



University of Essex

**School of Computer Science and Electronic Engineering**

**“Project Report”**

**Generating Synthetic Data in the context of Health Data, and comparing it  
to Real World Data using Artificial Intelligence Systems**

**CE903 Group Project**

**Supervised by: Dr Ana Matran-Fernandez**

**Submitted by:**

Team 15 Group Members	
Samuel Jularbal	2101224
Ahmed Ali	2201163
Priteshkumar Thakkar	2201235
Isreal Ufumaka	2204569
Abdul Wahid	2202396
Shikhar Sharma	2205456
Yuan Zan	2200557
Dharsigan Bharathidasan	2205532
Adedeji Adetunji	2201823
Ashu Berwal	2204914

# ABSTRACT

The aim of our project is to generate synthetic data using health datasets and compare it with real data. The Generative Adversarial Network (GAN) model is used for synthetic data generation. This model has two neural networks i.e., the generator network learns to generate synthetic data samples from the real data samples and the discriminator network learns to distinguish between the real and synthetic samples. With our trained model, we can apply any tabular data to generate its equivalent synthetic data. After data generation, we use various testing methods such as visualizing and comparing the real and synthetic generated data and apply machine learning classification models to test the samples. The results produced promising synthetic data but are still distinguishable from real data. Synthetic data helps in anonymizing the health dataset, ensuring data privacy, and avoids data duplication.

## **Key words:**

Synthetic Data Generation, Health data, General Adversarial Networks (GANS), Tabular Data, Data Privacy

# TABLE OF CONTENT

ABSTRACT.....	I
LIST OF FIGURES .....	IV
1. INTRODUCTION .....	1
1.1 Purpose of this Project.....	1
1.2 Scope of the Project.....	1
2. LITERATURE REVIEW .....	3
2.1 Synthetic and Private Smart Health Care Data Generation using GANs.....	3
2.2 A Synthetic Data Generation System for Healthcare Applications .....	4
2.3 Survey on Synthetic Data Generation, Evaluation Methods and GANs.....	5
2.4 Medical Image Synthesis for Data Augmentation and Anonymization Using Generative Adversarial Networks.....	6
3. METHODOLOGY .....	7
3.1 Data Analysis .....	7
3.1.1 Data Collection .....	7
3.1.2 Data preprocessing .....	8
4. MODEL DESIGN.....	9
4.1 Challenges with GANs, Pre-processing & Post-Processing .....	9
4.2 GAN Model Architecture.....	9
4.2.1 Generator .....	9
4.2.2 Discriminator .....	10
4.2.3 Model Architecture.....	10
5. IMPLEMENTATION.....	11
5.1 Libraries Used .....	11
5.1.1 Data Pre-processing .....	11
5.1.2 Data Visualization .....	11
5.1.3 Deep Model Learning.....	11

5.2 Generating Synthetic Data .....	11
5.3 Graphical User Interface (GUI).....	12
5.3.1 Main Window .....	12
5.3.2 Model Training Window .....	12
5.3.3 Data Generation .....	13
6. EVALUATION.....	14
6.1 Challenges of Evaluating Synthetic Tabular Data .....	14
6.2 Visualizing Synthetic Samples.....	14
6.3 Testing Synthetic Samples .....	16
6.4 Overall Evaluation.....	18
7 PROJECT MANAGEMENT .....	20
7.1 Project Management structure and methodologies .....	20
7.2 Project planning.....	21
7.3 Gantt Chart Representation .....	22
7.4 Tools for project management: .....	23
7.4.1 Jira: .....	23
7.4.2 Git lab: .....	24
7.4.3 Slack: .....	24
7.4.4 Zoom:.....	25
9. CONCLUSION.....	26
REFERENCES .....	27

# LIST OF FIGURES

Figure 1: Graphical User Interface Main Window .....	12
Figure 2: Training Window .....	13
Figure 3: Sample Generation Window .....	13
Figure 4: The Absolute Log Mean and STDs of the Real and Synthetic Dataset .....	14
Figure 5: The Distribution of Real vs Synthetic Data.....	15
Figure 6: Histogram of Real vs Synthetic Data .....	16
Figure 7: Testing Metrics of Synthetic Data using Classification ALgorithms.....	16
Figure 8: Confusion Matrix of Classification Results .....	17
Figure 9: The Wasserstein Distance and Jensen-Shannon Divergence metrics of the real and synthetic datasets .....	18
Figure 10: Project Structure .....	21
Figure 11: Gantt Chart Representation .....	22
Figure 12: Jira Software Backlog .....	23
Figure 13: Git Lab Work Directory .....	24
Figure 14: Slack Work Channels .....	25

# **1. INTRODUCTION**

Artificial intelligence (AI) has been used increasingly in recent years, and it has enormous potential in a variety of industries, including healthcare. AI can completely change how diagnoses are made, treatments are delivered, and even how healthcare services are offered in the future. To train and validate AI models, however, significant amounts of high-quality data are needed to accomplish these objectives. Synthetic data generation provides a practical and effective solution to various challenges faced in artificial intelligence for health, such as data scarcity, data privacy concerns, and bias in the training data. To fulfil some specific requirements or circumstances that might not be present in the actual data, synthetic data is produced. Because synthetic data are used as an alternate replica of real datasets, they can be helpful when creating any kind of real-world system without using real data. In many different areas, synthetic data is used as a filter to remove information that might otherwise risk the secrecy of some confidential data points. Datasets for many sensitive uses exist but are not accessible to the public, using synthetic data avoids these privacy concerns by using actual customer information without authorization or compensation [1].

## **1.1 Purpose of this Project**

This project's purpose is to develop and implement a General Adversarial Network (GAN) model to produce synthetic samples to address the challenges associated with training and validating machine learning models in healthcare. Such concerns this model aims to address are privacy, biases in training and small datasets. By creating synthetic data that can accurately replicate the features of real-world data, this project can help facilitate the development of AI-solutions in health care.

## **1.2 Scope of the Project**

The scope of generating synthetic data for health-related purposes includes creating a new dataset from existing medical datasets to address issues related to scarce and confidential data. The synthetic data generator aims to supplement existing datasets and create a new dataset that closely follows the statistical characteristics of the original dataset.

The GAN will use a generator to introduce enriched data features and variability, leading to the production of multiple representative data sets that can be used to train and validate AI models. Synthetic data can mitigate data privacy concerns related to sensitive personal information contained in medical records and address issues related to data imbalance and varying feature distributions that can introduce bias.

The scope of this project also includes analyzing the original dataset to understand its underlying structure, including missing values, data types, variable distribution, class imbalance, correlations, and trends. The synthetic dataset will be evaluated using visualization techniques, statistical evaluation methods and machine learning assessment.

## 2. LITERATURE REVIEW

### 2.1 Synthetic and Private Smart Health Care Data Generation using GANs

This research paper discusses the potential benefits of using machine learning algorithms in IoT-based applications such as smart healthcare, smart cities, smart farming, and personalized medicine. However, the main challenge is accessing large amounts of sensitive and private data, which local data protection laws can limit. Realistic synthetic datasets can address this challenge by providing enhanced user privacy and reducing the risk of exposure due to privacy-breaching attacks on ML models. Generative modeling, particularly GANs, is commonly employed for synthetic data generation and can be combined with privacy preservation solutions like differential privacy. The article focuses on the use of Boundary-Seeking GANs (BGANs) for generating synthetic smart healthcare data samples with user privacy guarantees [3]. GANs rely on training a discriminator to determine a measure of the difference between a goal distribution and generated distributions. GANs do not work for discrete data because, in their typical formulation, they depend on the produced samples being fully differentiable with respect to the generative parameters. We present a technique for training GANs with discrete data that computes importance weights for produced samples using the estimated difference measure from the discriminator, giving a policy gradient for training the generator. We refer to our approach as boundary-seeking GANs because the importance weights strongly relate to the discriminator's judgement limit. The healthcare data generated by BGAN also has 3v's (volume, velocity and variety) of big data. In addition to BGANs, they also used WGANs in our original collection of experiments. According to our research, BGANs are better suited for creating fake smart healthcare datasets because they agree more quickly and produce data of higher quality. Furthermore, our suggested method offers extra privacy protection by incorporating Differential Privacy in various contexts. These findings demonstrate that the suggested method can produce authentic examples of smart health care data while guaranteeing user anonymity.

The overall summary of this paper is a real-world smart health care dataset from consumers who are spread out regionally are gathered. Smart health care information reflects various dietary and exercise trends based on age, race, region, dietary tastes, and other variables. Suggested a technique based on GANs for creating artificial categorical and numerical time-series data in tabular form, along with techniques for creating privacy-preserving forms of the



synthesized data examples. For open use in study, they have developed realistic, privacy-preserving smart health care datasets with fine-grained dietary and exercise user profiles [3].

## **2.2 A Synthetic Data Generation System for Healthcare Applications**

This paper presents a method for generating synthetic sensor data that reflects human behavior found in real sensor datasets, which can be used to address the problem of limited availability and variety of real-world sensor-driven datasets. The approach employs hidden Markov models (HMMs) to model smart home data's sequential nature and sequence-generative nature. Real smart home datasets are used to train the model and generate synthetic data, and data similarity measures are used to validate the realism of the generated data. The paper also shows how synthetic data generation can be applied to the problem of generating synthetic data for semi-supervised activity recognition to improve algorithm accuracy when only a small amount of real annotated data is available. The main contribution of the paper is to provide synthetic data that preserves the underlying patterns and structures of real data and can be used to improve machine learning-based techniques by augmenting small, labelled training datasets. They compared one week of randomly chosen real data with one week of synthetically generated data for the same home and same time period using two common measures of distance for time series data, Euclidean distance and Dynamic Time Warping (DTW). This is done for each of the 10 smart home datasets and the average distance results are reported. Also compute the distances between one week of real data and one week of real data from another home within the same time period as well as the distances between one week of real data and week data of real data from the same home within another time of year.

To further examine the realism of the data, the authors use SynSys combined with semi-supervised activity recognition to augment a real smart home dataset. They hypothesize that training an activity recognition algorithm with synthetically augmented data will boost performance of the recognition algorithm. To perform semi-supervised learning, they first train the activity recognition algorithm on subsets of data extracted from one month of ground truth-annotated sensor-based activity data for 10 smart home sites. They then modify the algorithm to be semi-supervised using a self-training approach. The amount of synthetic data is constant in each case and corresponds to one month of generated data. The final 200,000 sensor events, approximately three weeks of real data, are held out for testing. They report the accuracy averaged over the 10 smart home sites [5].

## 2.3 Survey on Synthetic Data Generation, Evaluation Methods and GANs

This research paper presents a survey of synthetic data generation algorithms, with a focus on GANs for tabular data, and the evaluation of synthetic data quality. They highlighted the importance of generating high-quality synthetic data that mimics the underlying data distribution of real datasets, while also preserving privacy. Overview of GANs, their training, and the different architectures proposed in this paper. The main methods for synthetic data generation, including both traditional and deep learning methods are discussed.

The synthetic data generator creates a valid collection of instance models using a customized OCL constraint solver. The initial collection of instance models is potentially invalid, but the solver attempts to repair them based on the constraints specified in the overall approach and the underlying data schema. The solver is fed with the instance models from the initial collection, rather than building them from scratch, to increase the likelihood of reaching valid instance models. If the solver cannot fix an instance model within a predefined number of iterations, the instance model is discarded, and the data generator re-invokes to extend the initial collection. After obtaining the desired number of valid instance models, the data generator attempts to realign the sample with the desired statistical characteristics using an iterative process that subjects the instance models to corrective constraints. These corrective constraints provide cues to the solver on how to tweak an instance model to improve the statistical representativeness of the whole data sample. The tweaked instance model is compared with the original instance model, and if it satisfies all the validity constraints and improves overall representativeness, it replaces the original instance model. The iterative process may generate multiple corrective constraints, and they are treated as being soft. Additionally, the study intends to focus on unbalanced datasets and use machine-learning models to assess their performance in the minority class when synthetic samples are incorporated into their training [6].

## **2.4 Medical Image Synthesis for Data Augmentation and Anonymization Using Generative Adversarial Networks**

In [7], considering data diversity as a critical success in the training process of artificial intelligence models, they acknowledge the different shortcomings of datasets in health care such as imbalance cases for rare diseases. They propose a model to generate synthetic MRI images using GANs. Their work demonstrates firstly improved performance and secondly, the use of GANs in data anonymity. They concluded that this could help solve the issue of small available datasets and data restrictions in sharing health data.

### **3. METHODOLOGY**

For this project we will be doing the following steps to produce a model that can generate synthetic samples from a real dataset:

- 1) Data pre-processing: Pre-process the real-world health data to make it suitable for synthetic data generation. This includes removing identifying information such as names and addresses and any columns which expose privacy.
- 2) GAN architecture: Choose a suitable GAN architecture that is designed for health data. This may include modifications to the generator and discriminator to ensure that the synthetic data is both realistic and protects patient privacy.
- 3) Training: Trained the GAN model using real-world health data. During training, the generator learns to generate synthetic health data, while the discriminator learns to distinguish between the synthetic and real-world data.
- 4) Generate synthetic data: Once the GAN model is trained, use the generator to generate synthetic health data. The synthetic data should be evaluated to ensure that it is both realistic and protects patient privacy.
- 5) Evaluate the synthetic data: Evaluate the synthetic data to ensure that it has similar statistical properties as the real-world data. We use techniques such as data visualization, distribution comparisons and machine learning assessment.
- 6) Comparison: Compare the results of the AI system analysis on synthetic and real-world health data. The comparison should assess the performance of the AI system on both datasets and identify any differences or limitations in the synthetic data.

#### **3.1 Data Analysis**

##### **3.1.1 Data Collection**

The information was gathered from the Agency for Healthcare Research and Quality's (AHRQ) Healthcare Cost and Utilization Project (HCUP), a compilation of healthcare databases. (AHRQ). The information on inpatient hospital stays in member states was taken specifically from the State Inpatient Databases (SID).

Over 100,000 hospital admissions for diabetic patients between 1999 and 2008 are detailed in the collection. Demographic data, clinical variables, and hospital-specific statistics are all

variables in the dataset. The dataset's goal is to make it easier to conduct studies on hospitalizations and other outcomes connected to diabetes in the US [2].

The following are a few of the precise factors in the dataset:

- Information about a patient's age, gender, ethnicity, and weight status
- Information about the admission type, admission source and discharge has been provided
- Clinical information about the patients that which medical specialty the patient has been treated
- Many diseases have been verified finally the outcome is to find whether the patient is diabetic or not

The collection is frequently used in studies on outcomes and hospitalizations linked to diabetes in the US. The dataset, for instance, may be used by researchers to examine risk factors for hospital readmissions among diabetic patients or to examine variations in diabetes-related results between hospitals or geographical areas.

### **3.1.2 Data preprocessing**

As we want the GAN model to work with real datasets and show generalizability of the model, we will not preprocess the diabetes dataset used to train it. For data preprocessing, we will be adding a preprocessing step to our GAN model. This will allow the model to take in any unprocessed data fed into the model, which will be further explored in this paper's Model Architecture section. The model will clean the data as best as it can by removing null values, replacing missing string values, and encoding categorical values. This will allow the model to generalize and train on all data inputs by converting all object type data to processable training information. The training data will then be scaled before training the GAN model.

## 4. MODEL DESIGN

### 4.1 Challenges with GANs, Pre-processing & Post-Processing

The issues with GANs and their use on real-world data are abundant. GANs in their base form struggle to take mixed-type data variables and then simultaneously generate a mix of discrete and continuous outputs [9]. To alleviate this problem with GANs we have added a preprocessing step that can be applied to any form of unprocessed tabular datasets.

The preprocessor reads the input and selects whether the feature is discrete or continuous, treating all discrete inputs as one-hot vectors, whilst all continuous data is taken without further engineering. For string types with missing values, such as “?”, those values are replaced with `np.nan`, then all nan values are converted to 0. All unique values are replaced with categorical values with unique integers from 0 to the number of unique categorical values minus 1.

After the model has produced a set of synthetic samples, the samples are then post-processed to remove scaling and normalize to the synthetic data. A ‘process’ method scales the data back to its original range then converts the synthetic data to a pandas data frame and rounds the data to the nearest integer. For each column containing categorical data, the code replaces the unique integer values used during training with their corresponding original categorical values using the loaded feature information.

### 4.2 GAN Model Architecture

#### 4.2.1 Generator

The generator network,  $G$ , takes a random noise vector  $z$  as input and takes a generated sample  $\hat{x}$  as output, which is an estimate of a real data sample  $x$  [10]. The generator can be represented as  $G(z) = \hat{x}$ .

Using the keras library, a LeakyReLU and Batch Normalization layer is applied to the generator. The Generator then finally has a sigmoid activation function, where finally a random variable is selected from the based on the input data to begin the production of synthetic data.

### 4.2.2 Discriminator

The discriminator network,  $D$ , takes an input sample of  $x$  and outputs the scalar value  $D(x)$ , which is a probability representation of the authentic sample [10].

The discriminator of the model uses a Binary Cross Entropy loss function, with an Adam optimizer. Similar to the Generator, a LeakyRelu activation function is applied to the input, with a sigmoid activation function following afterwards.

### 4.2.3 Model Architecture

The model will then be trained through the generator and discriminators iteratively, with each iteration more realistic samples are output by minimizing the distribution of the generated samples to the real samples. The discriminator is trained to distinguish between the real and fake samples by maximizing the difference in probability of the real and fake samples.

Summary of the GANs architecture:

- $z$ : Random noise vector, sampled from a simple distribution (such as uniform or normal).
- $x$ : Real data sample, drawn from a complex distribution (the dataset)
- $\hat{x}$ : Generated sample, output of the generator network  $G(z)$ .
- $D(x)$ : Discriminator network output for a real sample  $x$ .
- $D(\hat{x})$ : Discriminator network output for a fake sample  $\hat{x}$ .
- $G(z)$ : Generator network output for a noise vector  $z$ .
- $\theta_D$ : Parameters of the discriminator network.
- $\theta_G$ : Parameters of the generator network.

With the network now fully connected, the input data is fed through a min-max scaler before performing the adversarial process. The model's output then follows a post-processing step, removing the previous min-max scaling to produce distinct discrete and continuous outputs and generating them in the correct columns.

## **5. IMPLEMENTATION**

### **5.1 Libraries Used**

The following libraries were used in the implementation of this project:

#### **5.1.1 Data Pre-processing**

- Pandas (version 1.5.3): A data manipulation library used for data cleaning, merging and reshaping.
- NumPy (version 1.24.2): A numerical computing library used for array manipulation and mathematical operations
- Scikit-learn (version 1.0): A machine learning library used for data pre-processing, feature selection, and model building

#### **5.1.2 Data Visualization**

- Matplotlib (version 3.7.1): A data visualization library used for creating static, interactive, and animated plots
- Table Evaluator (version 1.5.0): A table visualiser designed for evaluating real and authentic generated samples

#### **5.1.3 Deep Model Learning**

- TensorFlow (version 2.11): A machine learning library used for building and training deep learning models
- Keras (version 2.11): A high-level neural networks API used for building and training deep learning models

### **5.2 Generating Synthetic Data**

After designing the model (named GAN15 after the name of our group) we applied an unprocessed diabetes dataset to train GAN15 and produce synthetic samples. The GAN model then applies a preprocessing function to the training data. For training we take the entire dataset (101766 rows and 50 columns), dropping identification columns as they will not help with training the model.



After the generator and discriminators have been trained on the dataset, we can then call the training function applying the hyperparameters to the network. For the GAN15 model we found the following hyperparameters to be best for producing synthetic samples.

- Epochs: The model is trained using 4000 epochs
- Learning Rate: The model uses a learning rate of  $1e-3$
- Batch Size: The model is trained using a batch size of 32

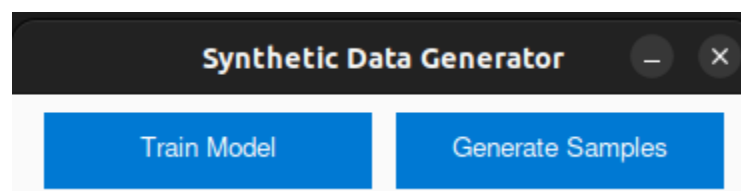
Following the training of the GAN15 network, a sampling function allows the user to select how many synthetic samples the network should produce. For testing we produce 10,000 synthetic samples to evaluate.

## 5.3 Graphical User Interface (GUI)

To make things simple to understand, we have made a simple Graphical User Interface (GUI). The GUI uses the PySimpleGUI library to create windows and elements and interacts with other Python modules to train a generative model and generate synthetic data.

### 5.3.1 Main Window

The GUI is composed of two windows: the main window and the train/generate windows. The main window has two buttons: "Train Model" and "Generate Samples". Clicking the "Train Model" button opens a new window where the user can input the parameters for training the model. The same applies to the "Generate Samples" button, which opens a window for generating new data.



*Figure 1: Graphical User Interface Main Window*

### 5.3.2 Model Training Window

When the user clicks the "Train Model" button, the GUI opens a new window where the user can input the parameters for the training process. The parameters include the input file,

number of epochs, learning rate, batch size, and whether or not to save the model. If the user chooses to save the model, they must provide a name for it. Once the user clicks "Start Training," the model is trained using the provided parameters. The loss curves and a visual evaluation of the generated data using the table evaluator module are displayed after the model has finished training.

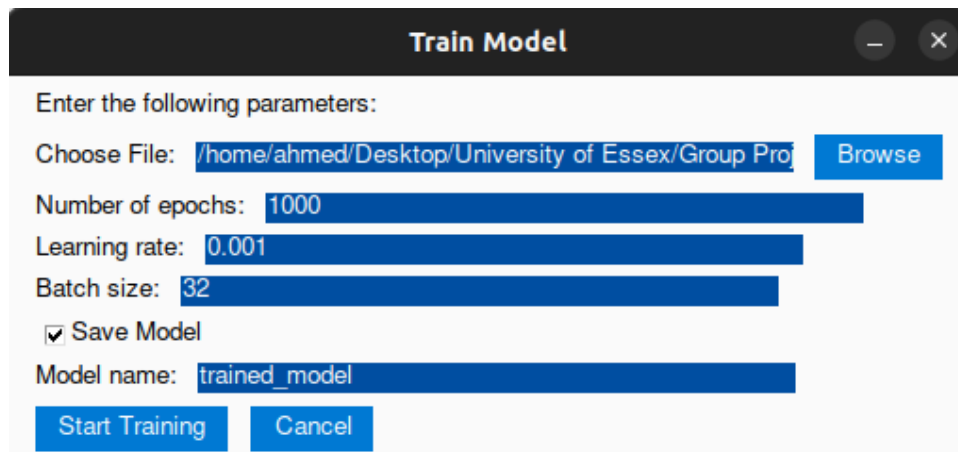


Figure 2: Training Window

### 5.3.3 Data Generation

When the user clicks the "Generate Samples" button, the GUI opens a new window where the user can provide the trained model path, the number of samples to generate, and the name of the output file. Once the user clicks "Generate Samples," the script generates the new data using the trained model and displays it in a popup window.

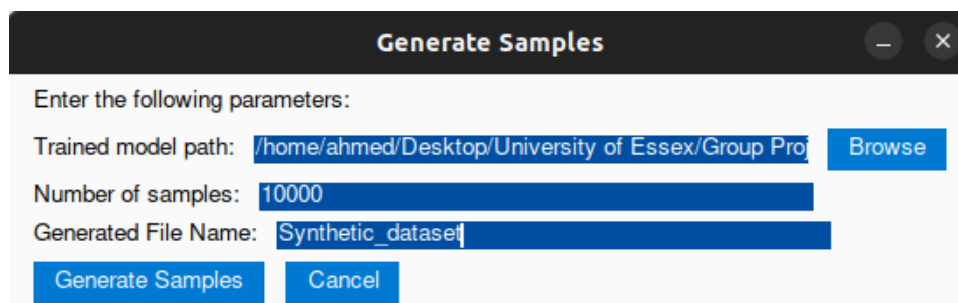


Figure 3: Sample Generation Window

## 6. EVALUATION

### 6.1 Challenges of Evaluating Synthetic Tabular Data

Unlike image data, the evaluation of synthetically generated tabular can be challenging [11]. The common testing metrics can be with the help of insights of experts or using likelihood fitness metrics, such as Jensen-Shannon divergence and Wasserstein Distance [11]. However, the application of these metrics is not standardized across research, making comparisons difficult for tabular synthetic data. Traditional machine learning scoring (like accuracy and F1-score) is not able to support the training of deep learning models [11].

To test the synthetic tabular data of the GAN15 model, we impose different internal testing methods. We perform a case of visual inspection, statistical analysis, and model-based assessments to review the performance of our GAN in producing synthetic samples.

### 6.2 Visualizing Synthetic Samples

The first method of testing the synthetic samples is by comparing them with the real samples and visualizing the differences between the two samples. Using the generated synthetic samples, we use the Table Evaluator library to visualize the distributions of the two data samples.

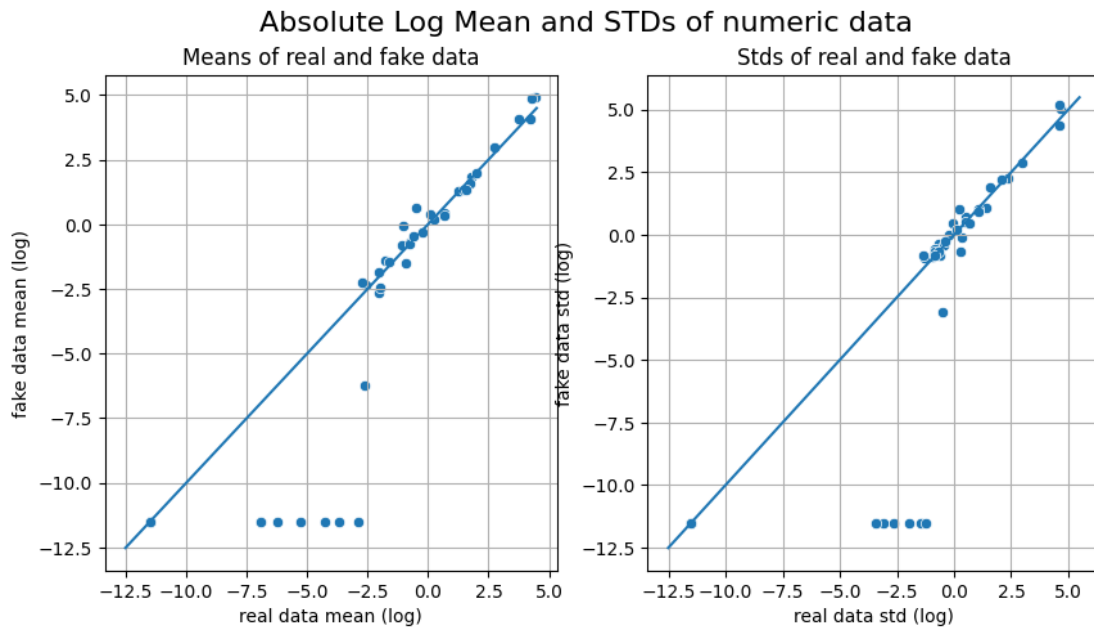


Figure 4: The Absolute Log Mean and STDs of the Real and Synthetic Dataset

We can visualize statistical insight of the data by viewing the similarity between the Absolute log Mean (Figure 3). With the insights into the Absolute Log Mean and Standard Deviation (STD) we can see the average magnitude of the differences between the two datasets. A smaller absolute log mean indicates that the synthetic data closely resembles the real data, while a larger absolute log mean signifies that the synthetic data deviates from the real data. Figure 4 shows that the Absolute log mean and the standard deviation of the real data against synthetic data resemble similar Means and STDs. We can infer that the data from the synthetic samples closely resemble that of the real data based on the averages of magnitude between the two datasets.

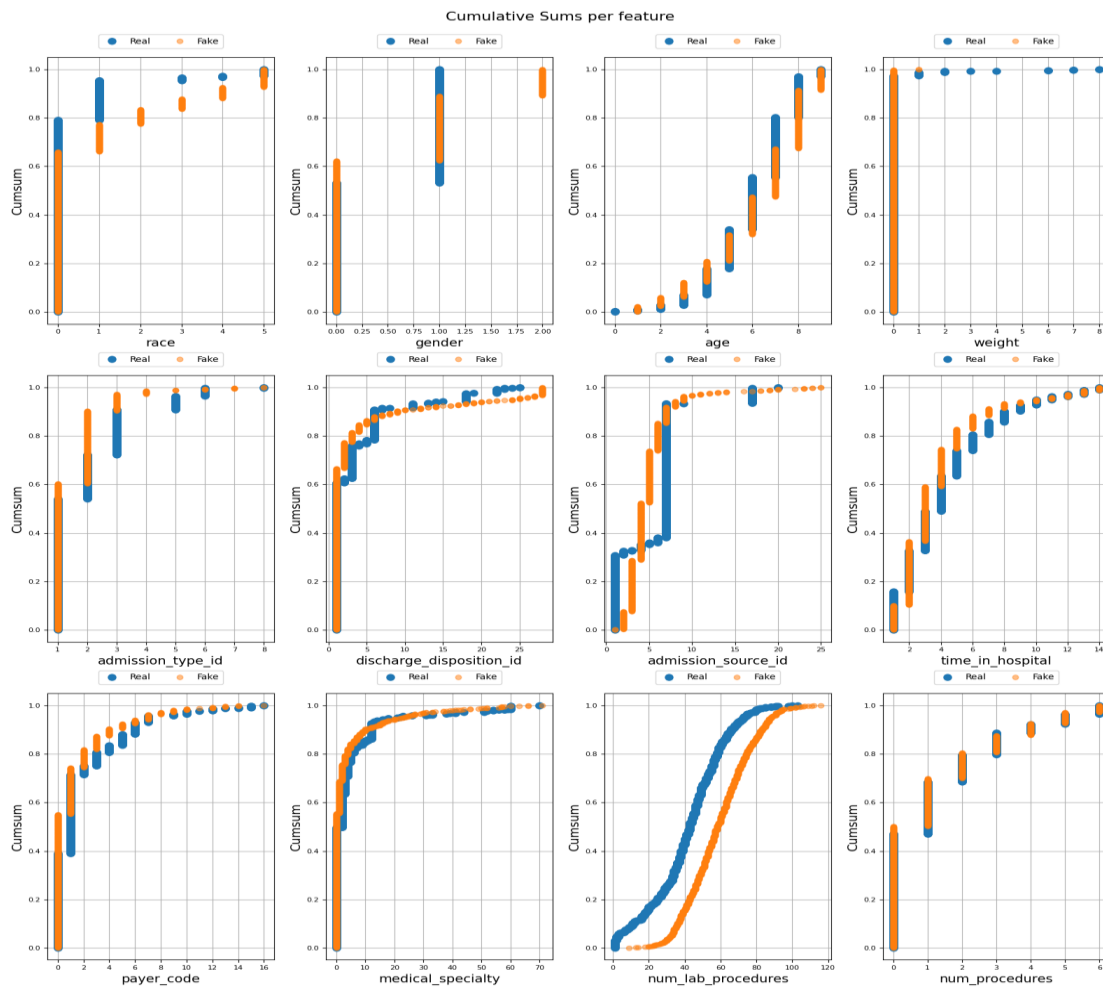


Figure 5: The Distribution of Real vs Synthetic Data

Taking just the first 12 features from the two datasets, we can see a side-by-side comparison in Figure 1 of the distributions of both. The results of the GAN15 model show promising outputs, in that the generated synthetic data does hold its own in both categorical and continuous columns. However, it must be noted that the synthetic samples have failed to interpret all the

data accurately. For instance, in the gender feature there are only two genders from the real dataset, but the GAN15 model has interpreted a third class for the feature. However, for continuous data and other categorical interpretations, the model has performed well in interpreting the distributions of the real data set and producing accurate samples.

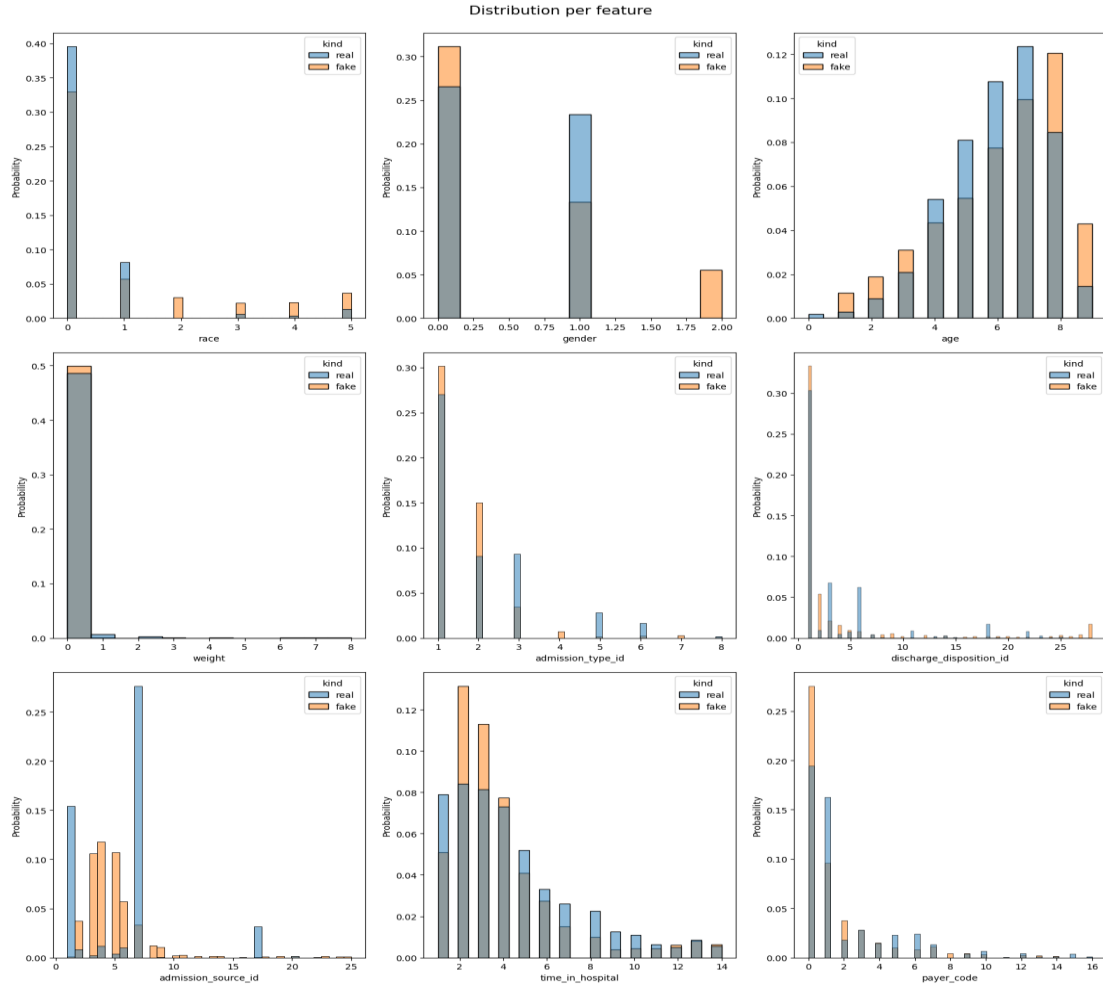


Figure 6: Histogram of Real vs Synthetic Data

## 6.3 Testing Synthetic Samples

Model	Accuracy	Precision	Recall	F1-Score	Support
Pca_svc	0.82	0.82	0.82	0.82	4000
K-Nearest Neighbour	0.78	0.78	0.78	0.78	4000
Decision Tree	0.98	0.98	0.98	0.98	4000
Logistic Regression	0.83	0.83	0.83	0.83	4000
Random Forest	1.00	1.00	1.00	1.00	4000
Naïve Bayes	0.66	0.75	0.65	0.62	4000

Figure 7: Testing Metrics of Synthetic Data using Classification Algorithms.

The method of testing the GAN15 model is through analyzing the results of the datasets being used in a machine learning model process. A merged processed dataset of both the GAN15 samples and real samples was compiled together to generate a dataset with the size of 20000 rows and 49 features. The final additional feature is the target label to determine between the real and synthetic data – the label ‘1’ marks real data, whereas the label ‘0’ marks synthetic data.

A multitude of machine learning classification models were picked to test whether it could determine the difference between the real and synthetic samples. The models that were chosen were: PCA SVC, K-Nearest Neighbour, Decision Tree, Logistic Regression, Random Forest and Naïve Bayes. Table 1 shows a compilation of all the scoring metrics from the resulting training and testing. The results of the classification models are that they were all able to successfully classify between the real and fake samples, showing that the synthetic samples had not been able to completely replicate the real dataset. The high accuracy suggests the machine learning models clearly told between the real and synthetic data.

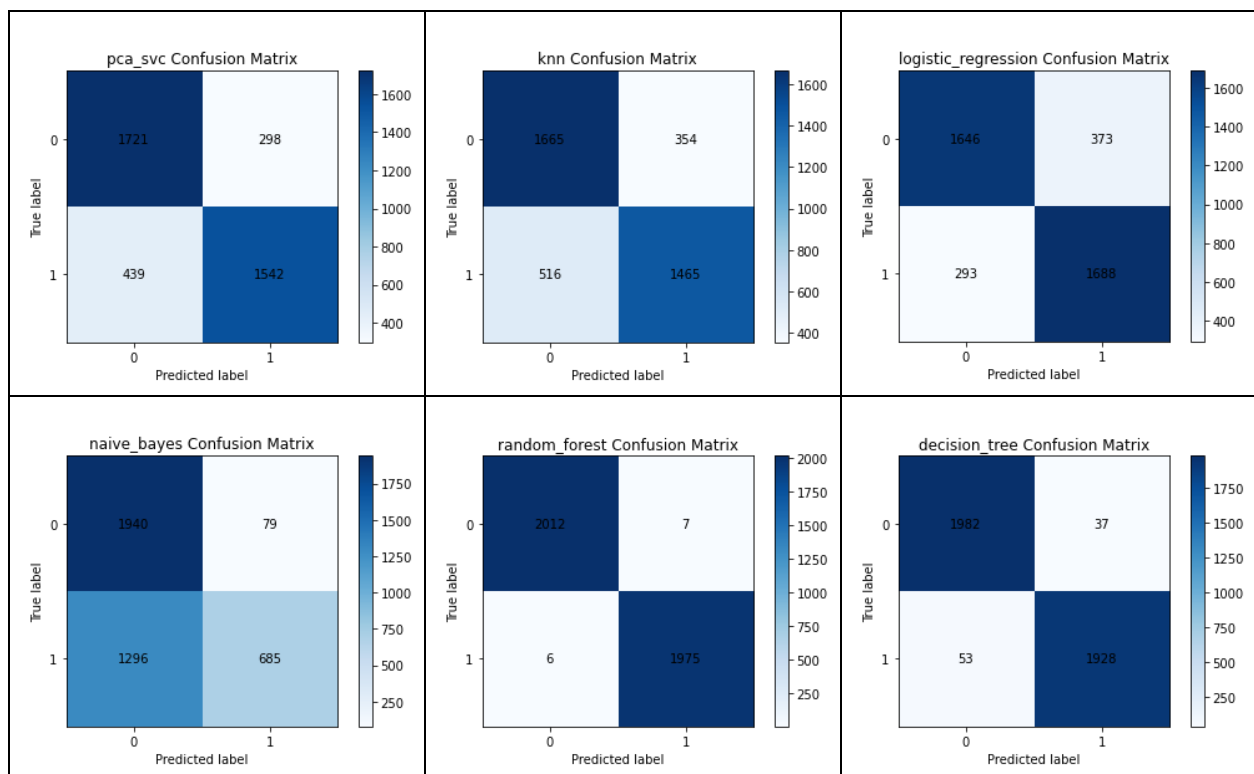


Figure 8: Confusion Matrix of Classification Results

Figure 8 shows a Confusion Matrix which evaluates the performance of the machine learning models. It shows the number of true positives, true negatives, false positives and false

negative predictions. The information reveals that most models correctly classify between the real and fake samples.

The final testing metric we used was the Wasserstein distance and Jensen-Shannon Divergence. As stated, these matrices are not standardized across all tabular synthetic data research, but it does provide a measurement for the distribution and probabilities for the datasets.

<b>Wasserstein Distance</b>	<b>4.991*</b>
<b>Jensen-Shannon Divergence</b>	<b>0.46</b>

*Figure 9: The Wasserstein Distance and Jensen-Shannon Divergence metrics of the real and synthetic datasets*

The Wasserstein distance shows the probability between two points in a trajectory [12]. It shows to be a better measurement of probability distribution compared to Euclidean distance in regard to comparing non-gaussian distributions between the datasets [12]. The Jensen-Shannon divergence shows robustness as a measure of distance comparison between two distributions. It can be used in quantifying the dissimilarity and similarity of distributions in the dataset [13].

Our interest regarding the measurement of the GAN15 model's synthetic data compared to the real data is trying to minimize the values in both Wasserstein distance and Jensen-Shannon Divergence. Both metrics do not have a set range and are to be used against other research that use similar metrics, however, as referenced before these measurements are not standardized across all research on synthetic tabular data.

## 6.4 Overall Evaluation

For the overall evaluation, based on the visual, statistical, and modelling aspects of testing, the synthetic data generated from the GAN15 is not perfect. The measurements show that there is discernability between the real and synthetic data – showing that the GAN15 model was not able to produce unrecognizable synthetic data.

With that in mind, the GAN15 model was successful in producing synthetic data that was close to representing the real dataset. The GAN15 model bypassed the challenges of fitting discrete and continuous features into a GANs network and was successful in producing data samples close to the distributions of the real dataset.

The GAN15 model further displayed its ability to generate several good samples as only two classification models showed an elevated level of prediction compared to the rest, as such not all the classification models were able to distinguish between the real and synthetic samples.

Whilst the GAN15 model samples were not perfect for the diabetes dataset, it showed that it is possible to generate high-quality samples in line with the original datasets distributions and produce samples with high scalability.



# 7 PROJECT MANAGEMENT

## 7.1 Project Management structure and methodologies

For managing this project, we followed the following two methodologies,

- 1) CRISP-DM Reference model
- 2) Agile

The chosen project management structure for this project will incorporate two methodologies: CRISP-DM Reference Model and Agile. The CRISP-DM Reference Model is a widely accepted methodology for managing data-driven projects in the field of Data Science. On the other hand, Agile methodology is based on the principle that software development is a collaborative effort of knowledge workers, who work together in teams to achieve project goals.

The CRISP-DM methodology comprises of six distinct phases, each of which plays a crucial role in completing a data project successfully. In the first phase, known as Business Understanding, the focus is on defining the data problem and identifying the business objectives. The second phase, Data Understanding, involves collecting the initial data, analyzing its properties, and verifying its quality. In the third phase, Data Preparation, the data is constructed, cleaned, and formatted. The fourth phase, Modelling, involves selecting the appropriate modeling technique, generating test and train sets, and building the model. The fifth phase, Evaluation, focuses on evaluating the results and reviewing the entire process. Finally, in the Deployment phase, plans are made for deploying the model, monitoring its performance, and producing a final report.

CRISP-DM is a widely used methodology for data-driven projects such as those involving AI and ML. It is a structured approach that involves several steps, from defining the problem to producing an output based on the data. However, it may face challenges in scaling ML projects with huge complexity or big data. To overcome this issue, we will combine CRISP-DM with Agile project management. Agile is a flexible and adaptive approach that allows for quick responses to changes in project developments. Specifically, we will use the Scrum method of Agile, which involves working on product backlogs, having regular meetings, and working in weekly sprints. This will allow us to be flexible in changing project direction when problems

occur, such as unworkable datasets or out-of-scope questions. However, since this is a data-centric project, obtaining regular feedback from customers or interviews may be difficult. By combining both methodologies, we aim to provide a structured approach to our project while still being flexible and adaptive to changes.[8]

## 7.2 Project planning

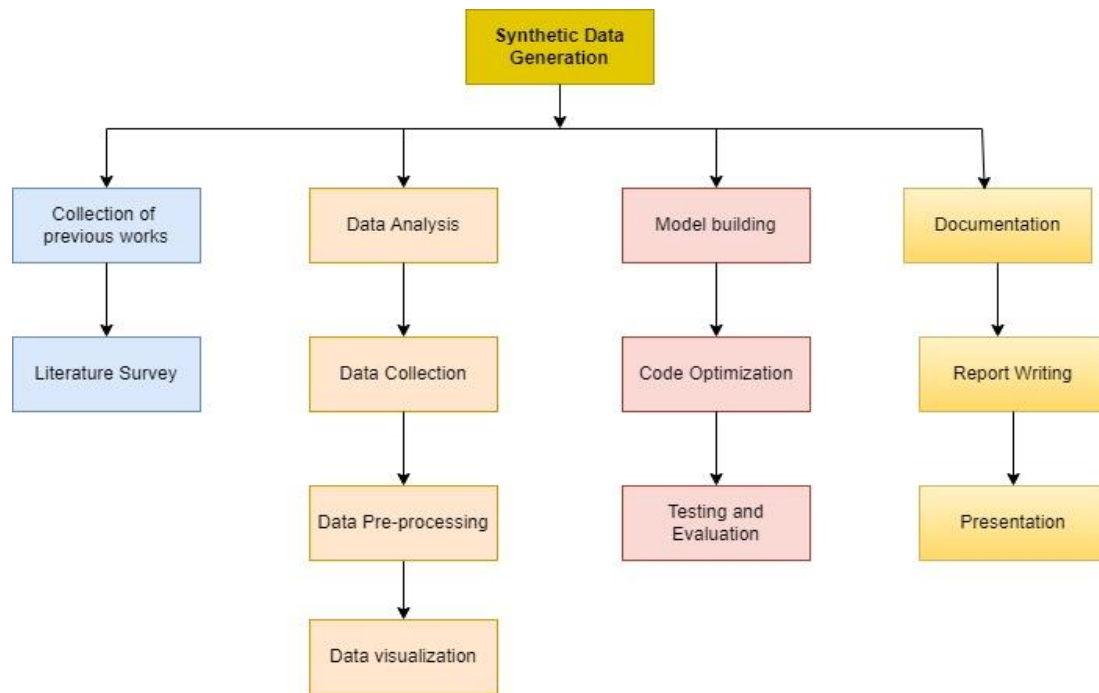


Figure 10: Project Structure

We divided our group into two teams which are Exploratory Data Analysis (EDA) team and GANs Team. The EDA team is involved in data collection, data preprocessing and data visualization. The GANs team will work on the GANs model and test it.

**EDA team Member** list are as follows:

- Isreal Ufumaka
- Dharsigan Bharathidasan
- Priteshkumar Thakkar
- Shikhar Sharma
- Ashu Berwal

**GANs team Member** list are as follows:

- Samuel Jularbal
- Ahmed Ali
- Abdul Wahid
- Yuan Zan
- Adedeji Adetunji

After the allocated tasks were completed, both teams worked together on project report writing. Based on the Gantt Chart mentioned below, the tasks are completed within the allocated timeframe.

### 7.3 Gantt Chart Representation

Gantt chart is a graphical representation for managing and monitoring tasks efficiently for any given project. It helps to allocate the tasks for each team member based on their availability and performance. It helps to utilize the available time properly [8].

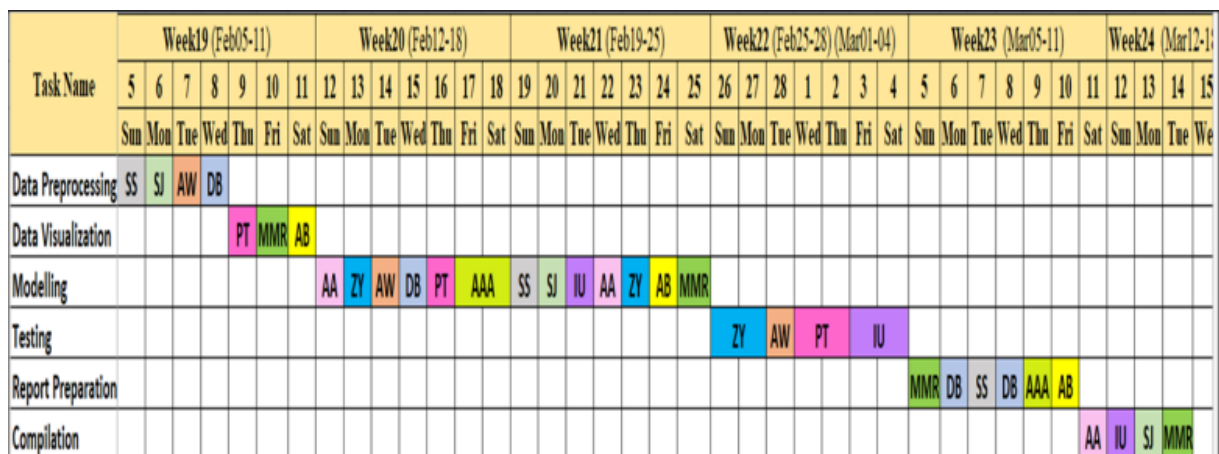


Figure 11: Gantt Chart Representation

#### Gantt Chart Explanation:

- Our project is divided into six subtopics which are Data Pre-processing, Data Visualization, Modelling, Testing, Report Preparation and Compilation.
- The team members name is abbreviated as follows: AA-Ahmed Ali, AAA - Adedeji Adetunji, AB-Ashu Berwal, AW-Abdul Wahid, DB-Dharsigan Bharathidasan, IU-

Isreal Ufumaka, PT- Priteshkumar Thakkar, SJ- Samuel Jularbal, SS- Shikhar Sharma, ZY- Zan Yuan.

- As per the Gantt chart observation, the whole project can be completed in 38 days.

## 7.4 Tools for project management:

### 7.4.1 Jira:

Jira is a project management tool used for creating and allocating tasks among the team members. It improves communication between the team members by providing transparency in the workflow of the project.

It acts as a platform for showcasing the completed tasks, backlogs, issues within the team members.

The team members can add their comments on the comment section to express their ideas on their own tasks or other team member's tasks.

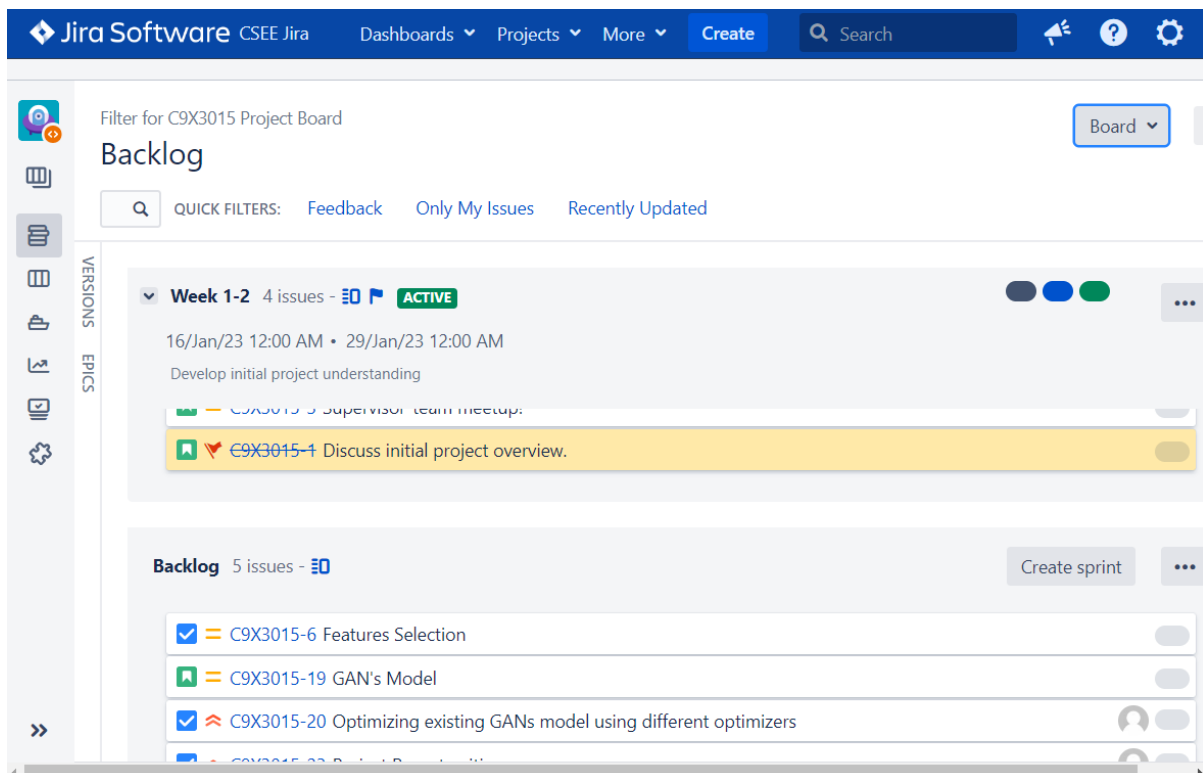


Figure 12: Jira Software Backlog

## 7.4.2 Git lab:

Git lab is a code management platform used in several projects. In that each team member can add their source code files which can be accessed or reviewed by other team members. It helps to improve the performance of the code by sharing the comments on the uploaded files.

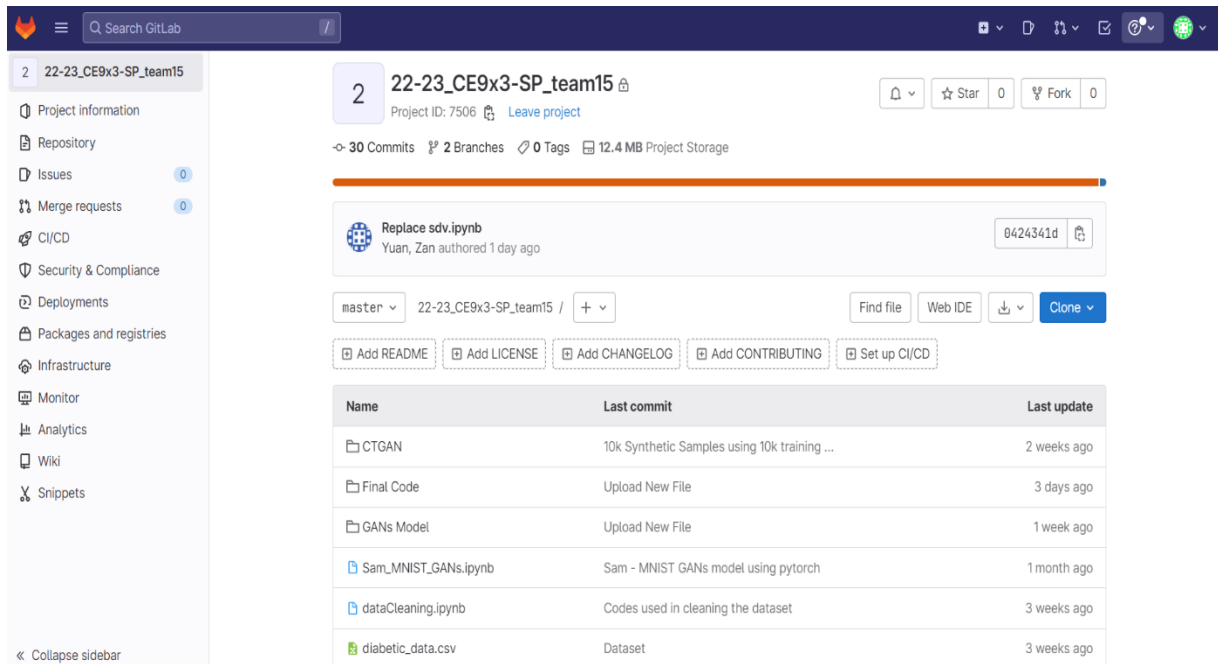


Figure 13: Git Lab Work Directory

## 7.4.3 Slack:

Slack is a communication tool to interact internally within the group members to share the issues, views about the allocated tasks. In slack a member can either text individual team members by sending a direct message or send a common text in the group chat. Slack is also used to make an internal voice or video calls to a specific person in a team or to a group for sharing their comments about the project.

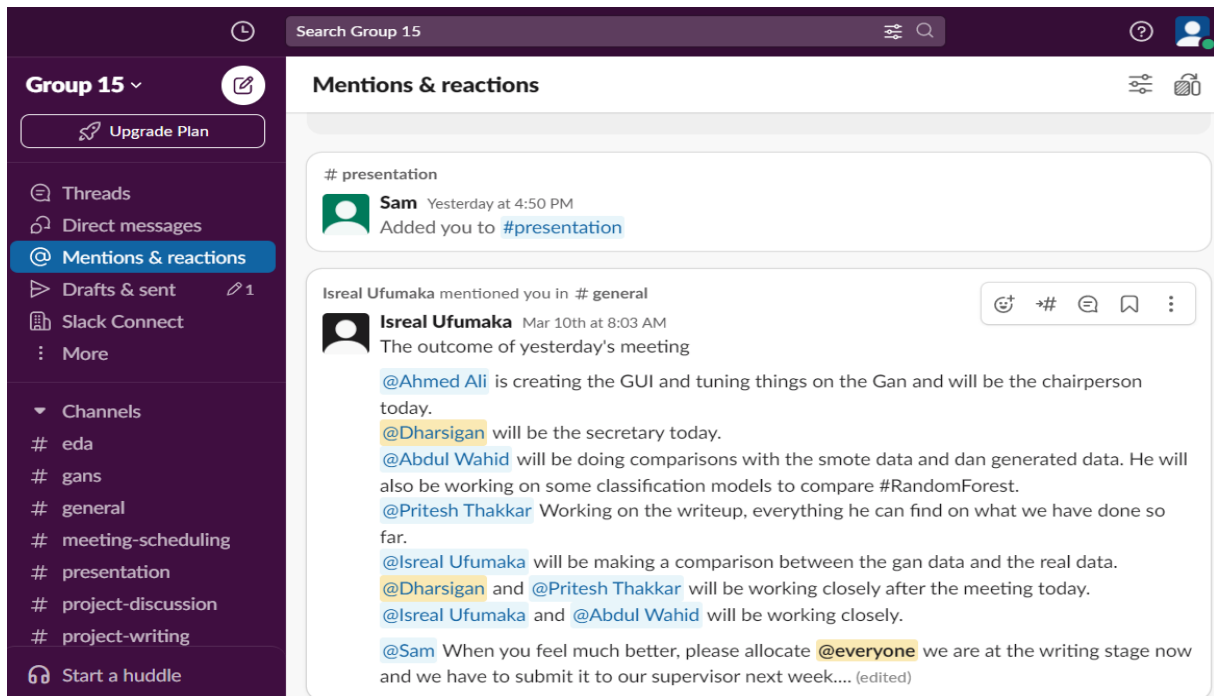


Figure 14: Slack Work Channels

#### 7.4.4 Zoom:

Zoom is another online video communication platform which is used to make arrange a formal meeting with the team members and with the project supervisor for discussing the progress of the project and to share the views/issues that needs to discuss with in the group.

We organized weekly meetings with team members unable to attend the in-person meetings to share the project's progress for that week.

## 9. CONCLUSION

In conclusion, this project's focus on designing and developing a General Adversarial Network (GAN) has been successful in producing synthetic data generation. This work contributes to the mounting challenges of data scarcity, privacy concerns and biases from AI-driven healthcare data. This group has been able to produce a robust GAN model (GAN15) to produce promising synthetic healthcare data closely resembling real-world data whilst addressing the issues presented. As a result, the successful implementation of this GAN model and future research has the potential to revolutionize the use of healthcare data in new AI models, affect how treatments are delivered and how healthcare services are provided – leading to a more efficient and promising development of healthcare.

## REFERENCES

- [1] Wikipedia,2022, “Synthetic Data” [online], [https://en.wikipedia.org/wiki/Synthetic\\_data](https://en.wikipedia.org/wiki/Synthetic_data) [Accessed: 05 March 2023]
- [2] Clore,John, Cios,Krzysztof, DeShazo,Jon & Strack,Beata. (2014). Diabetes 130-US hospitals for years 1999-2008. UCI Machine Learning Repository. <https://doi.org/10.24432/C5230J>.
- [3] Imtiaz S, Arsalan M, Vlassov V and Sadre R, “Synthetic and Private Smart Health Care Data Generation using GANs” [online], <https://ieeexplore.ieee.org/abstract/document/9522203> [Accessed :06 March 2023]
- [4] Hjelm D, Jacob A, Che T, Trischler A, Cho K and Bengio Y, “Boundary-Seeking Generative Adversarial Networks” [online], <https://arxiv.org/pdf/1702.08431.pdf> [Accessed : 07 March 2023]
- [5] Dahmen J and Cook D [online],” SynSys: A Synthetic Data Generation System for Healthcare Applications” [online], <https://www.mdpi.com/1424-8220/19/5/1181> [Accessed: 08 March 2023]
- [6] Danker F and Ibrahim M [online], “Fake It Till You Make It: Guidelines for Effective Synthetic Data Generation “ <https://www.mdpi.com/2076-3417/11/5/2158> [Accessed:09 March 2023]
- [7] Shin, HC. *et al.* (2018). Medical Image Synthesis for Data Augmentation and Anonymization Using Generative Adversarial Networks. In: Gooya, A., Goksel, O., Oguz, I., Burgos, N. (eds) Simulation and Synthesis in Medical Imaging. SASHIMI 2018. Lecture Notes in Computer Science(), vol 11037. Springer, Cham. [https://doi.org/10.1007/978-3-030-00536-8\\_1](https://doi.org/10.1007/978-3-030-00536-8_1)
- [8] “Software Requirements Specification Document”, <https://cseejira.essex.ac.uk/secure/attachment/52982/RequirementsAndPlanByGroup15.pdf>
- [9] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling Tabular data using Conditional GAN," arXiv preprint arXiv:1907.00503, 2019.



- [10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, pp. 2672-2680, 2014.
- [11] H. Lara and M. Tiwari, "Evaluation of Synthetic Datasets for Conversational Recommender Systems," *arXiv preprint arXiv:2212.08167*, Dec. 2022, cs.CL. [Online]. Available: <https://arxiv.org/abs/2212.08167>.
- [12] M. A. Abdullah, A. Pacchiano, and M. Draief, "Reinforcement Learning with Wasserstein Distance Regularisation, with Applications to Multipolicy Learning," *arXiv preprint arXiv:1802.03976*, 2018.
- [13] J. Lin, "Divergence measures based on the Shannon entropy," in *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 145-151, Jan. 1991, doi: 10.1109/18.61115.