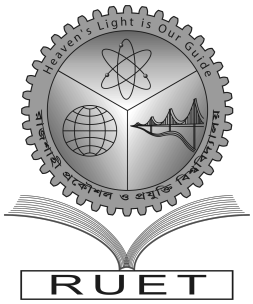
Heaven's Light is Our Guide



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Rajshahi University of Engineering & Technology, Bangladesh

Generative Adversarial Network based Synthetic Data Generation System

Authors

Md. Aukerul Moin Shuvo
Roll No. 1603061
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology

Md. Shohanoor Rahman
Roll No. 1603112
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology

Supervised by

Barshon Sen
Assistant Professor
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology

ACKNOWLEDGEMENT

It is only with the help of the Almighty that we have been able to write this entire thesis book. It's difficult to do without His grace. It is His will that we move on, therefore may He lead the way.

Our deepest appreciation and highest regard go to Barshon Sen, Assistant Professor in the Department of Computer Science and Engineering at Rajshahi University of Engineering Technology, for his invaluable assistance during the course of this study. His guidance and help were crucial at every stage. He encouraged us and provided the groundwork for us to become independent researchers. His presence and generosity were a pleasant reprieve at times of mental pressure. His method of inspiration was the driving force for this thesis. We are really fortunate to have him as our supervisor.

Also, we'd want to express our sincerest gratitude to the rest of the faculty for their time, expertise, and effort in creating a conducive atmosphere for academic research.

Finally, we'd like to express our appreciation to our wonderful parents.

November 5, 2022
RUET, Rajshahi

Md. Aukerul Moin Shuvo
& Md. Shohanoor Rahman

Heaven's Light is Our Guide

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Rajshahi University of Engineering & Technology, Bangladesh

CERTIFICATE

This is to certify that this thesis report entitled ***"Generative Adversarial Network based Synthetic Data Generation System"*** *submitted by* ***Md. Aukerul Moin Shuvo, Roll:1603061*** *and* ***Md. Shohanoor Rahman, Roll:1603112*** *in partial fulfillment of the requirement for the award of the degree of Bachelor of Science in Department of Computer Science & Engineering of Rajshahi University of Engineering & Technology, Bangladesh is a record of the candidates' own work carried out by them under my supervision. This thesis has not been submitted for the award of any other degree.*

Supervisor

_____

Barshon Sen
Assistant Professor
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology
Rajshahi-6204

External Examiner

_____

Md. Azmain Yakin Srizon
Lecturer
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology
Rajshahi-6204

ABSTRACT

The widespread availability and use of data have also raised concerns recently. Occasionally, data may be difficult to locate, such as highly secret or sensitive information. Real data have been being substituted with synthetic data to protect the privacy and for long-term sustainability. Generative adversarial networks (GANs) are becoming popular due to their ability to preserve statistical properties. Much effort has been put into establishing new GAN models but little has been done to preprocess data to increase GAN performance or to evaluate synthetic data quality. Our thesis work solely focuses on improving tabular data preparation before feeding it to the generative model. For preprocessing, we performed a thorough exploratory data analysis (EDA) of the training data, paying close attention to every detail in order to identify any issues. We performed missing value analysis, feature distribution analysis, statistical description, and class imbalance analysis and fixed the difficulties by comparing the probable solutions, such as comparing power and log transformations to reduce the skewness of feature distributions. We have used Conditional Tabular GAN (CTGAN) model to generate synthetic tabular data with better tuning by altering the hyperparameter values. We've extensively examined our created synthetic data in several dimensions to build a standard evaluation method. A similar EDA was applied on the generated data, conducted principal component analysis (PCA) in 2 ways, checked the correlations, extracted major features in 3 different ways, and validated the results of synthetic data with real data to ensure that both are conveying similar information. In addition, the statistical features of synthetic data were evaluated using four distinct statistical tests, such as the Kolmogorov–Smirnov test (KS-test), logistic detection, et cetera, along with a comparison of mutual information. Lastly, popular machine learning algorithms were used to evaluate the performance of synthetic data and validated it against real data. For example, the extreme gradient boost (XGBoost) model performed best for both real and synthetic data, with accuracy of 0.982 and 0.963 respectively.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Introduction

Real data is always preferred for making decisions, capturing insights, making predictions, etc. by researchers, data scientists, data analysts, data engineers and so on. while working with a problem. However, actual data is expensive because it is hard to obtain, gather and methodize to make it usable and because there is no universal scale for measuring quantitative and qualitative quality. In addition to these drawbacks, real data include personally identifiable information (such as age, income, medical history, an organization's personnel list, pay, etc.) that makes them very vulnerable to disclosure.

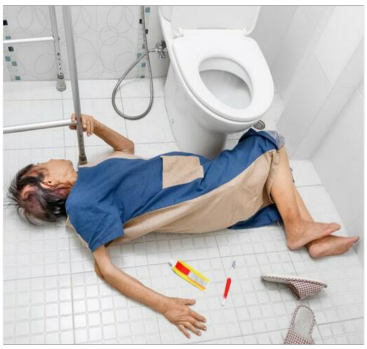
More so, there is a dearth of true data regarding unusual occurrences, or black swan events, such as meteor showers, vehicle accidents, and so on. In addition, data on sensitive events like heart attacks that occur in people's homes is very scarce, making it difficult for researchers, data engineers, scientists, and developers to create solutions that might really benefit humanity.

Therefore, synthetic data was introduced as a means of evading these problems associated with genuine data. Synthetic data is information that is not found in natural data and is created by an algorithm rather than from natural occurrences [1]. Figure 1.1 below depicts algorithmically created human faces. Since a large amount of data is required for many deep learning models, they may be utilized in a similar fashion in place of or in addition to actual data. For instance, a data set may have 10,000 rows, but ten times that amount of information is required to generalize a conclusion for a model [2].

Figure 1.1: Synthetic People [1]

Due to a lack of actual data, it is not feasible to train certain models. However, we may utilize synthetic data to do so. These advantages highlight the value of synthetic data in protecting the confidentiality of personal information. Figure 1.2 depicts a number of situations where it is possible to employ synthetic data.

Rare Golden Tiger



Heart Attack in Private Place

Falling Stars

Figure 1.2: Use Case of Synthetic Data (Collected from internet).

1.2 Problem Identification

Idealistically, data scientists would utilize actual data for all purposes. Real data would be the preferred resource for training and evaluating models, generating software test data, and presenting monthly sales figures since it reflects actual occurrences. However, actual data has

several disadvantages, including the requirement to locate, collect, and manage it. In addition, the amount of genuine data is often insufficient for the needs of the assignment. Furthermore, actual data may be subject to stringent regulations, such as when it includes personal information. Synthetic data may provide help in a number of these circumstances. The purpose of data synthesis is to generate new, never before seen data. This may serve as a distinct dataset for model development. The goal of this thesis is to develop sophisticated methods to produce synthetic data, such as Generative Adversarial Networks (GANs). The results of GANs have been shown to be successful. Even though it is well established that GANs may produce novel data, such as pictures, they have not been widely used for tabular data types such as discrete and categorical information, text, time series, etc. There are a number of factors to think about and methods to produce synthetic data [3]. Most generators need to know some kind of distributional criteria for real-world data [4]. However, simple methods can't promise that your fake data will have the same distribution, correlation, usability, and other characteristics that are really present in the real thing.

Therefore, the goal of this thesis's research is to develop a data pre-processing technique that will allow GANs to efficiently and effectively generate high-quality synthetic data, including but not limited to text data, discrete data, continuous data, time-series data, etc., that closely mimics the meaning and variability of real data in terms of its patterns, distributions, correlations, and confidentiality.

Additionally, an evaluation system that guarantees the integrity and usefulness of the synthetic data has been proposed.

1.3 Motivation

The first and most important goal of this thesis was to find a solution to the issue of insufficiently big data sets that are needed to properly train deep learning models so that they may effectively generalize a given problem.

The second reason was to address the dearth of actual data on critical situations, such as a heart attack or a panic attack occurring in a bedroom or other personal space.

The third reason was to protect the confidentiality of private and sensitive information such as bank account data, age, passport number, and other similar information.

Purpose of this thesis was to propose an effective real-data preprocessing system for generating high-quality, factual synthetic data using GANs. The subsequent purpose included evaluating the synthetic data across multiple statistical dimensions and comparing them to real data using a variety of machine learning methods. Using this approach, simulated data may be analyzed to the same depth as genuine data.

1.4 Goals

This study was done with the goals of improving existing systems for generating high-quality synthetic data by including techniques like pre-processing of real data before feeding them to a GAN network and a thorough assessment system for generated synthetic data in as many dimensions as feasible.

1.5 Benefits, Ethics and Sustainability

The development of machine learning, artificial intelligence, and big data has spawned new problems, such as the theft of personal information. Certain concerns have led to new legislation [5] that aims to make these data production practices more sustainable. Additionally, conventional software engineering, artificial intelligence, data science, and data analytics sometimes need vast quantities of data. In light of these facts, a data generating method that benefits all academics, companies, and people to develop more sustainable new technologies is required. These advantages mirror the United Nations' sustainable development objectives, especially the ninth goal: *"Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation"*. [6].

1.6 Research Objectives

The objectives of this thesis are as follows:

(a) To offer an improved approach for data pre-processing to produce synthetic data.

(b) To provide accurate and credible synthetic tabular data.

(c) To provide an improved assessment method for synthetic tabular data in several dimensions.

1.7 Delimitations

This thesis does not focus on developing a software tool, but rather on developing a process to produce synthetic data relatively effectively and rapidly, as well as building an appropriate assessment mechanism to measure the quality of generated synthetic data. In order to help the GAN model train more effectively, this also emphasizes preparation strategies for the input. However, the creation of other forms of synthetic data, such as photographs, movies, etc., is beyond the scope of this study.

1.8 Thesis Organization

The **second chapter** introduces the foundations of machine learning, deep learning, and GANs as they pertain to the synthesis of synthetic data. The **third chapter** describes the research approach used to produce synthetic tabular data, as well as its implementation. The **fourth chapter** includes a study of synthetically created data and a comprehensive assessment system. The **last chapter** provides a summary of the study, research questions, and responses, as well as a discussion of the larger future objective of this research.

1.9 Conclusion

This chapter describes what synthetic data are, their use scenarios, and the forms of challenges that may be overcome by using them. In addition, we have described the problem we attempted to resolve throughout the thesis work, our reason for doing so, and the objectives of this thesis work. Along with the constraints, we have also addressed the relevance, ethics, and sustainability of this thesis work. This chapter basically presents the book's outline and explains the intent of our research work.

Chapter 2

Background Study and Literature Review

2.1 Introduction

To comprehend our contributions, various machine learning-related topics must be understood. In order to facilitate a better understanding of our technique, we will now expound on certain fundamental machine learning and Generative Adversarial Networks (GANs) ideas.

2.2 Machine Learning

Machine learning provides computer systems the ability to learn by data and advanced statistical strategies, while not the need of being explicitly programmed. It uses statistical learning and optimization strategies that allow computers to analyze data sets and determine patterns to form nearly-accurate predictions [7]. Machine learning is regarding extracting knowledge from data. It's an exploration field at the intersection of statistics, artificial intelligence, and computing and is additionally called predictive analytics or statistical learning. Alike the human brain gains knowledge and understanding, machine learning depends on input, like training data or knowledge graphs, to grasp entities, domains and therefore the connections between them [8]. It looks for patterns in data, thus it will later create inferences supported by the examples provided. The goal of machine learning is to permit computers to learn autonomously while no human intervention or assistance and modify the reference model consequently.

Human intervention or reinforcement influences the performance of machine learning algorithms to variable degrees. The later subsections section discusses the four most significant machine learning models.

2.2.1 Supervised Learning

Supervised learning is more controlled and less biased than Unsupervised Learning. The user provides pairs of inputs and intended outputs to the algorithmic program; therefore, the algorithmic program develops a way to create the desired output given an input [9]. Specifically, the algorithmic program is capable of creating a result for an unknown input without human interaction. Let's consider the following situation as an illustration: the user provides the algorithmic program with a significant number of emails (the input) and data indicating whether these emails are spam (which is the desired output) or not. Given a brand-new email, the algorithmic program can tell whether it is a spam or not.

The usual algorithmic program for supervised machine learning consists of about three components.

(a) **A decision process**: A formula of computations or other procedures that ”guesses” what reasonable pattern your algorithmic process is attempting to find based on the data provided [10].

(b) **An error function**: A method for evaluating the accuracy of an estimate by comparing it to known instances (where available). Did the choice method dig right? If not, how does one quantify “how bad” the miss was?

(c) **A technique for updating or optimizing**: An approach in which the algorithmic program's observation of a miss modifies how the decision-making procedure affects the end conclusion, such that the next time the number of mistake will be reduced [10].

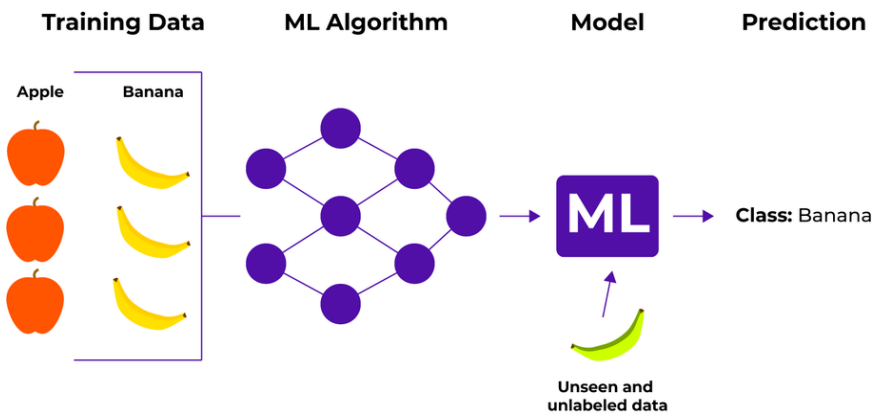


Figure 2.1: Example of How Supervised Learning Works (Image taken from internet)

In the image example of figure 2.1, we want to make a machine learning model that can estimate whether there is an apple or a banana on a conveyor belt. We must therefore use pictures of both apples and bananas, and for each picture, it must be stated whether it is an apple or a banana that is depicted. Once the machine learning model is trained, it can then be used to make predictions on new data that does not have a label. The important thing is that there is a label present during the training of the model, otherwise we cannot do supervised learning.

2.2.2 Unsupervised Learning

The algorithm is given unlabeled data from which it pulls undiscovered patterns or insights. The system never comprehends the correct output with confidence [11]. Instead, it derives conclusions from datasets on the expected result.

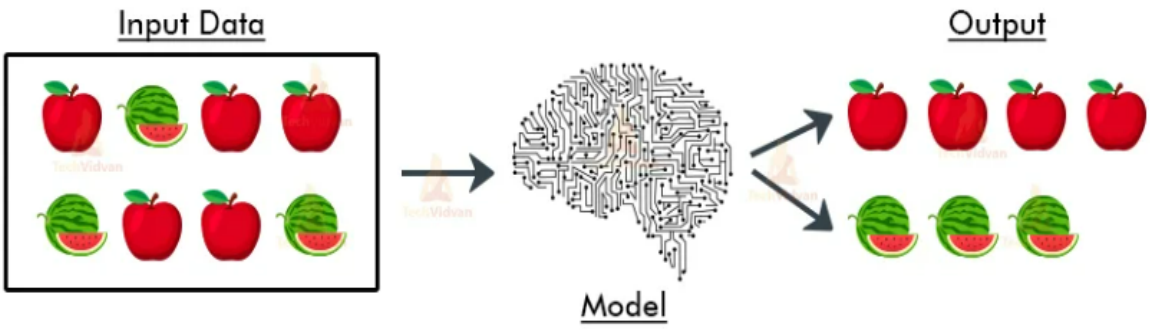


Figure 2.2: Example of Unsupervised Learning (Image taken from internet).

In the image example of figure 2.2, we want to make a machine learning model that can differentiate between apples and watermelons but we have not labelled which one is apple and which one is watermelon. Once the machine learning model is trained, it can then be used to make cluster of different fruits. The important thing is that there is no label present during the training of the model and the machine will just cluster the fruits of similar kind

2.2.3 Semi-supervised Learning

Given a set of partially labeled data, the algorithm completes its task by utilizing the labeled data to determine the parameters for interpreting the unlabeled data. It employs a smaller labeled data set during training to facilitate classification and feature extraction from a larger unlabeled data set [12].

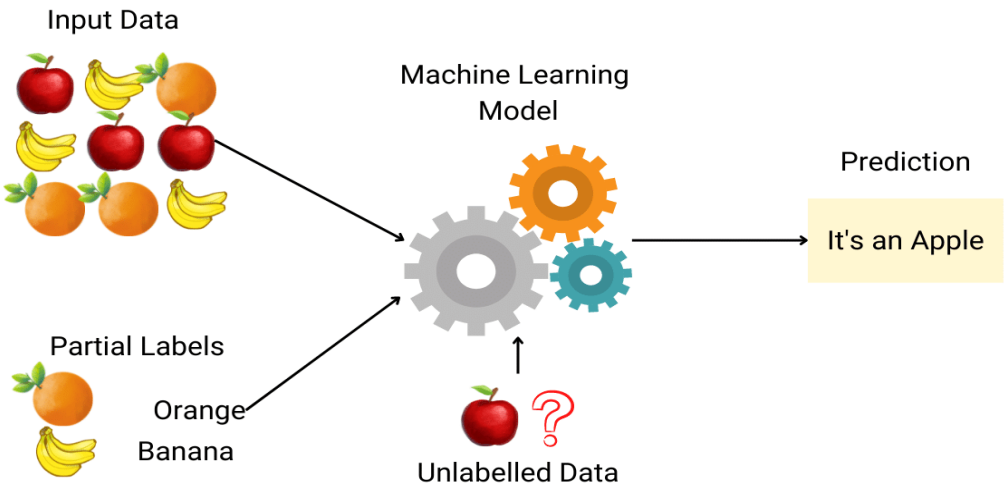


Figure 2.3: Example of Semi-supervised Learning (Image taken from internet)

Here is an example in figure 2.3 to make it clear. There is a bucket consists of three fruits, apple, banana, and orange. We have captured the image of all three but labeled only the orange and banana images. Here, the machine first will classify the new apple image as not a banana and not orange. Then we will observe these predictions and label them as apples. Then retraining the model with that label will give it the ability to classify apple images as an apple.

2.2.4 Reinforcement Learning

It is a Machine Learning algorithm that allows software agents and machines to automatically determine the ideal behaviour within a specific context to maximize its performance. It does not have a labelled dataset or results associated with data so the only way to perform a given task is to learn from experience. For every correct action or decision of an algorithm, it is rewarded with positive reinforcement whereas, for every incorrect action, it is rewarded with negative reinforcement. In this way, it learns which actions are needed to perform and which are not. Reinforcement learning can, therefore, help in industrial automation as well as the gaming sector primarily [13].

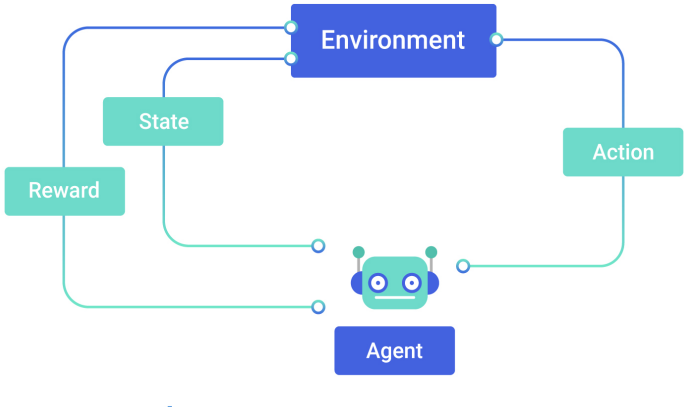


Figure 2.4: Example of How Reinforcement Learning Works (Image taken from internet)

2.3 Artificial Neural Networks (ANNs)

The operation of biological neurons in the brain is the basis for neural networks. These neurons get a particular input and turn out a particular output looking on the input. A neural network integrates several layers of neurons to construct complex structures capable of learning complex nonlinear functions [14]. The parameters of the model are the learned weights and thresholds of the neurons, which are acquired during training.

In the next sub-section, we will discuss about backpropagation and gradient descent.

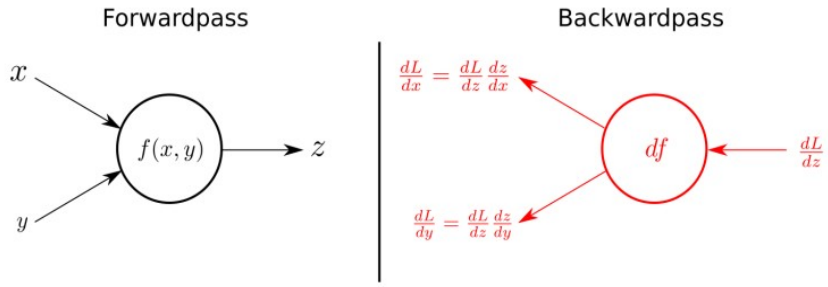


Figure 2.5: Forward Pass and Backward Pass in a Neural Network [15]

2.3.1 Backpropagation and Gradient Descent

The ability of neural networks to learn several parameters, even upto millions, is one of their key components. Backpropagation and gradient descent makes this possible. Two steps are required to train a neural network. The network receives some data at first and outputs a result. Second, the network's weights are adjusted in accordance with the measurement of the answer's inaccuracy or loss. The key is contained in this update. Iteratively calculating the gradients of the weights with respect to the loss and changing the network parameters allow a network to gradually converge to a minimum of the loss [16]. When the loss reaches its lowest point, the network can no longer be trained to perform better.

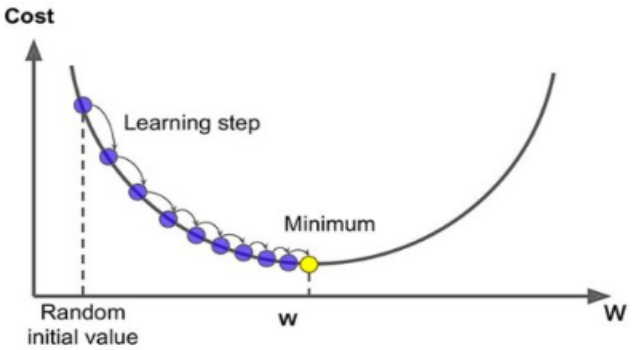


Figure 2.6: Example of Gradient Descent Algorithm [17]

The value of the parameter w is modified by computing the gradient with respect to the loss ('cost' in the figure) and multiplying it by a learning step. This is carried out until convergence, i.e., when w reaches its minimum.

2.4 Deep Learning

Deep learning involves a series of interconnected layers that are capable of autonomously learning complicated mappings quickly and effectively. Through its hidden layer design, a deep learning technique progressively learns classes, initially developing low-level classes like as letters, then relatively higher-level classes such as words, and lastly higher-level classes such as sentences [18].

Using a mix of data inputs, weights, and biases, deep learning neural networks imitate the human brain. These elements work together to accurately identify, classify, and describe data objects.

Deep neural networks are comprised of many layers of linked nodes, each of which builds on the previous layer to improve prediction or classification [18]. This progression of processing through the network is known as forward propagation. The visible layers of a deep neural network are the input and output layers. In the input layer, the deep learning model receives the required data, and in the output layer, the final prediction or classification is created [18].

Backpropagation is a strategy that utilizes algorithms, such as gradient descent, to quantify mistakes in predictions and then alters the weights and biases of the function by reversing through the layers to train the model. Forward propagation and backpropagation enable neural networks to create predictions and correct errors. The accuracy of the algorithm grows continuously with time.

2.5 Deep Generative Models

Unsupervised learning is used to create a generative model, which may be used to learn any form of data distribution [19]. In order to produce new data points with some variances, generative models try to learn the true data distribution of the training set. Multilayer neural networks that can produce samples in accordance with the distribution of the data describe deep generative models [19].

Deep generative models are neural networks with numerous hidden layers that have been trained

using samples to approximate complex, high-dimensional probability distributions. After the model has been adequately trained, we may use it to estimate each observation's likelihood and to generate fresh samples from the underlying distribution [20].

2.6 Autoencoder

It is necessary to comprehend the autoencoder before moving on to the variational autoencoder. The encoder q and the decoder p are the two components that make up an autoencoder. A latent variable of size n called z is produced by the encoder from an input called x. Formally, the encoder is represented as q [21]. As an illustration, consider the MNIST [22] dataset, which consists of 28×28 images of handwritten digits. The encoder receives a 28×28-dimensional image as input, which is then converted into a lower dimension of size n. This corresponds to the general case where n is smaller than the dimension of x. information is compressed into $z's$ lower dimension. In contrast, the decoder p attempts to piece together the initial input x from the latent variable z. The network frequently has a structure that is totally different from the encoder network. The symbol stands for the decoder's parameters. Formally, the decoder is written as p.

The so-called reconstruction loss, which is frequently the mean-squared error or cross-entropy between the input and output, is used to train these networks. It indicates intuitively how much the produced data resembles the input data.

The main issue with standard autoencoders for generation is that sampling this space is difficult and interpolation may be useless since the latent space of the encoded values may have gaps [23]. Clusters may be seen in the figure 2.3, which makes this obvious. The autoencoder's decision to use these segments makes sense because they facilitate the latent vector $z's$ decoding by the decoder [21] [23]. It does, however, preclude point interpolation and random sampling from the latent space.

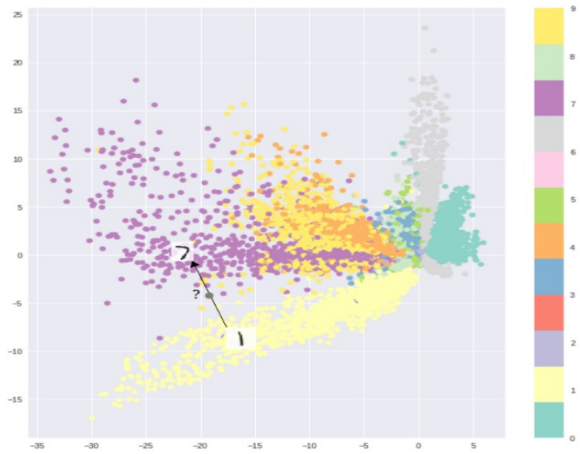


Figure 2.7: Latent Space of Autoencoder Trained on MNIST, Optimized Solely for Reconstruction Loss

By sampling the gaps in a latent space with discontinuities, new information is essentially given to the decoder. The decoder will provide erroneous and obscure data because it has no concept to what these latent vectors correspond to. There are different kinds of autoencoders. In the next section, we have explained variational autoencoder because it has become popular as generative model.

2.7 Variational Autoencoders (VAEs)

The variational autoencoder was created as a solution to the demands of contemporary data sampling, such as being able to handle intricate models and huge datasets. While GANs can create samples that are incredibly realistic, they have little control over the traits you want to include or exclude. On the other hand, a variational autoencoder allows for very precise feature selection on created data as well as the modification of existing data. Because there are no gaps in the latent space, the variational autoencoder fundamentally differs from the conventional autoencoder. This enables interpolation and transformations on actual data, making them ideal for generative jobs. This is accomplished through an unexpected process that involves splitting the latent vector of size n into two vectors of size n, one vector of means, and one vector of standard deviations. Each value in the vector of means has a corresponding value at the same

index in the vector of standard deviations. Together, they establish a probability distribution, often a Gaussian, from which a sample is taken to provide information to the decoder.

Let's say that the latent space has a length of 2, and the encoder produces two vectors: one with two means, $[0.3, 0.75]$, and the other with two standard deviations, $[0.09, 0.04]$. With those means and standard deviations, we sample two Gaussians to obtain the latent vector $z : [0.34, 0.74]$. As a result, the decoder sees samples from the distribution of the encoding during training in addition to seeing the specific encodings of the input data. This implies intuitively that the decoder discovers that a certain output relates not just to a specific point but also to the region around it, which contains data points that are very similar [24]. Since they have no boundaries, they can end up in the latent space as a number of little 'islands' continuing the problem of the gaps [25]. In order to enable smooth interpolation and arbitrary sampling, we would prefer for these distributions to be as similar to one another as possible while still remaining distinct [26]. The Kullback-Leibler Divergence (KL-divergence) [27], which is a component of the loss function, is added to do this. A measure of how divergent two probability distributions are called the KL-divergence. In order to minimize this, the probability distribution's parameters must be optimized and must closely mirror the real distribution.

2.8 Generative Adversarial Networks

In 2014, Ian Goodfellow et al. presented GANs [28]. Two hostile "players" participate in a minimax game and present a novel brain architecture based on game theory in their essay. They circumvent some of the challenges of explicitly modeling a data distribution, such as the requirement to simulate complicated probabilistic calculations that emerge during maximum likelihood estimation using intricate functions (such as in the VAE). The GAN framework consists of a generative model competing against a discriminative model that learns to differentiate between samples generated by the generative model and samples generated from actual data. Due to the competition between the two sides, it is no longer possible to discriminate between fake and genuine data points [29].

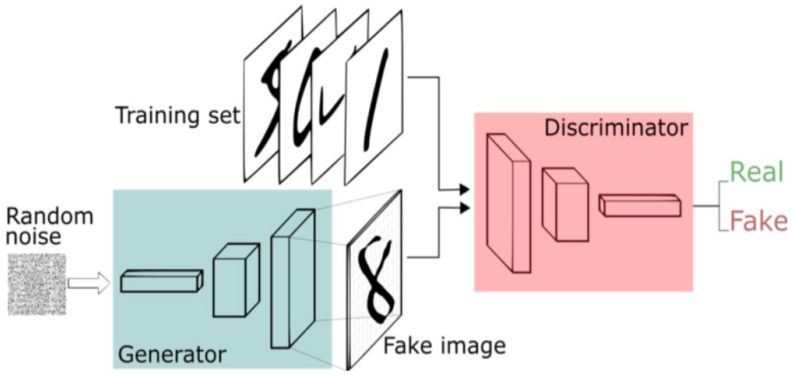


Figure 2.8: GAN Architecture with Samples Taken from MNIST Images [28]

The generator learns to map some random noise to the distribution of the actual data after receiving some random noise as input. The discriminator receives the generator's output together with a real image. The discriminator then tries to distinguish between actual and fraudulent images [30].

2.8.1 Generator Discriminator

A generator G and a discriminator D make up the machine learning model known as the Generative Adversarial Network developed by Goodfellow et al. [28]. After an adversarial game in which the two models, D and G, are simultaneously trained, this approach produces a generative network. The discriminative model D calculates the likelihood that a sample comes from training data rather than G, while the generative model G records the data distribution. Formally, the goal of this adversarial game can be expressed as:

$$\min_G \max_D E_{x\ \mathrm{pdata}}[\log D(x)] + E_{z\ \mathrm{p(z)}}[\log(1 - D(G(z)))] \tag{2.1}$$

Where, the generator function $G(z)$ maps samples from $p(z)$ to the data space while it is trained to confuse the discriminator to believe that these samples come from the original data distribution P_{data} .

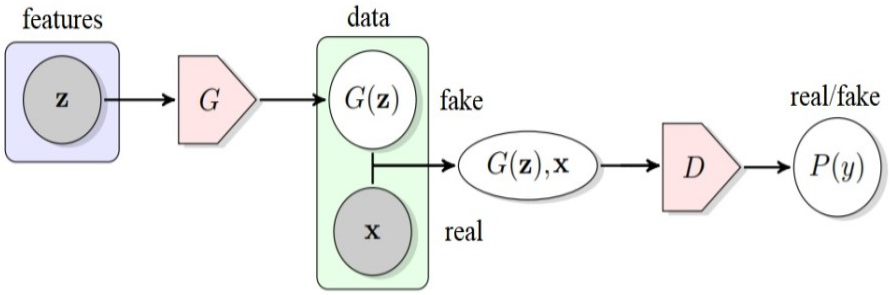


Figure 2.9: Generative Adversarial Networks' structure [28]

The discriminator model can be misled into estimating with a high probability $P(y)$ that these samples come from an original data distribution by the generator model, which is taught to produce synthetic data $G(z)$ [30].

In plainer terms, Goodfellow et al. [28] described in the paper that the generative model can be compared to a group of forgers who are attempting to create counterfeit money and utilize it covertly. The discriminative model, meanwhile, is comparable to the police in that it looks for counterfeit money. Both parties are motivated by the competition in this game to refine their techniques until they are unable to tell the difference between real money and counterfeit money.

Further investigations have revealed other GAN model flaws since the publication of the original article by Goodfellow et al [31]. On the downside, it does not give the option to choose what data to generate or to produce categorical data. On the other hand, training GANs is known to be difficult for machine learning models.

Mirza et al. released Conditional Generative Adversarial Networks [32], one of the first methods to enhance the GAN model. A study demonstrates the ability to regulate the data generated by simply adding a one-hot encoded vector specifying the class to generate as an input to the generator and discriminator. Arjovsky et al. presented the Wasserstein GAN a few years later, saying it would give the original GAN model training stability and interpretability [33]. Later studies have shown that the Wasserstein technique can enable the GAN to produce categorical

data by simply having a corresponding Softmax output in the generator network with a dimensionality equal to the number of discrete values for each categorical variable.

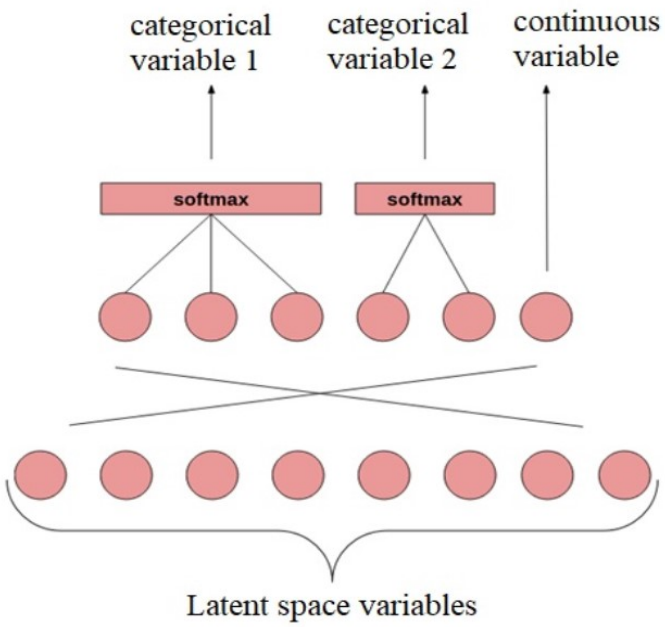


Figure 2.10: A Generator for Categorical and Continuous Variables Blended Together [28].

There is one continuous variable in this situation along with two categorical variables that each have one of three possible values.

Other alternative GAN designs have since made a significant and beneficial contribution to improving the original model. One of these contributions is the Bidirectional GAN, a model with the ability to access the latent space representation of the data, which is suggested in the study Adversarial Feature Learning [34]. a study that is driven by the notion that generative models' latent spaces hold a wealth of semantic data. Therefore, accessing these semantic latent representations may be advantageous for jobs where semantics are important for feature representation as well as for regulating what data to generate.

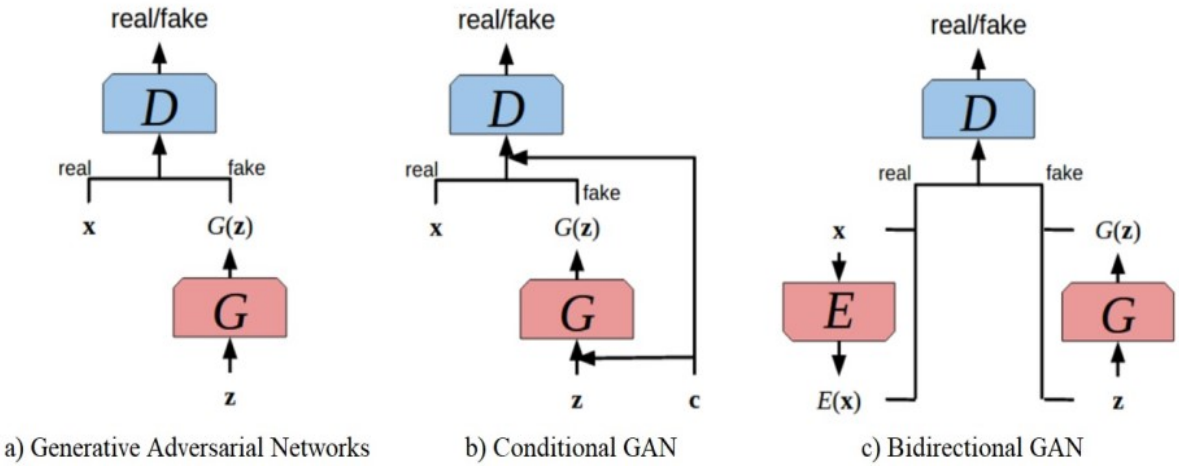


Figure 2.11: Simplified Structures of a) an Original GAN Framework , b) a Conditional GAN Introducing an Additional Input of Data , and c) a Bidirectional GAN, which Introduces an Encoder E that Should Learn to Invert the Generator G .

2.9 Conclusion

We've covered the basics of machine learning and the various learning methods in this chapter. As a precursor to this, we discussed neural networks, the fundamental building blocks of deep learning. We then went on to explain auto-encoders, generative adversarial networks (GANs), and the other deep generative models that are being used to generate synthetic data. These background studies have helped us to do further thesis work. In the next chapter we have discussed about GAN models which can generate synthetic tabular data.

Chapter 3

GANs for Tabular Data

3.1 Introduction

We have only covered GANs in general up to this point, which is frequently used to process images. This thesis, however, focuses on GANs used with tabular data, therefore that is what will be covered in-depth in this part. GANs for tabular data have a number of additional challenges while generating data, including changing the data to be suitable for GAN use, handling many data kinds at once, and performing a deterministic inverse translation of the created data. Thus, this involves a few more steps than discriminative models and GANs for picture data. We will go over the procedures for training GANs on tabular data in this part. The two most used data kinds, categorical and continuous, are the main focus of this research. No distinction is made between nominal and ordinal data, for instance, gender and education will both have the same encoding even though education obviously has a hierarchy with higher and lower levels.

3.2 Tabular GAN (TGAN)

The architecture known as TGAN was put up by Xu et al. [35] for the creation of tabular data. The authors wanted to offer a toolkit that could produce any dataset with categorical and continuous data. They accomplish this using a generator that traverses over the columns and predicts the value for the subsequent column based on previous outputs. The generator has an Recurrent Neural Network (RNN) with Long Short Term Memory (LSTM) cells. To obtain the final output, the LSTM output is passed through a number of dense layers, including an attention layer.

A completely connected network serves as the discriminator, which is simpler. Additionally, they identify the issues non-Gaussian distributed inputs pose for neural networks and provide a probabilistic distribution strategy to solve these issues. They apply a Gaussian Mixture model (GMM) to each numerical column separately, to be more precise. Mode-specific normalization for numerical variables is the term used to describe this method. For each numerical column C_i, m components are used to train the GMM. The means and standard deviations of the m components, $\mu_i(1), ...\mu_(m)$ and $\sigma_i(1), ...\sigma_i(m)$, respectively, represent the output of fitting the GMM. They calculate the probability originating from each of the m Gaussian distributions as a vector, $u_i, j(1), ...u_i, j(m)$ for each value in column C_i, j. A normalized probability distribution over m Gaussians is therefore represented by the final vector, u_i, j. The data are finally normalized using the mean and standard deviation of the Gaussian to which they most likely belong: $v_i, j = (c_i, j(i_k)) = 2i(k)$. These values are clipped to [0.99, 0.99] as the very last step before being fed into the neural network. Each value c_i,$j$is ultimately represented by the union of u_i, j and vi, j.

Let's look at figures 3.1 and 3.2 to get a sense of this encoding in order to give it a little more context. Each piece of data is encoded according to the likelihood that it belongs to one of the four Gaussians and its position inside the Gaussian that is considered to be the most likely. Because they don't employ argmax, their solution is not affected by the issue with the derivative of argmax being 0 in relation to the issues with categorical data.

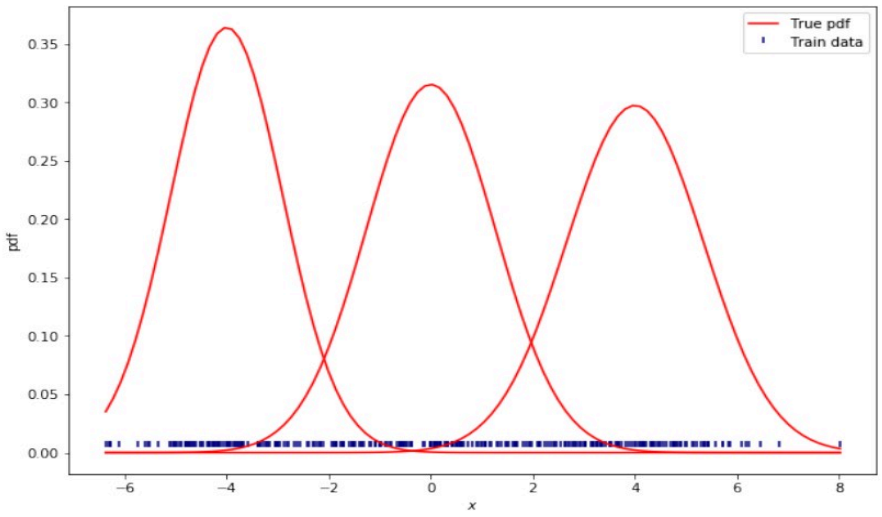


Figure 3.1: A set of data points created by sampling three Gaussians 100 times each. The result is a data distribution that is very hard to model with a single Gaussian.

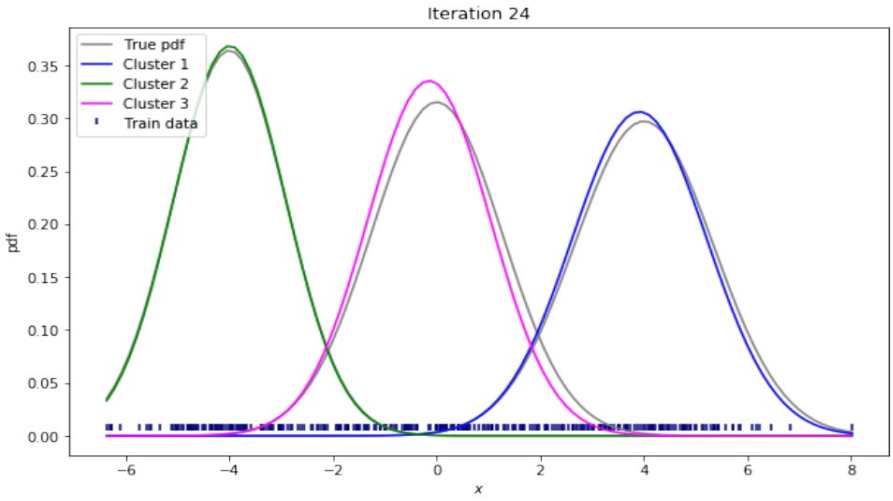


Figure 3.2: Three Gaussians fitted to the sampled data gives us three almost identical Gaussians. If this were modeled using a single Gaussian, sampled data would be very different than the real data.

They continue to use the typical softmax output instead. They use label smoothing to the real data to close the gap between a softmax distribution and a one-hot encoded real value, making it more difficult for the discriminator to distinguish between the two. This appears to work rather well in practice, and they obtain some pretty spectacular outcomes. In addition, they suggest a regularization method using a diversity vector. The overall distance between one sample and all other samples in the mini-batch using a predetermined metric makes up each dimension in this vector. This provides the discriminator with some meta data regarding that particular sample. It is difficult to say what and how much of an impact this addition has, though, because they do not directly analyze it. In order to avoid overfitting, they use the standard GAN losses but add a Kullback–Leibler divergence (KL-divergence) component to the generator loss. This probably has the desired impact on sample diversity but degrades sample quality, as was stated earlier sections. The TGAN generator's loss function is

$$LG = -Ez \sim U(0,1) \log D(G(z)) + \sum_{i=1}^{nc} KL(ui', ui) + \sum_{i=1}^{nd} KL(di', di)) \tag{3.1}$$

where d is the one-hot vector of categorical columns and n_c and n_d denote the number of continuous and categorical columns, respectively.

3.3 Medical GAN (MedGAN)

MedGAN [36] adopts a completely different strategy. They suggest converting categorical values to a latent continuous representation that the generator can learn using an autoencoder. The decoder lies between the generator and the discriminator and converts the continuous output of the generator to the format of the real data using a pretrained autoencoder (which consists of decoder and encoder). Their application had some restrictions. For instance, they simply implemented and evaluated count and binary variables. Although count variables appear to lack a predetermined limit, they are essentially ordinal integer numbers. They also don't support managing both data kinds simultaneously; instead, they support handling them in distinct models.

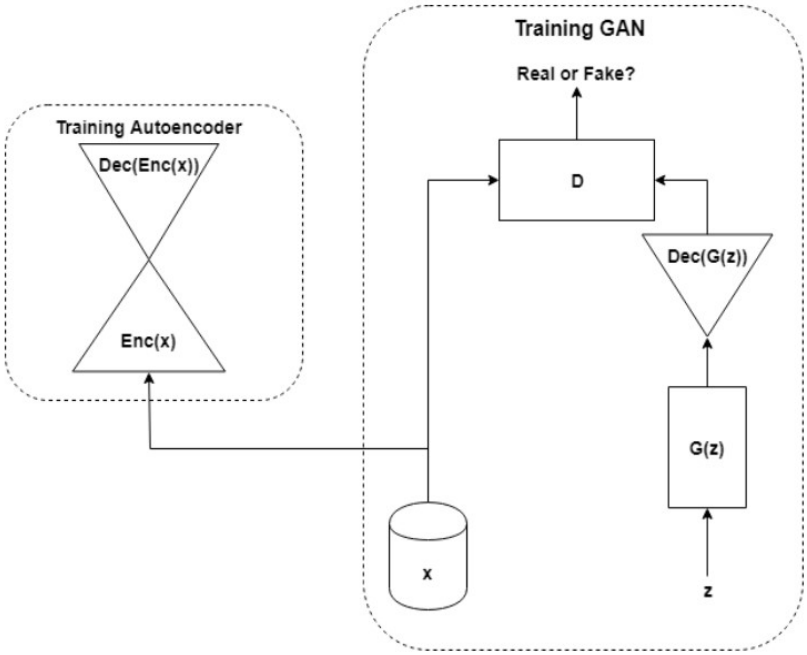


Figure 3.3: Architecture of MedGAN

First, the autoencoder is trained, which is visible left. Then, the GAN is trained with the decoder of the autoencoder situated between the generator and discriminator.

Let's discuss about their contributions before delving into the finer points.

(a) They suggested utilizing autoencoders to get around categorical values in GANs.

(b) They suggested a method known as minibatch averaging, in which they provide the discriminator additional input by using the average of a minibatch. This accomplishes the same goal as the diversity vector used in TGAN and reduces modecollapse.

(c) In their generator G, they added skip connections to the dense layers. These are the same types of connections found in ResNet and other cutting-edge discriminative models, and they enable the model to look back at older data while also enhancing the gradient flow.

They offer results in their findings for both classifier F1 scores and privacy capabilities. Additionally, they evaluate on a number of artificial baselines like sampling from a Bernoulli distribution [37], based on presence in the real dataset, and randomly flipping a binary value in 10% of the cases. For the two data types mentioned, MedGAN seems to outperform the evaluated alternatives like VAE, and Restricted Boltzmann Machines (RBM). They do, however, appear to outperform the VAE and RBM, which is significant on its own. Additionally, they carry out a rather thorough assessment of privacy protection. Depending on the datatype, each component of the MedGAN is made up of multilayer perceptrons with a varying level of activation. Both the Enc and the Dec employ ReLU for count variables. They employ tanh in the Enc and sigmoid activations in the Dec for binary values. This strategy appears to be effective for MedGAN. Due to the model's limited functionality with different data types,we needed more, and its performance was significantly worse than TGAN and TableGAN, we decided not to build upon MedGAN.

3.4 TableGAN

TableGAN [38] employs convolutional neural networks in a totally different manner to attempt to capture correlations in two dimensions. We'll go over their strategy step by step. Data encodings in TableGAN are much simpler than they are in TGAN. Categorical values are converted to numerical values, which are then scaled by minimax to a value in $[-1, 1]$ (i.e. $[a, b, c] \rightarrow [1, 2, 3] \rightarrow [-1, 0, 1]$) for use with tanh in the final layer. Just scale continuous columns to a value between -1 and 1. The Deep Convolutional GAN (DCGAN), one of the main forces behind high quality picture synthesis, served as the foundation for the architecture of the TableGAN. While the discriminator D uses LeakyReLU for all layers except the final one, which

employs sigmoid, the generator G uses ReLU for all intermediate layers, as is typical for convolutional models. They introduce classifier C, which is conceptually the same as discriminator D. However, C uses one of the columns as labels that it attempts to predict rather than classifying the sample as real or fake. This appears to improve sample quality and coherence in their tests. They cite the example that someone with a cholesterol of 50 cannot be diagnosed with diabetes since that level of cholesterol is too low. In addition to the regular GAN loss, this classifier C's loss also includes an additional loss known as information loss. The output of the final layer before the sigmoid in the discriminator is used to evaluate the information loss, which is strongly connected to the Wasserstein distance. The absolute difference between the logit means, which is how the Wasserstein distance is determined, is how the information loss is defined.

$$Lmean = \parallel E[fx]x \sim pr - E[fG(z))]z \sim pz \parallel 2 \tag{3.2}$$

$$L\sigma = \parallel \sigma[fx]x \sim pr - \sigma[fG(z))]z \sim pz \parallel 2 \tag{3.3}$$

$$L^{G}info = Lmean + L\sigma \tag{3.4}$$

where the standard deviation is represented by and the logits in the final discriminator layer by f. Therefore, if $lmean = 0$ and $l = 0$, the discriminator will not be able to tell the difference between real and fake data because they will both appear the same. The authors point out that there is a requirement to regulate the privacy of the real data points, and that the degree to which a distribution of synthetic data is identical to distribution of real data determines how private it is. They manage this by modifying LG information to

$$Linfo^{G} = max(0, Lmean - \sigma mean) + max(0, L\sigma - \delta\sigma) \tag{3.5}$$

where the hyperparameters mean and are used to regulate privacy. As long as L is less than for both the mean and standard deviation losses, this regularization sets $LGinfo = 0$. The loss is unbounded and the synthetic data distribution will attempt to resemble the true data distribution if the mean and standard deviation are both 0. With a substantial correlation between sample quality and mean and, this adaption seems to achieve its intended goal. We picked TGAN over TableGAN because it performs better in situations with categorical data than TableGAN does in our testing.

3.5 Conditional Tabular GAN (CTGAN)

Although it is a follow-up to, Conditional Tabular GAN [35] does not immediately go on where that study left off. This study presents a new tabular GAN architecture as well as updated data pretreatment techniques. Confusionally, they have also used the name TGAN for this architecture, but they have now changed it to CTGAN in the accompanying Python package, therefore we will use this nomenclature as well. The study gets off to a solid start by outlining five pressing issues that must be solved in order to successfully synthesize tabular data.

(a) **Mixed data types**: Mixed data types are common in real data. A GAN must be able to apply both softmax and tanh on the output to produce this data simultaneously.

(b) **Non-Gaussian distributions**: In contrast to tabular data, where continuous data frequently exhibit non-Gaussian distributions with lengthy tails, values in photographs typically follow a Gaussian-like distribution that can be easily normalized to $[-1, 1]$. With tanh and sigmoid, the gradient locations are frequently flat, leading to a condition known as gradient saturation. Gradient-based learning is lost by a model when gradient saturation takes place.

(c) **Multimodal distributions**: Multiple modes are frequently present in tabular continuous data. In their test datasets, they discover that $\frac{57}{123}$ continuous columns contain numerous modes. With the help of some specific preprocessing techniques, which will be described in a moment, this and the prior issue can be solved.

(d) **Learning from vectors with sparse one-hot encoding**: They described Gumbel-softmax is now used to address the reoccurring softmax problem .

(e) **Columns with extreme category imbalance**: In many categorical columns, the dominant category is represented on more than 90% of the rows, which is referred to as a highly imbalanced distribution. Missing the minor classes has little effect on the distribution overall and is frequently overlooked by the discriminator. In essence, this is a type of mode-collapse, for which they use PacGAN, in which a discriminator is shown a collection of samples in order to assess whether or not they are all fake.

With a few minor modifications, the data preprocessing methods are largely the same as those of TGAN . All continuous values are still encoded, but the variational Gaussian mixture model is used in place of the conventional Gaussian mixture model (VGMM). Additional weights, or parameters, $u^{(1)}, ..., u^{(m_i)}$, are included in the VGMM model. Please keep in mind that C_i stands for column i and that $c_{i,j}$ stands for a value in column C_i. We first make an attempt to determine how many modes there are in the Ci distribution. To do this, we employ the VGMM. This results in the probabilistic model $P_{C_i}(C_{i,j})$ and a Gaussian Mixture with m_i components that has means of $\mu_i^{(1)}, ..., \mu_i^{(i)}$, standard deviations of $\sigma_i^{(1)}, ..., \sigma_i^{(i)}$, and weights of $u_i^{(1)}, ..., u_i^{(i)}$.

(a) To start, we attempt to calculate the number of modes present in the C_i distribution. To do this, we employ the VGMM. In turn, this generates the probabilistic model $P_{C_i}(C_{i,j})$ and a Gaussian Mixture with m_i components, means $\mu_i^{(1)}, ..., \mu_i^{(i)}$, standard deviations $\sigma_i^{(1)}, ..., \sigma_i^{(i)}$ and weights $u_i^{(1)}, ..., u_i^{(i)}$.

$$PCi(ci, j) = \sum_{k=1}^{mi} u^{(k)} N(ci, j; \mu i^{(k)}, \phi i^{(k)}) \tag{3.6}$$

(b) Determine the probability mass function (PMF) for the value of ci,j that was taken as a sample from each of the mi modes as

$$Cat(k; [\bar{\beta}(i, j)^{(k)}]k = 1....mi), where, \bar{\beta}(i, j)^{(k)} = \frac{u^{(k)} N(ci, j; \mu i^{(k)}, \phi i^{(k)})}{PCi(ci, j)} \tag{3.7}$$

(c) Sample $k_{i,j}$ and transform it into a one-hot representation i, j using $Cat(k; [\bar{\beta}(i, j)^{(k)}]k = 1....m)$

(d) As in TGAN, the data $c_{i,j}$ is finally normalized with the mean and standard deviation of the most likely Gaussian by using the formula $a_{i,j} = (c_{i,j} - \mu_i^{(k)})/4\phi_i^k$. Following that, the value is cut to $[-1, 1]$, which encompasses 99.99% of all samples. Then, $a_{i,j}$ and i, j stand in for value $c_{i,j}$.

Three more suggestions are made to enhance training. The first one is a conditional vector that, using one-hot encoding, enables conditioning on a specific value of a particular column. Without delving into details, this strategy aids in increasing the variety of samples by occasionally explicitly conditioning on a certain value of a column, compelling the generator G to imitate this. Normally, it is simple for the generator to disregard this number if a column contains a

class that makes up the majority (let's say $> 90\%$), as the critic does not require the minor class to be present. G will be penalized if it deviates from the required class as a result of this explicit conditioning. This conditional vector is made up of all the categorical columns that have been one-hot encoded and concatenated, where the only 1 is in the class that is requested. Let's assume that there are two columns, $D1 = 1, 2$, and $D2 = 1, 2, 3$, and that $D1 = 2$ is the condition. Then, the one-hot encoded vectors of the columns, $m1 = [0, 1]$ and $m2 = [0, 0, 0]$, are merged to form $cond = [0, 1, 0, 0, 0, 0]$. This also pertains to their second contribution, where they extend the loss function to include the cross entropy between the resulting one-hot encoding of the categorical columns and the conditional entropy.

The final statement, which asks how to choose which class to condition on and with what probability, is closely related to the first two. This is accomplished by randomly choosing one of the classified columns first. Calculate the PMF of the values in that column, where each value's mass is equal to the logarithm of the column's frequency. The last step is to choose a random number from the PMF and set it to 1. Set the conditional vector's other values all to 0: $cond = m_1 \oplus ... \oplus m_N$is the total number of categorical columns, denotes the joining of two vectors, and 1 represents the chosen value in the chosen columns. The network's architecture has also evolved. The network is now made up of just a few dense layers with batch normalization after the authors discovered that the LSTM cells were not necessary to achieve the desired performance, which dramatically reduced training time. Additionally, they switched the GAN architecture from the standard GAN to the WGAN-GP base, even though this isn't mentioned elsewhere in the study.

3.6 Conclusion

In this chapter, we have covered a variety of GAN models that may be used to produce synthetic tabular data. We have detailed the models' capabilities, shortcomings, and our own observations on their usefulness and abilities. This chapter pictures the recent works of GAN-based synthetic tabular data, which allowed us to fill in the blanks left by earlier studies and advance our own research work. In the following chapter we have presented our own methodology to produce synthetic tabular data.

Chapter 4

Methodology

4.1 Introduction

This chapter provides a comprehensive breakdown of the investigative techniques that we have used during the course of this inquiry. Detailed information is provided about the processes for acquiring data, techniques for analyzing data, and methods for processing data. Clarification is then provided on the GAN model that was used by us in order to generate the fake data. There is also discussion over quality assurance for synthetic data.

4.2 Overview of the Proposed System

Our proposed system comprises mostly of the following five components: data collection, exploratory data analysis (EDA) of the gathered data, data preprocessing, GAN model training using the preprocessed data, and the synthetic data evaluation method. These main building blocks include certain subcomponents as well. In the following parts, each of the blocks is described in depth.

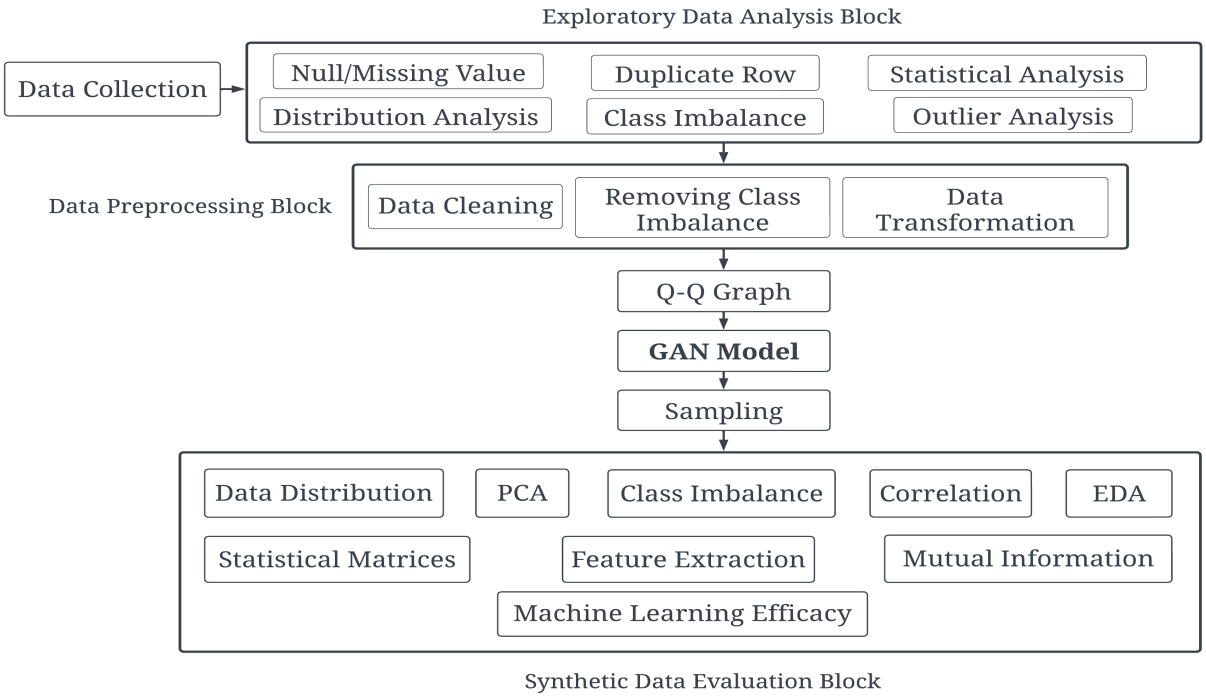


Figure 4.1: Overview of the Proposed System

4.3 Data Collection & Description

For the most part, case studies, observations, surveys, and experiments make up the bulk of the data gathering methods used in quantitative research. The overarching goal of this thesis is to develop a standardized approach to pre-processing data and to conduct thorough quality assessments of synthetic data in as many dimensions as feasible. Both of these objectives are intertwined with one another. A big bankruptcy data set is used to put the approach through its paces. It has about 7000 rows and 96 characteristics. For the years $1999 - 2009$, we scoured the Taiwan Economic Journal for relevant information [39]. According to the business rules of the Taiwan Stock Exchange, a firm is considered bankrupt if it cannot pay its debts. Each financial institution has a single record that matches the criteria. The likelihood of a company going bankrupt is influenced in some way by each and every variable in the data set. Some examples of such metrics are the effective tax rate, the cash flow per share, the sales per share, the net income growth, the total asset turnover, the current liability to asset ratio, the cash flow to sales ratio, the gross profit to sales ratio, and so on.

4.4 Data Analysis

The term "Data Analysis" refers to the intricate process of doing early exploratory investigations on data in order to discover patterns and anomalies, test hypotheses, and provide predictions based on a statistical representation [40]. It is vital to first achieve a level of factual comprehension, and then to collect as many insides as is practically possible. In exploratory data analysis, the first step is to make sense of the data before actually utilizing them. Our study of the data includes, among other things, the extraction of information from data sets, the examination of missing values, the observation of duplicate data, the description of data, and the investigation of data distribution and class imbalance.

4.4.1 Data Set Specification

We examined the reliability of the data and considered whether it had been utilized in previous studies or not.

(a) **Source:**

Deron Liang: deronliang@gmail.com

& Chih-Fong Tsai: cftsai@mgt.ncu.edu.tw

National Central University, Taiwan

The data was obtained from UCI Machine Learning Repository:

https://archive.ics.uci.edu/ml/datasets/Taiwanese+Bankruptcy+Prediction

(b) **Relevant Papers:**

This dataset has been used in the following journal: Financial Ratios and Corporate Governance Indicators in Bankruptcy Prediction: A Comprehensive Study. European Journal of Operational Research, vol. 252, no. 2, pp. 561-572 by Liang, D., Lu, C.-C., Tsai, C.-F., and Shih, G.-A. (2016). [39]

https://www.sciencedirect.com/science/article/pii/S0377221716000412

(c) **Attribute Information:**

This data set contains 6819 number of instances and 96 features. Updated feature names and description to make the data easier to understand (Y = Output feature, X = Input features) are given below in the table 4.1 .

Table 4.1: Data Set Attribute List

Symbol	Feature Name
Y	Y Bankrupt?: Class label
X1	ROA(C) before interest and depreciation before interest: Return On Total Assets(C)
X2	ROA(A) before interest and % after tax: Return On Total Assets(A)
X3	ROA(B) before interest and depreciation after tax: Return On Total Assets(B)
X4	Operating Gross Margin: Gross Profit/Net Sales
X5	Realized Sales Gross Margin: Realized Gross Profit/Net Sales
X6	Operating Profit Rate: Operating Income/Net Sales
X7	Pretax net Interest Rate: PreTax Income/Net Sales
X8	Aftertax net Interest Rate: Net Income/Net Sales
X9	Nonindustry income and expenditure/revenue: Net Nonoperating Income Ratio
X10	Continuous interest rate (after tax): Net IncomeExclude Disposal Gain or Loss/Net Sales
X11	Operating Expense Rate: Operating Expenses/Net Sales
X12	Research and development expense rate: (Research and Development Expenses)/Net Sales
X13	Cash flow rate: Cash Flow from Operating/Current Liabilities
X14	Interestbearing debt interest rate: Interestbearing Debt/Equity
X15	Tax rate (A): Effective Tax Rate
X16	Net Value Per Share (B): Book Value Per Share(B)
X17	Net Value Per Share (A): Book Value Per Share(A)
X18	Net Value Per Share (C): Book Value Per Share(C)
X19	Persistent EPS in the Last Four Seasons: EPSNet Income
X20	Cash Flow Per Share
X21	Revenue Per Share (Yuan ¥): Sales Per Share
X22	Operating Profit Per Share (Yuan ¥): Operating Income Per Share
X23	Per Share Net profit before tax (Yuan ¥): Pretax Income Per Share
X24	Realized Sales Gross Profit Growth Rate
X25	Operating Profit Growth Rate: Operating Income Growth
X26	Aftertax Net Profit Growth Rate: Net Income Growth
X27	Regular Net Profit Growth Rate: Continuing Operating Income after Tax Growth
X28	Continuous Net Profit Growth Rate: Net IncomeExcluding Disposal Gain or Loss Growth

Symbol	Feature Name
X29	Total Asset Growth Rate: Total Asset Growth
X30	Net Value Growth Rate: Total Equity Growth
X31	Total Asset Return Growth Rate Ratio: Return on Total Asset Growth
X32	Cash Reinvestment %: Cash Reinvestment Ratio
X33	Current Ratio
X34	Quick Ratio: Acid Test
X35	Interest Expense Ratio: Interest Expenses/Total Revenue
X36	Total debt/Total net worth: Total Liability/Equity Ratio
X37	Debt ratio %: Liability/Total Assets
X38	Net worth/Assets: Equity/Total Assets
X39	Longterm fund suitability ratio (A): (Longterm Liability+Equity)/Fixed Assets
X40	Borrowing dependency: Cost of Interestbearing Debt
X41	Contingent liabilities/Net worth: Contingent Liability/Equity
X42	Operating profit/Paidin capital: Operating Income/Capital
X43	Net profit before tax/Paidin capital: Pretax Income/Capital
X44	Inventory and accounts receivable/Net value: (Inventory+Accounts Receivables)/Equity
X45	Total Asset Turnover
X46	Accounts Receivable Turnover
X47	Average Collection Days: Days Receivable Outstanding
X48	Inventory Turnover Rate (times)
X49	Fixed Assets Turnover Frequency
X50	Net Worth Turnover Rate (times): Equity Turnover
X51	Revenue per person: Sales Per Employee
X52	Operating profit per person: Operation Income Per Employee
X53	Allocation rate per person: Fixed Assets Per Employee
X54	Working Capital to Total Assets
X55	Quick Assets/Total Assets
X56	Current Assets/Total Assets
X57	Cash/Total Assets
X58	Quick Assets/Current Liability

Symbol	Feature Name
X59	Cash/Current Liability
X60	Current Liability to Assets
X61	Operating Funds to Liability
X62	Inventory/Working Capital
X63	Inventory/Current Liability
X64	Current Liabilities/Liability
X65	Working Capital/Equity
X66	Current Liabilities/Equity
X67	Longterm Liability to Current Assets
X68	Retained Earnings to Total Assets
X69	Total income/Total expense
X70	Total expense/Assets
X71	Current Asset Turnover Rate: Current Assets to Sales
X72	Quick Asset Turnover Rate: Quick Assets to Sales
X73	Working capitcal Turnover Rate: Working Capital to Sales
X74	Cash Turnover Rate: Cash to Sales
X75	Cash Flow to Sales
X76	Fixed Assets to Assets
X77	Current Liability to Liability
X78	Current Liability to Equity
X79	Equity to Longterm Liability
X80	Cash Flow to Total Assets
X81	Cash Flow to Liability
X82	CFO to Assets
X83	Cash Flow to Equity
X84	Current Liability to Current Assets
X85	LiabilityAssets Flag: 1 if Total Liability exceeds Total Assets, 0 otherwise
X86	Net Income to Total Assets
X87	Total assets to GNP price
X88	Nocredit Interval

Symbol	Feature Name
X89	Gross Profit to Sales
X90	Net Income to Stockholder's Equity
X91	Liability to Equity
X92	Degree of Financial Leverage (DFL)
X93	Interest Coverage Ratio (Interest expense to EBIT)
X94	Net Income Flag: 1 if Net Income is Negative for the last two years, 0 otherwise
X95	Equity to Liability

(d) **Data Set Activity Overview:**

The following is a brief summary of the actions associated with the data set:

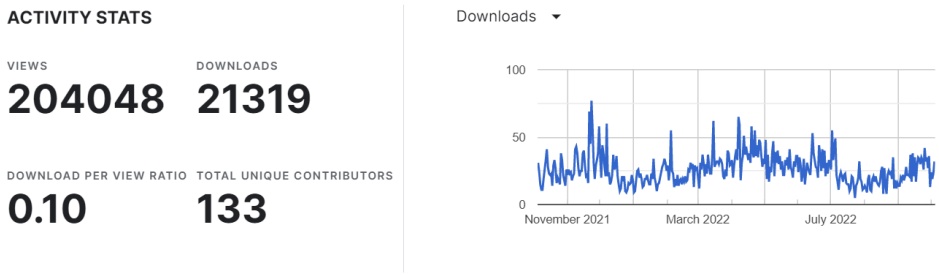


Figure 4.2: Activity Overview of the Data Set [41]

4.4.2 Missing Value Analysis

When a data set is missing information for a particular variable of interest, this is known as missing data. The data analysis or model's robustness may be compromised, depending on the amount of missing data volume. During exploratory data analysis and before using the data in a machine learning or deep learning model, finding missing values is a crucial initial step. As a consequence, they should be cleaned or eliminated before being given to the models to ensure accurate results. Finding the missing data points is an indication of quality compromise. There is more than one approach to handling missing values. Dropping rows with missing values is an easy solution, but it might result in significant data loss. In all likelihood, there are more efficient alternatives. For univariate procedures, statistical methods such as mean and mode imputation may be used [42]. Interpolation methods such as linear polynomial quadratic may also be employed [43]. Additionally, there are superior methods [44], such as the regression

Bayesian model, for imputation. Conditional dependency and independence are used in these models to determine the potential causal links between the variables [45]. For larger data sets and greater dimensionality, these models perform better than their predecessors in the vast majority of instances [46]. To fill in the blanks in our data set, we used a Bayesian network based approach.

4.4.3 Duplicate Data Deletion

When there is duplicate information in a database, it signifies that the information has been recorded more than once. Unique data are highly required for greater generalizability of model, which might hinder the split between train test and validation sets. If duplicate information is not removed, the model may begin to favor some types of information over others [47]. For example, due to duplication of data a model will overfit on the training data and will not generalize well. Due to duplication, same data will be passed twice before every back propagation. So, basically going forward is happening twice and updating the parameters once, which is not a good way of training a model.

4.4.4 Statistical Data Description

Each column in a data collection has its own properties, such as mean, median, standard deviation, etc. The data description contains all the information about the data sets and their contents, allowing anybody to use the data for research and comprehend the data structures. Some significant features for data description are: standard deviation, quantiles, outliers, variance, distribution etc [48]. They symbolize the useful insights of data and how they must be processed before a machine learning or deep learning model can be imposed on them [49]. These explanations are essential for understanding the decision-making process behind the model used to generate the synthetic data. These explanations also aid in avoiding frequent pitfalls [50]. We have characterized the selected data in as many statistical procedures as feasible in order to do better pre-processing and create data of high quality.

4.4.5 Class Imbalance

In a categorical choice variable, class imbalance occurs when one mode is disproportionately represented in comparison to the others. Specifically, class imbalance must be addressed before

a model can be trained, since doing so prevents it from becoming too specialized for the over-represented class [51] [52]. Under-sampling, over-sampling, and hybrid strategies that use both may all be used to avoid a skewed sample of one particular class [53].
We have utilized Synthetic Minority Oversampling Technique (SMOTE) oversampling to eliminate class imbalance since undersampling procedures reduce the over-represented category and lose a great deal of data if the imbalance is too great.

SMOTE begins by randomly selecting a minority class instance a and then locating its k closest minority class neighbors. The synthetic instance is then generated by selecting at random one of the k closest neighbors b and joining it to a to produce a line segment in the feature space. The created synthetic instances are a convex combination of the two selected examples a and b [54].

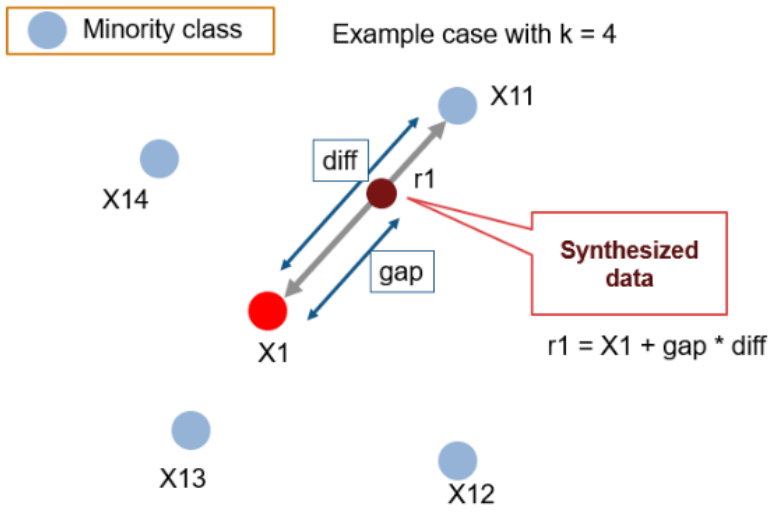


Figure 4.3: SMOTE Technique [55]

4.5 Data Preprocessing

Every effective Machine Learning project begins with the preparation of data [56]. Due to the unstructured nature of real-world data, data cleaning is a crucial aspect of creating machine learning models [57]. In order for machine learning models to work well, the raw data must

undergo some type of pre-processing [58].

4.5.1 Data Cleaning

The accuracy and reliability of an analysis are directly proportional to the quality of the data used [59]. When doing data analysis, poor quality inputs lead to inaccurate conclusions and wasted time. In order to make sure that only reliable information is utilized in the analysis, data cleaning is a crucial step [60].

The process of data cleansing entails more than merely eliminating erroneous information. It's the backbone of data analytics and an essential part of machine learning [61]. Major data cleaning steps of our methodology are discussed below.

4.5.1.1 Handling missing data

The presence of some missing values is typical in big databases. It's possible that the person responsible for keeping track of the data either forgot to enter them or didn't start collecting the missing data variables until much later. The management of missing data is a prerequisite to any work with datasets [62]. Several methods described in the Missing Value Analysis portion of this book may be used to complete them.

4.5.1.2 Outlier Filtering

Outliers can not be disregarded since they contain crucial data information, but they may also distract the machine learning model from the majority of the data sample [63]. Comparing data with or without an outlier, might provide surprising insights that are difficult to see at first glance. Working with outliers requires a robust system capable of handling outliers, or outliers may be eliminated [64].

4.5.1.3 Feature Transformation

Every model, whether it be for supervised learning or unsupervised learning, classification or regression must use feature transformation [65]. Creating new features from current features that may aid in increasing the model performance is known as Feature Engineering, another name for this process. These novel traits may not be interpretable in the same way as the original features, but they could have more explanatory value in another domain. As a bonus, feature transformation can be used for feature reduction [66]. Numerous methods exist for this, including linear combinations of the original attributes and the use of non-linear functions. Feature Transformation expedites the convergence of machine learning algorithms, which is a major benefit.

Linear and Logistic regression are two examples of Machine Learning models that make the assumption that the variables are normally distributed [67]. True dataset variables are more likely to have a skewed distribution. It is possible to map skewed distribution by applying transformations to normal distribution, which improves the efficiency of machine learning models [68]. Forms of Feature Transformation are discussed here.

(a) **Log Transformation**: When data undergoes a log transformation, it is transformed such that it more closely follows a normal distribution, but not perfectly [69]. Any characteristics with negative values are omitted from this transformation. Right-skewed information often benefits from this. Right-skewed information often benefits from this. This Transforms the feature from the Addictive Scale into a multiplicative scale, or a linear distribution [70].

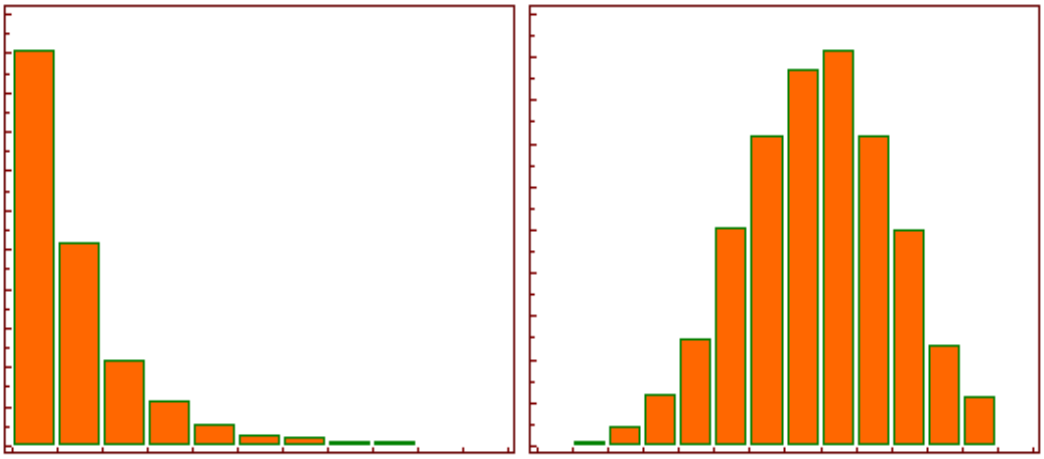


Figure 4.4: Effect of Log Transformation [69]

(b) **Power Transformation**: In order to get a more "Gaussian" distribution of results, power transformations are used [71]. Box-Cox and Yeo-Johnson transformations are the types of Power Transformation. Input data must be exclusively positive for use with Box-Cox (not even zero is acceptable). If a feature's value is zero or negative, Yeo-Johnson can help.

Box-Cox Transformation: The specific examples of this transformation are $\sqrt{x}$, $\sqrt{y}$ and $\log(x)$. [72]

Yeo-Johnson Transformation: The Yeo-Johnson is a modification of the Box-Cox transformation [73].

4.5.1.4 Quantile-Quantile Plot

In order to determine whether the provided transformation is consistent with a normal distribution, a Quantile-Quantile plotplot(Q-Q Plot) analysis is performed. If the feature follows a Normal distribution, its values should lie in a straight line with a slope of 45 degrees (y=x) when displayed against the theoretical quantiles in Q-Q plots [74].

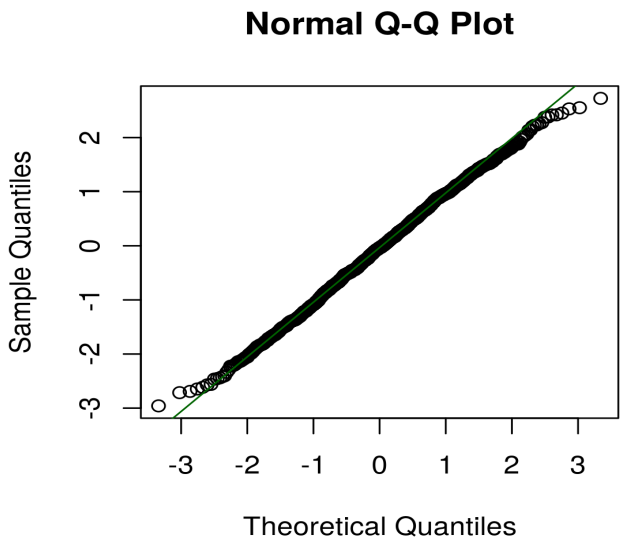


Figure 4.5: Normal Q-Q Plot [75]

4.6 Overview of Used GAN Model

CTGAN is a common technique that leverages the GAN architecture to model tabular data and conditionally select rows from the resultant model to produce Synthetic Data [76]. Currently, this may be regarded cutting-edge since it has outperformed other techniques for the given issue. Several of the most important ideas presented in the CTGAN article are described here.

1. **Multimodal Distribution Representation in Continuous Variables**
   A mode in a data distribution is a region of significant data concentration. A multi-modal distribution is a distribution whose probability distribution curve has many peaks. A Vanilla GAN model cannot account for all the modes of continuous variables, and adding traditional normalization approaches leads to Mode Collapse difficulties, in which a Generator generates samples around a single node, which significantly limits its learning [77]. CTGAN architecture implements a Mode Specific Normalization approach to address the aforementioned difficulty. Below is a high-level summary of the strategy [76]. Each continuous value is ultimately represented by a One Hot Vector indicating the mode and a normalized scalar giving the model value.

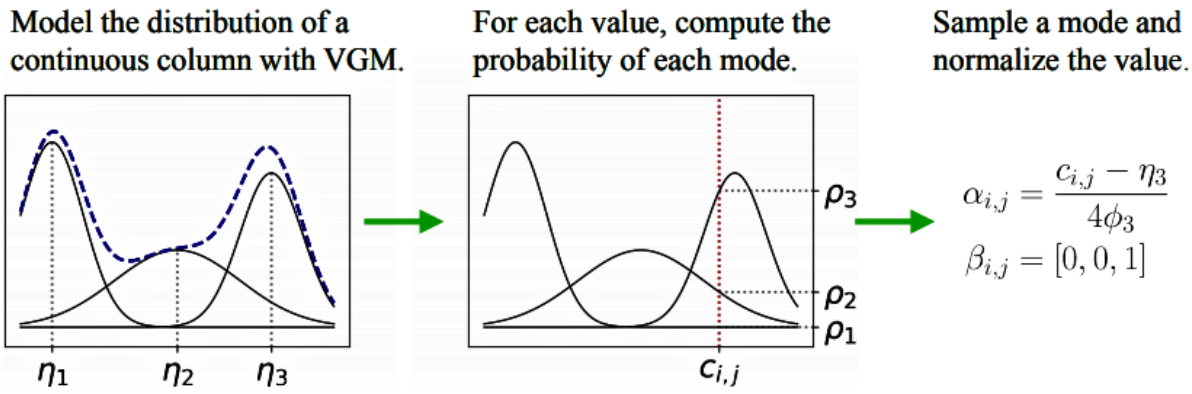


Figure 4.6: Mode Specific Normalization [76]

2. **Training via Sampling and Conditional Generators**: CTGAN presents the Training by Sampling and Conditional Generators [76] method to resample the training data such that all categories in discrete variables have an equal probability of being included in the sample from which GAN learns.

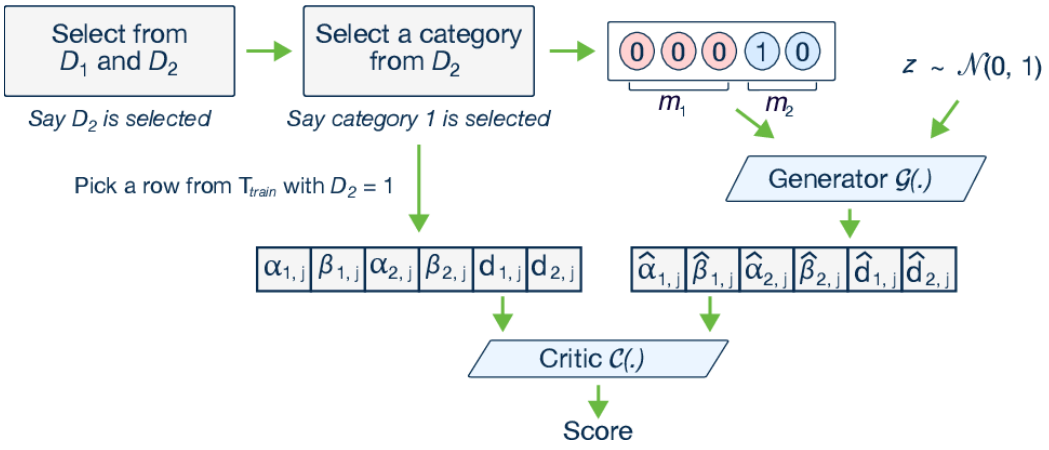


Figure 4.7: Training by Sampling and Conditional Generators [76]

3. **Maintain Linear and Nonlinear Correlation among Attributes**: Complex linear and nonlinear correlations between variables must be accounted for and maintained in the synthetic data [76].

   Both the Generators and Critics models include Fully Connected Hidden Networks to represent the association between qualities. In addition to the hidden layers, the design employs mix activation functions such as Leaky Relu and Dropout approaches to avoid any overfit.

4.7 Synthetic Data Evaluation

There is not yet a standard method through which the outputs of different synthetic data models can be compared. From a technical standpoint, previous work has mostly concentrated on identifying and quantifying the different fit measures that may be used to evaluate the gap between the actual and synthetic distributions [78].

Alternative approach for evaluating is to devide the data into a train set and a test set, and then only utilizing the train dataset to generate synthetic data that can be evaluated. Standard ML models are then formed using both the original training data and the newly created synthetic data. These models are then compared based on their test data evaluation results.

Our approach for evaluating synthetic data consists of various strategies with varying dimensions, discussed next.

4.7.1 Exploratory Data Analysis (EDA) of Synthetic Data

This covers Missing Value Analysis and Class Imbalance Analysis of Synthetic data and also comparison of these analysis with actual data.

4.7.2 Data Distribution Comparison

A data distribution is a function that displays the potential values for a variable and the frequency with which they occur. In addition to the input values that may be seen, the distribution of an event includes all conceivable values.

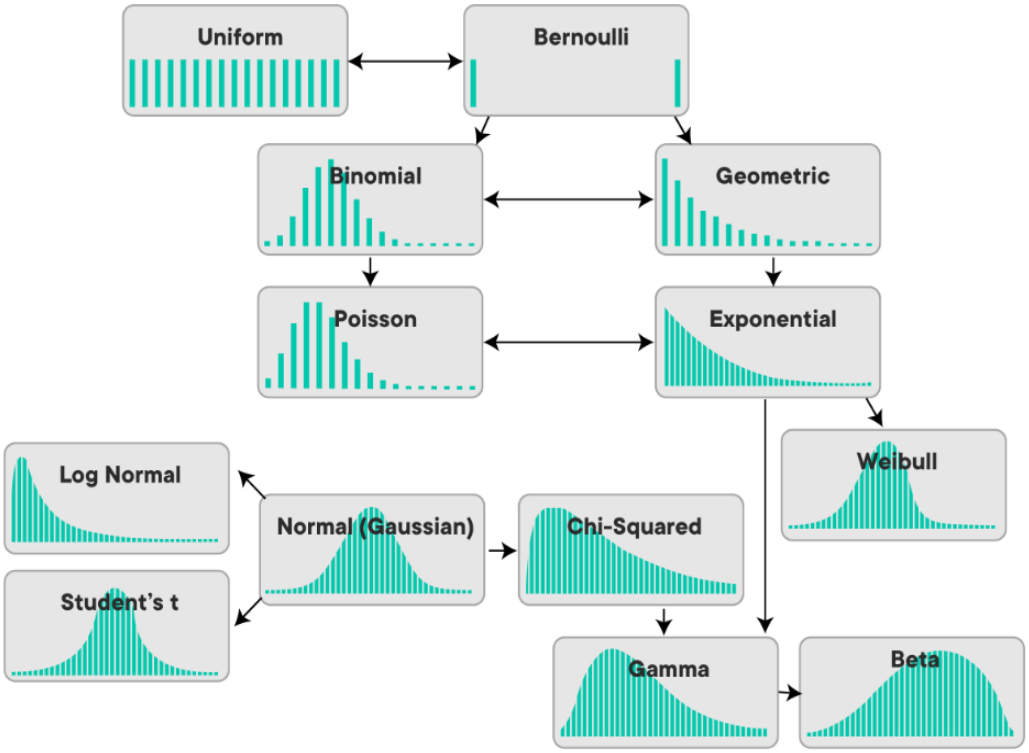


Figure 4.8: Different Types of Data Distributions [79]

(a) **Uniform Distribution:** The Uniform Distribution is the simplest distribution to explain. It is when all the outcomes are equally likely.

(b) **Bernoulli Distribution:** The Bernoulli Distribution essentially represents the probability of success of an experiment. An example would be tossing a coin where the 2 outcomes are given a probability of occurrence.

(c) **Poisson Distribution:** The Poisson Distribution represents the probability of n events in a given time period when the overall rate of occurrence is constant. In simpler terms, it gives a probability of how often something might happen. We can use the amount of mail a person receive everyday. A Poisson Distribution would give the probability of how many pieces of mail a person receive each day.

(d) **Gaussian or Normal Distribution:** The Gaussian or normal distribution is the most common distribution that we will come across. This follows a bell shape, which is the name we may recognize it by, and is found in many real world data, such as height and weight.

4.7.3 Principal Component Analysis (PCA) and Comparison

Principal component analysis (PCA) is a method for lowering the dimensionality of such huge datasets, hence improving interpretability while minimizing information loss. It achieves it by generating new variables that are uncorrelated and progressively optimize variance [80].

4.7.3.1 First Principal Component and Second Principal Component

The first main component is the spatial direction along which projections vary the most [81]. The second main component is the direction that greatest variation amongst all orthogonal to the first direction [82].

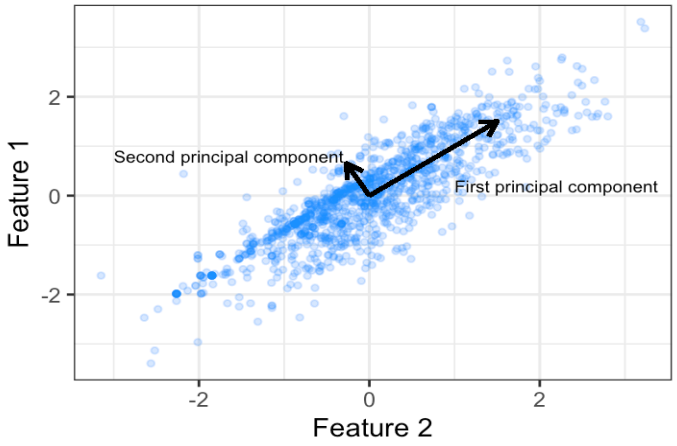


Figure 4.9: First Principal Component and Second Principal Component [81]

4.7.4 Feature Correlation Comparison

A correlation between two features may be seen as a connection between them. Thus, when one feature's value changes, the other feature's value often changes in a predictable way [83].

Correlation heatmaps are a sort of diagram that illustrates the strength of numerical attribute connections. The purpose of correlation plots is to determine which qualities are connected and the strength of this association. Typically, a correlation plot has a number of numerical characteristics, each represented by a column. The rows illustrate the link between each attribute

pair. Positive cell values indicate a positive association, whereas negative cell values denote a negative relationship. Correlation heatmaps may be used to identify possible links between variables and assess the strength of these connections. In addition, correlation plots may be used to determine linear and nonlinear correlations and to identify outliers. The color-coding of the cells facilitates the quick identification of links between characteristics. It is possible to utilize correlation heatmaps to identify both linear and nonlinear connections between features.

Here is an example of a correlation heatmap designed to illustrate the linear connection between characteristics.

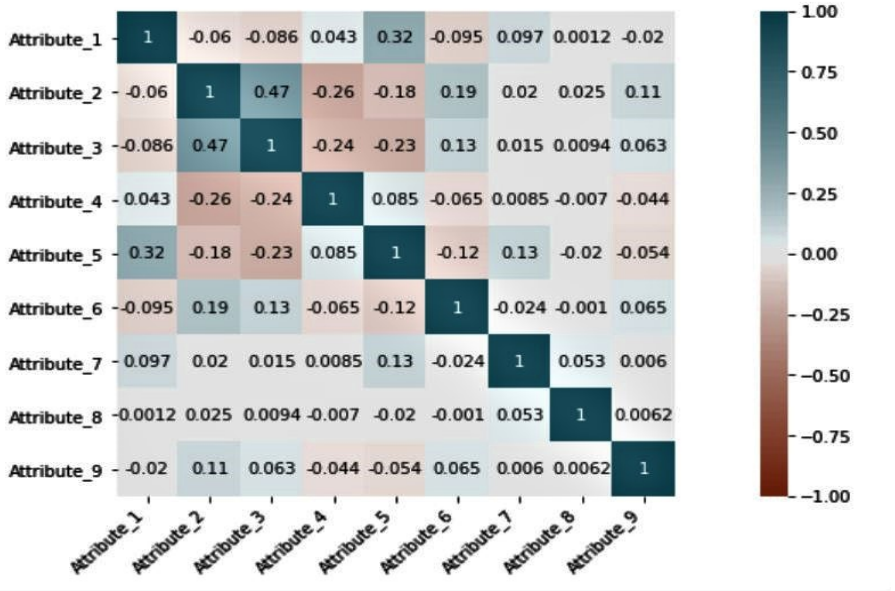


Figure 4.10: Correlation Matrix with Heatmap [83]

4.7.5 Statistical Metrics

These are comparisons between tables using different statistical tests. Some of them function by comparing many columns at once, while others operate by comparing each column separately and then producing an aggregated result. This comprises of the below matrices.

4.7.5.1 KSComplement

This metric compares the distributions of continuous columns using the empirical Cumulative Density Function (CDF) and the Kolmogorov–Smirnov test (KSComplement) on two samples. The output of each column is 1 minus the KS Test D statistic, which reflects the greatest difference between predicted and observed CDF values [84]. The Kolmogorov–Smirnov test is a nonparametric goodness-of-fit test and is used to determine whether two distributions differ, or whether an underlying probability distribution differs from a hypothesized distribution [84].

4.7.5.2 CSTest

This metric use the Chi-Squared test (CSTest) to compare the distributions of two discrete columns. The output of each column is the CSTest p-value, which indicates the probability that the two columns were taken from the same distribution [85].

4.7.5.3 LogisticDetection

Logistic regression detection, also known as Logistic Detection, is a statistical analytic technique for forecasting a binary result, such as yes or no, using a series of past observations. Predictions for a dependent variable may be generated using a logistic regression model by examining the interplay between one or more independent variables. The scikit-learn Logistic Regression classifier is the basis of the Logistic Detection metric.

4.7.5.4 SVCDetection

Support Vector Classifier Detection (SVCDetection) is a supervised machine learning technique often used for classification problems. The way SVC works is by projecting the data points into a high-dimensional space and then locating the best hyperplane to use as a boundary between the two groups [86]. The scikit-learn classifier used to create this detection measure is a Support Vector Classification model.

4.7.6 Comparison of Major Features

The purpose of feature selection is to pick fewer features than the total number of features while maintaining equivalent accuracy. Selected key characteristics using many methodologies and compared them between synthetic and actual data. The techniques that are involved discussed below.

4.7.6.1 Recursive Feature Elimination with Cross-Validation(RFECV)

Recursive Feature Elimination with Cross-Validation Features are prioritized using the model's coefficient or feature important attributes, and RFE seeks to reduce dependencies and co-linearity by iteratively deleting a small number of features in every loop [87].

4.7.6.2 Chi-Squared Feature Selection

The Chi-Squared statistics are computed using the following formula,

$$\tilde{\chi}^2 = \sum_{k=1}^{n} \frac{(O_k - E_k)^2}{E_k} \tag{4.1}$$

where "O" refers to observed or actual value and "E" stands for anticipated value assuming the two categories are independent. If they are independent, the O and E values will be low, however if they are related, the Chi-squared value will be high. These Chi-Square statistics are modified by the degree of freedom, which changes according to the number of levels of the variable and the number of levels of the class variable [88]. Chi-Square is a fairly straightforward method for selecting univariate features for classification. It does not account for the relationships between features. This method is well suited for categorical variables and has extensive use in textual data. It should be noted that Chi-Square may also be used for numerical variables after discretization.

4.7.6.3 Variance Threshold Feature Selection

Variance Threshold is a feature selector that eliminates from the dataset any low variance characteristics that are of little modeling use [89]. It may be used for unsupervised learning since it only considers the features (x) and not the outputs (y). Default Threshold Value is 0

(a) If Variance Threshold equals zero (Remove Constant Features)

(b) If Variance Threshold is greater than zero (Remove Quasi-Constant Features)

4.7.7 Mutual Information Comparison

The mutual information between two independently selected random variables is a quantitative measure of the strength of that connection. More specifically, it evaluates the typical amount of data sent by one random variable about another. Using the following formula, we can determine this: The formula for Information Gain, $IG(S, a)$ is: $H(S) - H(S|a)$. Specifically, H(S) is the entropy of the data set before any modification (explained above), H(S | a) is the entropy of the data set given the variable a, and IG(S, a) is the information for the data set S for the variable a for a random variable [90].

4.7.8 Machine Learning Efficacy

These metrics are used to train a Machine Learning model on synthetic data and then assess its performance on actual data. Due to the requirement that these metrics assess the performance of a Machine Learning model on the dataset, they are only applicable to datasets that reflect a Machine Learning issue [91].

4.8 Conclusion

In this chapter, a comprehensive outline of our thesis work has been presented. We have covered data collection, data analysis, and data preparation, amongst other topics. Then, we have

explained the GAN model that we used to create synthetic tabular data. Finally, we have discussed about our proposed multi-dimensional data evaluation method to prove the quality of synthetic tabular data. The following chapter consists the implementation of these processes we have mentioned in this chapter.

Chapter 5

Implementation

5.1 Introduction

The last chapter provided a long theoretical discussion, and this chapter follows it up with an in-depth technical description of how the techniques that we presented have been implemented. Implementation entails four main stages.

(a) EDA of real data

(b) Data Preprocessing

(c) Data Generation Model

(d) Synthetic Data Evalution

5.2 Implementation Tools

The programming language that we used was Python version 3.7, and we used the Kaggle Cloud platform to carry out the implementation on a GPU Tesla 37C P-100. TensorFlow, Keras, NumPy, Pandas, Matplotlib, Seaborn, Sci-kit Learn, and a number of other frameworks were used throughout this thesis.

5.3 Exploratory Data Analysis

Exploratory Data Analysis of the data includes Data Set Information, Missing Value Analysis, Checking for Duplicate Data, Statistical Data Description, Feature Distribution, and Class Imbalance Check.

5.3.1 Data Set Information

We used information from the Taiwan Economic Journal, the most widely read publication in Taiwan. It covers a decade's worth of information (1999-2009) [39]. This dataset was produced via a survey effort by academics and industry experts, making it ideal for repurposing in future studies. A total of 6819 occurrences and 96 characteristics can be found in the dataset [39]. The dataset might be used to help shape the next generation of data production tools for the financial technology sector.

	Bankrupt?	ROA(C) before interest and depreciation before interest	ROA(A) before interest and % after tax	ROA(B) before interest and depreciation after tax	Operating Gross Margin	Realized Sales Gross Margin	Operating Profit Rate	Pre-tax net Interest Rate	After-tax net Interest Rate	Non-industry income and expenditure/revenue	Continuous interest rate (after tax)	Operating Expense Rate
0	1	0.370594	0.424389	0.405750	0.601457	0.601457	0.998969	0.796887	0.808809	0.302646	0.780985	1.256969e-04
1	1	0.464291	0.538214	0.516730	0.610235	0.610235	0.998946	0.797380	0.809301	0.303556	0.781506	2.897851e-04
2	1	0.426071	0.499019	0.472295	0.601450	0.601364	0.998857	0.796403	0.808388	0.302035	0.780284	2.361297e-04
3	1	0.399844	0.451265	0.457733	0.583541	0.583541	0.998700	0.796967	0.808966	0.303350	0.781241	1.078888e-04
4	1	0.465022	0.538432	0.522298	0.598783	0.598783	0.998973	0.797366	0.809304	0.303475	0.781550	7.890000e+09

Figure 5.1: Portion of the Data Set [39]

5.3.2 Missing Value Analysis

Before we could begin modeling , missing values in the data set are a tremendous obstacle. Numerous machine learning techniques require that missing data be imputed or eliminated before continuing. Fortunately, the data set has no missing values.

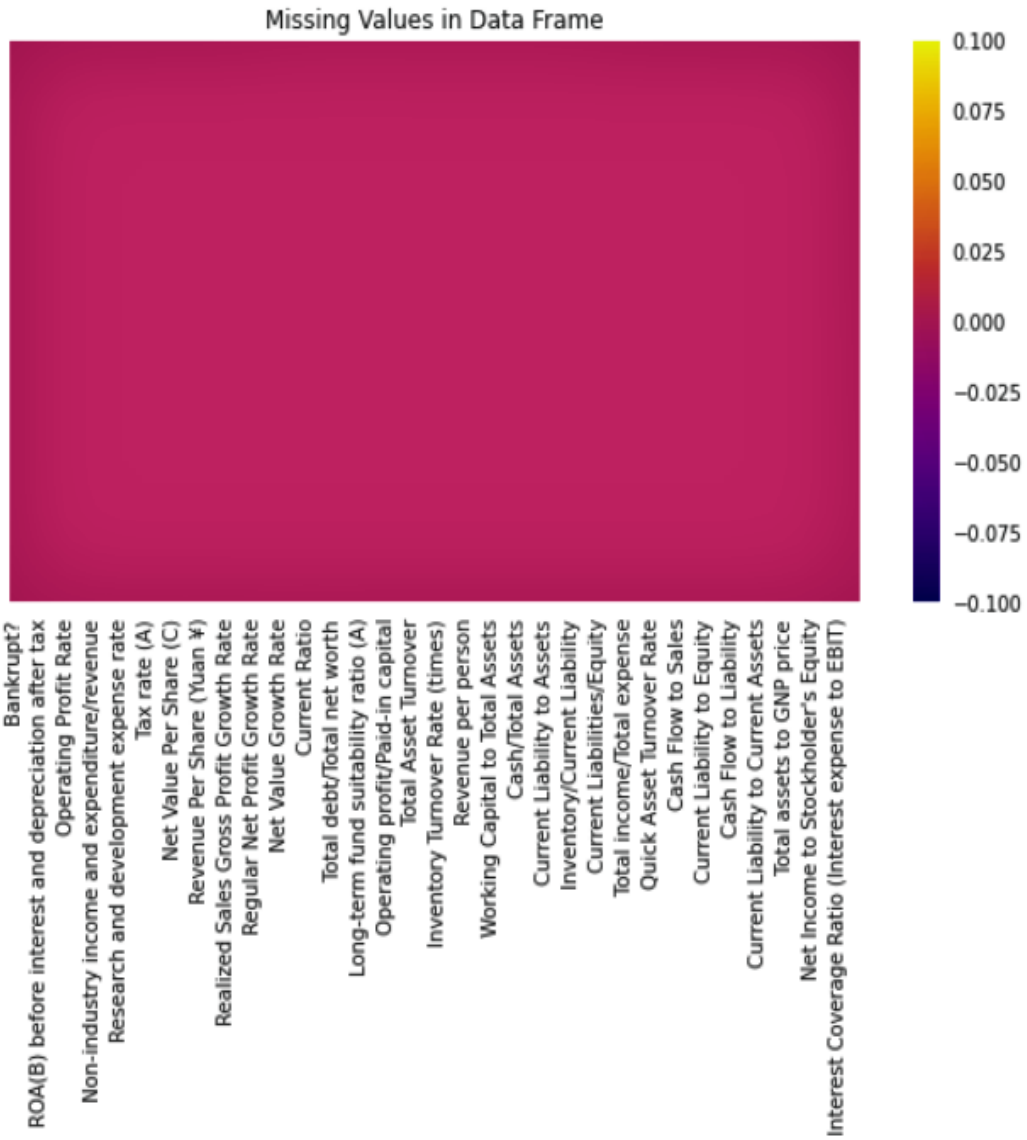


Figure 5.2: Number of Missing Values which is Zero, Represented by All Red Color Throughout the Figure.

In the figure 5.2, the color scale indicates that the color red corresponds to zero. After analyzing missing value in the data set, all of the features were colored red, indicating that there is no missing value in the data set.

5.3.3 Checking for Duplicate Data

Incorporating redundant information into a Machine Learning Model is a certain way to introduce bias into the resulting analysis and prediction. Therefore, it is important that our data set contains no duplicates. To this end, we performed a comprehensive search for instances of duplicate information and came up empty.

5.3.4 Statistical Data Description

The Statistical Summary provides information and a summary of the data. It provides information about the data set's values. This covers the location of the mean and the skewness of the data. The data's center or the position of a trend may be determined using measures of location such as mean and median. Spread measures indicate the data set's dispersion or diversity. This information might be crucial. For instance, test results within a certain range may be considered normal, but those within a different range could suggest a problem. The figure below depicts a statistical overview of our chosen data set

	Bankrupt?	ROA(C) before interest and depreciation before interest	ROA(A) before interest and % after tax	ROA(B) before interest and depreciation after tax	Operating Gross Margin	Realized Sales Gross Margin	Operating Profit Rate	Pre-tax net Interest Rate	After-tax net Interest Rate	Non-industry income and expenditure/revenue
count	6819.000000	6819.000000	6819.000000	6819.000000	6819.000000	6819.000000	6819.000000	6819.000000	6819.000000	6819.000000
mean	0.032263	0.505180	0.558625	0.553589	0.607948	0.607929	0.998755	0.797190	0.809084	0.303623
std	0.176710	0.060686	0.065620	0.061595	0.016934	0.016916	0.013010	0.012869	0.013601	0.011163
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.476527	0.535543	0.527277	0.600445	0.600434	0.998969	0.797386	0.809312	0.303466
50%	0.000000	0.502706	0.559802	0.552278	0.605997	0.605976	0.999022	0.797464	0.809375	0.303525
75%	0.000000	0.535563	0.589157	0.584105	0.613914	0.613842	0.999095	0.797579	0.809469	0.303585
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

8 rows × 96 columns

Figure 5.3: Statistical Data Description for a Portion of Data Set

5.3.5 Feature Distribution

A statistical dataset's distribution shows how often certain values fall within different ranges and how widespread those values are. The distribution provides a mathematical function with parameters that may be used to estimate the probability of any observation in the sample space. The feature distribution provides insight into the types of characteristics included in the data set and the range of values that can be expected for each. If the values are concentrated or spread out, it can be noticed. Since deep learning models learn from data, the distribution is critical. If they're fed false information, their conclusions will be incorrect. The quality of a deep learning model depends on the quality of its input data. Having almost similar training and validation subsets is ideal.

In this below figure 5.4, we can see feature distributions for all of the 96 features in the data set. Most of the features are highly skewed.

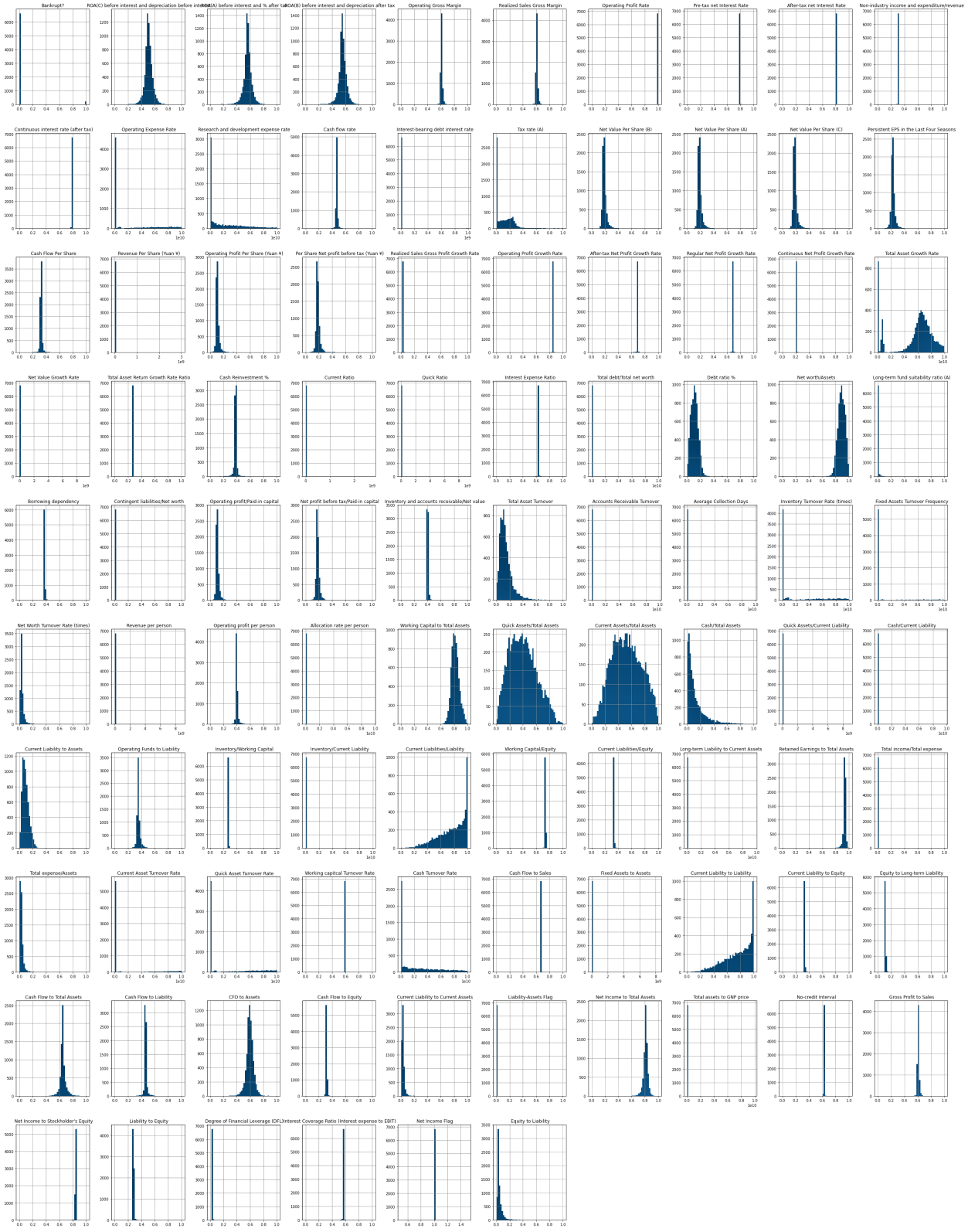


Figure 5.4: Feature Distributions of Taiwan Bankruptcy Data Set

5.3.6 Class Imbalance Check

Class Imbalance arises in data sets with an unbalanced number of observations. In other words, a binary classification task might have a large number of items from one class and a negligible number from another. This might also occur in a multi-classification situation if the great majority of data are grouped in one category or if one category is grossly underrepresented relative to the others as $96.77\%(6599)$ data is of survived company and $3.23\%(220)$ data is of bankrupted company

The imbalance issue is not explicitly defined, therefore there is no "official threshold" to indicate that we are in fact dealing with class imbalance, although a ratio of 1 to 10 is often unbalanced enough to warrant balancing procedures.

The majority of machine learning techniques assume that data are uniformly distributed. Consequently, when there is a class imbalance, machine learning classifiers tend to be more biased towards the majority class, resulting in inaccurate categorization of the minority class.

We have an issue with class imbalance in our data collection, as seen in the diagram below.

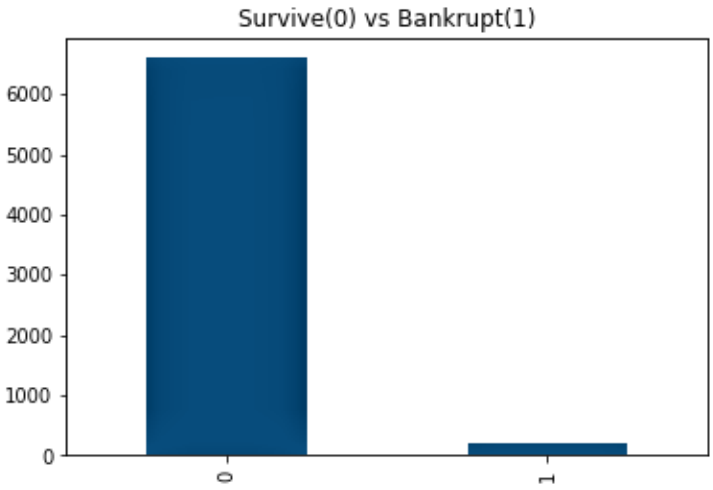


Figure 5.5: Class Imbalance in the Taiwanese Bankruptcy Prediction Data Set

5.4 Data Preprocessing

Exploratory data analysis revealed that the data set has issues such as skewed feature distribution and class imbalance. We have dealt with these problems under the Data Pre-processing subsection.

5.4.1 Feature Transformation

We have used Log Transformation and Power Transformation and compared their performances to rectify the asymmetry in the feature distribution. As for normal skewness, Power Transformation performed better than Log Transformation. We have so far used the Power Transformation outcomes. Quantile-Quantile Graph (Q-Q Graph) is used to illustrate the difference between the two transformations in the following diagram.

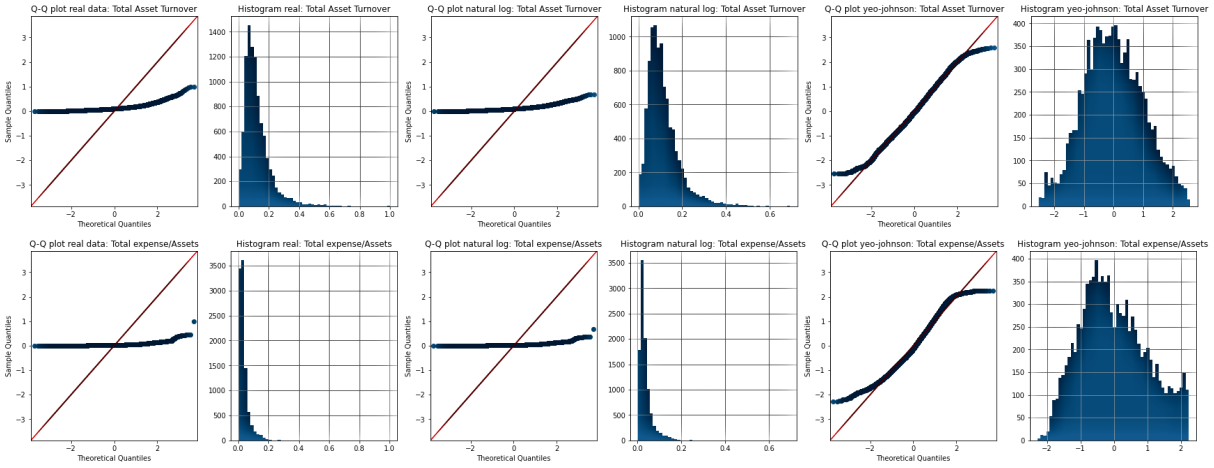


Figure 5.6: Q-Q graph comparison before(Left) and after Log(Middle) and Power(Right) Transformation

On the left side of Figure 5.6, we can see the Q-Q graph (X axis: Theoretical Quantiles, Y axis: Sample Quantiles) and feature distribution (X axis: Feature Value, Y axis: Cumulative Distribution Function) for a feature titled 'Total Asset Turnover' where the distribution is extremely skewed. Therefore, in order to eliminate the skewness and create a normal distribution, we've used two kinds of feature transformation methods: Log Transformation and Power Transformation: Yeo-Johnson. Figure 5.6 depicts the distributions after transformation for log trans-

formation and power transformation at the center and far right, respectively. We have shown the same things for another feature named 'Total Expense/Assets' on the bottom portion of the figure 5.6 .

5.4.2 Removal of Class Imbalance

We have used the SMOTE oversampling strategy [92] to rectify the issue of Class Imbalance. The outcome of SMOTE oversampling is shown below.

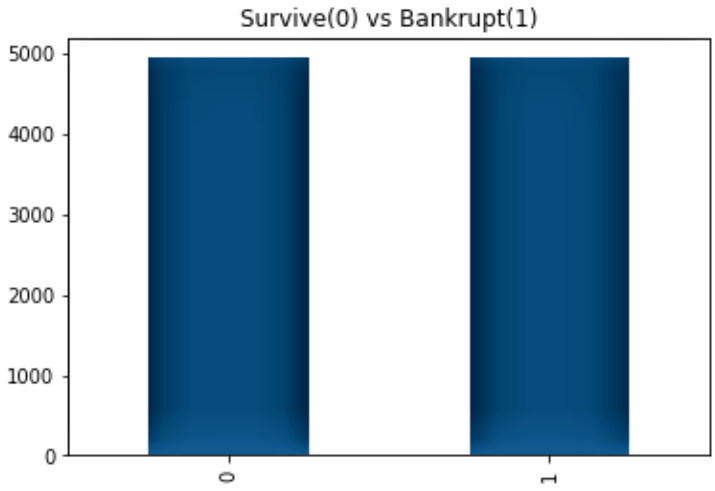


Figure 5.7: Class Balance Result After SMOTE Oversampling

5.5 Data Generation Model

We have used CTGAN [76] as our data generation model. In the generator, CTGAN uses Batch-Normalization [93] and ReLU activation function [94]. After two hidden layers, the synthetic row representation is generated using mixed activation functions. The scalar values are generated by tanh, while the mode indicator and discrete values are generated by Gumbel Softmax. In the Discriminator, CTGAN uses Leaky ReLU activation function and dropout on each hidden layer. CTGAN uses the PacGAN framework with 10 samples in each pack to prevent mode collapse. We train the model using WGAN loss with gradient penalty and use Adam optimizer.

The CTGAN model has been tuned using these hyperparameters in the below table.

Table 5.1: CTGAN Hyperparameter List

Hyperparameter Name	Value
Batch Size	500
Epochs	10000
Embedding Dimension	128
Generator Dimension	(256, 256)
Discriminator Dimension	(256, 256)
Generator Learning Rate	0.0005
Discriminator Learning Rate	0.0005

5.6 Synthetic Data Sampling

Up until this point, we have trained the model using the preprocessed Taiwan Bankruptcy Prediction Data Set as well as our very own distinctive values for each of the hyperparameters. In order to get a table from the CTGAN model that has been trained, we will need to perform some sort of sampling. We have acquired something close to the same number of synthetic data samples as we obtained from the real data set.

5.7 Conclusion

This chapter resembles the execution of our proposed system step by step. We have shown how we have obtained and analyzed data, as well as identified the flaws that have arisen from our exploratory data analysis and presented our solutions. We have ALSO discussed the generative model, its parameters which we have used to train the model and finally sampled some synthetic data.

Chapter 6

Result and Performence Analysis

6.1 Introduction

Synthetic data is one factor that, when used properly, may enhance the data-centric approach. Synthetic data are data created artificially and not acquired from actual occurrences. It duplicates the statistical components of genuine data without having any personally identifying information, so protecting the privacy of people. Our synthetic data thesis focuses solely on this. How can we assure that the synthetic data adheres to the same quality criteria as the original data was our major concern. Equally as important as retaining the statistical features of the original data is ensuring that it adheres to a stringent data quality standard.

In this chapter, we will guide you through a comprehensive assessment of performance in which we compare the quality standards of synthetic data to those of actual data using a number of methodologies and dimensions.

6.2 Exploratory Data Analysis (EDA) of Synthetic Data

In this section, we evaluated the quality of the Synthetic Data by doing exploratory data analysis on it.

6.2.1 Missing Value Analysis: Synthetic Data

The modeled synthetic data is complete in every respect. There is not a single missing value. The lack of missing values is indicative of the high quality of synthetic data. In addition, we have looked for duplicates in both the original training data and the produced data and have not discovered any, which is a very promising indicator of having successfully acquired high-quality data. The missing value analysis produces the following diagram.

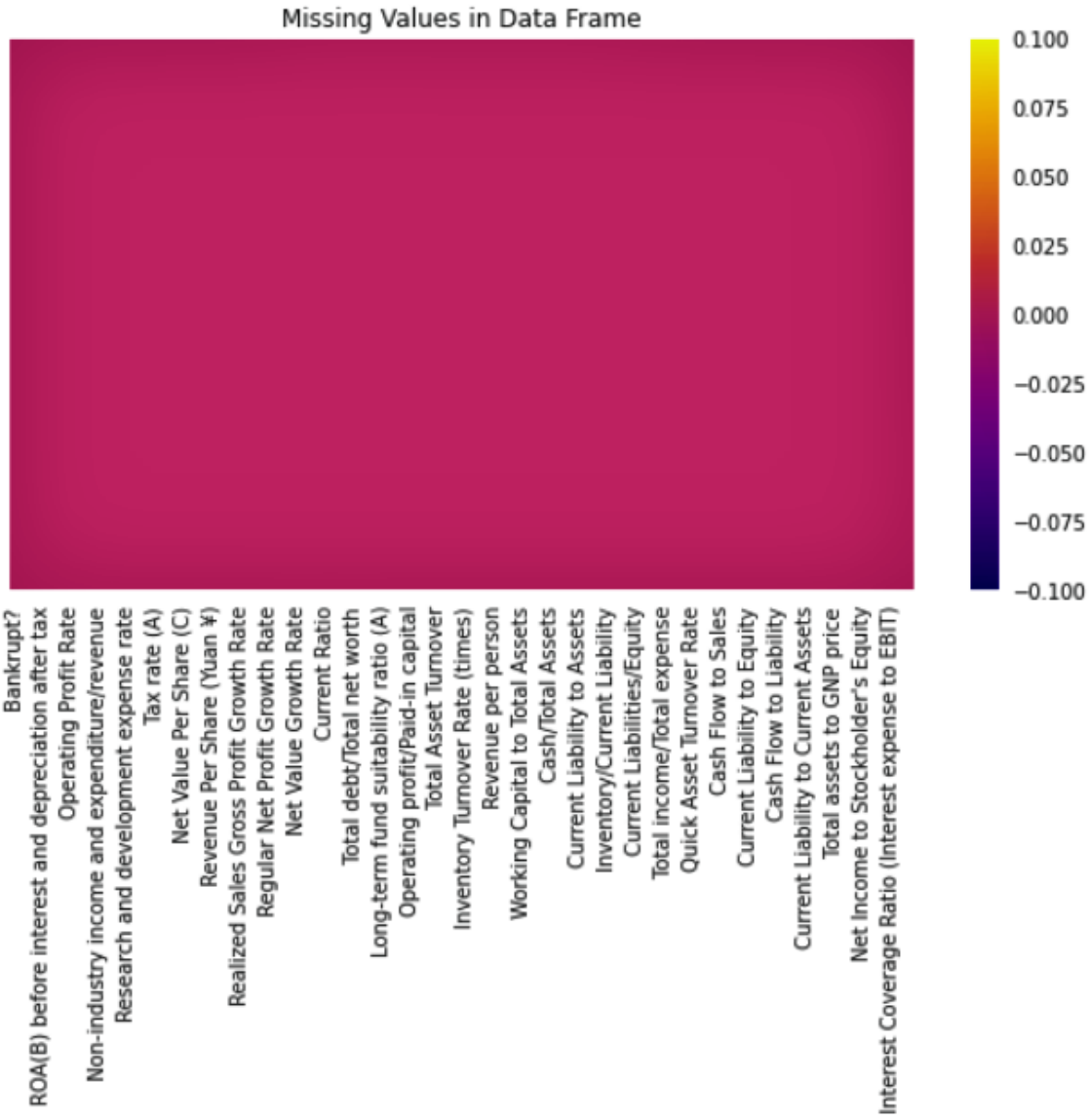


Figure 6.1: Number of Missing Values for a Portion of Synthetic Data which is Zero, Represented by All Red Color Throughout the Figure.

In the figure 5.2, the color scale indicates that the color red corresponds to zero. After analyzing missing value in the synthetic data set, all of the features were colored red, indicating that there is no missing value in the synthetic data set.

6.2.2 Class Imbalance Check

The synthetic data's class distribution matches that of the data used to generate it perfectly, hence there is no class imbalance in the generated data which also proves the class balance quality of synthetic data.

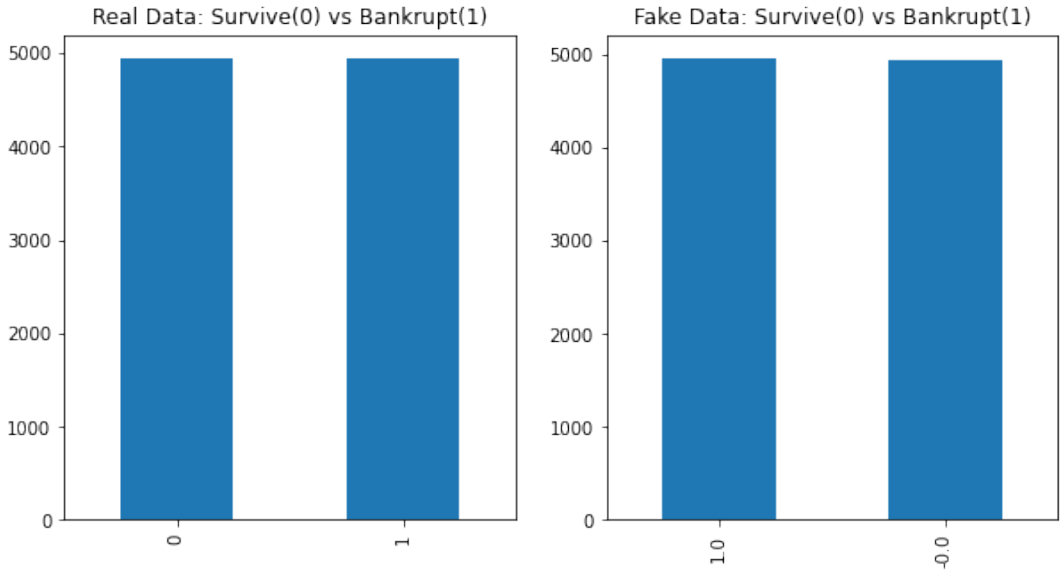


Figure 6.2: Class Imbalance Analysis in Real & Synthetic Data

6.2.3 Feature Distribution Comparison

Feature Distribution is a function that enumerates all potential Data values. It may be either discrete or continuous. Several frequent Probability Distribution functions provide probabilities for the occurrence of a feature's several possible outcomes. The probability distribution underlying the function may be used to calculate the probability density function. This helps us provide confidence intervals to the likely range of Data values. We may also compute the probability that an extreme value will occur. Comparing the feature distributions of actual and

synthetic data leads us to the conclusion that they are identical. This demonstrates that the synthetic data originated from the same distribution as the original data, as well as the probability of occurrence for the varied possible outcomes of the attributes.Similarly to the actual data, Synthetic Data recorded the same modality. Very little distance exists between the distributions of the actual data and the synthetic data, indicating that the synthetic data caught the distribution of the real data pretty well.

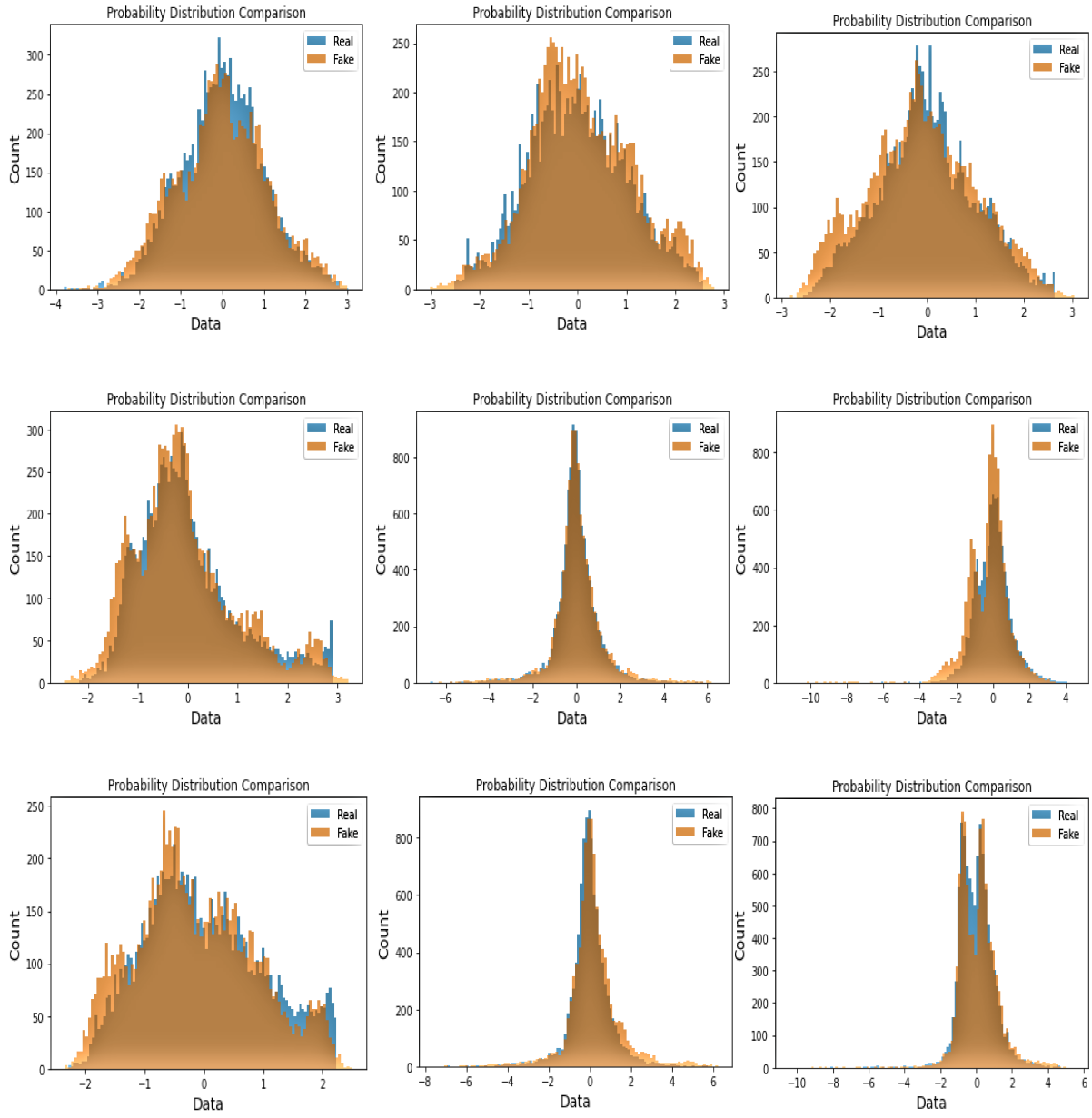


Figure 6.3: Distribution Comparison of Some Features: Real & Synthetic Data

6.3 Principal Component Analysis (PCA) Comparison

Principal Component Analysis (PCA) is a method that converts high-dimensional data into lower-dimensional data while maintaining as much information as feasible [95]. PCA is quite beneficial when dealing with data sets including several characteristics. PCA extracts meaning from data using mathematics. PCA comprehends which portion of our data is significant and calculates the quantity of information included inside the data. The more information there is, the larger the variance, and vice versa.

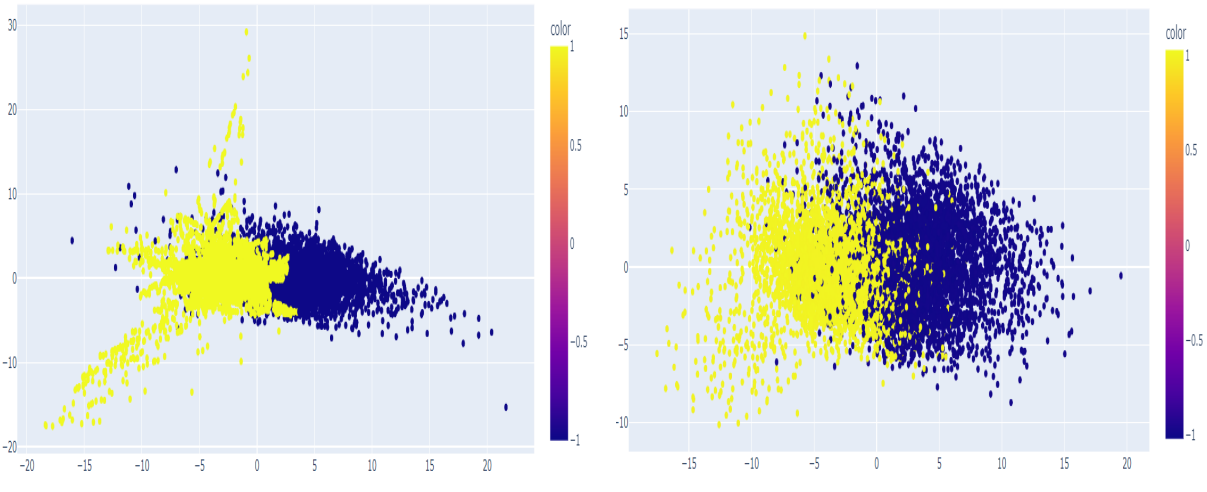


Figure 6.4: Two Most Descriptive PCA: Real(Left) & Synthetic(Right) Data

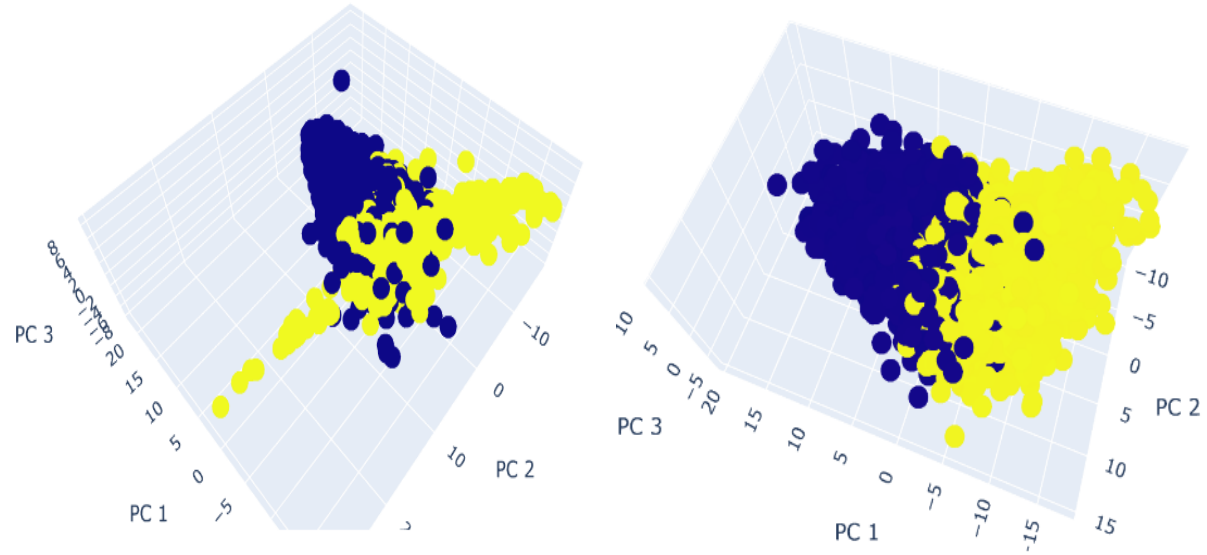


Figure 6.5: Three Most Descriptive PCA: Real(Left) & Synthetic(Right) Data

Here, we have done PCA on both Original and Synthetic data to determine if the first two and third main components of both datasets include comparable types of information. From the graphs, it is clear that the first two and third principal components of both the real and synthetic data carry a nearly same proportion of variance.

Table 6.1: Percentage of Captured Variance by Original and Synthetic Data

PCA	**Variance**	
	Original Data	**Synthetic Data**
Two Most Descriptive Principal Components	37.09%	36.76%
Three Most Descriptive Principal Components	48.74%	47.12%

6.4 Feature Correlation Comparison

Correlation is a statistical method for determining the association between two or more data set variables or features. By using Correlation several things can be learned such as:

(a) A number of qualities are reliant on a single attribute or on a single cause for a number of traits.

(b) Attributes may be linked to one another in one or more ways. When two characteristics are highly correlated, it may be possible to infer the third characteristic based on the second.

Numerous modeling strategies include correlation as a fundamental quantity. Correlations may be broken down into three distinct categories:

(a) **Positive Correlation**: For a feature A and feature B to be positively correlated, either feature A must rise or feature B must decrease for there to be a change in the other feature. Both characteristics are in constant motion together, and their connection is linear and represented by a range of 0 to 1.

(b) **Negative Correlation**: A negative correlation between two features indicates that while feature A grows, feature B will decrease and vice versa, represented by a range of negative values up to -1.

(c) **No Correlation**: No correlation is no relationship between the features.

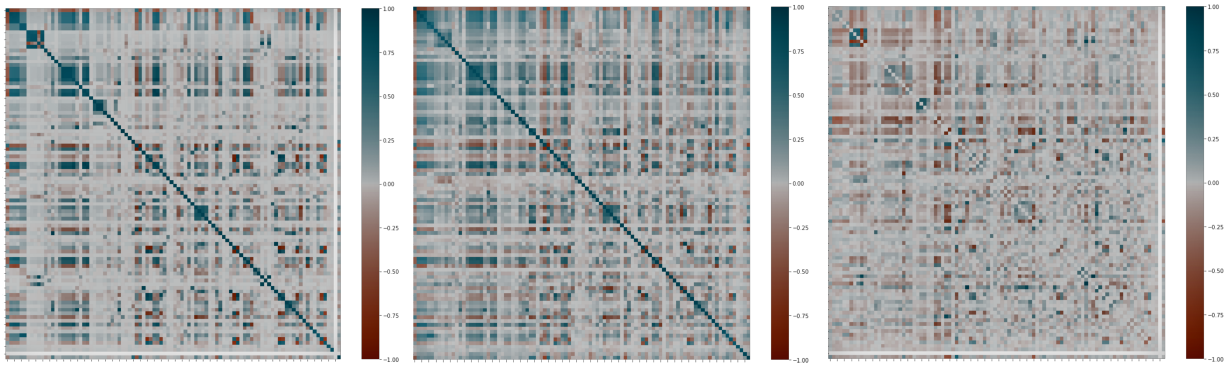


Figure 6.6: Feature Correlations Comparison Between Real & Synthetic Data

As shown in the figure 6.6, correlations in the original data are retained to a large degree in the synthetic data and the difference between them is very low. This guarantees that the resulting data closely resembles the original training data, preserving the consistency of the correlations. Maintaining constant correlations is crucial since we may utilize synthetic data in place of original data, and changing correlations will not provide the same insights as original data for any kind of application.

Additionally, we have examined the strongest three correlations found in both the real and synthetic data sets. They're quite close to one another, and their distribution stays almost unchanged from the source data.

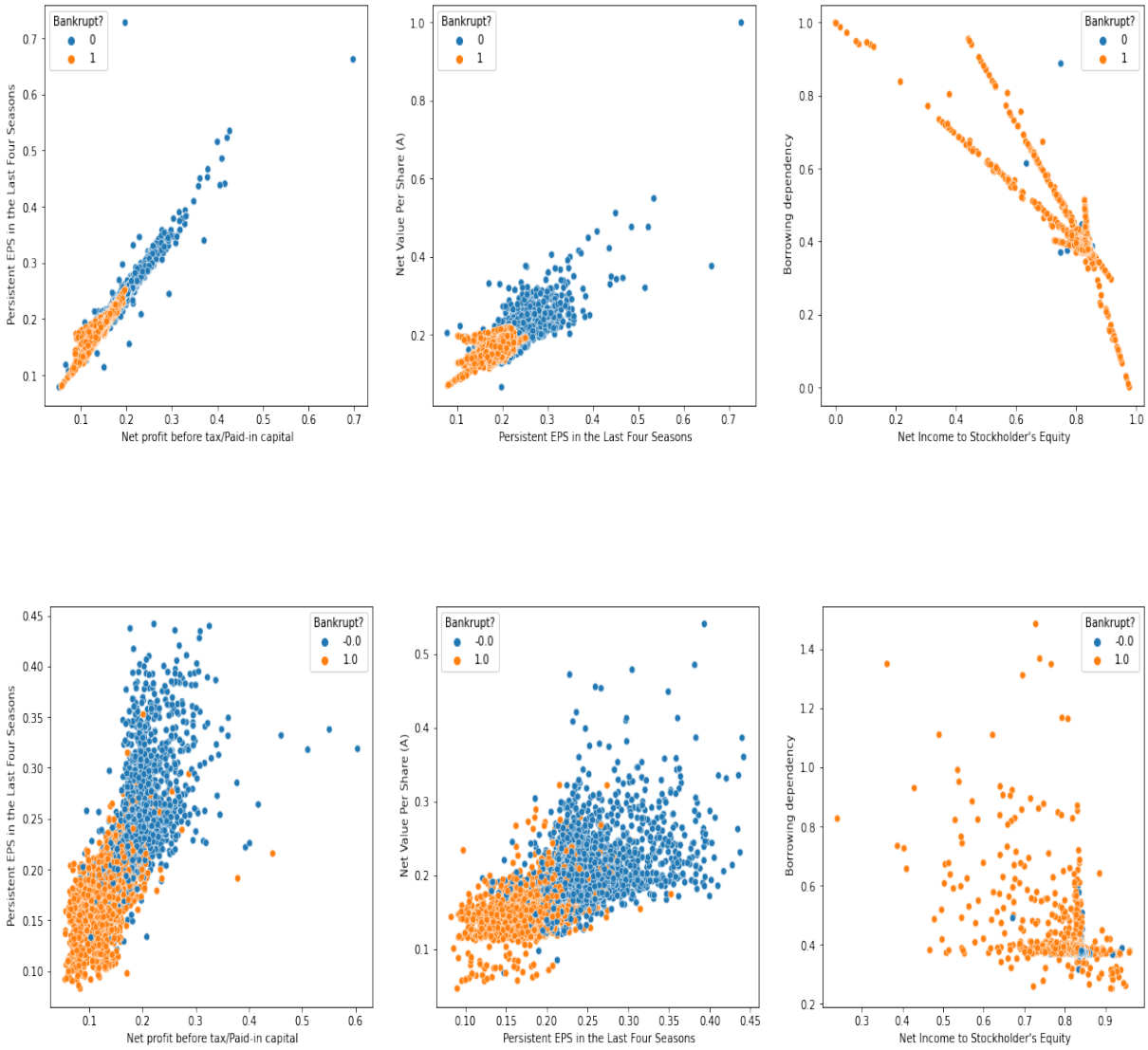


Figure 6.7: Top 3 Correlation Analysis: Real (Upper) & Synthetic (Lower) Data

6.5 Statistical Metrics

This collection of metrics compares the real and synthetic data using several sorts of statistical tests. In the simplest case, these metrics compare various aspects of the actual data to the corresponding feature of the synthetic data, and then return the average test result.

Table 6.2: Different Types of Statistical Tests on Synthetic Data

Metric	Name	Raw Score	Normalized Score	Minimum Value	Maximum Value	Goal
Logistic Detection	Logistic Regression Detection	0.983	0.98	0.0	1.00	Maximize
SVC Detection	SVC Detection	0.997	0.99	0.00	1.00	Maximize
KSTest	Kolomogorov-Smirnov Statistic	0.864	0.86	0.00	1.00	Maximize
KSTest Extended	Inverted Kolmogorov-Smirnov Statistic	0.883	0.88	0.00	1.00	Maximize

The best approach to conceptualize logistic regression is as a classification-specific version of linear regression. The major distinction between linear and logistic regression is that the range of logistic regression is confined between 0 and 1. The purpose of the logistic regression detection test is to maximize the score, and the fact that the synthetic data scored extremely near to 1 demonstrates the high quality.

Using SVC to separate two classes is predicated on locating support vectors (i.e., representative training data points) to establish the bounding hyperplanes where the margin between both planes is maximal. The number of support vectors increases as the complexity of the issue increases. Consequently, the mean score of all support vectors is calculated. As the goal is to maximize this score, the table reveals that the score is pretty close to 1, indicating the excellent quality of synthetic data.

The objective of the KSTest/KSTestExtended Test for Distributional Adequacy is to test for

discrepancies in the form of two distributions or to compare them to the predicted statistical distribution. This test is used to determine whether a sample originates from a population with a specified distribution. The K-S or Kolmogorov-Smirnov test is used to evaluate the performance of classification models. Specifically, K-S is a measurement of the degree of separation between the positive and negative distributions. K-S should be high $(Max = 1.0)$ when the fit is excellent and low $(Min = 0.0)$ when the fit is not good. When the K-S value falls below 0.05, the lack of fit is considered statistically significant. The fact that the KSTest scores are quite high indicates that the distributions of synthetic data are a good match for the real data, as seen inn the table.

6.6 Comparison of Major Features

We used a variety of feature selection techniques on both the real and synthetic data to identify key elements that helped us decide whether or not to declare bankruptcy. It demonstrates if the synthetic data have prioritized the same features as the real data and arrived at the same conclusion.

6.6.1 Recursive Feature Elimination with Cross-Validation (RFECV)

The RFE method, or Recursive Feature Elimination, is a wrapper technique. The algorithm identifies which properties are most important by fitting a model to the data and analyzing the results. After determining the significance of a feature, it gradually eliminates less significant elements in subsequent iterations. Until the ideal amount of characteristics is reached, they are gradually deleted. With an emphasis ranking, Recursive Feature Elimination with Cross-Validation highlights the features that matter most.

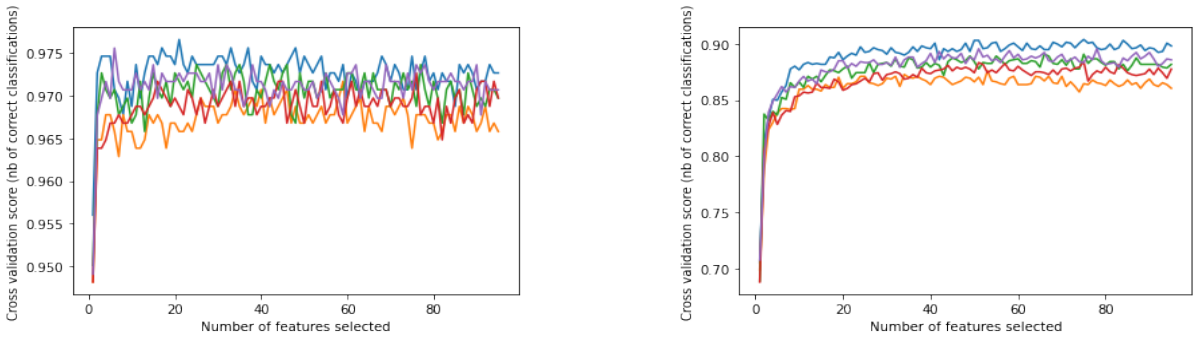


Figure 6.8: RFECV Feature Elimination: Real (Left) & Synthetic (Right) Data

According to the preceding graphs, RFECV indicates that real data requires just 32 of the total 96 characteristics to convey the relevant insights, whereas synthetic data requires 41 features. Compared to 96 characteristics, this deviation in the number of optimal features is negligible.

6.6.2 Chi-Squared Feature Selection

Chi-Square measures the correlation between two characteristics. In the table 6.3, the features picked using the Chi-Squared Feature Selection Method and their respective scores are shown. We observed the top 10 features with the highest scores, and found that 80% of features are essential in both actual and synthetic data.

Table 6.3: Top 10 Features Selected by Chi-Square Feature Selection

Real Data		**Synthetic Data**	
Specs	**Score**	**Specs**	**Score**
Fixed Assets Turnover Frequency	1.73E+12	Cash/Current Liability	1.28E+21
Cash/Current Liability	7.11E+11	Current Asset Turnover Rate	8.10E+12
Total assets to GNP price	2.28E+11	Operating Expense Rate	1.94E+11
Research and development expense rate	1.89E+11	Total Asset Growth Rate	1.88E+11
Total Asset Growth Rate	1.76E+11	Research and development expense rate	1.70E+11
Fixed Assets to Assets	1.32E+11	Cash Turnover Rate	1.69E+11
Net Value Growth Rate	1.09E+11	Fixed Assets to Assets	1.14E+11
Operating Expense Rate	1.08E+11	Net Value Growth Rate	8.30E+10
Cash Turnover Rate	9.00E+10	Fixed Assets Turnover Frequency	3.75E+04
Revenue per person	7.67E+10	Tax rate (A)	2.56E+02

6.6.3 Variance Threshold Feature Selection

The Variance Threshold selection for features filters out those with minimal variance. It is possible to utilize this feature selection technique for unsupervised learning since it focuses just on the features (X) and not on the intended results (Y). All features having a variance in the training set below this level will be discarded. We used this method on both the real and

simulated data with the same threshold, and we retrieved the characteristics that were over the threshold.

Table 6.4: Top Features Selected by Variance Threshold Feature Selection

Feature Selection: Real Data	**Feature Selection: Synthetic Data**
Operating Expense Rate	Operating Expense Rate
Research and development expense rate	Research and development expense rate
Interest-bearing debt interest rate	Interest-bearing debt interest rate
Revenue Per Share (Yuan ¥)	Revenue Per Share (Yuan ¥)
Total Asset Growth Rate	Total Asset Growth Rate
Net Value Growth Rate	Total Asset Growth Rate
Current Ratio	Current Ratio
Quick Ratio	Quick Ratio
Total debt/Total net worth	Total debt/Total net worth
Accounts Receivable	Accounts Receivable
Turnover	Turnover
Average Collection Days	Average Collection Days
Inventory Turnover Rate (times)	Inventory Turnover Rate (times)
Fixed Assets Turnover Frequency	Fixed Assets Turnover Frequency
Revenue per person	Revenue per person
Allocation rate per person	Allocation rate per person
Quick Assets/Current Liability	Quick Assets/Current Liability
Cash/Current Liability	Cash/Current Liability
Inventory/Current Liability	Inventory/Current Liability
Long-term Liability to Current Assets	Long-term Liability to Current Assets
Current Asset Turnover Rate	Current Asset Turnover Rate
Quick Asset Turnover Rate	Quick Asset Turnover Rate
Cash Turnover Rate	Cash Turnover Rate
Fixed Assets to Assets	Fixed Assets to Assets
Total assets to GNP price	

From the table 6.4, we can see the optimal number of features for real data is 24 whereas optimal number of features for synthetic data is 23. All the 23 features of synthetic data are present in the optimal features of original data also. The generation model just missed one feature according to variance threshold feature selection method which is very less with respect to 96 features in the original data.

6.7 Mutual Information (MI) Comparison

The amount of knowledge that may be gained about one random variable by knowing another is quantified by a quantity called mutual information. We used this statistical method to compare the mutual information included within the same features in the original and synthetic data.

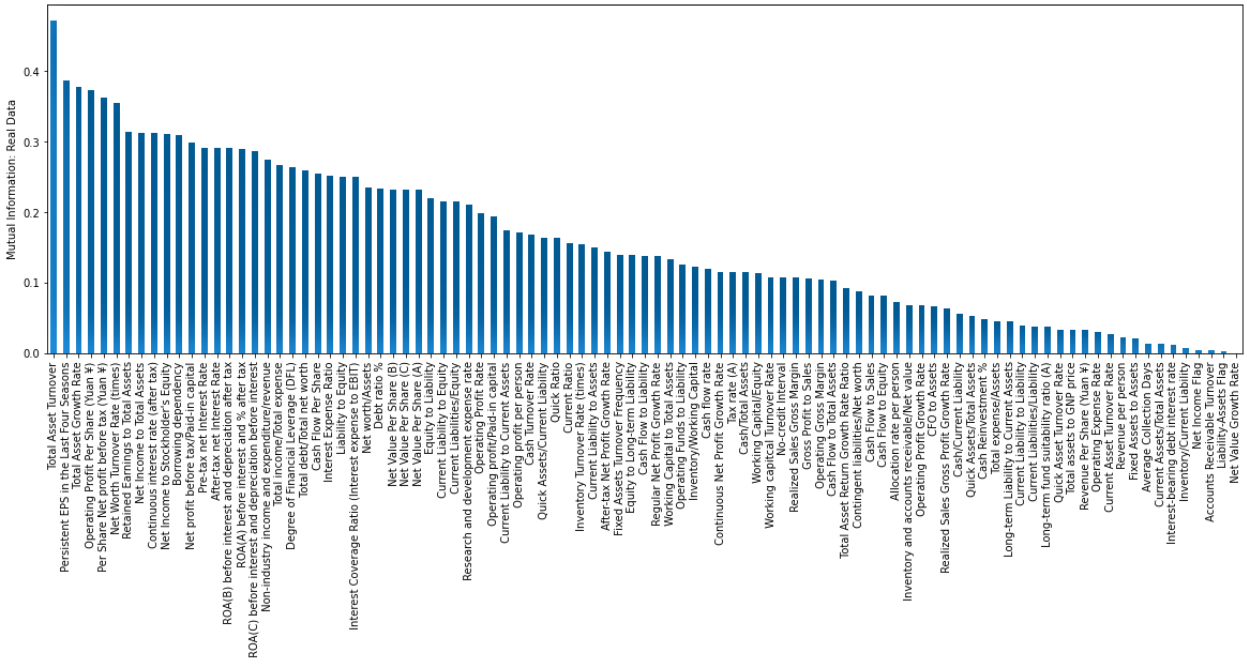


Figure 6.9: MI of Features: Real Data

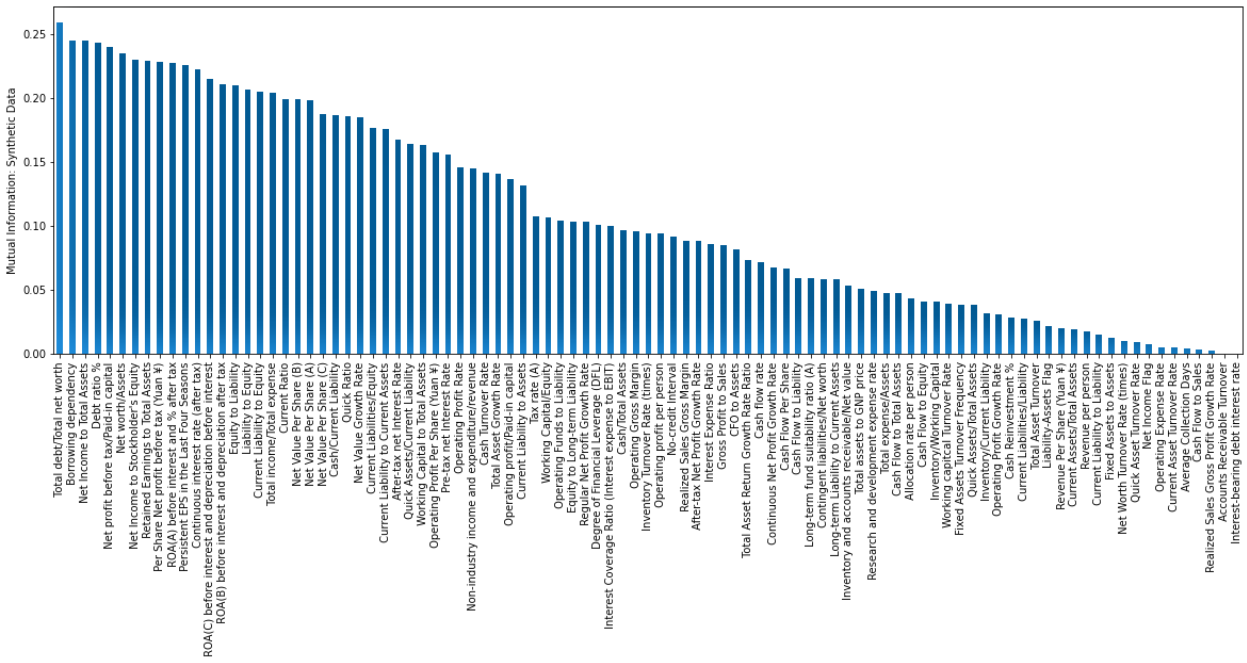


Figure 6.10: MI of Features: Synthetic Data

Figure 6.9 and figure 6.10 demonstrate that the mutual information among the characteristics are very consistent, a condition that guarantees the closeness of synthetic data to the original data.

6.8 Machine Learning Efficacy

This set of metrics will determine whether it is feasible to substitute actual data with synthetic data data in order to solve a machine Learning Problem by training a machine learning model on synthetic data and then assessing the score it achieves when assessed on real data.

Table 6.5: Machine Learning Model Efficacy

Real Data		**Synthetic Data**	
Model	**Best Score**	**Model**	**Best Score**
SVM	0.527173	SVM	0.509846
XGBoost	0.982219	XGBoost	0.962677

Real Data		Synthetic Data	
Model	**Best Score**	**Model**	**Best Score**
Random Forest	0.961103	Random Forest	0.955138
Logistic Regression	0.59214	Logistic Regression	0.560061
Naive Bayes Gaussian	0.594059	Naive Bayes Gaussian	0.590585
Decision Tree	0.951808	Decision Tree	0.912553

The table 6.5 demonstrates how machine learning models performed with both actual and fake data. We can observe that the outcomes are comparable. XGBoost scored the best with both actual and synthetic data, as did other models, despite changes in score for the same model. Based on the outcome, we may conclude that our synthesized data is a viable alternative to the original data.

6.9 Performance Analysis

Initially, we have performed the same exploratory data analysis (EDA) on synthetic data as we did on actual data. The EDA has shown that the synthetic data do not include any missing values, duplicate rows, or class imbalance issues, all of which are indicative of the quality of the synthetic data. Then, using principal component analysis, we analyzed the first major $2and3$ components for both actual and synthetic data and found that the recorded variance percentages were almost identical. Along with the top 3 correlations, we have examined all 96 feature correlations in both data sets and found that the difference is minimal. We have also examined the feature distributions visually and statistically using 4 distinct statistical tests, with both yielding extremely encouraging findings for synthetic data in comparison to the original data. Then, we identified key features to determine whether the synthetic data set exhibited the same focus on features as real data. We have used 3 different feature selection techniques like recursive feature elimination with cross-validation (RFECV), chi-squared feature selection and variance threshold feature selection. All of the 3 feature selection strategies have shown that the relevance of features in synthetic data is comparable to that of actual data. We have analyzed the

mutual information from both synthetic and actual data to see whether they contain a comparable amount of information. The comparison revealed that the information is quite similar, which confirms the quality of the synthetic data we create. Finally, we have trained prominent machine learning models using both real and synthetic data data to determine if synthetic data may substitute for actual data in the artificial intelligence research field. We have performed a head-to-head assessment of the accuracy of both data sets and discovered that their accuracy scores are extremely close, with the extreme gradient boost method providing the highest accuracy for both. Consequently, based on this exhaustive evaluation of synthetic data, we can conclude that our proposed system has created accurate and high-quality synthetic data.

6.10 Conclusion

In this chapter, we first conducted exploratory data analysis (EDA) using synthetic data. The EDA has revealed that synthetic data have no missing values, duplicate rows, or class imbalance concerns, indicating their quality. We then used principal component analysis to compare the variance percentages. Along with the top 3 correlations, we investigated all feature correlations in both data sets and discovered that the difference is minor. We have also compared feature distributions graphically and quantitatively using statistical tests, and synthetic data performed well with respect to the actual data. Next we have determined key features to see whether the synthetic data set focused on features like actual data. Recursive feature elimination with cross-validation (RFECV), chi-squared feature selection, and variance threshold feature selection algorithms have proved that synthetic data key features are equivalent to real data. The comparisonof mutual information showed that the synthetic data is identical. Finally, we have trained some popular machine learning models using both real and synthetic data to see whether synthetic data can replace real data in artificial intelligence research and found that both data sets' accuracy scores are quite similar. Based on this extensive evaluation of synthetic data, we can say our proposed system has generated precise and high-quality synthetic data.

Chapter 7

Conclusion and Future Works

7.1 Introduction

The goal of this thesis is to aid the development of a synthetic data generation system in a way that generates quality synthetic data preserving similar patterns and a similar statistical distribution of a real dataset. Most of the existing tools and approaches to generate synthetic data propose a new GAN model and simply are not focused on proving the quality of the synthetic data. Besides, most of the related studies do not use new machine learning technologies or do not address the generation of synthetic data of diverse types. Better preprocessing system, generative adversarial networks and a detailed evaluation system can be used to unravel these limitations. Therefore, the goal of this thesis's research is to develop a data pre-processing technique that will allow GANs to efficiently and effectively generate high-quality synthetic data, including but not limited to text data, discrete data, continuous data, time-series data, etc., that closely mimics the meaning and veracity of real data in terms of its patterns, distributions, correlations, and confidentiality. Additionally, an evaluation system that guarantees the integrity and usefulness of the synthetic data produced.

Our experimental findings and an analysis of their performance were presented in the preceding chapter. We have summarized our whole research work in this chapter, and then discussed the caveats of our study. Following this is a discussion of potential future works, and then the chapter concludes.

7.2 Thesis Summary

We used the "Taiwanese Bankruptcy Prediction Data Set" as our starting point. The data was then put to a comprehensive exploratory data analysis, including missing value analysis, duplicate row checking, statistical analysis, feature distribution analysis, class imbalance check, outlier analysis, etc. The research has shown a number of issues, including the data set's mostly skewed feature distributions and a difficulty with class imbalance.

In the implementation phase, we tackled the problem of skewed feature distributions by transforming the features. Using a q-q graph, we also examined several transformation algorithms and identified the most effective one for future usage. Then, we fixed the class imbalance issue by using the SMOTE oversampling strategy, which is much preferable than other sampling methods.

The preprocessed data were then fed into the CTGAN model, which was fine-tuned by modifying the hyperparameter values. After training the CTGAN model on the Taiwanese bankruptcy prediction dataset, we sampled an equal number of synthetic as well as original data.

Lastly, we have performed thorough evaluations across several aspects. Initially, we performed exploratory data analysis on the synthetic data set, which replicated the exploratory data analysis performed on the real data. This synthetic data collection has no missing values, duplicate rows, or issues with class imbalance, as revealed by exploratory data analysis. We have analyzed the distributions of synthetic and actual data. There was almost no difference in the distributions. PCA, feature correlations, mutual information, and major features demonstrated that the relevance of features in the synthetic data is equivalent to that of the actual data.

Statistical measures and the machine learning efficacy indicate that our created synthetic data may be utilized successfully in place of original data without affecting any performance criteria.

7.3 Contributions

Most significantly, our thesis work contributes by:

(a) The first contribution is an improved strategy for data preparation prior to feeding the GAN model.

(b) The second contribution is the fine-tuning of CTGAN model to generate quality synthetic data.

(c) The third contribution proposes a thorough evaluation technique for evaluating synthesized tabular data in multiple dimensions.

7.4 Limitations

While our study has been thorough, it does have certain limitations, which we detail below.

(a) One of the limitation of synthetic data is to generate a relational data like preserving the male-female characteristics or city-country relations etc.

(b) Major feature extraction has showed some score difference for synthetic data features with respect to the original data..

7.5 Future Works

The results achieved by our proposed methodology are really promising. Certainly, there is many area for development and several opportunities for new experiments. We have mentioned several prospective horizons below.

7.5.1 Using GANs for Solving Class Imbalance Problem

We began by resolving the issue of the class imbalance by using the SMOTE oversampling method, which is a statistical and distance-based approach to the data generation process. Nevertheless, this technique is capable of being improved and developed further via the use of GANs, which may potentially increase the quality of synthetic data.

7.5.2 GANs for Big Data

We are fully embracing the age of big data, in which billions of new data are produced each and every day. The vast majority of this massive amount of data is being produced by various social media sites. Big data is a challenge for privacy protection but is beneficial for models and solutions that are based on deep learning. The generation of high-quality synthetic big data using of GANs is also one of the overarching goals of this thesis work.

7.5.3 GANs for Heterogeneous Data

To this day, several distinct kinds of synthetic data may be produced by using a variety of individual generative models. Generative models still have a long way to go before they can overcome the difficulty of employing a unified generation model with data that is heterogeneous (including texts, images, and other types of data altogether). The elimination of this obstacle is one of our major long-term goals.

7.6 Conclusion

We have started by explaining synthetic data, its uses, and how to produce it. We then formalized the problem, our motivation, goals, and objectives. After that, we have covered the fundamentals of machine learning, deep learning, and generative models like GANs and VAEs. Next, we explored synthetic tabular data models, their abilities and limitations. We then presented our proposed system. Our starting step was the collection of ”Taiwanese Bankruptcy

Prediction Data Set." Missing value analysis, duplicate row checking, statistical analysis, feature distribution analysis, class imbalance check and exploratory data analysis were performed on the data . The data set's missing values, skewed feature distributions and class imbalance problems were among the flaws identified by the analysis and resolved them. Skewed feature distributions were resolved in the implementation phase by transforming the features. We also tested different transformation methods on q-q graphs to find the best one for future use. Then, we rectified class imbalance by utilizing the SMOTE oversampling approach, which is better than other sampling methods. methods. The preprocessed data were supplied into the CTGAN model, which was tuned by changing hyperparameter values. Finally, we have conducted thorough evaluations in numerous areas. We started by exploratory data analysis on the synthetic data set, which duplicated the exploratory data analysis on the actual data. This synthetic data set contains no duplicate rows or missing values or class imbalance, according to the exploratory data analysis. The distributions of synthetic and real data were examined. There was very little variation in the distributions. PCA, feature correlations, mutual information, and significant features showed that synthetic data features are as relevant as real data features.Statistical metrics and machine learning effectiveness demonstrated that our synthetic data may replace original data without impacting performance requirements in the field of artificial intelligence. There were some score difference for some major features in the synthetic data which we are aiming to solve in the future along with our other future goals like generating big data, heterogeneous data etcetera.

REFERENCES

[1] S. I. Nikolenko *et al.*, *Synthetic data for deep learning*. Springer, 2021.

[2] A. Munappy, J. Bosch, H. H. Olsson, A. Arpteg, and B. Brinne, "Data management challenges for deep learning," in *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 140–147, IEEE, 2019.

[3] H. Surendra and H. Mohan, "A review of synthetic data generation methods for privacy preserving data publishing," *International Journal of Scientific & Technology Research*, vol. 6, no. 3, pp. 95–101, 2017.

[4] J. Eno and C. W. Thompson, "Generating synthetic data to match data mining patterns," *IEEE Internet Computing*, vol. 12, no. 3, pp. 78–82, 2008.

[5] G. D. P. Regulation, "General data protection regulation (gdpr)—final text neatly arranged," *Retrieved April*, vol. 27, p. 2019, 2018.

[6] "The Sustainable Development Goals Report 2022." `https://unstats.un.org/sdgs/report/2022`. Accessed: 2022-10-05.

[7] Z.-H. Zhou, *Machine learning*. Springer Nature, 2021.

[8] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electronic Markets*, vol. 31, no. 3, pp. 685–695, 2021.

[9] R. E. Schapire and Y. Freund, "Foundations of machine learning," 2012.

[10] P. Cunningham, M. Cord, and S. J. Delany, "Supervised learning," in *Machine learning techniques for multimedia*, pp. 21–49, Springer, 2008.

[11] Z. Ghahramani, "Unsupervised learning," in *Summer school on machine learning*, pp. 72–112, Springer, 2003.

[12] X. J. Zhu, "Semi-supervised learning literature survey," 2005.

[13] M. A. Wiering and M. Van Otterlo, "Reinforcement learning," *Adaptation, learning, and optimization*, vol. 12, no. 3, p. 729, 2012.

[14] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications," *Proceedings of the IEEE*, vol. 109, no. 3, pp. 247–278, 2021.

[15] F. Kratzert, "Understanding the backward pass through batch normalization layer," *Flair of Machine Learning [online]*, 2016.

[16] S.-i. Amari, "Backpropagation and stochastic gradient descent method," *Neurocomputing*, vol. 5, no. 4-5, pp. 185–196, 1993.

[17] S. Bhattarai, "What is gradient descent in machine learning," *A TECH BLOG*, 2018.

[18] S. Leijnen and F. v. Veen, "The neural network zoo," *Multidisciplinary Digital Publishing Institute Proceedings*, vol. 47, no. 1, p. 9, 2020.

[19] R. Salakhutdinov, "Learning deep generative models," *Annual Review of Statistics and Its Application*, vol. 2, pp. 361–385, 2015.

[20] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan, "Do deep generative models know what they don't know?," *arXiv preprint arXiv:1810.09136*, 2018.

[21] M. Tschannen, O. Bachem, and M. Lucic, "Recent advances in autoencoder-based representation learning," *arXiv preprint arXiv:1812.05069*, 2018.

[22] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[23] W. Wang, Y. Huang, Y. Wang, and L. Wang, "Generalized autoencoder: A neural network framework for dimensionality reduction," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 490–497, 2014.

[24] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[25] R. Lopez, J. Regier, M. I. Jordan, and N. Yosef, "Information constraints on auto-encoding variational bayes," *Advances in neural information processing systems*, vol. 31, 2018.

[26] D. P. Kingma, M. Welling, *et al.*, "An introduction to variational autoencoders," *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019.

[27] C. W. Fox and S. J. Roberts, "A tutorial on variational bayesian inference," *Artificial intelligence review*, vol. 38, no. 2, pp. 85–95, 2012.

[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[29] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F.-Y. Wang, "Generative adversarial networks: introduction and outlook," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 4, pp. 588–598, 2017.

[30] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[31] T. D. Akinosho, L. O. Oyedele, M. Bilal, A. O. Ajayi, M. D. Delgado, O. O. Akinade, and A. A. Ahmed, "Deep learning in the construction industry: A review of present status and future innovations," *Journal of Building Engineering*, vol. 32, p. 101827, 2020.

[32] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[33] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, pp. 214–223, PMLR, 2017.

[34] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *arXiv preprint arXiv:1605.09782*, 2016.

[35] L. Xu and K. Veeramachaneni, "Synthesizing tabular data using generative adversarial networks," *arXiv preprint arXiv:1811.11264*, 2018.

[36] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, "Generating multi-label discrete patient records using generative adversarial networks," in *Machine learning for healthcare conference*, pp. 286–305, PMLR, 2017.

[37] B. Dai, S. Ding, and G. Wahba, "Multivariate bernoulli distribution," *Bernoulli*, vol. 19, no. 4, pp. 1465–1483, 2013.

[38] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim, "Data synthesis based on generative adversarial networks," *arXiv preprint arXiv:1806.03384*, 2018.

[39] D. Liang and C.-F. Tsai, "Taiwanese bankruptcy prediction data set," 2016.

[40] S. G. Heeringa, B. T. West, and P. A. Berglund, *Applied survey data analysis*. chapman and hall/CRC, 2017.

[41] "Company Bankruptcy Prediction." `https://www.kaggle.com/datasets/fedesoriano/company-bankruptcy-prediction`. Accessed: 2022-10-05.

[42] J. A. Torres Munguía, "Comparison of imputation methods for handling missing categorical data with univariate pattern," *Revista de Métodos Cuantitativos para la Economía y la Empresa*, vol. 17, pp. 101–120, 2014.

[43] M. Noor, A. Yahaya, N. A. Ramli, and A. M. M. Al Bakri, *Filling missing data using interpolation methods: Study on the effect of fitting distribution*, vol. 594. Trans Tech Publ, 2014.

[44] M. R. Stavseth, T. Clausen, and J. Røislien, "How handling missing data may impact conclusions: A comparison of six different imputation methods for categorical questionnaire data," *SAGE open medicine*, vol. 7, p. 2050312118822912, 2019.

[45] G. Chhabra, V. Vashisht, and J. Ranjan, "A comparison of multiple imputation methods for data with missing values," *Indian Journal of Science and Technology*, vol. 10, no. 19, pp. 1–7, 2017.

[46] A. Jadhav, D. Pramod, and K. Ramanathan, "Comparison of performance of data imputation methods for numeric dataset," *Applied Artificial Intelligence*, vol. 33, no. 10, pp. 913–933, 2019.

[47] R. Zhang, M. Indulska, and S. Sadiq, "Discovering data quality problems," *Business & Information Systems Engineering*, vol. 61, no. 5, pp. 575–593, 2019.

[48] R. L. Ott and M. T. Longnecker, *An introduction to statistical methods and data analysis*. Cengage Learning, 2015.

[49] P. Schratz, J. Muenchow, E. Iturritxa, J. Richter, and A. Brenning, "Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data," *Ecological Modelling*, vol. 406, pp. 109–120, 2019.

[50] A. R. S. Parmezan, V. M. Souza, and G. E. Batista, "Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model," *Information sciences*, vol. 484, pp. 302–337, 2019.

[51] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, pp. 1–54, 2019.

[52] F. Thabtah, S. Hammoud, F. Kamalov, and A. Gonsalves, "Data imbalance in classification: Experimental evaluation," *Information Sciences*, vol. 513, pp. 429–441, 2020.

[53] E. Rendon, R. Alejo, C. Castorena, F. J. Isidro-Ortega, and E. E. Granda-Gutierrez, "Data sampling methods to deal with the big data multi-class imbalance problem," *Applied Sciences*, vol. 10, no. 4, p. 1276, 2020.

[54] A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla, "Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary," *Journal of artificial intelligence research*, vol. 61, pp. 863–905, 2018.

[55] T. Pan, J. Zhao, W. Wu, and J. Yang, "Learning imbalanced datasets based on smote and gaussian distribution," *Information Sciences*, vol. 512, pp. 1214–1233, 2020.

[56] J. Bell, *Machine learning: hands-on for developers and technical professionals*. John Wiley & Sons, 2020.

[57] J. Brownlee, *Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python*. Machine Learning Mastery, 2020.

[58] J.-M. Jo, "Effectiveness of normalization pre-processing of big data to the machine learning performance," *The Journal of the Korea institute of electronic communication sciences*, vol. 14, no. 3, pp. 547–552, 2019.

[59] L. Sürücü and A. MASLAKÇI, "Validity and reliability in quantitative research," *Business & Management Studies: An International Journal*, vol. 8, no. 3, pp. 2694–2726, 2020.

[60] H. Shan and E. I. Gubin, “Data cleaning for data analysis,” in *Молодежь и современные информационные технологии: сборник трудов XVI Международной научно-практической конференции студентов, аспирантов и молодых учёных, 3-7 декабря 2018 г., г. Томск.—Томск, 2019.*, pp. 387–388, 2019.

[61] T. A. Runkler, *Data analytics*. Springer, 2020.

[62] C. K. Enders, *Applied missing data analysis*. Guilford Publications, 2022.

[63] J. Brownlee, “Data preparation for machine learning,” 2022.

[64] H. Liu, S. Shah, and W. Jiang, “On-line outlier detection and data cleaning,” *Computers & chemical engineering*, vol. 28, no. 9, pp. 1635–1647, 2004.

[65] H. Zhuang, X. Wang, M. Bendersky, and M. Najork, “Feature transformation for neural ranking models,” in *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pp. 1649–1652, 2020.

[66] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, “A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction,” *Journal of Applied Science and Technology Trends*, vol. 1, no. 2, pp. 56–70, 2020.

[67] W.-C. Lin and C.-F. Tsai, “Missing value imputation: a review and analysis of the literature (2006–2017),” *Artificial Intelligence Review*, vol. 53, no. 2, pp. 1487–1509, 2020.

[68] D. Petelin, “Gaussian processes for machine learning,” 2006.

[69] F. Changyong, W. Hongyue, L. Naiji, C. Tian, H. Hua, L. Ying, *et al.*, “Log-transformation and its implications for data analysis,” *Shanghai archives of psychiatry*, vol. 26, no. 2, p. 105, 2014.

[70] D. Curran-Everett, “Explorations in statistics: the log transformation,” *Advances in physiology education*, vol. 42, no. 2, pp. 343–347, 2018.

[71] W. W. Chen and R. S. Deo, “Power transformations to induce normality and their applications,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 66, no. 1, pp. 117–130, 2004.

[72] A. C. Atkinson, M. Riani, and A. Corbellini, "The box–cox transformation: Review and extensions," *Statistical Science*, vol. 36, no. 2, pp. 239–255, 2021.

[73] J. Raymaekers and P. J. Rousseeuw, "Transforming variables to central normality," *Machine Learning*, pp. 1–23, 2021.

[74] S. S. Dhar, B. Chakraborty, and P. Chaudhuri, "Comparison of multivariate distributions using quantile–quantile plots and related tests," *Bernoulli*, vol. 20, no. 3, pp. 1484–1506, 2014.

[75] P. Guha and B. Chakraborty, "On a multivariate generalization of quantile-quantile plot," in *BOOK OF ABSTRACTS*, p. 62, 2009.

[76] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional gan," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[77] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, "Veegan: Reducing mode collapse in gans using implicit variational learning," *Advances in neural information processing systems*, vol. 30, 2017.

[78] A. Figueira and B. Vaz, "Survey on synthetic data generation, evaluation methods and gans," *Mathematics*, vol. 10, no. 15, p. 2733, 2022.

[79] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[80] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[81] R. Bro and A. K. Smilde, "Principal component analysis," *Analytical methods*, vol. 6, no. 9, pp. 2812–2831, 2014.

[82] M. Ringnér, "What is principal component analysis?," *Nature biotechnology*, vol. 26, no. 3, pp. 303–304, 2008.

[83] M. A. Hall, "Correlation-based feature selection of discrete and numeric class machine learning," 2000.

[84] V. W. Berger and Y. Zhou, "Kolmogorov–smirnov test: Overview," *Wiley statsref: Statistics reference online*, 2014.

[85] M. F. Zibran, "Chi-squared test of independence," *Department of Computer Science, University of Calgary, Alberta, Canada*, vol. 1, no. 1, pp. 1–7, 2007.

[86] D. M. Tax and R. P. Duin, "Support vector data description," *Machine learning*, vol. 54, no. 1, pp. 45–66, 2004.

[87] X.-w. Chen and J. C. Jeong, "Enhanced recursive feature elimination," in *Sixth International Conference on Machine Learning and Applications (ICMLA 2007)*, pp. 429–435, IEEE, 2007.

[88] A.-M. Bidgoli and M. N. Parsa, "A hybrid feature selection by resampling, chi squared and consistency evaluation techniques," *World Academy of Science, Engineering and Technology*, vol. 68, pp. 276–285, 2012.

[89] M. A. Munson and R. Caruana, "On feature selection, bias-variance, and bagging," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 144–159, Springer, 2009.

[90] B. Gierlichs, L. Batina, P. Tuyls, and B. Preneel, "Mutual information analysis," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 426–442, Springer, 2008.

[91] D. Rankin, M. Black, R. Bond, J. Wallace, M. Mulvenna, G. Epelde, *et al.*, "Reliability of supervised machine learning using synthetic data in health care: Model to preserve privacy for data sharing," *JMIR Medical Informatics*, vol. 8, no. 7, p. e18910, 2020.

[92] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[93] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pp. 448–456, PMLR, 2015.

[94] Y. Li and Y. Yuan, “Convergence analysis of two-layer neural networks with relu activation,” *Advances in neural information processing systems*, vol. 30, 2017.

[95] M. Greenacre and J. Blasius, *Multiple correspondence analysis and related methods*. Chapman and Hall/CRC, 2006.