Evaluating ML Models Using Data Generated by GAN

Amirhossein Mashghdoust
Department of Computer Science
University of Manitoba
Winnipeg, Canada
Student Id: 07957769
mashghda@myumanitoba.ca

Teresa Seethanaboyina
Department of Computer Science
University of Manitoba
Winnipeg, Canada
Student Id: 07910431
seethant@myumanitoba.ca

***Abstract*—Synthetic data generation is turning into a crucial and demanding task in the field of Artificial Intelligence and Machine learning. Generative Adversarial Networks (GANs) are the more renowned and effective way of generating highly realistic variations of real-world data compared to other techniques, thereby our choice for the project. It finds non-linear relations between features of data based on feature activation functions. We have produced tabular synthetic data using GAN from a real-world airline passenger dataset. The data is divided into categories: only real, combination, and only synthetic. We train different variations of Machine Learning models on these types of data and compute the accuracies. We then analyze as to which ML models work well with the synthetic data compared to just the real data.**

***Index Terms*—synthetic data, Generative Adversarial Networks, tabular, accuracy, ML model**

I. Introduction

Data collection, processing, and management have become integral parts of the modern economy. Thereby the increased demand for real-world high-quality data. Real data has become the heart of research in the field of artificial intelligence, providing new insights, and novel discoveries which are beyond regular human abilities. To address the data sensitivity issues that come along, rigid data privacy rules are in place, resulting in increased difficulty in gathering and labeling real data for training AI models. The solution is Synthetic data, the computer-generated information for testing and training AI models that have become indispensable in the data-driven era. Its production is less expensive, comes pre-labeled, and bypasses many of the logistical, ethical, regulatory and privacy issues that come with training deep learning models on real-world samples [1]. This sort of information enables growth and may involve different kinds of data, including uncommon scenarios that are infrequent in actuality.

Synthetic data generation methods

- *Based on the statistical distribution:* This method involves generating numbers by examining the actual statistical patterns, to replicate similar factual information. In instances where real data is unavailable, this factual information can be utilized.
  A data scientist can produce a dataset with a random distribution sample by having a comprehensive knowledge of the statistical distribution in actual data. This can be accomplished using various distribution techniques such as the normal distribution, chi-square distribution, and exponential distribution. The accuracy of the model developed is significantly impacted by the proficiency of the data scientist in implementing this method. [2].
- *Based on an agent to model:* This technique enables the development of a model that elucidates observed actions, and it generates random information using the same model. This involves matching real data to the established data distribution, and organizations can employ this approach for generating synthetic data.
  In addition to this, alternative machine learning techniques may be employed to conform to the distributions. However, if the data scientist aims to forecast the future, using a decision tree may result in overfitting due to its simplicity and exhaustive depth.
  Moreover, in some instances, a portion of actual data may be accessible. In such circumstances, businesses can adopt a hybrid approach by creating

a dataset founded on statistical distributions and generating synthetic data through agent modeling derived from real data. [2].

- *Using deep learning:* Techniques for generating synthetic data involve utilizing deep learning models, such as a Variational Autoencoder or Generative Adversarial Network model.

  – VAEs are a form of unsupervised machine learning models with encoders that compress and condense real data, while decoders examine this data to create a representation of the original information. The primary purpose of utilizing VAEs is to guarantee that the input and output data remain highly comparable. The objective is to decrease the reconstruction error through numerous training iterations. [2].
  – Generative Adversarial Network (GAN) is a type of neural network comprising two interconnected networks. One network called the generator, generates synthetic representations, while the other network, known as the discriminator, learns to differentiate actual images from synthetic ones produced by the generator. Over time, the generator improves its ability to generate realistic synthetic data [3].

II. BACKGROUND

Generative Adversarial Networks (GANs) are a method of generative modeling that produces a novel collection of information based on the training data, which resembles the training data. GANs are comprised of two primary components (i.e., two neural networks) that compete against each other and can grasp, replicate, and examine the differences within a dataset.

- Generative – To acquire knowledge about a generative model, which depicts how information is generated in the form of a probability model. In plain language, it clarifies how data is generated visually.
- Adversarial – The training of the model is done in an adversarial setting.
- Networks – Usage of deep neural networks for training purposes [4].

Intuition: GANs consist of two significant elements, namely the Generator and Discriminator. The generator's function is akin to that of a thief who produces counterfeit samples based on authentic ones and misleads the discriminator into identifying them as real. On the other hand, the discriminator functions as a police officer whose job is to detect any irregularities in the samples generated by the generator and classify them as real or fake. This competition between the two components continues until an optimal level of perfection is reached, where the generator triumphs by tricking the discriminator with fake/synthetic data.
Components:

- Discriminator – This approach is supervised, indicating that it is a basic classifier that forecasts whether the data is real or fake. It is trained on authentic data and provides input to a generator.
- Generator – This method utilizes unsupervised learning techniques to produce synthetic data that is derived from real data. It is also a neural network that contains hidden layers, activation functions, and loss functions. Its purpose is to create phony images by receiving feedback and deceiving the discriminator, which is unable to differentiate between real and fake images. Once the generator successfully fools the discriminator, the training process ends, and a generalized GAN model is formed [4].

Figure [**?**] shows the architecture and working of GAN with multiple layers.

The generator's goal is to produce images that deceive the discriminator, while the discriminator's goal is to make accurate judgments. They engage in a game where they try to exploit each other's weaknesses. This results in the networks improving themselves by addressing their vulnerabilities, as their opponent is constantly exploiting them. Whenever a successful strategy is employed against the opponent, the opponent is required to adapt to it. As a result, the system continually challenges itself to become better by engaging in a game against a progressively more formidable opponent. This game between generator and discriminator is the minimax game and can be described with the following equation [4] [5] [6].

$$\mathbb{E}_x[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}}[\log(1 - D(G(\boldsymbol{z})))]$$

- D(x) is the discriminator's estimate of the probability that real data instance x is real.
- E_x is the expected value over all real data instances.
- G(z) is the generator's output when given noise z.
- D(G(z)) is the discriminator's estimate of the probability that a fake instance is real.
- E_z is the expected value over all random inputs to the generator (in effect, the expected value over all generated fake instances G(z)) [4].

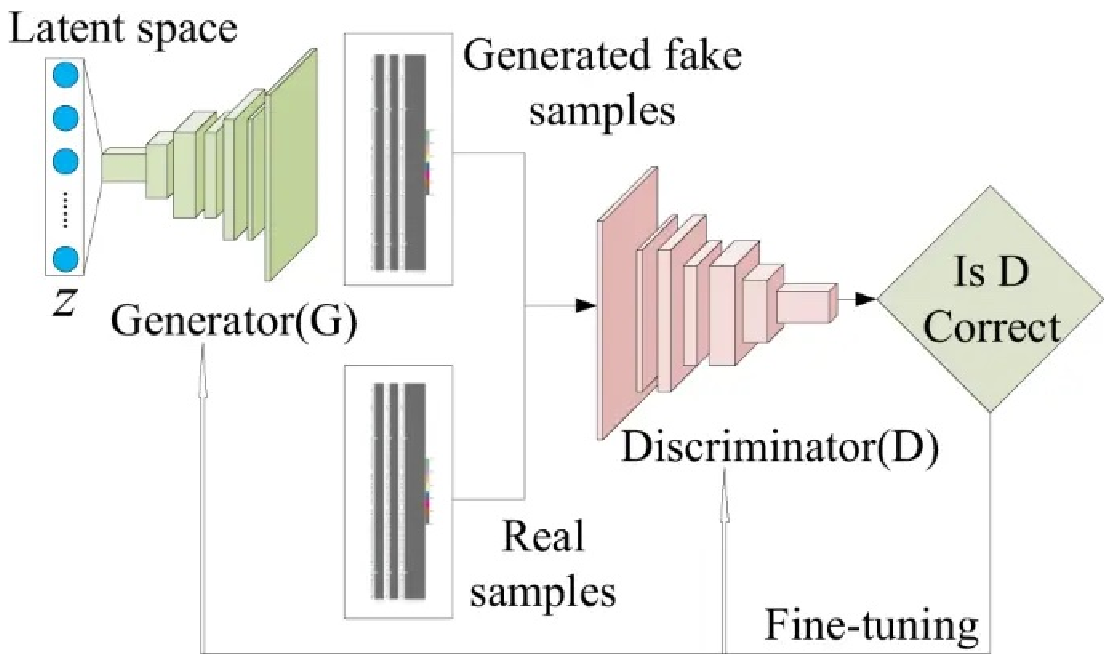

Fig. 1. Working of GAN [15]

The equation can be separated into two segments. The initial section is related to the discriminator. The purpose of the discriminator neural network is to receive an actual image as input and generate a scalar value $D(x)$, ranging from 0 to 1. This scalar value represents the likelihood of the input image x being a legitimate photograph of a face belonging to the dataset. A value of 1 indicates that the image is real, while a value of 0 indicates that the image is fake.

The network produces a score $D(G(z))$ for the fake images generated by the noise vector z that is fed into the generator $G(z)$. If the output is 1, it indicates that the generated image is real, and if it is 0, it indicates that the image is fake.

Our aim is twofold: we want the discriminator to correctly identify images from the dataset x as real with an output value close to 1, and we also want it to identify generated images $G(z)$ as fake with an output value close to 0 [5].

The generator's loss is not directly connected to the loss function, but it is connected to the discriminator which produces the output indicating whether the generated image is fake or real. If the discriminator outputs 0, meaning the image is fake, then the generator loss penalizes the generator for creating a sample that the discriminator identified as fake.

After the loss is calculated, the generator weights are updated through backpropagation via the discriminator network. This is crucial because the generator's parameters are highly dependent on the discriminator's parameters. The feedback received by the generator is used to create more realistic images. [6].

A. *Comparisions*

1) GANs and VAEs:

- GANs are known for producing visually more refined outcomes while learning natural image representation. However, VAEs are equally appealing as they provide both generative and inference models. To overcome this limitation, BiGANs were introduced, which include an inference model with GANs.
- The reason for the lack of sharpness in images generated by VAEs is often said to be due to the use of inference models during training that is not powerful enough to capture the actual posterior distribution. [12].
- Autoencoder networks are designed to reconstruct the input as the output. However, they are also regularized in such a way that the intermediate layers capture a meaningful representation of the input. This means that they are not simply an identity operation. In contrast, generative models are trained to generate new outputs from a latent space that closely resemble the input data [13].

2) GANs and Parallel Intelligence:

- GANs can be used to generate artificial-scene images, which can greatly expand the diversity of image datasets. By combining these generated images with real datasets, it is possible to train vision models with greater generalization ability.
- They can facilitate both predictive learnings for artificial systems and feedback learning for real systems.
- Combining GANs with parallel learning allows for the generation of large amounts of data and can be used for predictive learning through computational experiments [14].

III. Motivation

There is a lot of work being done to come up with new techniques in order to synthesize data more effectively, some of which we have seen in the Introduction section. One of the optimum solutions is using Generative Adversarial Networks (GANs) for generating synthetic data because they generate highly realistic variations of real-world data compared to other techniques. There have also been several successful attempts to make the GAN work more efficiently. But what Machine Learning models are compatible with the synthetically generated data? Can we distinguish how the ML models fit with real data and synthetic data? GANs are more researched with image data, but how do tabular data generated by GAN fit with ML models? Driven by these queries, we conducted our experiment to explore tabular synthetic data generated by GAN. Train and test machine learning models with synthetic data (generated by GAN) and then compare the models to find the most suitable one for our synthetic dataset. Compare how each of the ML models works with only real, only synthetic, and combination datasets.

IV. Related Works

Maayan et al. [16] introduce a technique for augmenting medical image data using GANs. They suggest a two-step training process in which classical data augmentation is first used to increase the size of the training set, followed by the application of GAN techniques to further increase the diversity and quantity of data through synthetic data augmentation. Xu, Lei, and Kalyan Veeramachaneni [17] propose a Tabular GAN (TGAN), which is a specific kind of GAN designed to generate tabular data such as educational or medical records. TGAN utilizes deep neural networks to create synthetic tables of high quality, and it can generate both continuous and discrete variables simultaneously. In the paper, 'Modeling tabular data using conditional gan' [18], Lei Xu et al. design CTGAN to handle Tabular data, which typically contains a mix of discrete and continuous variables. This can make modeling difficult due to challenges such as multiple modes in continuous variables and imbalanced distribution in discrete variables. CTGAN overcomes these challenges by using a conditional generator.

V. Dataset

We have used an airline passenger satisfaction dataset from Kaggle [7] which contains a survey on air passenger satisfaction. It is a tabular dataset with 103903 records consisting of 25 features. The dataset includes a mix of continuous features like Age of passenger, Flight distance, Departure delay in minutes, Arrival delay in minutes, and discrete features like Type of travel (personal or business travel), Class (business, economy, economy plus), Customer type (regular or non-regular airline customer). We aim to predict which of the two levels of satisfaction with the airline the passenger belongs to 'Satisfaction' / 'Neutral or dissatisfied'.

VI. Our Experiment

A. System Specification

We ran our code in Google Colab on Windows OS with 12GB RAM and 25 GB disk space. We used *Python* language and *keras* library.

B. Producing Synthetic Data

1) Preprocessing Data:

- Removing incomplete data: We removed the inconsistent records which contained missing values in any of the features.
- Choosing informative features
- One-hot encoding: We converted the categorical features to numerical values so that we have encoded values to train the GAN.
- Normalizing the data: Since each feature now had a different range of acceptable values, we normalized the data. So, all our values now lie between 0 and 1. Data normalization was performed by computing the infinite norm of each column (max value) using Python. We finally had 103594 records with 23 features each.

2) Training Discriminator: We trained the Discriminator on a real dataset using forward propagation without backpropagation for a certain number of epochs. The dataset provided did not contain any noise and only consisted of real images. The Discriminator used the instances generated by the Generator as negative output for the fake images. The following summarizes discriminator training:

- Classification of both real and fake data.
- The loss function used for the discriminator network helps to enhance its performance by penalizing it when it wrongly identifies real images as fake or fake images as real.
- The discriminator loss is used to update the weights of the discriminator.

We produced the fake data with 23 random features with normal distribution. Then we trained the discriminator using 103594 real data and 103594 fake data. It includes one dense hidden layer with 50 nodes, *tanh* activation function, and runs over 100 epochs.

3) Training Generator: To train the generator, we input random noise to generate fake outputs. During this process, the discriminator is not updated. However, when training the discriminator, the generator is not updated. The generator takes time to learn how to transform the noise input into meaningful data and requires multiple epochs to generate high-quality outputs. Training a generator involves the following internally:

- Produce a generator output by feeding a random noise sample to the generator.
- The discriminator predicts whether the output from the generator is real or fake.
- Calculate discriminator loss.
- Calculate gradients by performing backpropagation through both the discriminator and generator.
- Use these gradients to update generator weights.

We defined a generator with input layer 50 (latent space) and one hidden layer with 32 nodes and a linear activation function, *leaky relu*. The reason is that we want the values of that layer to cover the range of $-\infty$ to $+\infty$ so the output of the last layer can have all the values 0 to 1. The output of the generator is a value between 0 and 1 since we used a *sigmoid* function.

4) Generating Synthetic Data using GAN: We train the GAN by connecting the discriminator to the generator and locking the weights of the discriminator and setting the output of the GAN to 1 (real data). This is done for 1000 epochs. After training GAN we can use the trained generator layer to generate synthetic data. It takes a vector with 50 random values (latent space) and gives us synthetic data.
The generated samples from the Generator are fed into the Discriminator to determine whether they are real or fake, and the Discriminator provides feedback to the Generator. The Generator has then trained again based on the feedback from the Discriminator, and this process continues iteratively until the Generator produces samples that the Discriminator cannot distinguish from real ones.
We produced 10000 records of synthetic data. These synthetically generated data are continuous, so we used a self-defined function to discretize the produced data because most of the features in the real dataset are discrete. We denormalized the data by multiplying each feature by the max value of that column in the real data, rounded it, and normalized it again. Thus we obtained data just like the real data.

C. Training and Testing Machine Learning Models

1) Three types of train datasets:

- Real Data: We used 10000 records of the real data.
- Synthetic data: We used 10000 records of the synthetic data produced by GAN.
- Combination data: We combined the above to samples for a 20000 records sample.

2) Modelling: We trained the following ML models using the above 3 types of datasets.

- Support Vector Machine (SVM): The goal of the support vector machine (SVM) algorithm is to identify a hyperplane in an N-dimensional space (where N is the number of features) that can clearly separate the data points based on statistical methods [8].
  - SVM - RBF kernel: This technique generates a nonlinear function of the features, which transforms the data points to a higher-dimensional feature space. In this space, we can apply a linear decision boundary to separate the classes.
  - SVM - Linear kernel: When the data is perfectly linearly separable i.e.the data points can be classified into 2 classes by using a single straight line(if 2D). In such cases, we can use Linear SVM.
  - SVM - Sigmoid kernel: The algorithm takes input data and assigns them a value of either 0 or 1, making it possible to separate them with a straight line. [9].
- The k-nearest neighbors algorithm involves classifying an object by taking a majority vote from its neighboring objects, where k is the number of closest neighbors considered for the vote. The object is assigned to the class that is most frequent among its k-nearest neighbors. [10].

  - KNN – k=3
  - KNN – k=5
- Neural Networks consist of layers of nodes, including an input layer, one or more hidden layers, and an output layer. Each node, which is like an artificial neuron, is connected to another node and has a weight and a threshold. If the output of a node exceeds the threshold value, it becomes active and sends data to the next layer of the network. If not, no data is passed to the next layer. These networks depend on training data to learn and enhance their precision as time goes on [11].
  - Neural Network – 2 layers
  - Neural Network – 3 layers

We used 25893 completely new records as test data to calculate the accuracy of each model.

VII. Results

Table I presents the list of accuracies of the various ML models on which our experiment was conducted. This includes the 3 types of datasets used to train the models. Figure 2 shows the graphical comparisons of the same.

A. *Observations*

The classic ML Models, SVM and KNN do not see much improvement in the accuracy of the Real dataset and Combination dataset because these models are less dependent on the number of data (when compared with Neural Networks). Among the various kernels of SVM, RBF seems to suit better for just the synthetic data. Whereas, the Sigmoid kernel clearly improves the accuracy of the real dataset to the combination data. The variations of KNN models perform very similarly to each other. KNN seems to be less effective compared to the SVM model.

On the other hand, the more complex ML Model of neural networks has increased the accuracy of the model trained with real data compared to the combination data.

If we get the same accuracy for two models with the same structure, we prefer the one trained with more data since it can be more reliable. Thus this method probably can improve models with almost the same accuracies.

VIII. Drawbacks and Future Works

The process is computationally intensive. Training GAN models is time-consuming. Activation functions should be picked wisely since we want some specific range for the synthetic data. Setting the parameters of the GAN networks and the ML models can be depended on the dataset. Thus expanding our conclusion needs more investigation.

We have worked on a single dataset. This same approach can be applied to various other tabular datasets and models can be tested for their accuracies, providing a better understanding of how the ML models work with synthetic data. Using stronger systems with better configurations and specifications, we can train the models with more epochs and different parameters, providing more robust conclusions.

IX. Conclusion

We have synthesized data to match a real-world dataset using GAN. We have identified ML models that have the potential to work well with GAN-generated data. There is no one-fit-all ML model. but experimenting with more datasets will help to derive more robust conclusions. With a few changes and enhanced system specifications, our approach has the potential to be extended to other tabular data containing both discrete and continuous data Ex: Spotify songs dataset, Educational dataset, or any similar tabular data.

References

[1] Kim Martineau, “What is synthetic data?”, 2023. [Online]. Available at: https://research.ibm.com/blog/what-is-synthetic-data. [Accessed Apr. 19, 2023].

[2] Turing, “Synthetic Data Generation: Definition, Types, Techniques, and Tools”, 2023. [Online]. Available at: https://www.turing.com/kb/synthetic-data-generation-techniques#types-of-synthetic-data. [Accessed Apr. 19, 2023].

[3] Datagen, “What is Synthetic Data Generation?”, 2023. [Online]. Available at: https://datagen.tech/guides/synthetic-data/synthetic-data-generation/. [Accessed Apr. 20, 2023].

[4] Raghav Agrawal, “An End-to-End Introduction to Generative Adversarial Networks(GANs)”, 2021. [Online]. Available at: https://www.analyticsvidhya.com/blog/2021/10/an-end-to-end-introduction-to-generative-adversarial-networksgans/. [Accessed Apr. 20, 2023].

[5] datahacker.rs, “GANs – Face editing with Generative Adversarial Networks”, 2022. [Online]. Available at: https://datahacker.rs/005-gans-face-editing-with-generative-adversarial-networks/. [Accessed Apr. 20, 2023].

[6] Nilesh Barla, “Generative Adversarial Networks and Some of GAN Applications: Everything You Need to Know”, 2023. [Online]. Available at: https://neptune.ai/blog/generative-adversarial-networks-gan-applications. [Accessed Apr. 20, 2023].

[7] TJ KLEIN, “Airline Passenger Satisfaction”, 2020. [Online]. Available at: https://www.kaggle.com/datasets/teejmahal20/airline-passenger-satisfaction. [Accessed Apr. 20, 2023].

[8] Rohith Gandhi, “Support Vector Machine — Introduction to Machine Learning Algorithms”, 2018. [Online]. Available at: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47. Accessed Apr. 20, 2023.

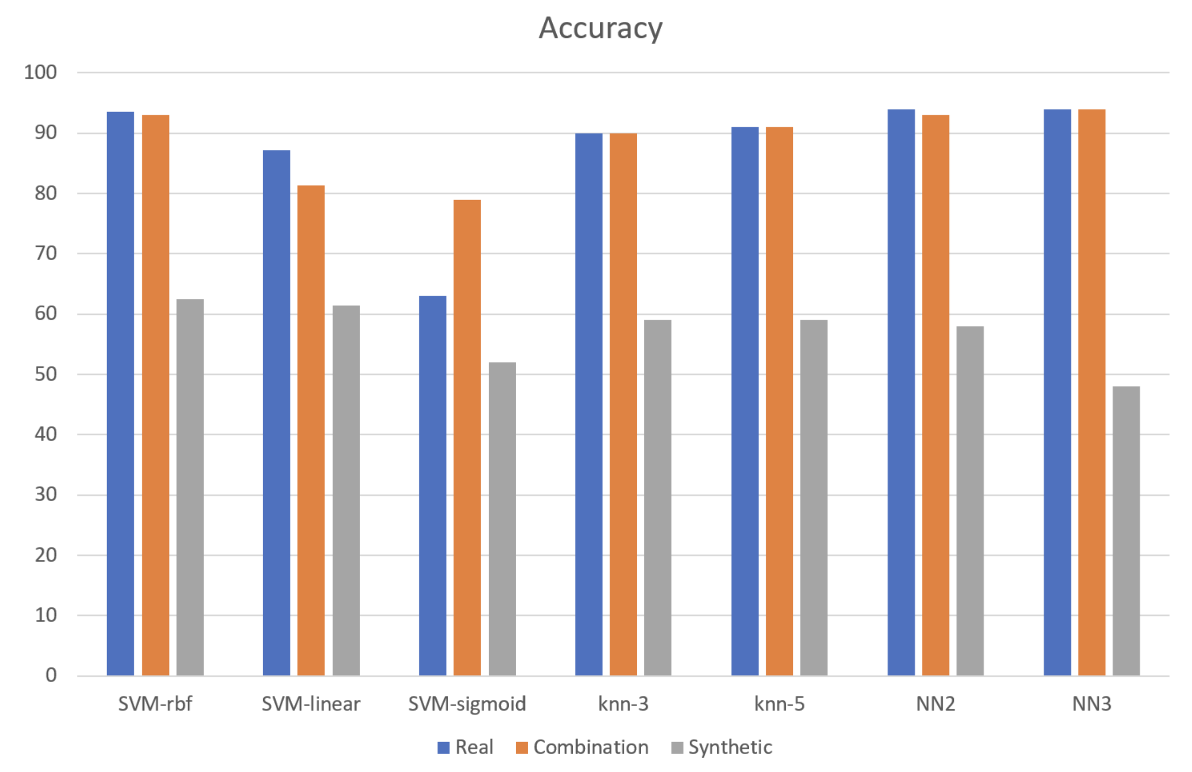

Fig. 2. Accuracies(in percentage) of ML Models on 3 types of data

TABLE I
ML MODEL ACCURACIES

Model with variation	Real Data (10000 Real)	Combination data (10000 Real + 10000 Syn)	Synthetic data (10000 Syn)
SVM - RBF	93.53%	93.01%	62.49%
SVM - Linear	87.20%	83.27%	61.48%
SVM - Sigmoid	34.72%	79.74%	52.06%
KNN – k=3	90.88%	90.87%	59.39%
KNN – k=5	91.17%	91.14%	59.39%
Neural Network – 2 layers	94.06%	93%	58.35%
Neural Network – 3 layers	94.39%	4.51%	48.67%

[9] Anshul Saini, “Support Vector Machine(SVM): A Complete guide for beginners”, 2021. [Online]. Available at: https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/. Accessed Apr. 20, 2023.

[10] Wikipedia, “k-nearest neighbors algorithm”, 2023. [Online]. Available at: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm. Accessed Apr. 20, 2023.

[11] IBM, “What is a neural network?”, 2023. [Online]. Available at: https://www.ibm.com/topics/neural-networks. Accessed Apr. 20, 2023.

[12] Mescheder, Lars, Sebastian Nowozin, and Andreas Geiger. ”Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks.” International conference on machine learning. PMLR, 2017.

[13] Feigin, Yuri, Hedva Spitzer, and Raja Giryes. ”Cluster with GANs.” Computer Vision and Image Understanding 225 (2022): 103571.

[14] Wang, Kunfeng, et al. ”Generative adversarial networks: introduction and outlook.” IEEE/CAA Journal of Automatica Sinica 4.4 (2017): 588-598.

[15] Ritika, “What’s GAN (generative adversarial networks), how it works?”, 2022. [Online]. Available at: https://www.labellerr.com/blog/what-is-gan-how-does-it-work/.

Accessed Apr. 20, 2023.

[16] Frid-Adar, Maayan, et al. ”Synthetic data augmentation using GAN for improved liver lesion classification.” 2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018). IEEE, 2018.

[17] Xu, Lei, and Kalyan Veeramachaneni. ”Synthesizing tabular data using generative adversarial networks.” arXiv preprint arXiv:1811.11264 (2018).

[18] Xu, Lei, et al. ”Modeling tabular data using conditional gan.” Advances in Neural Information Processing Systems 32 (2019).