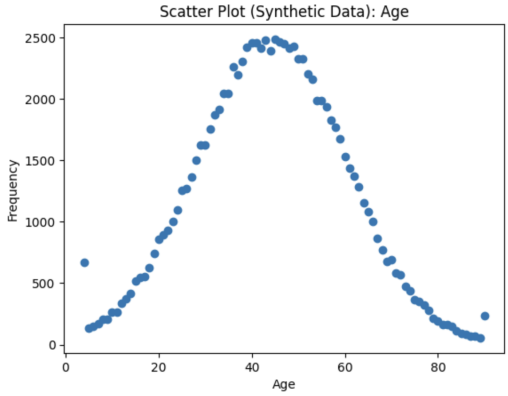
Figure 19: Minimum, Intermediate and Maximum KS Complement Score of Gaussian Copula

Scatter Plot Comparison of Real and Synthetic Data by Column

A comparison of scatter plots shows the relationship between related data points from two separate datasets, one real and the other synthetic. In this case, we're comparing all the columns mentioned in Table 1. The values from both datasets define the position of each point on the scatter plot, which represents a distinct data entry. By analyzing the dispersion and clustering of points in scatter plots from Figure 20 to Figure 30, We can evaluate the extent to which the synthetic data accurately reproduces the patterns and traits of the real data across those specific columns.

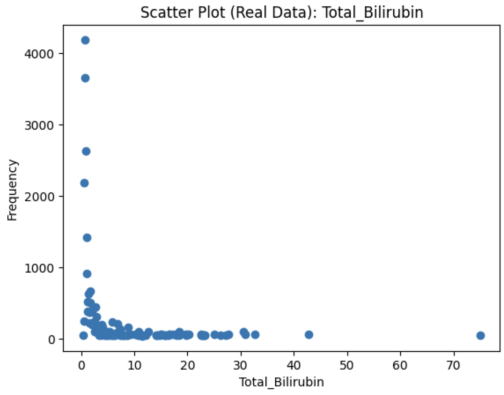


(a) Real Data Column

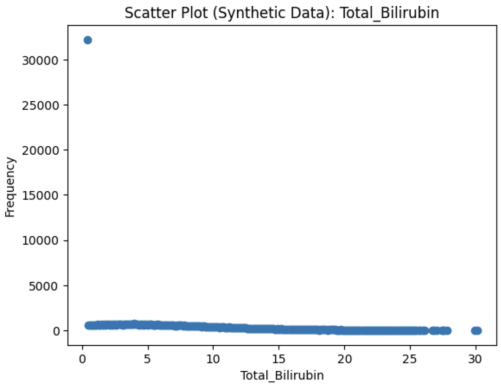


(b) Synthetic Data Column

Figure 20: *'Age'* column of the dataset (Gaussian Copula)

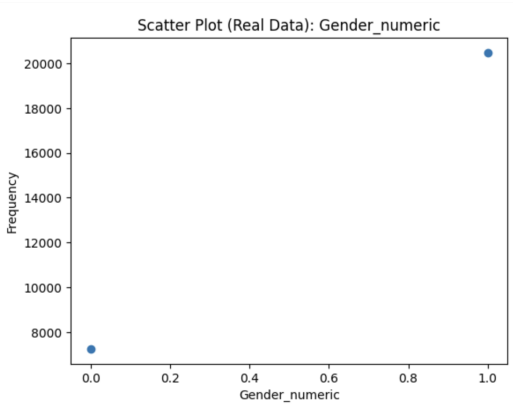


(a) Real Data Column

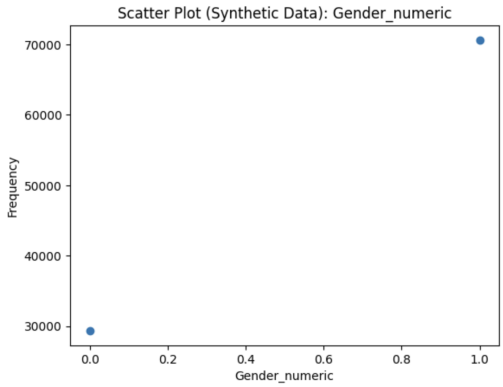


(b) Synthetic Data Column

Figure 21: *'Total_Bilirubin'* column of the dataset (Gaussian Copula)

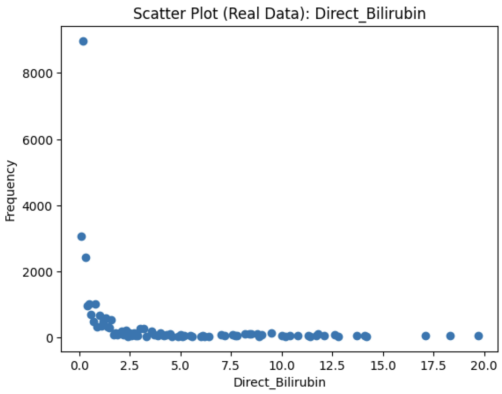


(a) Real Data Column

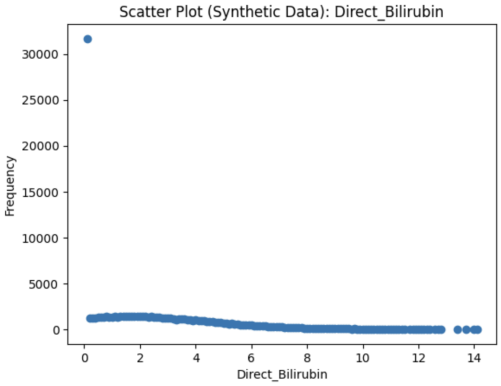


(b) Synthetic Data Column

Figure 22: *'Gender'* column of the dataset (Gaussian Copula)

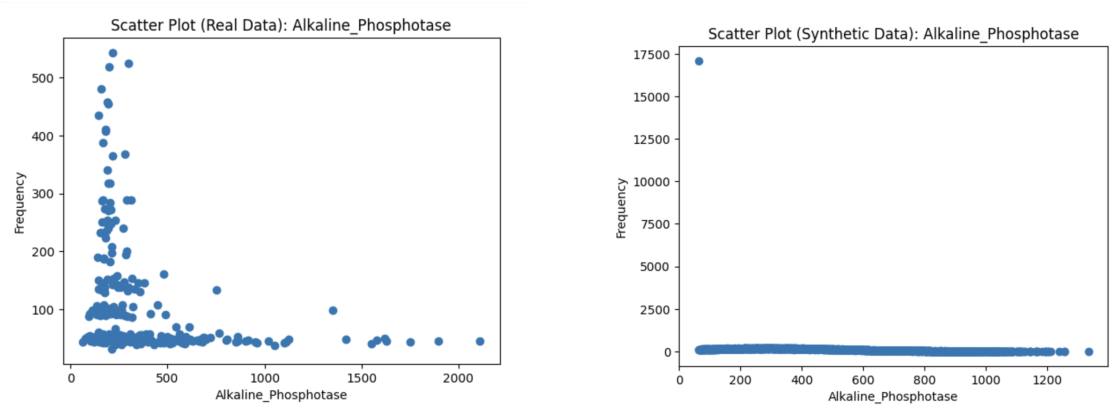


(a) Real Data Column



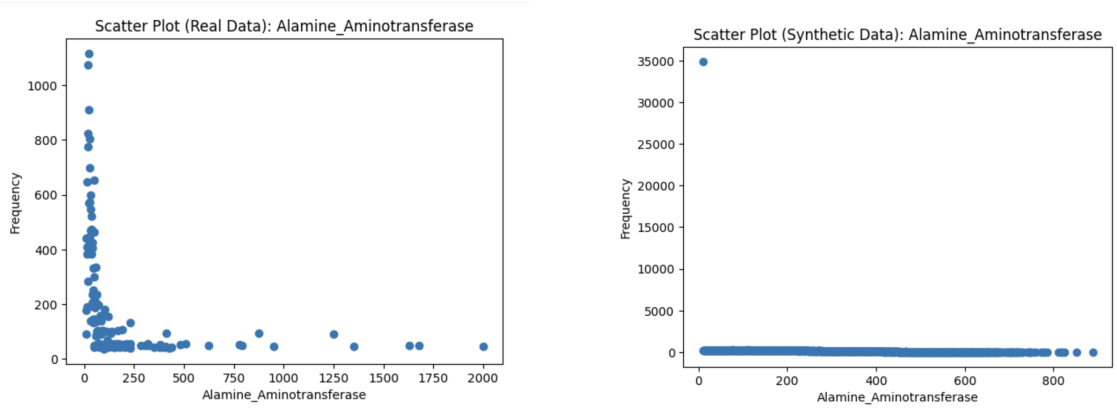
(b) Synthetic Data Column

Figure 23: *'Direct_ Bilirubin'* column of the dataset (Gaussian Copula)



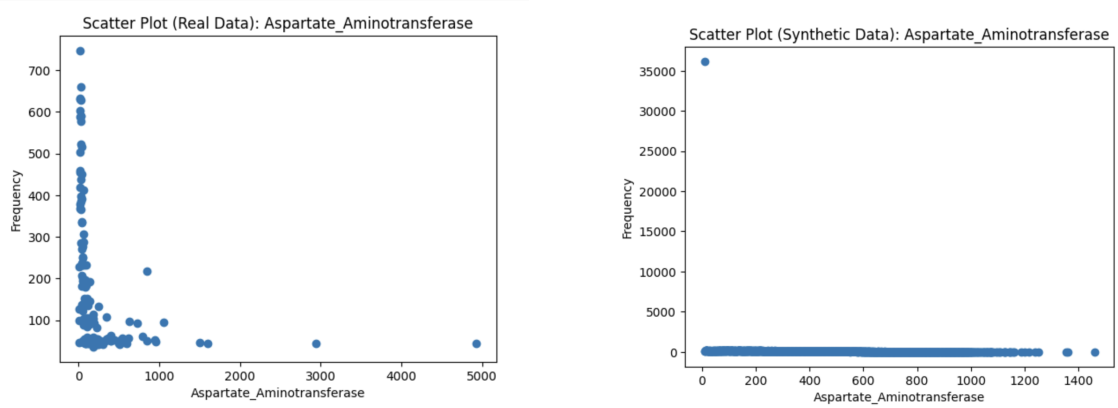
(a) Real Data Column (b) Synthetic Data Column

Figure 24: *'Alkaline_ Phosphotase'* column of the dataset (Gaussian Copula)



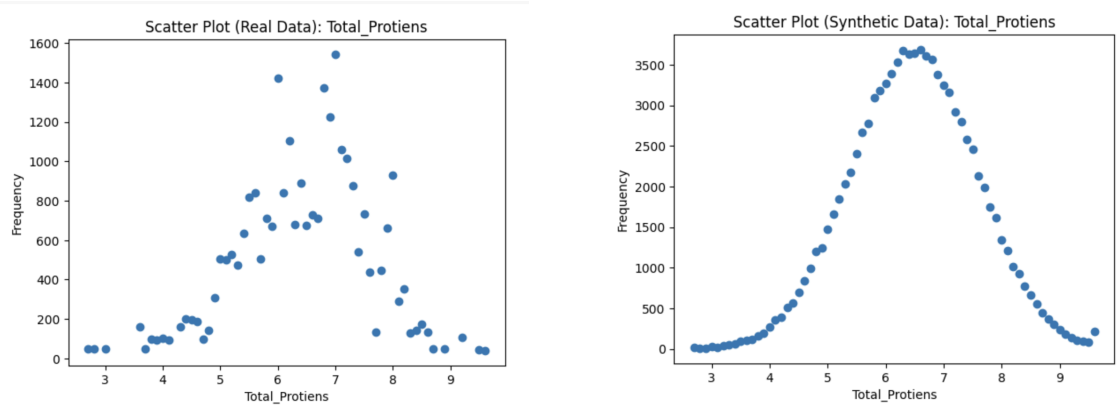
(a) Real Data Column (b) Synthetic Data Column

Figure 25: *'Alamine_ Aminotransferase'* column of the dataset (Gaussian Copula)



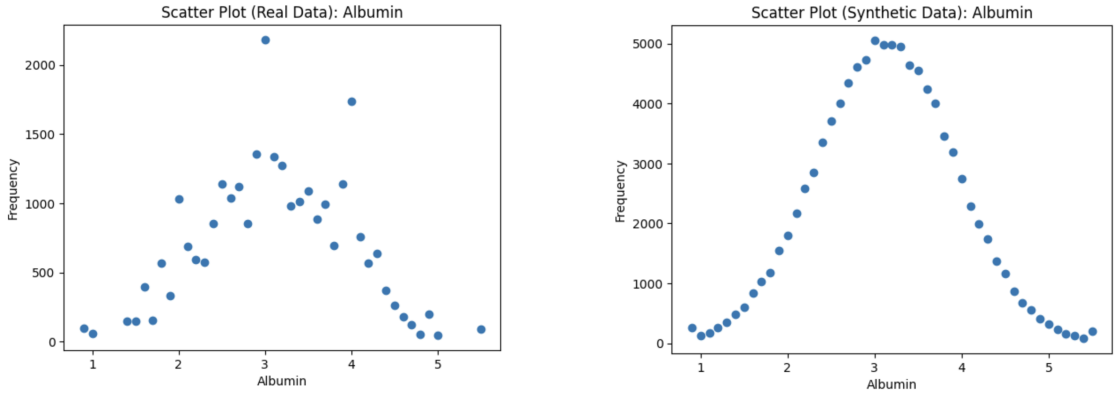
(a) Real Data Column (b) Synthetic Data Column

Figure 26: *'Aspartate_ Aminotransferase'* column of the dataset (Gaussian Copula)



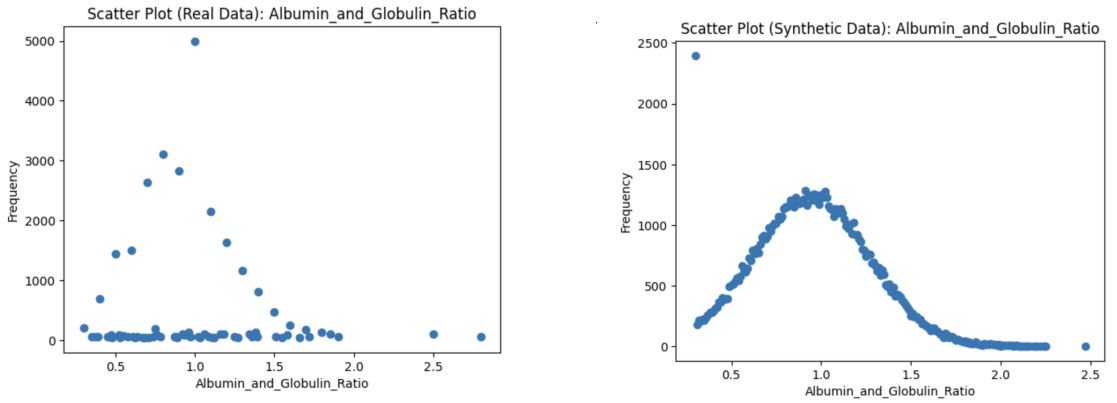
(a) Real Data Column (b) Synthetic Data Column

Figure 27: *'Total_ Protiens'* column of the dataset (Gaussian Copula)



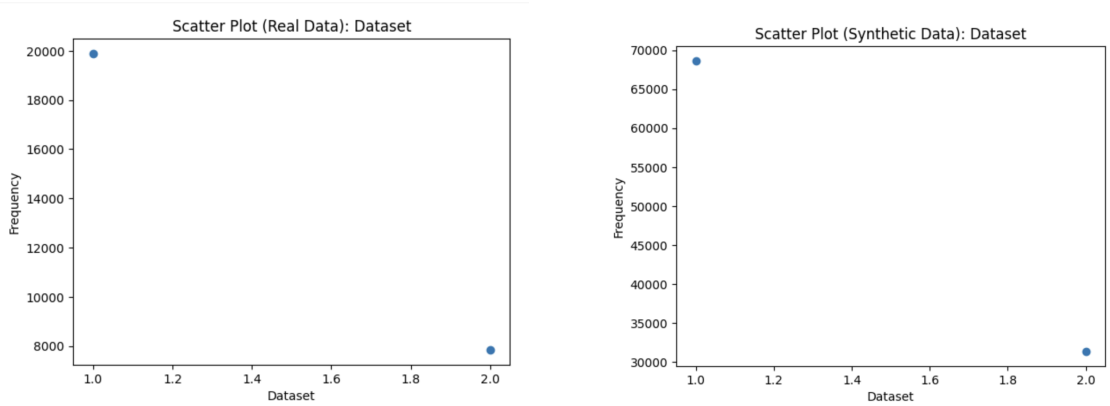
(a) Real Data Column (b) Synthetic Data Column

Figure 28: *'Albumin'* column of the dataset (Gaussian Copula)



(a) Real Data Column (b) Synthetic Data Column

Figure 29: *'Albumin_ and_ Globulin Ratio'* column of the dataset (Gaussian Copula)



(a) Real Data Column (b) Synthetic Data Column

Figure 30: *'Dataset'*(Classification) column of the dataset (Gaussian Copula)

4.2.3 Copula Generative Adversarial Network

The CopulaGAN synthesizer also employs diverse distribution modes, as elucidated in Section 3.2.3. In our analysis, we have specifically adopted the 'norm' (normal) distribution for all columns within the dataset. After the rigorous training of the CopulaGAN model, spanning 800 epochs as shown in Figure 31, a substantial dataset of 10,00,00 synthetic samples was generated.

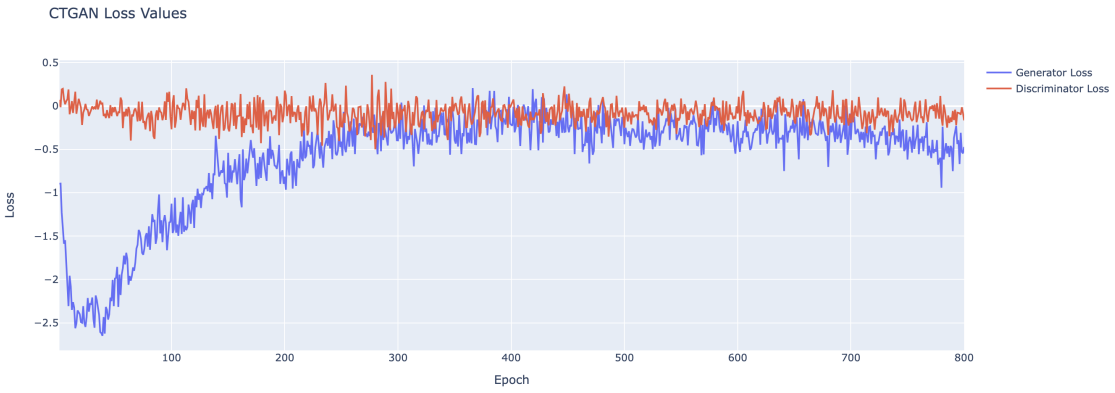


Figure 31: Copula GAN Generator and Discrimator Loss Value

The evaluation process involved the utilization of the KS Complement metric as mentioned in Section 3.3.2, a widely recognized and effective method for assessing the fidelity of synthetic data. This approach enabled an in-depth examination of the model's performance in generating synthetic data that closely aligns with the original dataset. The evaluation resulted in an overall average score of 0.94, as depicted in Figure 32.

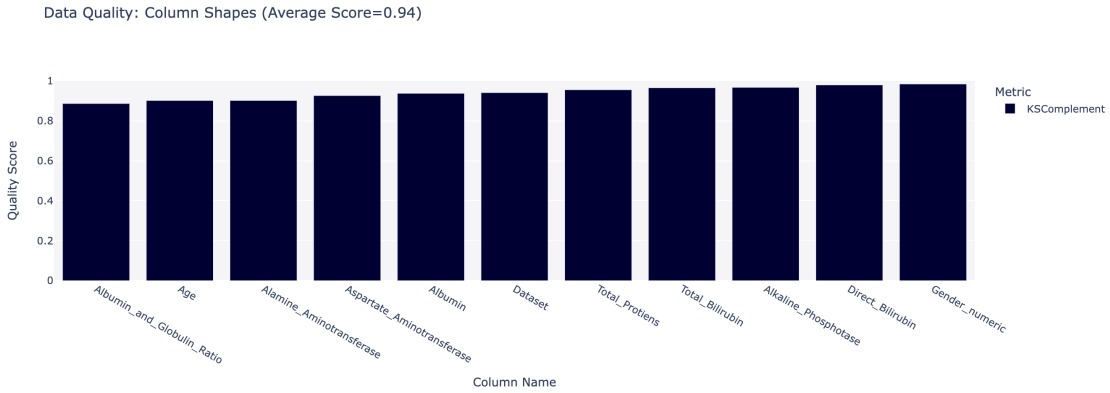
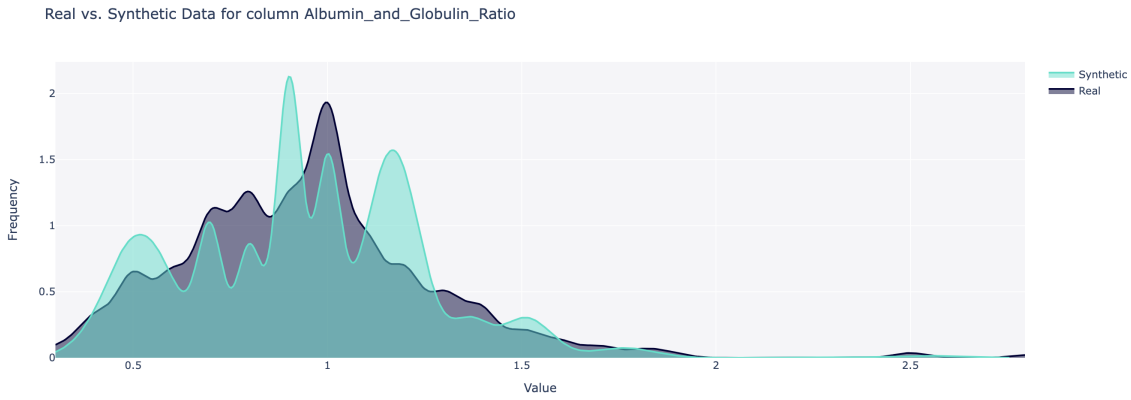
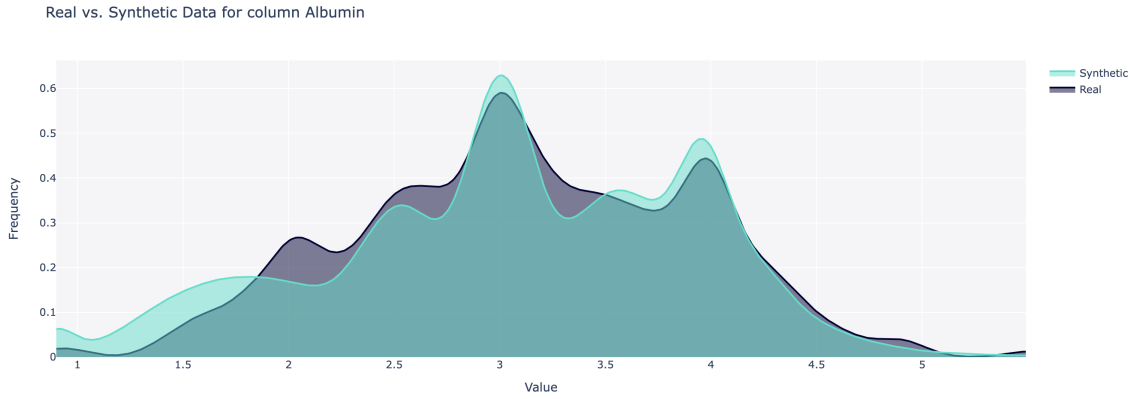


Figure 32: The KS Complement metric on CopulaGAN

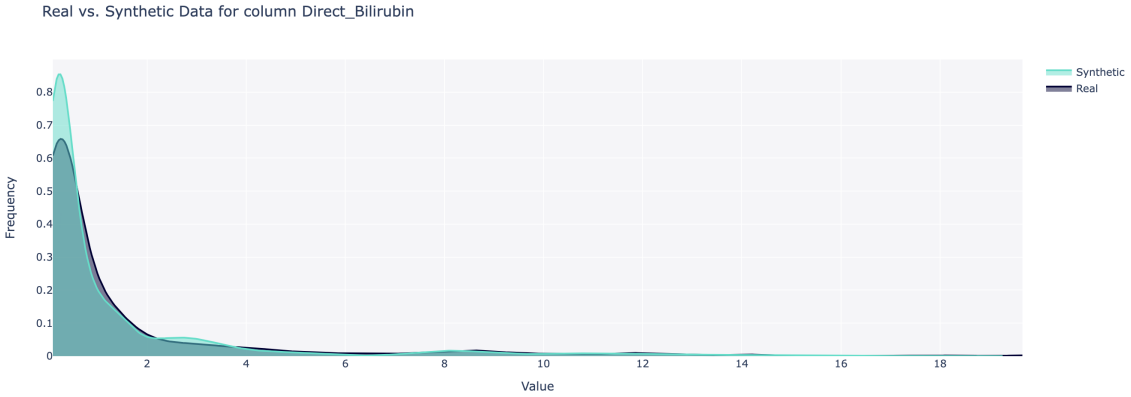
To gain a deeper understanding of the KS Complement scores across columns, we further visualized the minimum *i.e, Albumin_and_Globulin_Ratio* as shown in Figure 33a , intermediate *i.e, Albumin* as shown in Figure 33b, and maximum *i.e, Direct_Bilirubin* as shown in Figure 33c. These Histogram visualizations provide valuable insights into the distribution and alignment of the KS Complement metric across different attributes in the dataset as discussed in Section 3.3.1.



(a) Minimum KS Complement Score of CopulaGAN



(b) Intermediate KS Complement Score of CopulaGAN

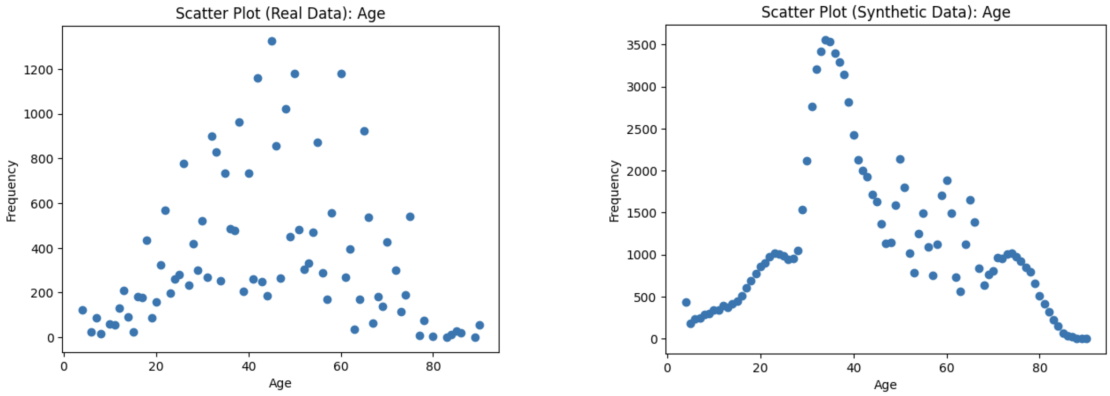


(c) Maximum KS Complement Score of CopulaGAN

Figure 33: Minimum, Intermediate and Maximum KS Complement Score of CopulaGAN

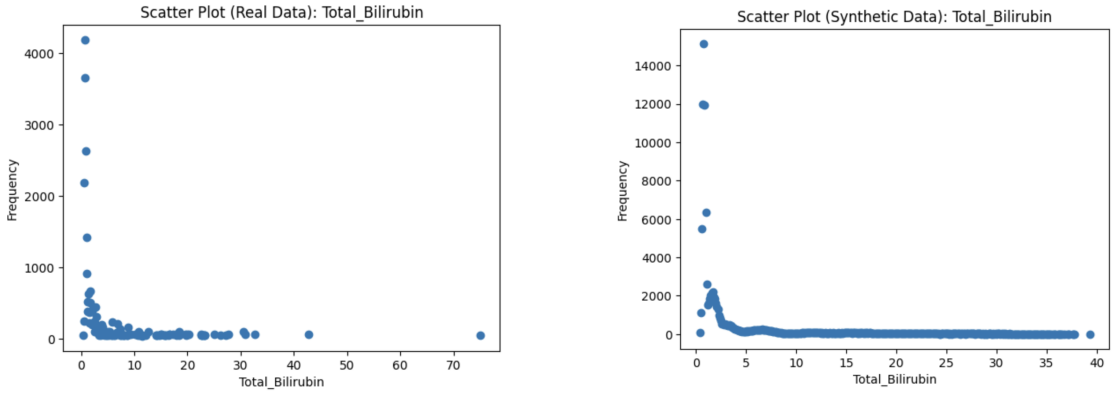
Scatter Plot Comparison of Real and Synthetic Data by Column

A comparison of scatter plots shows the relationship between related data points from two separate datasets, one real and the other synthetic. In this case, we're comparing all the columns mentioned in Table 1. The values from both datasets define the position of each point on the scatter plot, which represents a distinct data entry. By analyzing the dispersion and clustering of points in scatter plots from Figure 34 to Figure 44, We can evaluate the extent to which the synthetic data accurately reproduces the patterns and traits of the real data across those specific columns.



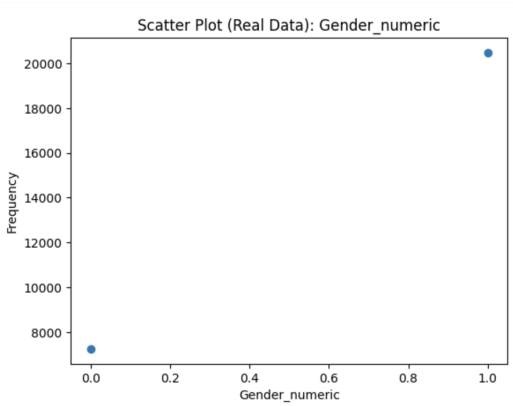
(a) Real Data Column (b) Synthetic Data Column

Figure 34: *'Age'* column of the dataset (CopulaGAN)

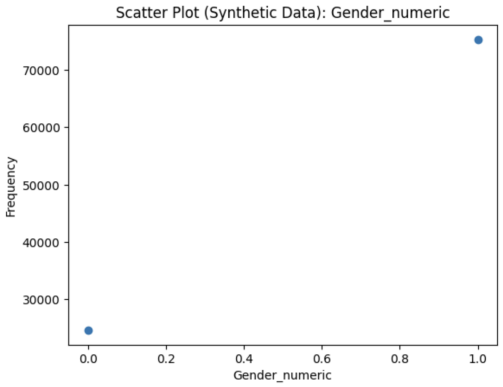


(a) Real Data Column (b) Synthetic Data Column

Figure 35: *'Total_Bilirubin'* column of the dataset (CopulaGAN)

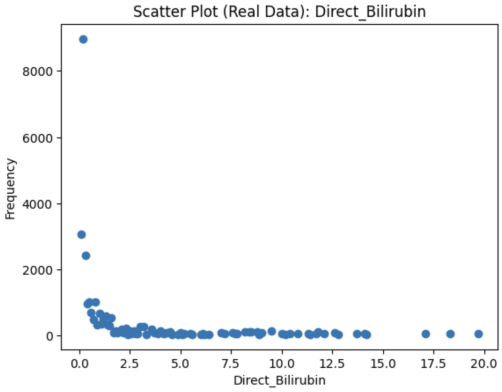


(a) Real Data Column

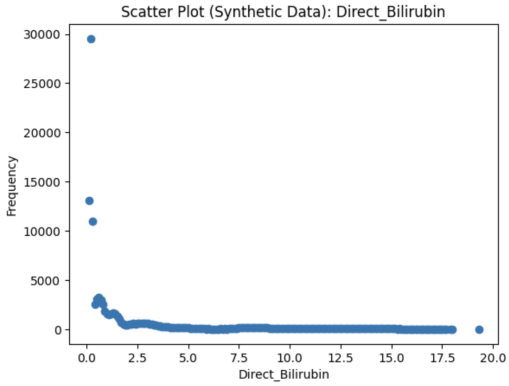


(b) Synthetic Data Column

Figure 36: *'Gender'* column of the dataset (CopulaGAN)

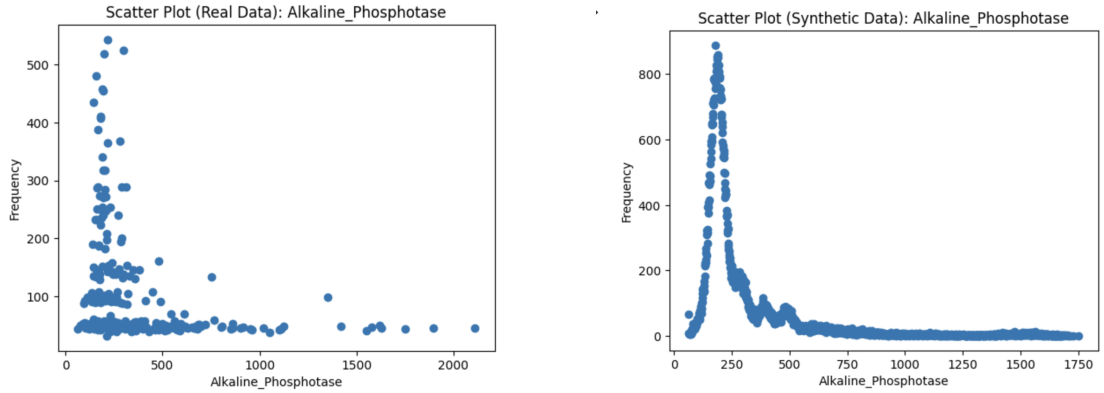


(a) Real Data Column



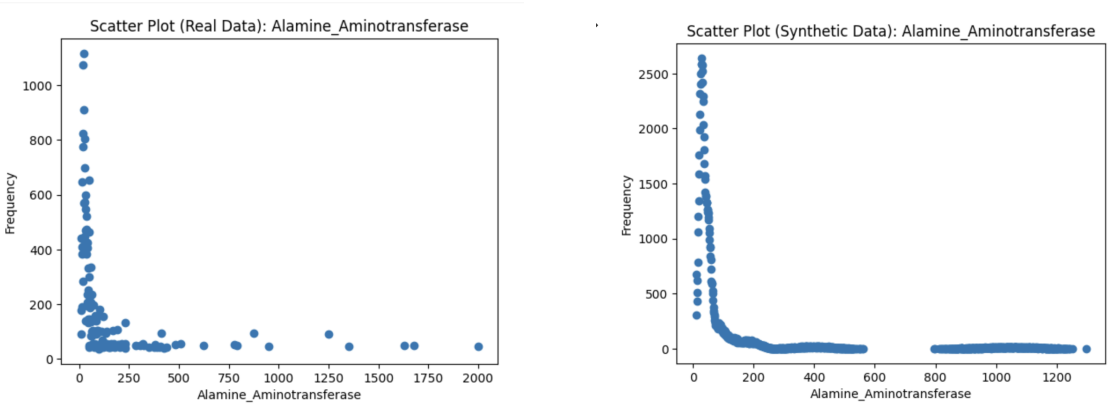
(b) Synthetic Data Column

Figure 37: *'Direct_ Bilirubin'* column of the dataset (CopulaGAN)



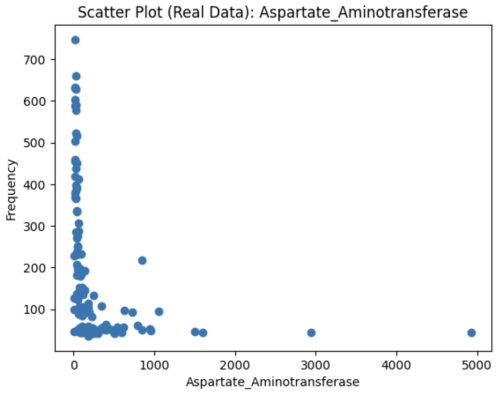
(a) Real Data Column (b) Synthetic Data Column

Figure 38: *'Alkaline_ Phosphotase'* column of the dataset (CopulaGAN)

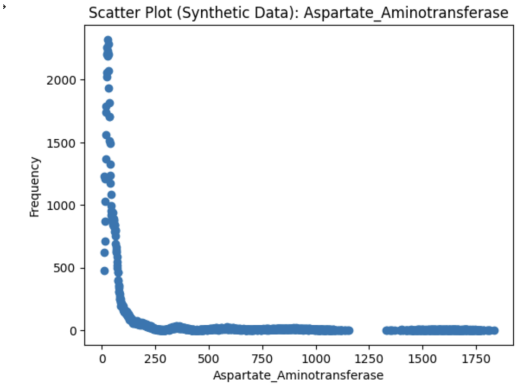


(a) Real Data Column (b) Synthetic Data Column

Figure 39: *'Alamine_ Aminotransferase'* column of the dataset (CopulaGAN)

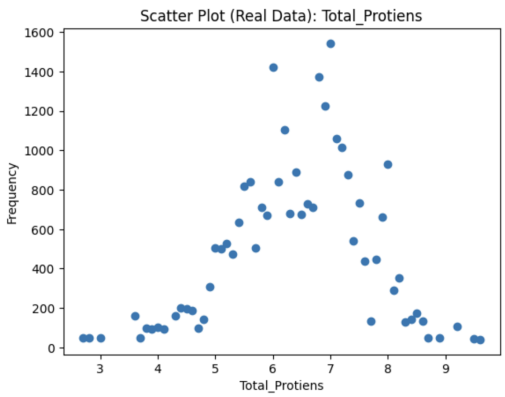


(a) Real Data Column

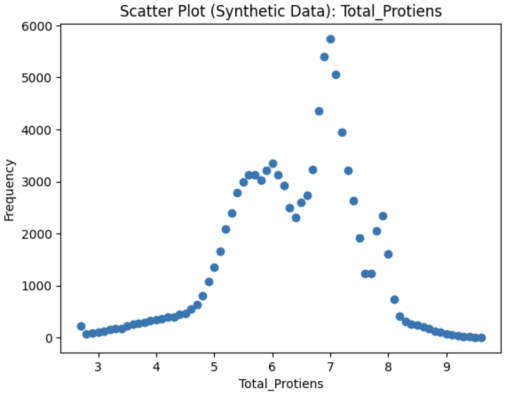


(b) Synthetic Data Column

Figure 40: *'Aspartate_ Aminotransferase'* column of the dataset (CopulaGAN)

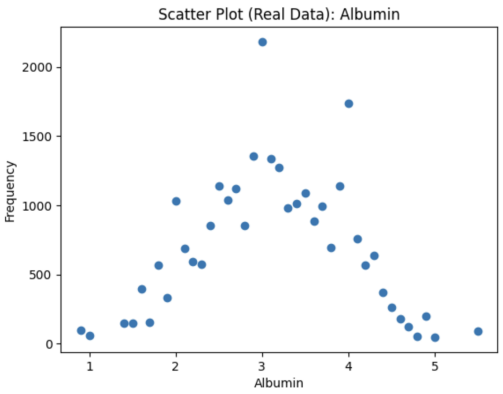


(a) Real Data Column

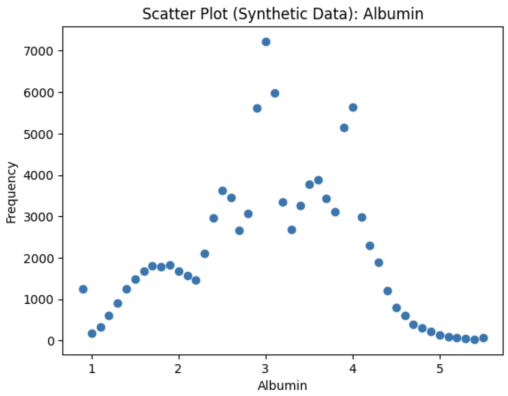


(b) Synthetic Data Column

Figure 41: *'Total_ Protiens'* column of the dataset (CopulaGAN)

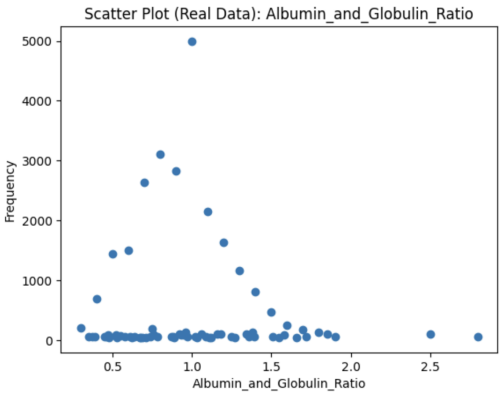


(a) Real Data Column

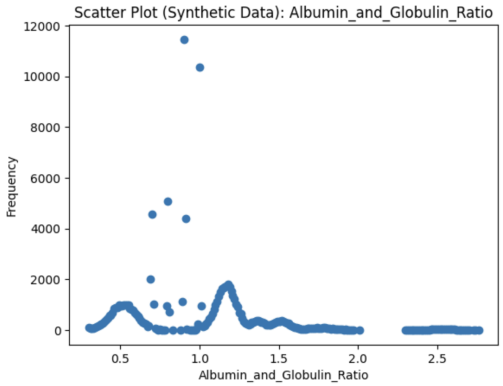


(b) Synthetic Data Column

Figure 42: *'Albumin'* column of the dataset (CopulaGAN)

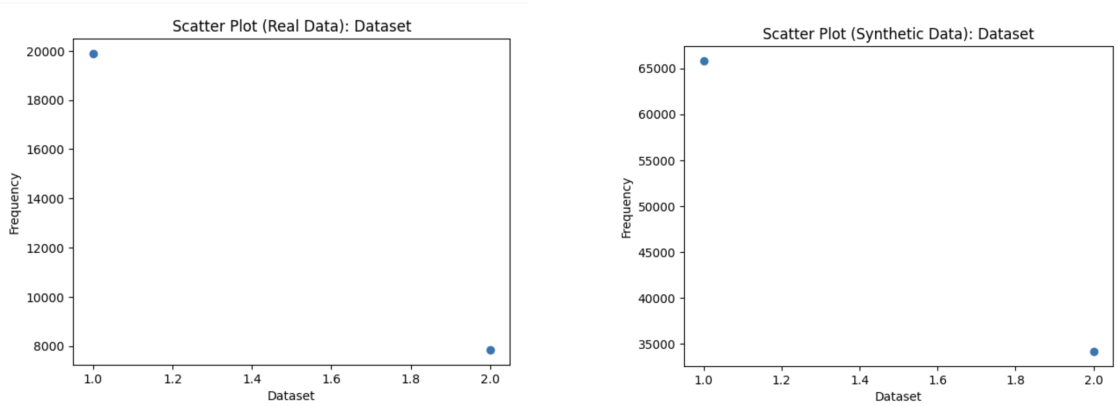


(a) Real Data Column



(b) Synthetic Data Column

Figure 43: *'Albumin_ and_ Globulin Ratio'* column of the dataset (CopulaGAN)



(a) Real Data Column (b) Synthetic Data Column

Figure 44: *'Dataset'*(Classification) column of the dataset (CopulaGAN)

4.3 Comparative Performance Analysis of CTGAN, Gaussian Copula, and Copula GAN Synthesizer

The present study employs a rigorous comparative analysis of performance, utilizing both the statistical metrics described in Section 3.3.2 and the detection metrics outlined in Section 3.3.4. The results are summarized in the provided Table 2, indicating that CTGAN emerges as the most optimal choice, exhibiting the highest scores among the three methods in both statistical evaluation and detection performance. Notably, CopulaGAN demonstrates competitive results and closely approximates the performance of CTGAN.

Table 2: The statistical and the detection metrics for CTGAN, GaussianCopula and CopulaGAN.

	Statistical Metrics	**Detection Metrics**
Synthesizer	**KSTest**	**Logistic Regression**
CTGAN	0.95	0.89
Gaussian Copula	0.83	0.65
CopulaGAN	0.94	0.89

4.4 Comparative Analysis of CTGAN and CopulaGAN with Gaussian Kernel Density Estimation Distribution

This approach differs from the one discussed in Section 3.2.3 where CopulaGAN was used with a 'norm' distribution. The inclusion of both the 'norm' and 'gaussian_KDE' parameters in this study aims to strike a balance between efficiency and sampling accuracy. Using the 'norm' distribution has its advantages as it allows training and sample generation faster, making it less resource-intensive. However Using the 'gaussian_KDE' distribution, training and sample generation is slower and it is resource extensive. As a result, it can generate upto only 10,000 sampless on a personal computer because of computational power limitation.

For the purpose of establishing a fair comparison, a set of 10,000 samples was generated utilizing the CTGAN model. Subsequently, this section presents a comprehensive comparison between CTGAN and CopulaGAN with the 'gaussian_kde' parameter, focusing on various criteria. These criteria encompass execution time and resource utilization analysis, assessment of generator and discriminator loss values and statistical metrics evaluation.

4.4.1 Execution Time and Resource Usage Analysis

With regard to model training time, Table 3 illustrates that the CTGAN outperforms CopulaGAN in terms of model training time requiring approximately 13.6 minutes. The sample generation process for CTGAN is remarkably quick, taking just 0.15 seconds with minimal memory usage. In contrast as shown in Table 4 CopulaGAN exhibits a longer model training time of around 17.6 minutes and significantly extended sample generation time of 25.7 minutes when generating 10,000 samples. Additionally, the memory usage associated with CopulaGAN is notably higher.

Furthermore, the tables provide a comprehensive set of additional metrics that contribute to the overall understanding of the models' performance characteristics.

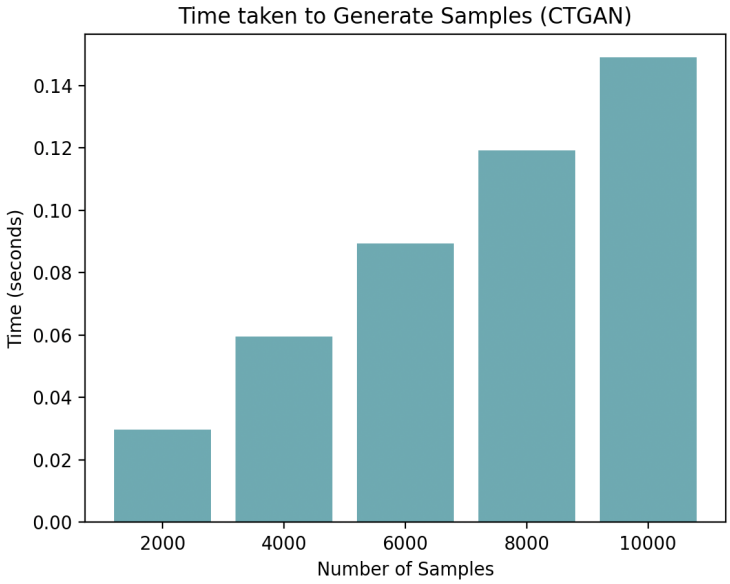
Table 3: Execution Time and Resource Usage Analysis of CTGAN

Metrics	**Model Training**	**Sample Generation**
Execution Time	819.92 sec	0.15 sec
User CPU Time	840.12 sec	0.14 sec
System CPU Time	435.92 sec	0.05 sec
Memory Usage	0.14 gb	0.06 gb

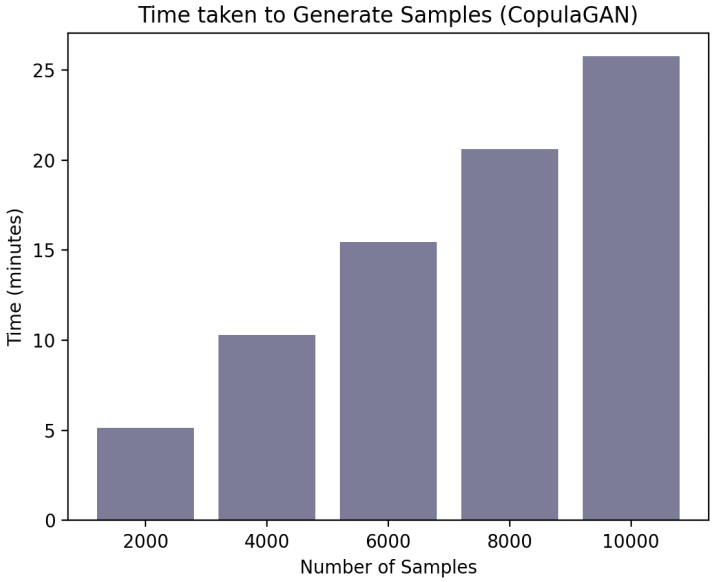
Table 4: Execution Time and Resource Usage Analysis of CopulaGAN

Metrics	**Model Training**	**Sample Generation**
Execution Time	1056.12 sec	1544.47 sec
User CPU Time	1074.58 sec	1661.43 sec
System CPU Time	501.59 sec	834 sec
Memory Usage	11.01 gb	4.18 gb

A bar plot study of the sample generation time distribution is shown in Figure 45. The Number of Samples are increment of 2000 up to a total of 10,000. In Figure 45a time taken is depicted in seconds and for CTGAN it takes a mere 0.02 seconds to generate 2000 samples and for 10,000 samples, it takes only 0.15 seconds. On the other hand, In Figure 45b time taken is depicted in minutes and for CopulaGAN it takes approximately 5.15 minutes to generate 2000 samples, while generating 10,000 samples extends to 25.75 minutes. This comparison underscores the significantly faster sample generation speed of CTGAN compared to CopulaGAN.



(a) Sample Generation Time Distribution: CTGAN



(b) Sample Generation Time Distribution: CopulaGAN

Figure 45: Sample Generation Time Distribution: Bar Plot Analysis

4.4.2 Generator and Discriminator Loss Value

In this research, the CTGAN and CopulaGAN synthesizer is trained on 800 epochs as shown in the Figure 46 and Figure 47. The CTGAN algorithm employs specific loss calculation formula as described in Section 3.3.3 which is also used by CopulaGAN synthesizer.The key to interpreting the loss values is to remember that the neural networks are adversaries. As one improves, the other must also improve just to keep its score consistent.

As described in Section 4.2.1, in case of 'Generator loss has stabilized at a negative value while the discriminator loss remains at 0 ' scenario it means that the generator has optimized, creating synthetic data that looks so real, the discriminator cannot tell it apart. The Figure 46 provides an illustration of this process, making it clear that the generator starts to stabilize at about 700 epochs and continues to improve after 800 epochs. Notably, the generator's negative loss value is -0.7 approx at 800 epoch.

In contrast, the Figure 47 shows a distinctive pattern, with the generator stabilizing at roughly 450 epochs and displaying a comparable negative loss value of roughly -0.7. This finding confirms that, as compared to CTGAN, CopulaGAN exhibits a noticeably improved stabilization performance.

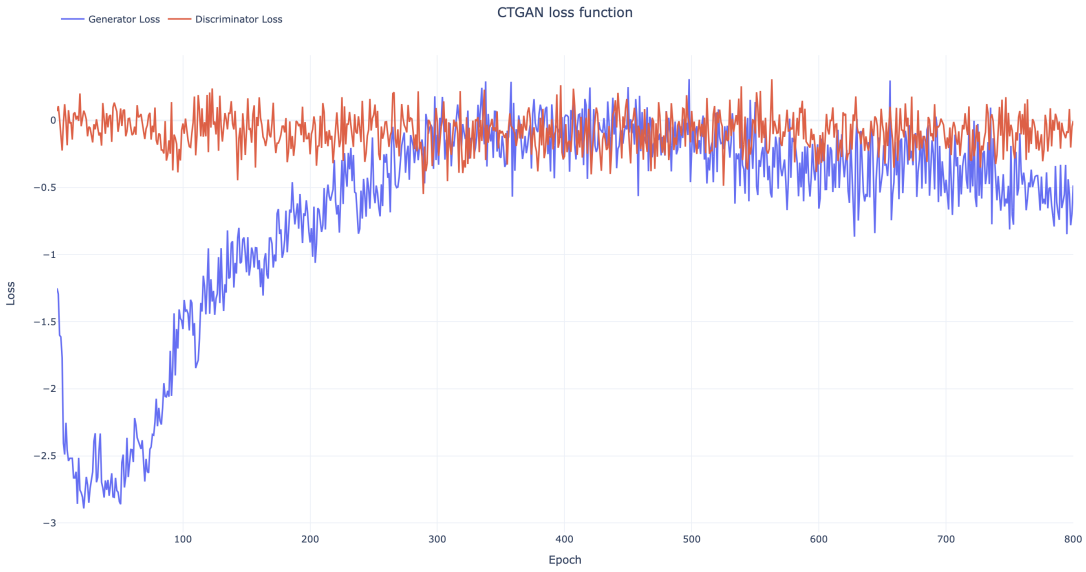


Figure 46: CTGAN Generator and Discrimator Loss Value

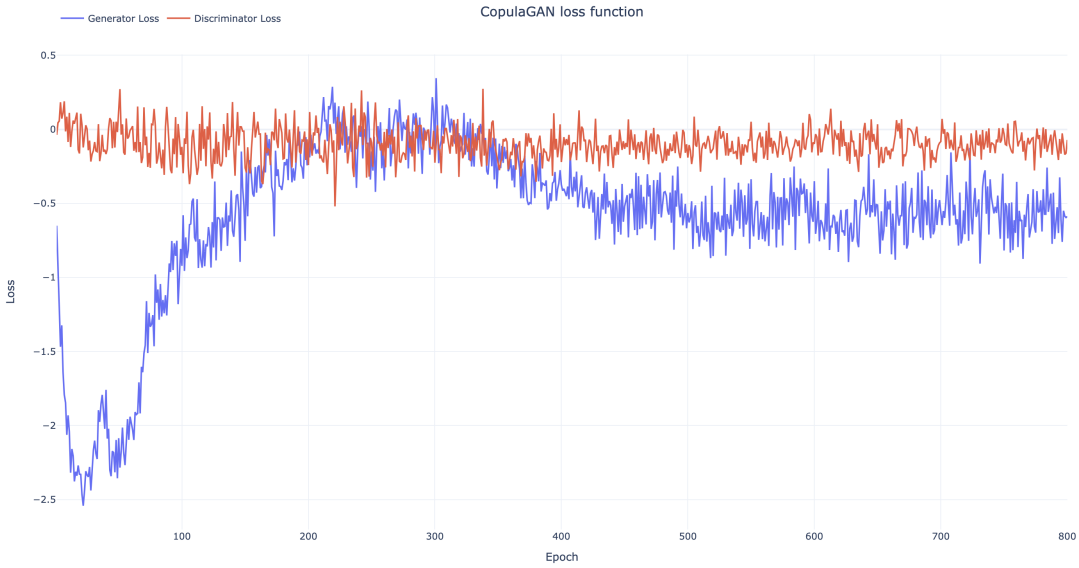


Figure 47: CopulaGAN Generator and Discrimator Loss Value with Gaussian KDE distribution

4.4.3 Statistical Metrics (KS Complement) Evaluation

The evaluation process involved the utilization of the KS Complement metric as mentioned in Section 3.3.2, a widely recognized and effective method for assessing the fidelity of synthetic data. This approach enabled an in-depth examination of the model's performance in generating synthetic data that closely aligns with the original dataset. The evaluation of CTGAN with 10,000 sample size resulted in an overall average score of 0.94, as depicted in Figure 48 and evaluation of CopulaGAN with 'gaussian_kde' and 10,000 sample size resulted in an overall average score of 0.96, as depicted in Figure 49. This observation underscores that the utilization of CopulaGAN with the 'gaussian_kde' distribution surpasses the accuracy of the CTGAN model, establishing itself as a high-performing model.

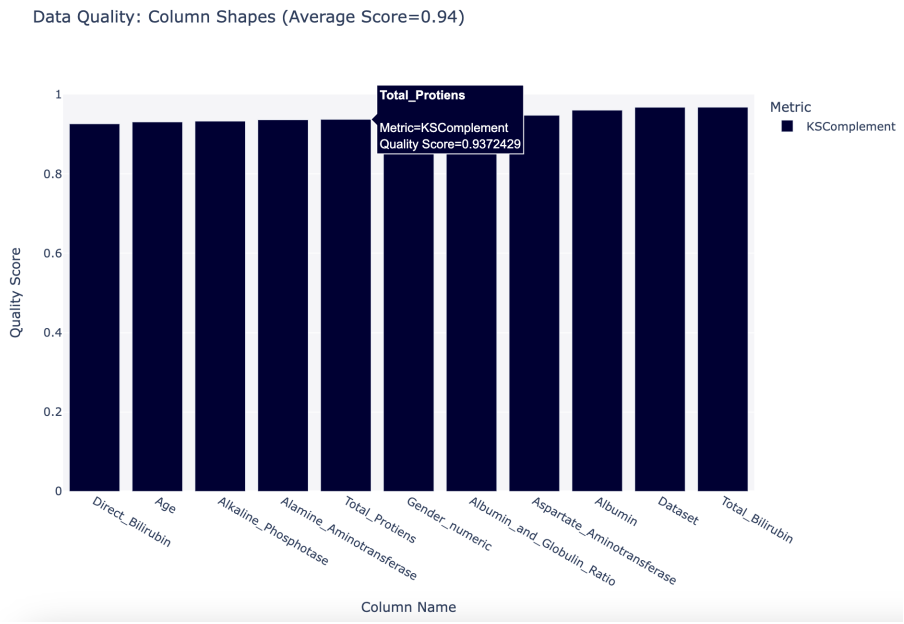


Figure 48: The KS Complement metric applied to CTGAN with a sample size of 10,000.

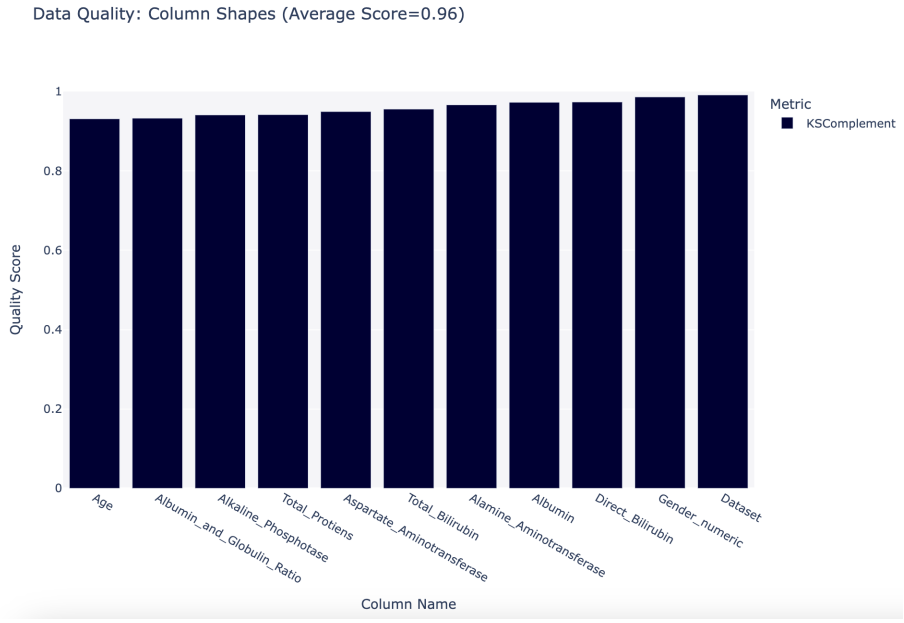
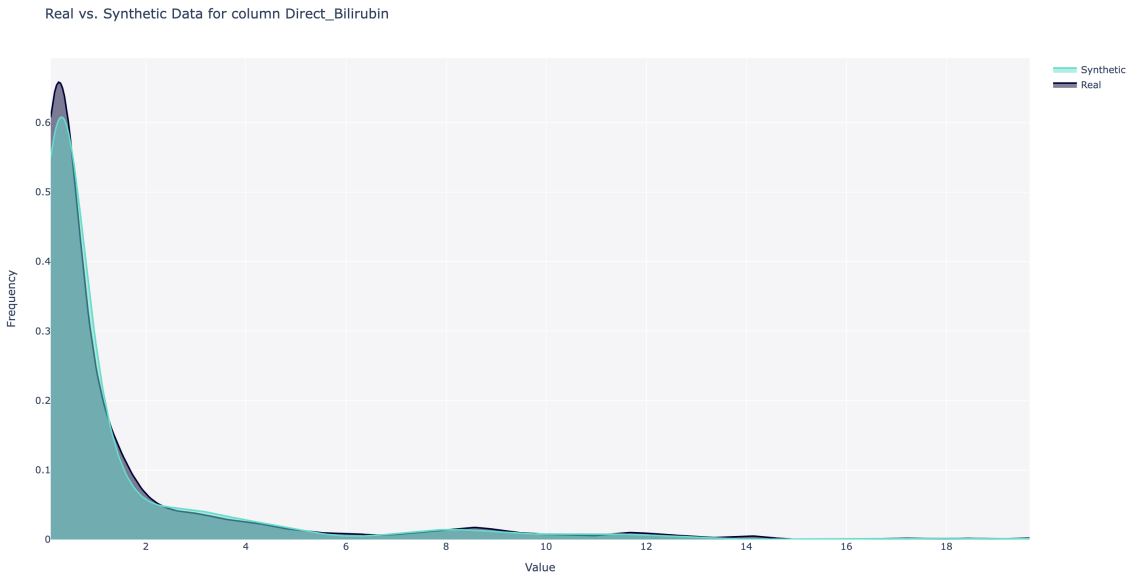


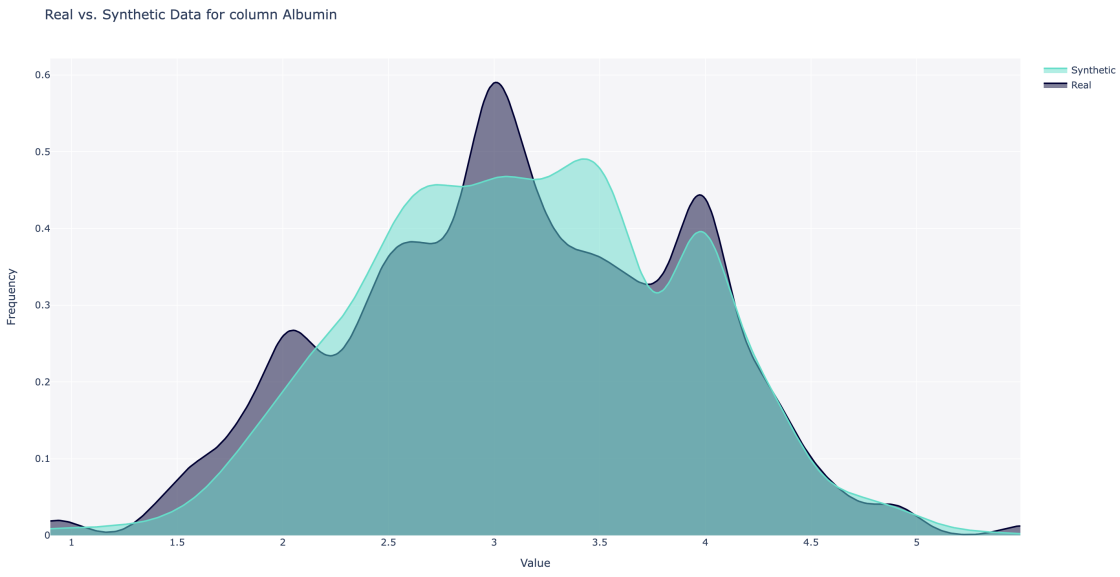
Figure 49: The KS Complement metric applied to CopulaGAN with a sample size of 10,000.

To gain a deeper understanding of the KS Complement scores across columns, we further visualized the minimum i.e, Direct_Bilirubin for CTGAN as shown in Figure 50a, minimum i.e, Albumin_and_Globulin_Ratio for CopulaGAN as shown in

Figure 51a, maximum i.e, Albumin for CTGAN as shown in Figure 50b and maximum i.e, Direct_Bilirubin for CTGAN as shown in Figure 51b. These Histogram visualizations provide valuable insights into the distribution and alignment of the KS Complement metric across different attributes in the dataset as discussed in Section 3.3

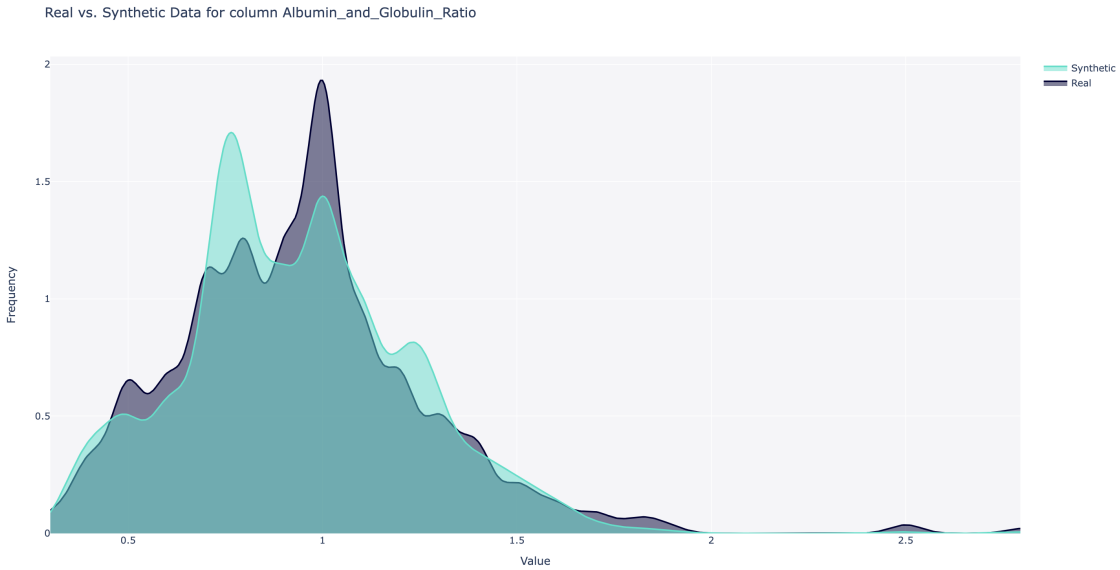


(a) Minimum KS Complement Score of CTGAN with a sample size of 10,000

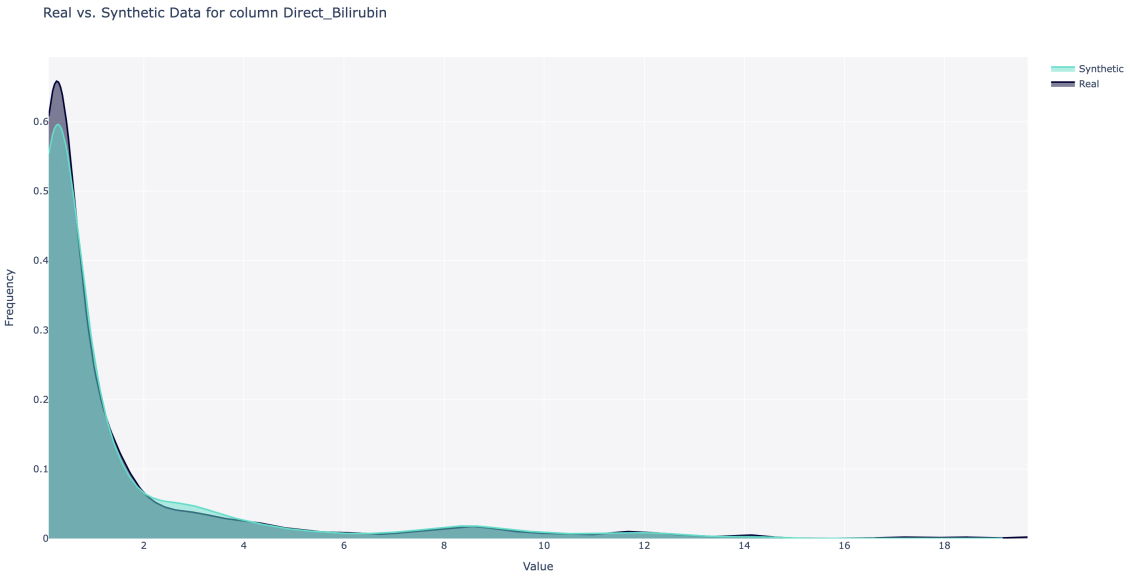


(b) Maximum KS Complement Score of CTGAN with a sample size of 10,000

Figure 50: Minimum and Maximum KS Complement Score of CTGAN with a sample size of 10,000



(a) Minimum KS Complement Score of CopulaGAN with a sample size of 10,000



(b) Maximum KS Complement Score of CopulaGAN with a sample size of 10,000

Figure 51: Minimum and Maximum KS Complement Score of CopulaGAN with a sample size of 10,000

5 Conclusion

This research examines popular Synthetic Data Generation models, namely CTGAN, Gaussian Copula, and CopulaGAN, for synthesizing Liver Patient Data. The "Indian Liver Patient Records" and "Liver Disease Patient Dataset 30K train data" datasets were used for generating synthetic samples.

Through visual representation and statistical analysis, we find that the analyzed algorithms, namely CTGAN, Gaussian Copula, and CopulaGAN, exhibit commendable performance in the domain of tabular synthetic data generation. Notably, CTGAN emerges as the standout performer, showcasing superior capabilities in this task. However, it's important to mention that CopulaGAN, especially when combined with the 'Gaussian_kde' distribution, achieves an even higher level of performance than CTGAN. This increased accuracy comes with a trade-off, as the training process requires much more time and resources. Nevertheless, all three algorithms demonstrate proficiency, indicating their viability and effectiveness in generating synthetic tabular data for diverse real-world applications.

For future work, my focus will be on improving the accuracy of all synthesizers through careful fine-tuning of hyperparameters and in-depth analysis of suitable parameters. Due to limited computational resources, I was able to generate only 10,000 samples using CopulaGAN with *'Gaussian_KDE'* distribution parameter. However, moving forward, I intend to explore methods to optimize and minimize the computational demands for more extensive sample generation.

Additionally, once the desired accuracy is achieved, I plan to train a classification algorithm using the generated data to classify liver cirrhosis in patients. By incorporating this phase, I anticipate that the model will have the opportunity to learn more effectively due to the availability of increased data samples.

References

[1] GBD 2017 Cirrhosis Collaborators. (2019). The global, regional, and national burden of cirrhosis by cause in 195 countries and territories, 1990–2017: a systematic analysis for the Global Burden of Disease Study 2017.

[2] Rockey, D. C., Caldwell, S. H., Goodman, Z. D., Nelson, R. C., & Smith, A. D. (2009). Liver Biopsy.

[3] Regev, A., Berho, M., Jeffers, L. J., Milikowski, C., Molina, E. G., Pyrsopoulos, N. T., Feng, Z. Z., Reddy, K. R., & Schiff, E. R. (2002). Sampling error and intraobserver variation in liver biopsy in patients with chronic HCV infection.

[4] Siddiqui, M. S., Yamada, G., Vuppalanchi, R., Van Natta, M., Loomba, R., Guy, C., Brandman, D., Tonanscia, J., Chalasani, N., Neuschwander-Tetri, B.,

& Sanyal, A. J. (2019). Diagnostic Accuracy of Non-Invasive Fibrosis Models to Detect Change in Fibrosis Stage.

[5] Zhang, D., Yin, C., Zeng, J., Yuan, X., & Zhang, P. (2019). Combining structured and unstructured data for predictive models: a deep learning approach.

[6] Chen, R. J., Lu, M. Y., Chen, T. Y., Williamson, D. F. K., & Mahmood, F. (2020). Synthetic data in machine learning for medicine and healthcare.

[7] Azizi, Z., Zheng, C., Mosquera, L., Pilote, L., & El Emam, K. (2019). Can synthetic data be a proxy for real clinical trial data? A validation study.

[8] Jadon, A., & Kumar, S. (2021). Leveraging Generative AI Models for Synthetic Data Generation in Healthcare: Balancing Research and Privacy.

[9] Alauthman, M., Aldweesh, A., Al-qerem, A., Aburub, F., Al-Smadi, Y., Abaker, A. M., Alzubi, O. R., & Alzubi, B. (2021). Tabular Data Generation to Improve Classification of Liver Disease Diagnosis.

[10] Hernandez, M., Epelde, G., Alberdi, A., Cilla, R., & Rankin, D. (2021). Synthetic data generation for tabular health records: A systematic review.

[11] Abedi, M., Hempel, L., Sadeghi, S., & Kirsten, T. (2020). GAN-Based Approaches for Generating Structured Data in the Medical Domain.

[12] Tucker, A., Wang, Z., Rotalinti, Y., & Myles, P. (2021). Generating high-fidelity synthetic patient data for assessing machine learning healthcare software.

[13] Imtiaz, S., Arsalan, M., Vlassov, V., & Sadre, R. (2021). Synthetic and private smart health care data generation using GANs.

[14] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., & Bharath, A. A. (2018). Generative Adversarial Networks: An Overview.

[15] Indian Liver Patient Records dataset. Retrieved from Kaggle: https://www.kaggle.com/datasets/uciml/indian-liver-patient-records (accessed on 21 July 2023).

[16] Liver Disease Patient Dataset. Retrieved from Kaggle: https://www.kaggle.com/datasets/abhi8923shriv/liver-disease-patient-dataset (accessed on 21 July 2023).

[17] Xu, L., Skoularidou, M., Cuesta-Infante, A., & Veeramachaneni, K. (2021). Modeling Tabular Data using Conditional GAN.

[18] Svensén, M., & Bishop, C. M. (2007). Pattern Recognition and Machine Learning.

[19] Li, Z., Zhao, Y., & Fu, J. (2020). SYNC: A Copula based Framework for Generating Synthetic Data from Aggregated Sources.

[20] Patki, N., Wedge, R., & Veeramachaneni, K. (2016). The Synthetic data vault.

[21] SDV Documentation. (n.d.). CopulaGANSynthesizer. Retrieved from https://docs.sdv.dev/sdv/single-table-data/modeling/synthesizers/copulagansynthesizer (accessed on 21 July 2023).

[22] Borji, A. (2021). Pros and Cons of GAN Evaluation Measures: New Developments. arXiv preprint arXiv:2103.09396.

[23] Theis, L., Oord, A. V. d., & Bethge, M. (2015). A note on the evaluation of generative models. arXiv preprint arXiv:1511.01844.

[24] Salimans, T., Ian, G., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training GANs. In Advances in Neural Information Processing Systems (pp. 2234–2242).

[25] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In Advances in Neural Information Processing Systems (pp. 6629–6640).

[26] Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 4401–4410).

[27] Bourou, S., El Saer, A., Velivassaki, T. H., Voulkidis, A., & Zahariadis, T. (2021). A Review of Tabular Data Synthesis Using GANs on an IDS Dataset.

[28] Berger, V. W., & Zhou, Y. (2006). Kolmogorov–Smirnov Tests.

[29] Levin, B. (1975). A Representation for Multinomial Cumulative Distribution Functions.

[30] SDV Metrics Documentation. (n.d.). Retrieved from https://docs.sdv.dev/sdmetrics/ (accessed on 21 July 2023).