UNIVERSITY OF TWENTE.

**Faculty of Electrical Engineering,**
**Mathematics & Computer Science**

# Privacy-Preserving Counterfactual Explanations To Help Humans Contest AI-Based Decisions.

**Dhivin Joshua Nelson**

**M.Sc. Thesis**
**August 2022**

**Supervisor**

prof. dr. Ir M.Van Keulen, UT
dr. D.K.Sarmah, UT
dr. B.Kamphorst, TNO
M Sc. Jasper van der waa, TNO

# Preface

This thesis recounts my work to get a Master's degree in Computer Science from the University of Twente, focusing on Data Science and Technologies. Two subfields of computer science have piqued my attention during my graduate studies: artificial intelligence (AI) and cryptography. These fields are becoming increasingly significant in our daily lives, and they provide us with complex and much-needed solutions.

Throughout my thesis, I am grateful for exploring the possibility of privacy-preserving explainable artificial intelligence. It has been more complicated than expected, necessitating reworking the research topics, but progress has been made. Furthermore, I am thrilled by the depth of knowledge I have gained from researching and writing my thesis.

I want to use this moment to express my gratitude to my supervisors and mentors. First, I want to thank my thesis supervisor, Prof. Maurice van Keulen, and Dipti Sarmah for participating in this study, and Bart Kamphorst and Jasper van der Waa for their insightful direction and participation in several brainstorming sessions. Second, I want to also express my gratitude to my family and friends for their support during my thesis.

Oh give thanks to the LORD, for He is good; for His steadfast love endures forever - 1 CHRONICLES 16:34 (Bible). I thank my Lord for His blessings and guidance throughout this study.

I hope you would enjoy the reading.

Dhivin J Nelson
*Enschede, August 2022*

# Abstract

Machine learning (ML)-enhanced telemedicine and chronic management systems that aid physicians or patients in their treatment may be a significant asset due to the escalating prevalence of diseases, particularly cardiovascular accidents. Adopting machine learning techniques might aid in the effort to make healthcare more accessible with increased efficacy and productivity. Machine learning-enabled chronic management systems, however, face some challenges.

First, the structure of Black-box AI models is highly complicated, and when applied to making decisions, it is difficult to understand how the model arrived at the result. The field of Explainable Artificial Intelligence (XAI) deals with making machine learning models more transparent and their predictions more explicable. More specifically, Counterfactual explanations are an XAI method that focuses on developing contrastive explanations by creating a hypothetical reality that opposes the observable facts. Counterfactual explanations answer the question," If X had not happened, Y would not have happened."

The second reason is that the importance of medical data and machine learning models has increased over time. As a result, there has been significant growth in norms and regulations governing their storage and use. Privacy-Enhancing Technologies (PET), combined with machine learning models, ensure that data and model privacy are not compromised.

It has not been thoroughly investigated how the use of counterfactual explanations might result in introducing new attack surfaces to the systems that are being modeled. For example, an attacker with malicious intent might use the information offered by explanations to their advantage to undertake a Membership Inference Attack (MIA) that would threaten the system's overall level of privacy [1]. This work will study the vulnerabilities of the counterfactual explanations, followed by performing a membership inference attack by generating multiple counterfactual explanations.

The use of Privacy-Enhancing Technologies, such as Differential Privacy, to protect privacy of individuals while generating counterfactual explanations, as well as the accompanying challenges and potential countermeasures against MIA is investigated in this research. In this study, synthetic datasets were used instead of the original data to construct counterfactual explanations, and the explanations were quantified using high-level performance metrics such as proximity, sparsity, and diversity to evaluate how useful they are in private and non-private contexts and relative entropy, which measures how much information is revealed as a result of these explanations.

The trials revealed that the performance of the synthetic data model degrades as the privacy budget shrinks. When the privacy budget is a more significant value, the synthetic data most accurately mimics the original data, and the counterfactual explanations provided are remarkably similar to those generated on the original data. On the contrary, a smaller privacy budget fails to simulate original data accurately, and the counterfactual explanations were ineffective; however, repeated queries revealed less information about the synthetic and original data demonstrating a good level of privacy. For a larger privacy budget, the synthetic data was exposed to a high level that always reveals few characteristics of the original data.

# Contents

# List of acronyms

**XAI** Explainable Artificial Intelligence

**PET** Privacy-Enhancing Technologies

**MIA** Membership Inference Attack

**DP** Differential Privacy

**CMS** Chronic Management System

**CVA** Cerebrovascular Accident

**DiCE** Diverse Counterfactual Explanations

Chapter 1

# Introduction

This chapter will address the health background and issues due to the chosen domain, followed by the use case scenario to explain to the reader the current challenges and the motivation which leads to the research question.

## 1.1 Health Background

According to WHO's 1970 definition, a stroke is "rapidly developing clinical evidence of focal (or global) impairment of brain function, with symptoms lasting 24 hours or more or leading to death, with no clear cause other than a vascular origin" [2]. Traditional stroke risk factors may be divided into two categories: those that can be modified and those that cannot. Hypertension, diabetes, high cholesterol, cardiovascular disease, lifestyle, smoking, and alcohol use are modifiable stroke risk factors, whereas age and gender are not modifiable factors [3] [4]. Stroke is the second biggest cause of death globally, killing 5.5 million people annually [4]. Stroke has significant economic and societal effects. Community-based education programs and the application of digital health technologies are essential techniques for the prevention of stroke [5].

Using digital health technology such as the Chronic Management System (CMS) would substantially reduce the burden of health care. However, CMS is not a substitute for physicians but rather a tool to assist healthcare practitioners or patients.

## 1.2 Use Case Scenario

Consider the following scenario to understand better why patients require a CMS: Mark has not been in the best of health as of late; he is a mature adult in his late 50s and has been feeling ill more frequently during the past several weeks and has a minor excess of body fat. He would want to visit a specialist for a checkup, but because medical professionals typically have large waiting lists, this may take a few weeks. A machine learning (ML)-enhanced CMS trained with big data containing information from different hospitals or nations can provide a preliminary diagnosis. Mark may have less of a need to continue making an appointment with the expert if the CMS produces a negative result. This would allow the medical professional to devote more time to treating needy patients. On the other side, if the CMS forecast is accurate and Mark is finally clinically diagnosed with a higher risk of having a stroke due to his lifestyle, this would most likely result in treatment plans that will need to be followed for the rest of his life.

## 1.3 Motivation

The scenario described in section 1.2 has two significant impediments to overcome. Firstly, why should Mark trust the result from a machine learning-enabled CMS? Moreover, there is no convincing evidence for Mark to challenge the decision if it turns out discriminatory. In order to counteract this obstacle, there are four approaches to contesting decisions made by the black-box AI model: (i) increasing the accuracy of the model by testing; (ii) ensuring perceptibility through explanation and confidence scores; (iii) training users; and (iv) monitoring the system [6]. This work will focus mainly on the Explainability of AI decisions.

The issue of trust becomes increasingly crucial as we outsource more decision-making to artificial intelligence (AI) and rely on AI to defend vital human commodities like safety, security, and healthcare [7]. To gain users' trust, the CMS must describe the decision-making process of the AI model or offer explanations for the AI's conclusion [8]. Research into XAI is on the rise because of the rising complexity of ML models. XAI is an area of study that aims to make black-box models transparent, interpretable, and understandable [9]. Explainable AI makes black-box models with complex learned functions comprehensible through transparent boxes or post-hoc explanations. Interpretable models approximate the black-box model in order to create a transparent box. Interpretable models include decision trees [10], Sparse Linear Models [11], Decision Rules [12], Rulefit [13]. On the other hand, post-hoc explanations attempt to explain the link between the feature values and the black-box prediction [1]. A disadvantage of XAI is that an increase in a model's transparency is often accompanied with a decrease in performance owing to the necessity for a less complicated classification system [14].

Users of CMS can be divided into two groups in a clinical setting: healthcare professionals and patients [15]. A healthcare professional is expected to have the expertise, whereas a patient's comprehension might vary greatly depending on their past education and experience with the condition. This work focused on contrastive post-hoc explanations since patients often anticipate a counter explanation to address the query, "Why this outcome and not that outcome?" [16]. Diverse Counterfactual Explanations (DiCE) [17], a post-hoc explanation method that generates contrastive/counterexamples is researched in this work. Its multi-objective nature enables it to generate a broad range of different and diverse counterfactual explanations from the original data. These counterfactual explanations are similar enough to the query instance but have a different prediction. Using the given scenario, Mark wants to understand the relationship between the features and the prediction by obtaining an explanation contrary to what was predicted.

The second impediment is that medical data privacy is a considerable concern. When it comes to the exchange of information, medical records, in particular, are accompanied by a considerable measure of concern [18]. Membership Inference (MIA) is an attack through which the adversary can discover if an individual record was included in the target model's training dataset or not through the confidence scores of the prediction [19]. The use of perturbing the outcome as a defense against MIA with strict privacy assurances is known as Differential Privacy (DP) [20]. In DP, statistical noise is used to restrict the amount of knowledge an adversary may learn due to the release of specific information [21].

*Aïvodji et al.* [1] stated that it remains an unresolved question whether DP will affect the robustness of the counterfactual explanations and the privacy of individuals, which motivates to research DP as a candidate to generate privacy-preserving counterfactual explanations.

## 1.4 Research Question (RQ)

The following study questions have been developed in order to investigate these potential trade-offs:

RQ 1 : *To what extent can we preserve privacy when generating multi-objective counterfactual explanations?*

Sub questions for the above mentioned research question are:

1.1 How vulnerable are multi-objective counterfactual explanations to membership inference attack?

1.2 Given the counterfactual explanation method *DiCE*, how effective is Differential Privacy in preserving the privacy of individuals?

1.3 How effective are the generated counterfactuals after applying Differential Privacy?

The following is how the remaining portion of this work is organized: A literature review on the more technical features of XAI may be found in chapter 2. Chapter 3 is a general review of counterfactual explanations. Chapter 4 discusses attacks on machine learning models. Chapter 5 is a general review of the various PET. In Chapter 6, an overview of the most recent advancements in Differential Privacy approaches is presented for the reader. The procedures for the tests are described beginning in Chapter 7, and then the experiments are carried out in Chapter 8. In the last chapter, Chapter 9, conclusions are formed from the acquired results, and the study's limits and future work are discussed.

Chapter 2

# Explainable AI

Artificial intelligence (AI) has seen a growth in the recent years, given the fact that its rising performance is unequaled by the solutions that are traditionally used. The inner workings of AI have become increasingly obscure as research and development of the technology proceeds. Although the growing opacity of the model does not pose a problem in all applications, decisions made by artificial intelligence may substantially influence human lives in some industries. For example, health care and law enforcement are two examples of fields where there is an inherent need to understand how an AI model came to its conclusion, this may assist in justifying or trusting the results, as well as avoiding inappropriate behavior such as prejudice [22]. Explainable AI can assist end-users in trusting that the AI is making good decisions, debate the decision, or build new hypotheses about the AI's capabilities.

## 2.1 Introduction

A real-life example to illustrate the social implications of using AI, An artificial intelligence system assigned ratings to instructors in Houston based education institution on how well their students performed on standardized tests compared to the state average [23]. Those staff who received excellent evaluations received accolades and sometimes even incentives. Those who performed poorly faced the possibility of being fired. The educators impacted by this AI system had the impression that it graded them incorrectly for no apparent reason. However, there was no way for them to verify whether or not the software was accurate or flawed. Explainable AI in ethical scenarios, if deployed, will assist end-users in trusting the AI system or contesting the decision made by the AI system. Figure 2.1 shows the contrast between the traditional machine learning setting and Explainable AI [7]. The traditional approach approximates a learned function with the data. The learned function is used to provide decisions for the tasks specified by the end-user. Whereas Explainable AI either approximates the complex learned function with a more straightforward function that the end-user interprets or uses post-hoc explanations to explain the reason behind the decision made by the complex learned function [10].

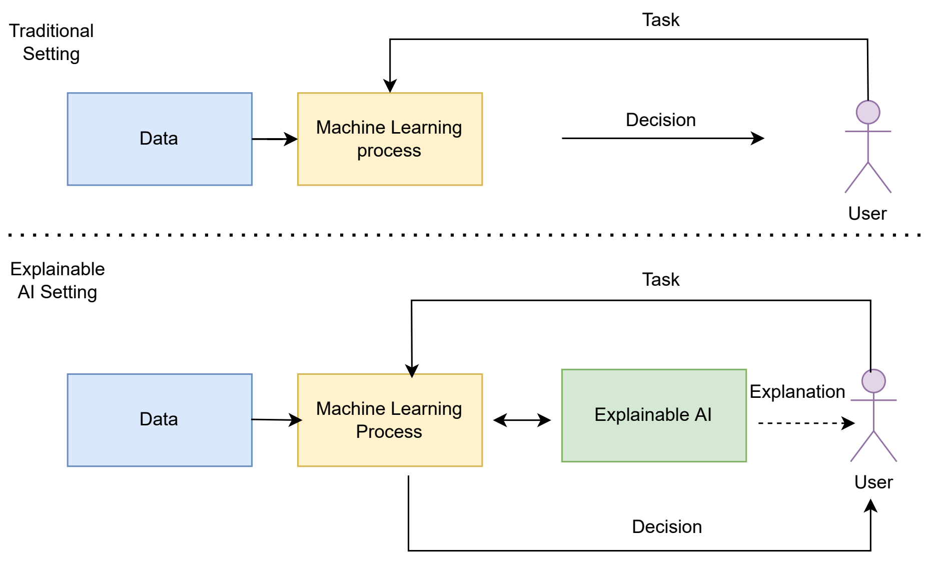


**Figure 2.1:** Traditional vs Explainable AI setting

### 2.1.1 Regulation

The existence of a ”right to explanation” in the EU General Data Protection Regulation (GDPR) and its consequences for companies and people have been highly contested topics of discussion. However, under the GDPR, there is no legally enforceable entitlement to explanation [24]. Relatively non-binding, Recital 71 in GDPR states that data processing techniques involving personal data ”should include specific information to the data subject, as well as the rights ”to obtain human intervention,” ”to express his or her viewpoint,” ”to obtain an explanation of the decision reached after such assessment,” and ”to challenge the decision” [25]. Given the societal implications of AI in crucial sectors, it is necessary, even if the right to explanations is not mandated.

## 2.2 Methods

Black-box models need additional techniques and approaches to determine how the system arrived at its conclusions [26]. The critical aspect for dividing XAI methods is the level of interpretation. If the explanations are for the original instance/current outcome, then it is local [27], and if the XAI method explains the entire black-box model, then it is global. The other distinction is that if the XAI method is applicable only for certain families of an algorithm (machine learning model), then it is model-specific. On the contrary, it is called model-agnostic [10]. Few examples are stated below for each category.

#### Global Model-Agnostic

Interpretable models are used when the black box model must be globally approximated. For example, a Decision tree can be trained on data and predictions from the black-box model. This resulting model (Decision tree) closely approximates the black-box model [10]. The decision rules from the tree can be used to explain the black-box decisions. The biggest challenge is *fidelity*, which measures how well the interpretable model approximates the black-box. Black-box models are complex; using a simple model to approximate a complex model is difficult. Therefore, it is not easy to achieve *global fidelity*.

### Local Model-Agnostic

*Ribeiro et al.* [27] proposed a local interpretable model explanation (LIME) to explain local individual black-box predictions. Lime creates a local approximation for the black-box model for the original instance. The local model should be a good approximation of the black-box model in that local region. The accuracy for approximation is known as *local fidelity*.

To understand the working of LIME, a local neighborhood of instances around the original instance is generated using an exponential smoothing kernel. Exponential smoothing kernel returns proximity score that helps sample around the original instance to provide faithful local explanations. For example, in Figure 2.2 the red cross denotes the original instance, and the exponential smoothing weights each sample around the red cross based on its proximity.

Next, a linear model (Interpretable model) was used to approximate the decision boundary of the black-box model in the local neighborhood (black-dashed line in Figure 2.2). Finally, the most influential features for the prediction in the local neighborhood are selected to generate explanations.

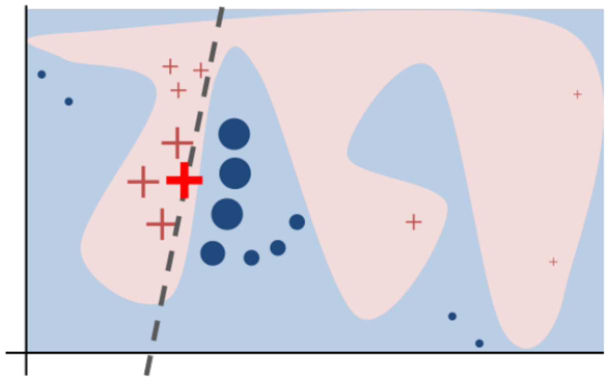


**Figure 2.2:** LIME [27]

### Neural Network Interpretation

In a neural network context, many layers of multiplication with weights and non-linear transformations are used to produce predictions. Depending on the design of the neural network, a single prediction might comprise millions of numerical computations. Unfortunately, our brains cannot trace the precise path from data input to prediction. Therefore, saliency maps are often used to emphasize the pixels in an image (original instance) critical to the network's classification to comprehend the findings of a neural network [28].

### Example-Based Explanations

Example-based explanation approaches choose specific examples of a dataset to explain the behavior of machine learning models.

The approaches mentioned above do not support a strategy for finding a contrastive explanation for the existing decision by the black-box model. Counterfactual explanations are example-based explanations that provide contrastive explanations [10]. Counterfactual explanations will help the end-user comprehend how to succeed in a specific activity. As a result, it stands out above other methods. Counterfactual explanations are covered in detail in the next chapter.

Chapter 3

# Counterfactual Explanations

Counterfactual explanations are post-hoc explanations that help understand the meaningful relationship (causal) between the features and prediction [17]. The GDPR does not seem to necessitate opening the ”black-box model” to explain the reasoning of the decision-making process to data subjects. With this in mind, counterfactuals may give information following the GDPR’s different informational standards while also giving some insight into the reasoning behind a particular decision [29].

To illustrate counterfactual explanations, in Table 3.1 the original instance is predicted with *Stroke* by a black-box AI model. If the end-user wishes to know how they can move to the other side of the decision boundary, that is *No Stroke*, counterfactual explanations will provide examples based on the desire of the end-user (see Figure 3.1). Counterfactual instance (explanation) generated helps the end-user to understand that, by reducing the average glucose level (diabetes) from 165.97 to 120.56 and BMI (obesity) from 31 to 24.9, the prediction changes to *No Stroke*.

| Example | | | | | | | |
|---|---|---|---|---|---|---|---|
| Instance | Gender | Age | Smoking status | Hypertension | avg gluscose level | BMI | Prediction |
| Original (x) | Male | 70 | 0 | 0 | 165.97 | 31 | Stroke |
| Counterfactual (x’) | Male | 70 | 0 | 0 | 120.56 | 24.9 | No Stroke |

**Table 3.1:** Counterfactual instance (explanation) for the original instance

A valid counterfactual instance (explanation) should be closer to the original instance, thereby promoting higher actionability. An important consideration while making a recommendation is which attributes can be modified (for example, high blood pressure, obesity, and diabetes) and which cannot be changed (e.g., gender, age). In a recommended counterfactual, the non-modifiable features should never be altered. When a modification is made to a legally sensitive attribute and the predicted outcome changes, this clearly shows that the data used to generate counterfactuals is biased.

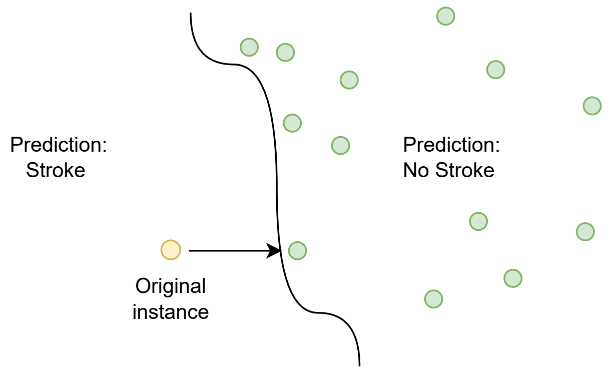


**Figure 3.1:** Counterfactual Explanations

## 3.1 Themes of research for Counterfactual Explanations

The themes involved during the generation of counterfactuals are as follows [30],

*Proximity* means that the original and counterfactual instance (example) should be close enough to produce different outputs. To explain this property, the original instance and counterfactual instance in Table 3.1 are close enough but produce different outputs. A distance metric $d$ is used to measure the distance between the original instance $(x)$ and counterfactual instance $(x')$. The counterfactuals (CF1, CF2, CF3) generated in Figure 3.2 are closer to the decision boundary when proximity is weighted high. Conversely, the produced counterfactuals are far from the decision boundary when proximity is weighted low.

*Actionability* means the counterfactual instance (example) should never change immutable features. To illustrate this example, the counterfactual instance in Table 3.1 preserves actionability by not changing *Age* and *Gender*.

*Data manifold closeness* means the generated examples should be close enough to the training data, thus making counterfactual instances to be valid, accurate, and not unrealistic for the end-user. In Figure 3.2 the dashed line represents data manifold.

*Sparsity* means the number of changes in features that produce a counterfactual instance with a different output. To illustrate this, in Table 3.1 the changes in feature *avg glucose level* and *BMI* produce a counterfactual instance with a different output. For a counterfactual to be as valuable as possible, the number of features that need to be altered should be kept to a minimum. It has been stated that individuals find it simpler to comprehend shorter explanations [31]; hence, sparsity should be taken into mind as an essential factor. When proximity is given a high weight, sparsity tends to be low. This indicates that there have been few modifications made to the features. On the other hand, if the proximity weight is set to a low value, the sparsity will be significant, and the features will undergo many modifications. When comparing the original instance to the counterfactuals, the sparsity can be determined by employing a distance metric that keeps track of the number of times the features of the counterfactuals have been altered.

Finally, *Multiple Diverse* counterfactual instances provide end-users with various options to choose which explanation best suits the end-user. *Mothilal et al.* [17] proposed the concept of mutiple and diverse counterfactuals and explained that the generated counterfactual explanations should follow two properties: *feasibility* and *Diversity*. A single counterfactual instance may not be feasible for the end-user; and in order to achieve feasibility multiple and diverse counterfactual explanations are generated. In Figure 3.2 multiple counterfactuals are generated (CF1, CF2, CF3) that promotes more feasibility than a single counterfactual generated in Figure 3.1.

There is a trade-off between proximity, sparsity, and diversity. When proximity is weighted high, sparsity is low, and diversity between the counterfactuals generated is low. On the other hand, if proximity is weighted low, sparsity is high, and diversity between the counterfactuals generated is high. Figure 3.2 illustrates the trade-off when proximity is high, diversity is low, and vice-versa.

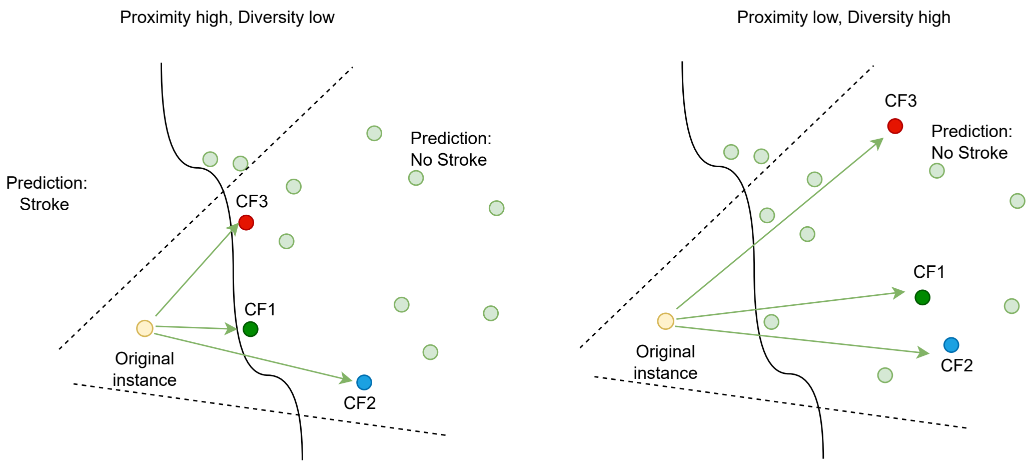


**Figure 3.2:** Themes

## 3.2 Generating Counterfactual Explanations

This section will explain the three most common counterfactual explanation methods followed by the research themes covered in the methods [10]. Table 3.2 illustrates counterfactual explanation methods followed by the themes in the research [30].

| Themes of Counterfactual explanations | | | | | |
|---|---|---|---|---|---|
| Methods | Proximity | Actionability | Data manifold closeness | Sparsity | Mutliple,Diverse |
| [29] UCA | ✓ | ✗ | ✗ | ✗ | ✗ |
| [17] DiCE | ✓ | ✓ | ✗ | ✓ | ✓ |
| [32] MOC | ✓ | ✓ | ✓ | ✓ | ✗ |

**Table 3.2:** Themes of research in counterfactual explanations

### 3.2.1 Generating single counterfactual explanations (UCA)

*Wachter et al.* [29] proposed an optimization algorithm to generate counterfactuals. Equation 3.1 is the optimization objective that denotes the *proximity* constraint mentioned in Table 3.2. Where $x$ is the original instance, $x'$ is the counterfactual instance, $y'$ is the desired prediction specified by the end-user, $f(x')$ is the output from the black-box model.

$$\arg\min_{x'} d(x, x') \text{ subject to } f(x') = y' \tag{3.1}$$

The Equation mentioned above is converted into an unconstrained optimization (see Equation (3.2)) where $\lambda$ is the regularizer term that balances between the desired output and distance function $d(x, x')$.

$$L(x, x', y', \lambda) = \arg\min_{d}\max_{\lambda} \lambda(f(x') - y') + d(x, x') \tag{3.2}$$

### 3.2.2 Generating multiple and diverse counterfactual explanations (DiCE)

*Mothilal et al.* [17] proposed diverse counterfactual explanations (DiCE), aiming to generate multiple and diverse counterfactual explanations, thus enabling end-users with many options. The optimization constraints are similar to Equation 3.2. The only difference here is the diversity of the generated counterfactuals. *Diversity* is measured using a determinantal point process that calculates the determinant of the matrix $k$, where $k$ measures the distance between the generated counterfactuals $dist(c_i, c_j)$ (see Equation 3.4). To avoid ill-defined determinants, the diagonal values in the matrix $k$ are perturbed. *Proximity* between the query instance and the multiple counterfactuals is measured using $dist(c_i, x)$. Equation 3.3 is the loss function specified for generating counterfactuals optimized using a gradient descent. $f(c_i)$ is the output from the black-box model for the multiple counterfactuals. $(f(c_i) - y^{'})$ calculates how close is the predicted outcome of the counterfactuals to the desired output $(y^{'})$ specified by the end-user. $\lambda_1$ and $\lambda_2$ are hyperparameters that balance *proximity* and *diversity*.

$$C(x) = \arg\min_{c_i....c_k} \frac{1}{k}\sum_{i=1}^{k}(f(c_i) - y') + \frac{\lambda_1}{k}\sum_{i=1}^{k} d(c_i, x) - \lambda_2 \text{ dpp\_diversity}(c_1...c_k) \tag{3.3}$$

$$\text{dpp\_diversity} = \det(k), \text{ where k} = \frac{1}{1 + \text{dist}(c_i, c_j)} \tag{3.4}$$

There is a trade-off between *proximity* and *diversity* when generating counterfactual instances. If *proximity* $\lambda_1$ is weighted high, then *diversity* $\lambda_2$ reduces and vice-versa. So the hyperparameters should be balanced such that counterfactual instances preserve both properties. The motivation specified by *Mothilal et al.* [17] to generate multiple and diverse counterfactual instances is *actionability*. A single counterfactual instance may not be actionable for the end-user, but diverse and multiple counterfactual instances increase the probability of *actionability*.

### 3.2.3 Generating multi-objective counterfactual explanations (MOC)

Multi-objective counterfactual explanations (MOC) satisfy *proximity*, *actionability*, *data manifold closeness*, and *sparsity* [32]. The first objective is to get the desired output specified by the end-user. The first objective is weighted using $\lambda$ (see equation 3.5). The second objective *(proximity)* is the distance between the counterfactual instance and original instance. The third objective is the number of feature changes done between the original instance and counterfactual instance *(sparsity)*. Finally, the fourth objective checks whether it is a plausible counterfactual, i.e., follows the underlying distribution *(data manifold closeness)*. The underlying data distribution is denoted by $X^{obs}$ and $E$ denotes explanation for the original instance $x$. The objective of Equation 3.5 is to decrease the distance between the original and counterfactual instances while also increasing the likelihood of obtaining a counterfactual with the desired output specified by the end-user.

$$E(x) = \min_{d}\max_{\lambda} \lambda(f(x') - y') + d(x, x') + g(x' - x) + l(x, X^{obs}) \tag{3.5}$$

Chapter 4

# Attacks on Machine Learning

An adversary's objective in privacy-related attacks is to get information that was not meant to be disclosed. This knowledge may pertain to the training data, the model, or even the extraction of information about data features. In this study, the attacks are classified as membership inference, property inference, and model extraction.

## 4.1 Membership Inference

A membership inference attack detects whether a particular data item was included in the target model's training or not [1]. The adversary queries the target model (machine learning model), receiving the prediction. An adversarial attack model uses the prediction and the real label to deduce the membership of the instance (see Figure 4.1).

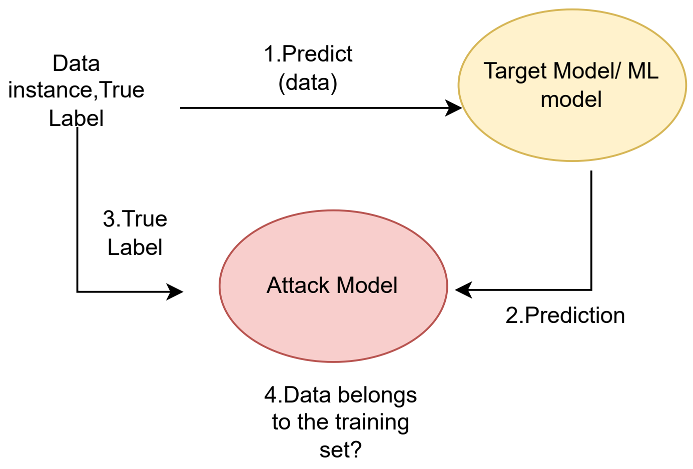


**Figure 4.1:** Membership Inference Attack

Membership inference attacks commonly leverage machine learning overfitting to differentiate between members and non-members of training data (see Figure 4.2). When a model is overfitted, it performs well on its training samples but not so well on new data. It is simpler for an adversary to conduct membership inference attacks if a machine learning model is overfitted [19]. In Figure 4.2, an adversary can construct an attack model to distinguish between the confidence scores of instances that a machine learning model is familiar with and instances that it is unfamiliar [19]. An example of the confidence-based membership inference attack is given below.

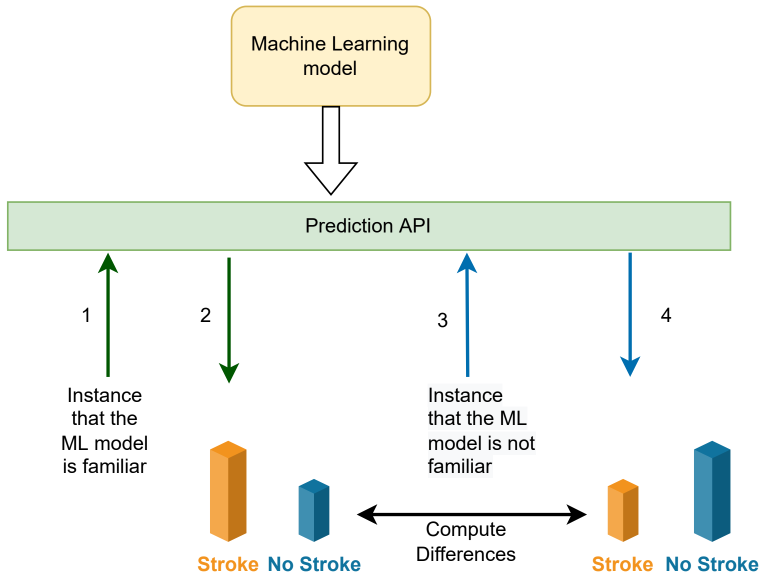


**Figure 4.2:** Exploit Model's Prediction

*Shokri et al.* [19] proposed an attack model for membership inference, a confidence-based attack exploiting the model's prediction. The adversarial creates *k* shadow models denoted by $f_{\text{shadow}}$ and trains each model using shadow datasets. For simplicity, the shadow datasets were generated using the marginal distribution of the features in the target data.

The attack strategy is as follows, each shadow model is queried with its own shadow dataset, $D$train, and a disjoint dataset $D$test (see Figure 4.3). For all *(x, y)* $\in$ $D$train, the prediction ***y*** $=$ $f_{\text{shadow}}$(x) is computed and added to a new record in the form of *(y,* ***y****, in)*. *y* denotes the confidence score of the true label, ***y*** denotes the confidence score of the predicted label, and *in* denotes that the record was present in training [19]. A similar process was followed for the disjoint dataset ($D$test), and the output was then added to a new record *(y,* ***y****, out)* where, *out* denotes that the record was not present in training. Finally, the attack model is trained on the records (*(y,* ***y****, in)* and *(y,* ***y****, out)*) to perform a membership inference.

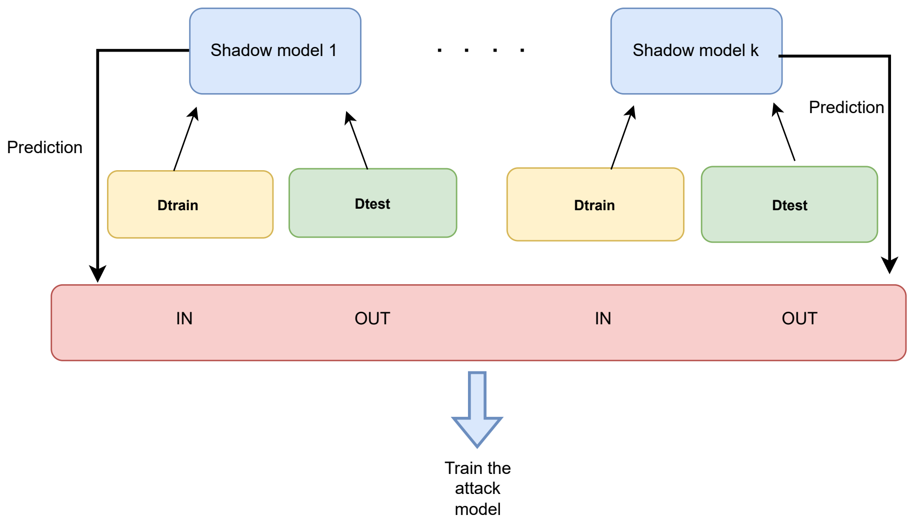


**Figure 4.3:** Training Attack Model for Membership Inference

## 4.2 Property Inference

Property inference is based on the premise that models trained with similar datasets or learning processes would have similar representations of functions [33]. The adversary aims to identify patterns in the target model parameters that expose characteristics that the model owner may not wish to disclose publicly.

Figure 4.4 shows how the adversary first trains a shadow model on the same challenge as the target model to build an attack model that predicts whether or not the target model has the property $P$. Shadow model is trained on a dataset similar to that of the target model but explicitly constructed to either have the property $P$ or not $(\widetilde{P})$. Finally, the Shadow model is used to create the attack dataset, which consists of $(F, P)$ and $(F, \widetilde{P})$, where $F$ represents the feature representation. Then, the attack model is trained on the attack dataset to distinguish whether or not the property belonged to the target model [33].

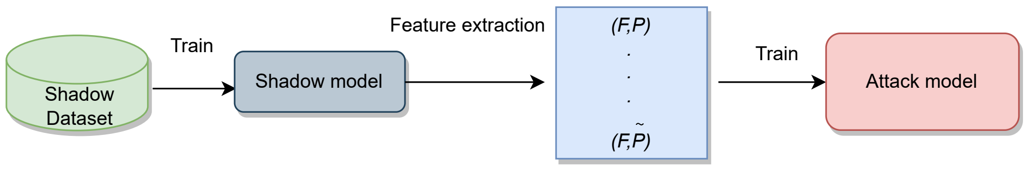


**Figure 4.4:** Property Inference Attack

## 4.3 Model Extraction

Model Extraction attack is an inference attack, where the adversarial aims to create a surrogate (replicate) model with high fidelity or accuracy that can be a substitute for the commercial API for the financial benefit [1]. These attacks are often categorized into *reconnaissance-motivated* and *theft-motivated model extraction* [34]. For example, in Figure 4.5, the theft-motivated adversary prefers the blue dashed decision boundary because of its high classification accuracy. In contrast, a reconnaissance motivated adversary prefers the dotted brown line because it closely approximates the target model's decision boundary.

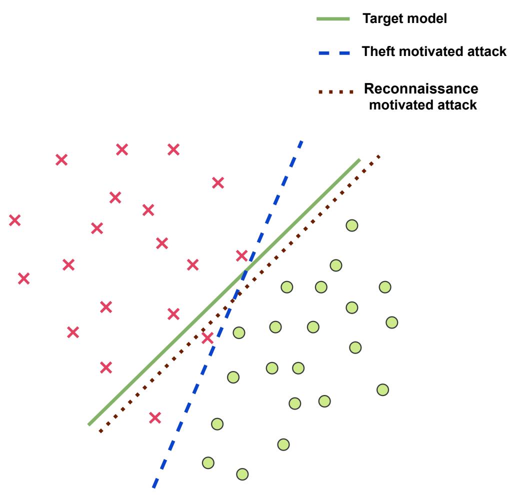


**Figure 4.5:** Types of Model Extraction

## 4.4 Adversarial Attack on Counterfactual Explanations

Machine learning models have been the focus of the attacks so far. Attack using counterfactual explanations (XAI) is addressed in this section.

The adversary utilizes the explanations to build the surrogate model in a counterfactual explanations-based model extraction attack [1]. For example, in a black-box situation, given a target model, its prediction API, and its explanation API, the explanation-based model extraction approach exploits the explanations of the original instance (current outcome) to generate a surrogate model (see Figure 4.6).

In the proposed work by *Aïvodji et al.* [1], the attacker builds an attack dataset $D_A$. For each $x \in D_A$, the attacker queries the prediction API and explanation API and receives counterfactual explanations for each query. The attacker then trains a surrogate model with these generated counterfactual explanations (Shadow Dataset) to create a surrogate model with high fidelity and accuracy. The framework that was used to generate counterfactuals is explained in section 3.2.2.

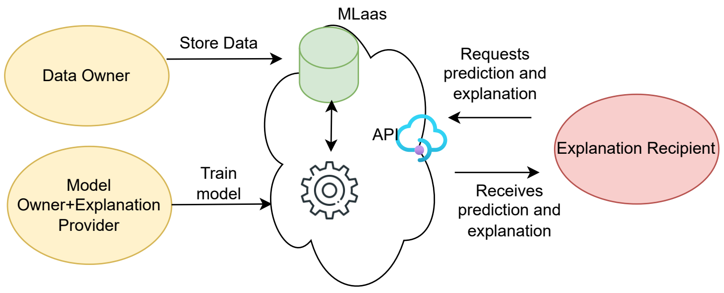

MLaas
Data Owner
Store Data
Requests prediction and explanation
API
Explanation Recipient
Model Owner+Explanation Provider
Train model
Receives prediction and explanation

**Figure 4.6:** Architecture for Model Extraction Attack

Chapter 5

# Privacy-Enhancing Technologies

Machine learning models require data to train. Data is often gathered by institutions specialized in information aggregation; however, data providers are typically hesitant to contribute data owing to privacy concerns. Therefore, in order to safeguard privacy and ownership during the collaborative training of the machine learning model, federated learning (FL) or cryptographic or perturbation approaches are employed. This chapter will discuss the cryptographic approaches, federated learning, and perturbation strategy.

## 5.1 Cryptographic approaches

Cryptographic approaches are used to solve the problem of protecting the confidentiality of data and models. Homomorphic encryption, for example, is a kind of cryptographic approach that enables users to do computations on encrypted data, which ultimately leads to a solution that is also encrypted and can only be deciphered by the owner of the secret key [35]. This assures that there is no breach of confidentiality while the computation is being done. In addition, there is a technique known as Secure Multiparty Computation, which makes it possible to compute a function or trade data without ever disclosing the data used in the process [36]. These cryptographic algorithms make it possible to train machine learning models using only encrypted data, ensuring both the model's and the data's integrity.

### 5.1.1 Homomorphic Encryption

A type of encryption known as homomorphic encryption (HE) uses the fact that some encryption techniques are homomorphic. This property makes it possible to execute operations on ciphertexts (encrypted data). The resulting output is legitimate in the same way if the data were not encrypted. For example, in Figure 5.1, the result of the summation function remains the same in the encrypted domain and the unencrypted domain. There are different types of Homomorphic encryption, partial, somewhat, and fully [35]. Partial homomorphic encryption (PHE) permits only a single operation to be performed numerous times on ciphertext. The operation may be either addition or multiplication, but never both. In contrast to PHE, somewhat homomorphic encryption (SHE) permits a limited amount of addition and multiplication operations. Finally, fully homomorphic encryption (FHE) enables an endless number of addition and multiplication operations on ciphertexts.

The most common homomorphic cryptosystem are ElGamal [37] and Pallier [38]. A practical example of the Pallier cryptosystem in the context of the neural network is given in this section.

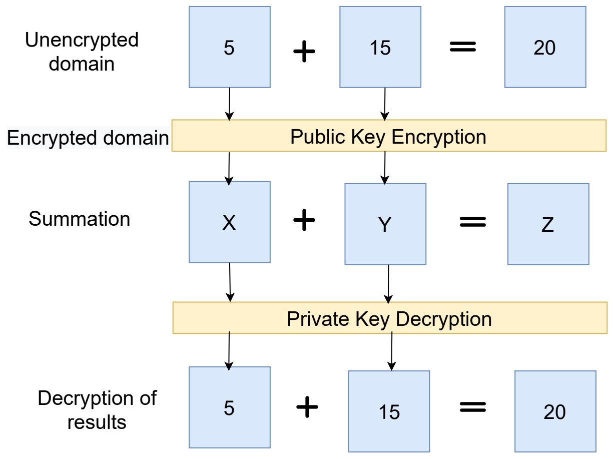


**Figure 5.1:** Homomorphic Property

A practical example of homomorphic encryption in a neural network to preserve data privacy is as follows, *Zhu et al.* [39] proposed a privacy-preserving neural network based on the homomorphic cryptosystem, and the experiment was performed on the MNIST dataset. To encrypt the MNIST dataset, the authors used the pallier homomorphic cryptosystem [40]. Pallier cryptosystem is a public-key encryption scheme that has the following stages, key generation, encryption, and decryption [38]. The key generation system generates a public key and a private key. The public key is used to encrypt the MNIST dataset, which results in the ciphertext of images. The client then sends the encrypted data to the server-side. At the server-side, features are extracted (convolution). In order to compute the activation function, *Zhu et al.* [39] designed an interactive protocol where the server computes the activation (ReLU) by interacting with the client. The client decrypts the result from convolution and sends positivity or negativity. If negative, the server squashes the output to zero; on the other hand, if positive performs the activation function (ReLU). An average pooling was used to calculate the average of each patch in the extracted features. A linear transformation was applied in the final classification stage, and the same process mentioned above was followed for the final activation function (ReLU). Finally, the output from the activation function is sent to the client; the client then decrypts the output using the private key generated from the pallier cryptosystem. Figure 5.2 below explains the entire process.

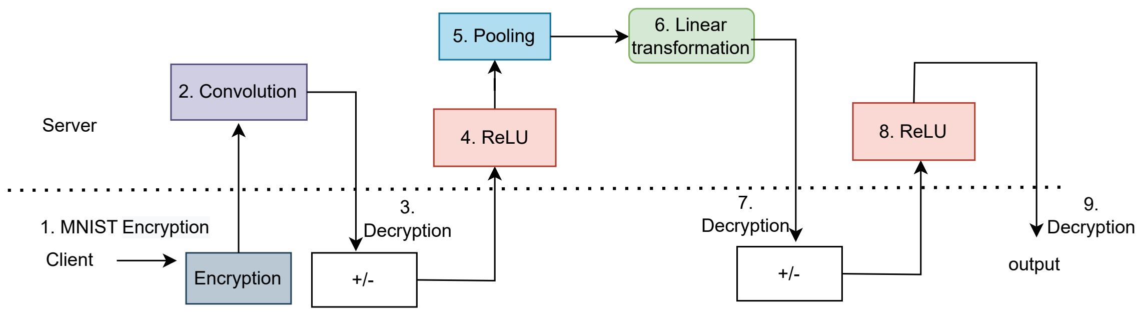


**Figure 5.2:** Homomorphic Encryption in Convolutional Neural Network

### 5.1.2 Secure Multi-Party Computation

Secure multiparty computation (S-MPC) protects the privacy of many participants' input (data) to calculate a joint output. Sub problem for secure multi-party computation is two-party computation (2PC) [41].

Homomorphic encryption and garbled circuits enable two parties to calculate the outcome of their input (data). The steps involved in a garbled circuit with respect to a Boolean logic are as follows: In order to generate the garbled circuit, a garbler encrypts the boolean circuit and then encrypts the truth table's output entry with the two input labels (Double-Key symmetric encryption). The table that has been encrypted is called a garbled table. Lastly, the garbler sends the input labels and the encrypted table to the other party. Finally, the evaluator deciphers the table with the input labels to get the output label [41].

Most MPC protocols use Shamir secret sharing, when an honest majority is needed in secure multiparty computation. When the secret is shared among $n$ parties, $t+1$ or more parties may come together to interpret the secret. This is an example of a secret sharing system. There is no way for parties lower than $t$ to get any information or decipher the secret [36].

For MPC and Homomorphic encryption, the most significant problem is the speed. In addition, it is necessary for all parties involved in MPC to communicate and be connected; hence the overhead costs of MPC are considerable [42].

## 5.2 Federated Learning

Remote execution is an alternative to a centralized training model because the data is not stored in a centralized location; Federated learning may decentralize the modeling process while still training a global model [43]. Federated learning can be used even if the information is kept on a large data center or as little as a single mobile phone. In Federated learning, local models are trained on each device, and the parameters are subsequently shared to train the global model.

A widely used framework is Federated Averaging (FedAvg) [44]. (1) Selected parties in the Federated averaging framework are initially provided with an up-to-date global model. (2) The global model is then updated using the local data. (3) Next, the server receives the local models. Finally, a new global model is generated by averaging all incoming local models. The above mentioned method continues as long as FedAvg has not reached the stated number of iterations. (see Figure 5.3).

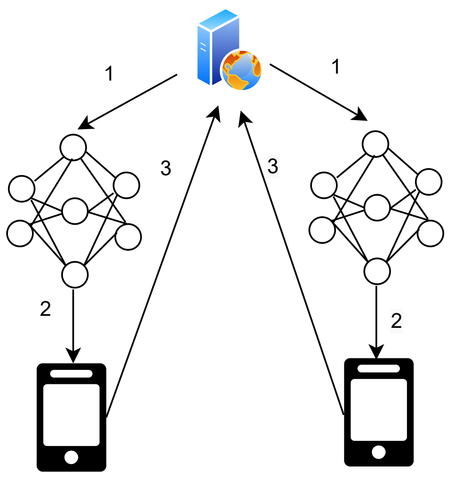


**Figure 5.3:** Federated Average

## 5.3 Perturbation Approach

Data perturbation can preserve individual record confidentiality by using a data security approach that introduces 'noise' into databases (data) [45]. Differential Privacy, the most frequent type of perturbation technique, offers a quantitative measure of privacy and is quite popular [46].

### 5.3.1 Differential Privacy

Differential Privacy is a guarantee made by the data holder to the data subject that the use of the data in any research or analysis would not impair the subject's privacy [47]. The basic rule of Information recovery stipulates that excessively exact responses to too many questions would undermine privacy; Differential Privacy seeks to extend this phenomenon as long as possible and produce noise in the output that obscures the involvement of a particular person in the data [47]. There is a trade-off between privacy and accuracy; a large amount of noise added will increase privacy but reduce the system's accuracy.

A mechanism is considered differentially private when an attacker looking at the output cannot comment on whether or not a specific instance was included in the dataset [48]. In addition, differentially private algorithms may protect against attacks like membership inference (Refer to section 4.1). The next chapter covers Differential Privacy in detail because of its relevance in this research.

Chapter 6

# Differential Privacy Details

Differential Privacy, introduced by *Dwork et al.* [47] states that, a mechanism is differentially private when looking at the output; the attacker cannot comment whether an individual instance was included in the dataset or not. When a mechanism is differentially private, it can protect against intrusions such as membership inference. Below is the generalized equation (6.1) for Differential Privacy. *D* and $D^{'}$ are two neighboring datasets that differ in only one row, $M$ is a randomized mechanism, and $S$ is a subset of the outcome space [47]. Differential Privacy guarantees that the outcome produced by both datasets *D* and $D^{'}$ will remain the same even if they differ in one record.

$$\Pr[M(D) \in S] \leq e^{\varepsilon} \Pr[M(D^{'}) \in S] \tag{6.1}$$

The Equation 6.1 can also be re-written as Equation 6.2 which is bounded by an $\varepsilon$ value known as privacy budget. Privacy budget controls how much the mechanism's ($M$) output can differ between the two neighboring dataset. If the ratio of the probabilities of the outcome of the two datasets, *D* and $D^{'}$ is small, then the adversarial attacker cannot learn whether an individual was present or not.

$$\frac{\Pr[M(D) \in S]}{\Pr[M(D') \in S]} \leq \varepsilon \tag{6.2}$$

Differential Privacy is closely related to *divergence* in statistics [49]. Divergence measures the difference between two distributions. Renyi divergence states that if the distribution over the outcome $M(D)$ and $M(D')$ is the same even if they differ in one instance, the mechanism is differentially private [49]. The Laplace mechanism is a classical approach for differential privacy, which adds noise sampled from the Laplace distribution to the output of the mechanism to perturb the results [50]. More details about the noise is covered in Section 6.1 and 6.3.

## 6.1 Privacy Parameters

This section discusses the trade-off between privacy budget ($\varepsilon$) and noise.

### 6.1.1 Privacy Budget

The most significant distance between a query on a dataset $D$ and the identical query on a neighboring dataset $D^{'}$ is denoted by the privacy budget. An alternative name for this metric is the "Epsilon" ($\varepsilon$).

A *small* privacy budget value means that an adversary cannot distinguish individuals based on the outputs of the mechanism $(M)$ on the dataset $D$ and the neighboring dataset $D'$.

On the contrary, when the privacy budget is a significant value, the ratio of the probabilities (see Equation 6.2) of observing the output of the mechanism $(M)$ on the dataset $D$ and $D^{'}$ is large. An adversary observing the output can distinguish whether an individual belongs in the dataset or not when the privacy budget is a significant value.

In order to maintain high levels of privacy, small values of privacy budget guarantee similar outputs for comparable inputs; on the other hand, significant values of privacy budget permit less similar outputs and hence less privacy.

For example, a count query on a database counts the number of individuals with an average glucose level higher than $160$. As a result, the count query returns a value of $10,000$. To make this mechanism of counting queries differentially private, we add noise. Assuming the sensitivity of the count query is 1, privacy budget is 0.1, and the noise is sampled from a Laplace distribution centered at 0 with a scale of 10. The output of the count query is $10021$, and the query results are different each run, but they are still relatively close to the actual outcome $(10,000)$.

### 6.1.2 Privacy Budget and Noise trade-off

Random noise in the mechanism's output efficiently means attaining differential privacy. The significant difficulty is to add enough noise to fulfill the criteria of Differential Privacy, but not so much that the response becomes too noisy to be of value.

If the privacy budget is increased from $0.1$ to $15$ from the previous example in Section 6.1.1, then the output of the differentially private count query is $10,000.04$, which is very close to the actual outcome. This means that with the increase in privacy budget, the noise value reduces, and the output is very close to the actual outcome.

### 6.1.3 Sensitivity

The degree to which a function's output shifts in response to a change in that function's input is referred to as the function's sensitivity $(\Delta_f)$. Sensitivity is a complicated subject that is an essential component in estimating the noise to accomplish Differential Privacy [50]. The sensitivity of a function should always be a bounded value, and in the case of unbounded value, clipping helps to bound the value of the function. Section 6.3.3 covers clipping in the context of neural networks.

## 6.2 Properties

This section analyzes the properties of differentially private mechanisms, which emerge from the notion of Differential Privacy [46].

- *Sequential composition* is a property of Differential Privacy that limits the total privacy cost incurred by disclosing numerous outcomes of differentially private mechanisms applied to the same input data [51]. If a mechanism $(F_1(D))$ is $\varepsilon_1$-differentially private and another mechanism $(F_2(D))$ is $\varepsilon_2$-differentially private, the combined mechanism is $(\varepsilon_1 + \varepsilon_2)$-differentially private. Adding up of the privacy budgets $(\varepsilon_1 + \varepsilon_2)$ enables to build more complex differentially private algorithms.

- *Post-processing* computation, states that differentially private computations may be released with confidence since any post computation will likewise be differentially private, as long as the privacy protection given by the differentially private mechanism is maintained [46]. Hence, it is perfectly safe to execute arbitrary calculations on the output of the differentially private mechanism.

## 6.3 Mechanism

This section discusses the different mechanisms to achieve Differential Privacy [46].

### 6.3.1 Laplace

Statistical noise distributed from the Laplace distribution is used to perturb the results of a numeric query to achieve $\varepsilon$-Differential Privacy [50].

**Definition 1 (Laplace Distribution)** *The Laplace distribution (centered at 0) with scale b is the probability distribution with probability density function:*

$$\text{lap}(D|b) = \frac{1}{2b}\exp\left(-\frac{|D|}{b}\right) \tag{6.3}$$

Noise to be added is given by the parameter ($b$), which is to be calibrated based on the privacy budget ($\varepsilon$) and the global sensitivity ($\Delta_1 f$).

**Definition 2 ($L1$ Global Sensitivity)** *The $L1$ global sensitivity ($\Delta_1 f$) of a function $f$ for two neighboring dataset $D$ and $D^{'}$ is:*

$$\Delta_1 f = \max\left\{\left\|f(D) - f(D^{'})\right\|_1\right\} \tag{6.4}$$

Differential Privacy is obtained by adding noise selected from the Laplace distribution. Each time a value has to be made private, a different quantity of noise is taken from the distribution and added to the output of the function ($f$). Definition 3 combines all the above definition, to make a mechanism differentially private. $Y$ is a random variable drawn from the Laplace distribution with a scale (b) given by $\frac{\Delta_1 f}{\varepsilon}$.

**Definition 3 (Laplace Mechanism)** *Given any function $f$, the Laplace mechanism is defined as:*

$$M(D, f(\cdot), \varepsilon) = f(D) + Y \tag{6.5}$$

### 6.3.2 Gaussian

The Gaussian mechanism is similar to the Laplace mechanism. However, the difference is that it achieves $(\varepsilon, \delta)$-Differential Privacy by adding Gaussian noise [47]. The other difference is that the Laplace mechanism uses $L1$ global sensitivity, whereas the Gaussian mechanism can utilize both $L1$ and $L2$ global sensitivity.

**Definition 4 (Approximate Differential Privacy)** *Approximate Differential Privacy for two neighboring dataset $D$ and $D^{'}$ is given by:*

$$\Pr[M(D) \in S] \leq e^{\varepsilon} \Pr[M(D^{'}) \in S] + \delta \tag{6.6}$$

The parameter $\delta$ denotes failure probability. With a probability of $1-\delta$, we get a guarantee that the mechanism is differentially private. Approximate Differential Privacy satisfies the *sequential composition* and *post-processing* properties. If a mechanism $(F_1(D))$ is $(\varepsilon_1, \delta_1)$-differentially private and another mechanism $(F_2(D))$ is $(\varepsilon_2, \delta_2)$-differentially private, the combined mechanism is $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$-differentially private thus allowing to build complex algorithms [46]. In order to make a mechanism differentially private, Gaussian noise with mean $0$ and variance $(\sigma^2)$ is added to the output of the function $(f)$. The variance $(\sigma^2)$ is calibrated using $L2$ sensitivity $(\frac{\Delta_2 f}{\varepsilon^2})$.

**Definition 5 (Gaussian Mechanism)** *Given any function $f$, the Gaussian mechanism is defined as:*

$$M(D, f(\cdot), \varepsilon, \delta) = f(D) + \mathcal{N}(\sigma^2), \text{ where } \sigma^2 = \frac{2\Delta_2 f \log(\frac{1.25}{\delta})}{\varepsilon^2} \tag{6.7}$$

### 6.3.3 Differential Privacy in Neural Networks

*Abadi et al.* [52] proposed a differential private stochastic gradient descent (DP-SGD), Where the noise sampled from Gaussian distribution is added to the computed gradient in each iteration and then back propagated to optimize the weights. Adding noise prevents an adversarial from reconstructing data from the weights [53]. Furthermore, preserving the privacy of training data.

Algorithm 1 represents the entire process to make a training process differentially private [52].

**Algorithm 1** An algorithm for differentially private stochastic gradient

```
Require: $\left\{(x,y)\cdots(x,y)_N\right\}$, samples. $c_g$, clipping for gradient. $\sigma^2$, noise scale. $T$, number of epochs
  for training. $L$, loss function. $\eta$, learning rate.
  Initialize $\theta_0$ randomly.
  for i = 1,...,$T$ do
      Take a mini-batch of random samples
      Compute Gradient, $g_i(x) \leftarrow \nabla_i L(\theta_i, y)$
      Clip Gradient, $\overline{g}_i(x) \leftarrow g_i(x) / \max(1, \frac{\|g_i(x)\|_2}{c_g})$
      Add Noise, $\tilde{g}_i(x) \leftarrow \overline{g}_i(x) + \mathcal{N}(0, \sigma^2 c_g^2 I)$
      Gradient Descent, $\theta_{i+1} \leftarrow \theta_i - \eta \tilde{g}_i(x)$
  end for
```

#### Compute Gradient

A neural network comprises of interconnected neurons, and the weight assigned to each link in a neural network is distinct. The neuron's activation function defines the output of a neuron. The initial forward propagation phase begins when the network is exposed to training data. Next, a loss function estimates the loss (or error) and compares the prediction result to the actual prediction. Finally, gradient descent allows the weights to be changed incrementally by computing the loss function's derivative (or gradient).

#### Clip Gradient

Due to non-linear activation functions, it is needed to bound the gradient to achieve Differential Privacy. The gradient vector is clipped using a threshold $c_g$ (see Figure 6.1). The clipping constant ensures that if the gradient is greater than the clipping constant, then it is scaled down to be the norm of $c_g$, and if smaller than the clipping constant, then the gradients are not clipped [52].

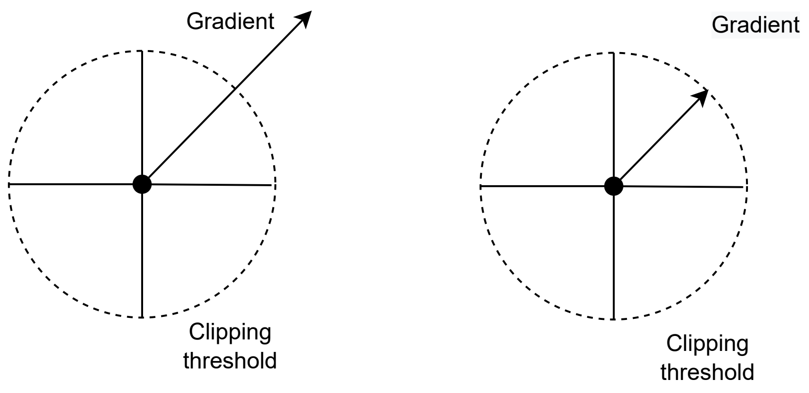


**Figure 6.1:** Gradient Clipping

### Noise

If the Gaussian noise with variance $\sigma^2$ is chosen from equation 6.7, Algorithm 1 is $(\varepsilon, \delta)$-Differentially private. For a batch-size of $L$, size of the dataset $N$ and sampling probability denoted by $q = \frac{L}{N}$, Algorithm 1 satisfies $(O(q\varepsilon), q\delta)$-differentially private using the strong composition theorem [47].

*Abadi et al.* [52] proposed the moments accountant method which proves that the Algorithm 1 is $(O(q\varepsilon\sqrt{T}), \delta)$-differentially private and provides much tighter bound and smaller privacy budget $(\varepsilon)$ compared to the strong composition theorem. The theorem for the moments' accountant method is given below.

**Theorem 1** *There exist constants $c_1$ and $c_2$ so that given the sampling probability $q = L/N$ and the number of steps $T$, for any $\varepsilon < c_1 q^2 T$, Algorithm 1 is $(\varepsilon, \delta)$-differentially private for any $\delta > 0$ if we choose noise such as:*

$$\sigma \geq c_2 \frac{q\sqrt{T \log(1/\delta)}}{\varepsilon} \tag{6.8}$$

## 6.4 Synthetic Data

Medicine and health informatics are two fields where it is hard to get as much data as other fields [54]. The collection process is both costly and time demanding, restricting the quantity of data that can be collected. As a result, the topic of constructing machine learning models for medical analytics continues to be particularly hard. The Generative Adversarial Network, or GAN, and its variants have shown excellent performance when modeling the underlying data distribution and producing high-quality samples [55]. However, the GANs may still reveal confidential information about the training sample. It is envisioned that training GAN with the principles of Differential Privacy, particularly differentially private stochastic gradient descent, would preserve the privacy of individuals against inference attacks [56].

GAN train two models simultaneously: a generator model (G) that creates fake samples and a discriminator model (D) that differentiates the samples created by the generator as real or fake. Equation 6.9 is the loss function to train the generator and the discriminator [57]. The generator aims to minimize the distance between the generated and original data (real), whereas the discriminator aims to maximize it. In Equation 6.9, $P_{real(x)}$ is the original data distribution, $P_{fake(z)}$ is the input noise distribution.

$$\min_{G} \max_{D} = E_{x \sim P_{real(x)}}[\log(D(x))] + E_{z \sim P_{fake(z)}}[\log(1 - D(G(z)))] \tag{6.9}$$

Figure 6.2 is the pictorial representation of the differentially private GAN. The generator creates fake data from input random noise, and the discriminator is trained on both the fake data and original data. The training of the discriminator is done by the techniques mentioned in section 6.3.3 thus making it differentially private. The generator parameters can also guarantee Differential Privacy because of the *post-processing* property (see section 6.2). It is safe to produce data once the training phase has been completed since the generator's settings ensure Differential Privacy. In short, we have a discriminator that is differentially private and makes the generator differentially private.

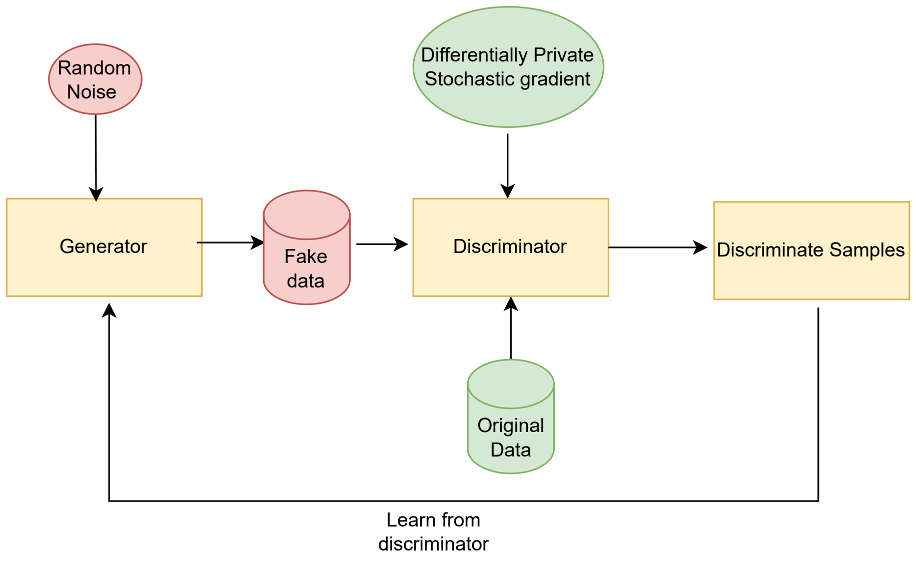


**Figure 6.2:** DP-GAN

A practical example of how differentially private synthetic data can hide information, *Xie et al.* [56] used Differential Privacy techniques along with GAN to generate synthetic data for the electronic health records, and the synthetic data safeguarded the count of the rareness of illnesses. This exemplifies how differentially private synthetic data may be used to safeguard information. Assuming that there is public-available data of a specific population, an adversary may derive statistical information and use it to their advantage. However, because Differential Privacy has the potential to alter statistical information, the adversary is unable to obtain the correct information.

The idea of differentially private synthetic data is utilized in this research to generate counterfactual explanations to hide statistical information and preserve privacy.

# Chapter 7

# Methodology

This chapter describes the data used for the experiments, the employed models, the privacy-preserving framework, the metrics used for assessment, and lastly, how the experiments were setup.

## 7.1 Data

The dataset used for the experiments is *Stroke Prediction Dataset*. The dataset is available freely in Kaggle [58]. Figure 7.1 describes the categorical features and Figure 7.2 describes the continuous and discrete features in the dataset.

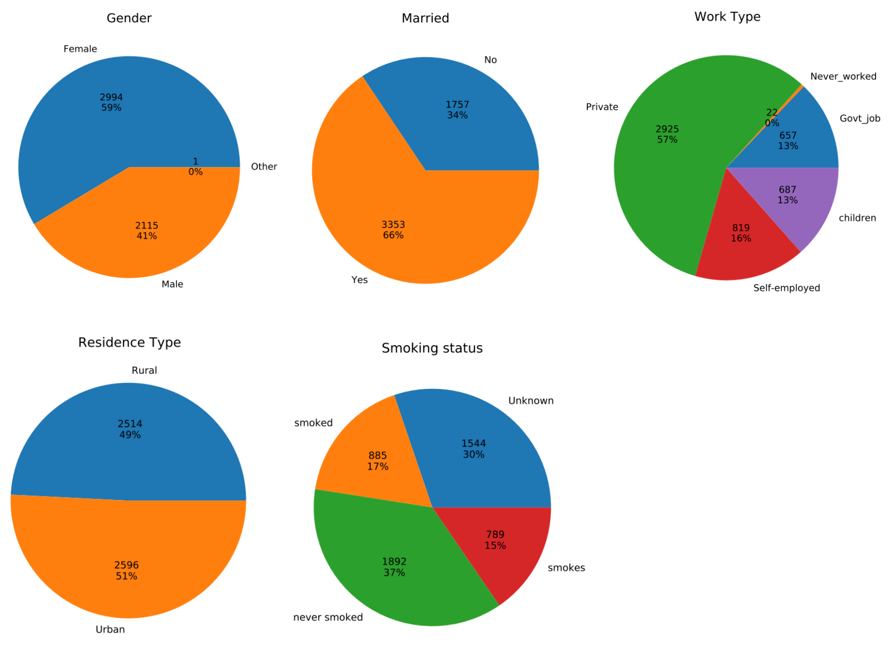


**Figure 7.1:** Categorical features

The *Stroke Prediction Dataset* consists of $5110$ rows and $11$ columns. Modifiable factors include *work type*, *residence type*, *smoking status*, *average glucose level*, *body mass index*, *hypertension*, and *heart disease*. Nonmodifiable factors include *age*, *gender*, and *marital status* (Refer to Appendix A.1). The target feature (outcome) is *stroke*. Out of $5110$ records, $4861$ records have the outcome *No*

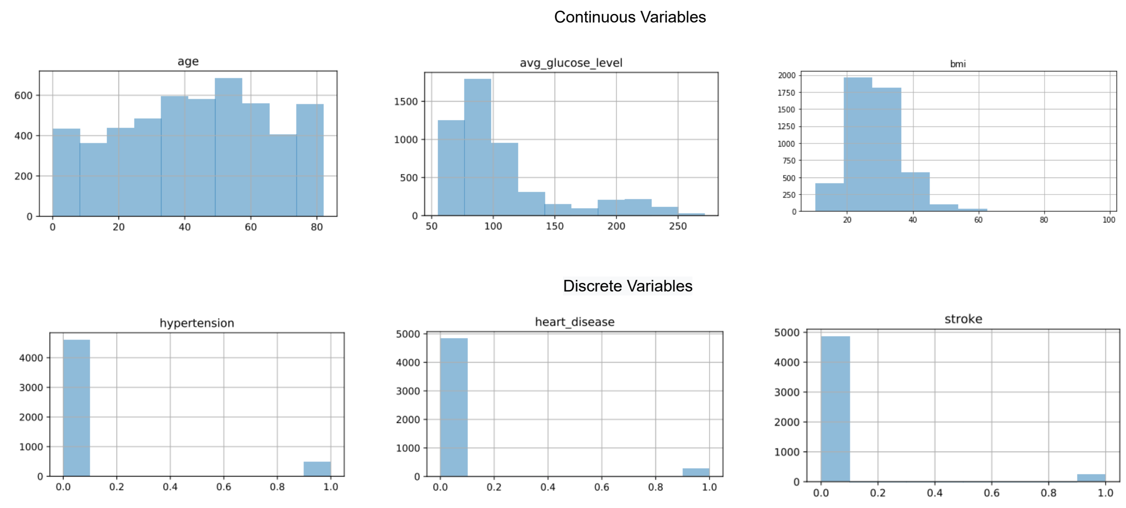


**Figure 7.2:** Continuous and Discrete features

*Stroke*, which is equivalent to $95$ percent, and only $249$ records have the outcome *Stroke*. Due to the class imbalance, data processing is necessary which is covered in the next section.

### 7.1.1 Data Processing

Categorical features, *work type*, *residence type*, *smoking status*, *gender*, and *marital status* are one-hot encoded [59]. The total features after one-hot encoding are $21$. To oversample the target feature (*Stroke*), minority random oversampling was used [59]. After the random oversampling, the total size of the dataset is $9722$. The features are reduced using the correlation coefficient with the target feature (*Stroke*) to choose the best-correlated features [60], which results in $12$ columns. The *Stroke prediction Dataset* is also normalized using min-max scaling; the reason behind the scaling is covered in section 7.2.3. The processed data is split into train and test using the $80/20$ ratio. The training data is referred to as original data in the following sections.

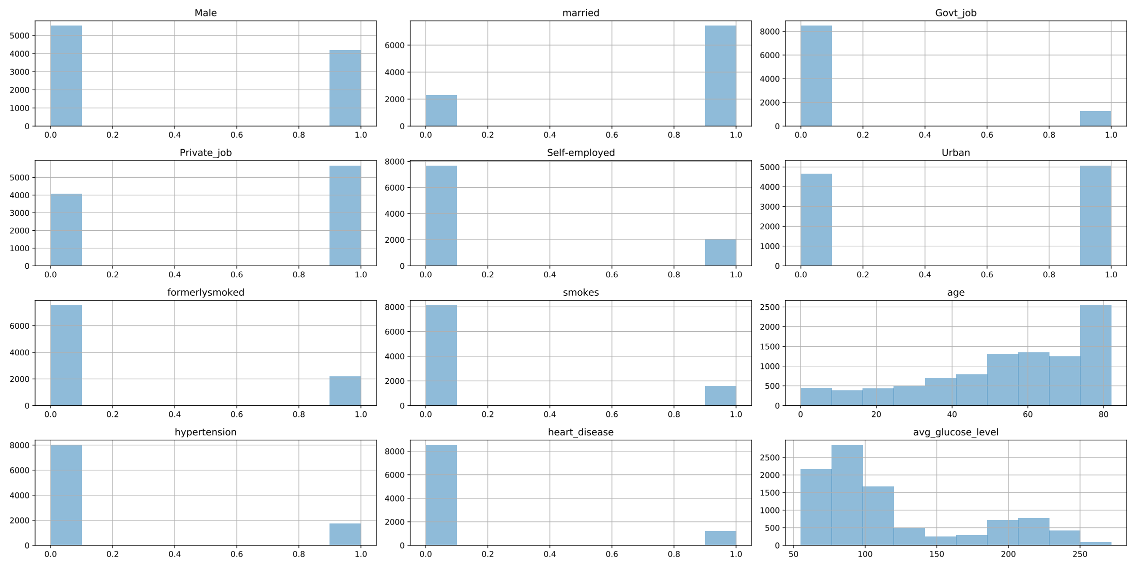


**Figure 7.3:** Continuous and Discrete variables after data processing

## 7.2 Models

This section describes the models used in the study, beginning with the model that was used to generate diverse and multiple counterfactual explanations, following that, the attack model that leverages counterfactual explanations to perform a membership inference attack; following that, the synthetic data model and then the data compression model to visualize the synthetic data.

### 7.2.1 Counterfactual Explanation Model

DiCE algorithm was used (Refer to section 3.2.2) to generate counterfactual instances, which form the basis of the explanations of *Stroke* prediction decision of the black-box model. The architecture of the black-box model can be found in Appendix A.2.2. DiCE is available as a free and open-source package [61]. The objective function of DiCE comprises several distinct parameters, the most important of which are *proximity* and *diversity*. When it comes to generating counterfactual explanations, the existing architecture utilizes the original data for generating the counterfactual explanations through random sampling, and the end-user determines the required levels of *proximity* and *diversity* and the number of counterfactuals to be generated. The counterfactual explanations generated for this research only change the modifiable features (Refer to Appendix A.1 to understand modifiable features) and use synthetic data in contrast to the original data. The algorithm and an example for DiCE can be found in Appendix A.2.3.

### 7.2.2 Membership Inference Attack Model

The membership inference attack suggested by *Shokri et al.* [19] depends on the construction of a shadow dataset (Refer to section 4.1) that is similar to the original data. In this study, the shadow dataset is made of counterfactual explanations produced by simple API queries.

The membership inference attack for this study is broken down into two parts: First, an adversary attacker accesses the explanation and prediction API with an instance and, as a result, obtains the prediction (Stroke Prediction) and the multiple counterfactual explanations. Next, the adversarial attacker uses the explanation to build a shadow dataset ($D^{\text{trainshadow}}$) and then builds a disjoint dataset ($D^{\text{testshadow}}$) from $D^{\text{trainshadow}}$ such that $D^{trainshadow} \cap D^{testshadow} = \emptyset$. All the records in $D^{\text{trainshadow}}$ are labeled $1$ based on the hypothesis of the adversary attacker that all the records would have been a part of the training of the target/victim model (black-box model); on the other hand, records in $D^{\text{testshadow}}$ is labeled as $0$, assuming the data did not belong in the training.

The second part involves training the attack model. The attack model is simply a supervised classification model that predicts, given an instance, whether it belongs in the training of the black-box model or not. The attack model is a Multi-layer perceptron built using the Keras library [62] [63], and consists of $6$ hidden layers trained with Rectified Linear Unit (ReLU) as the activation function, and the final activation function used is Softmax [64]. Negative log-likelihood as the loss function and root mean square propagation (RMSProp) with learning rate $0.0001$ was used to train the attack model [64]. The architecture of the membership inference attack can be found in Appendix A.2.4. Figure 7.4 is a pictorial representation of the membership inference attack.

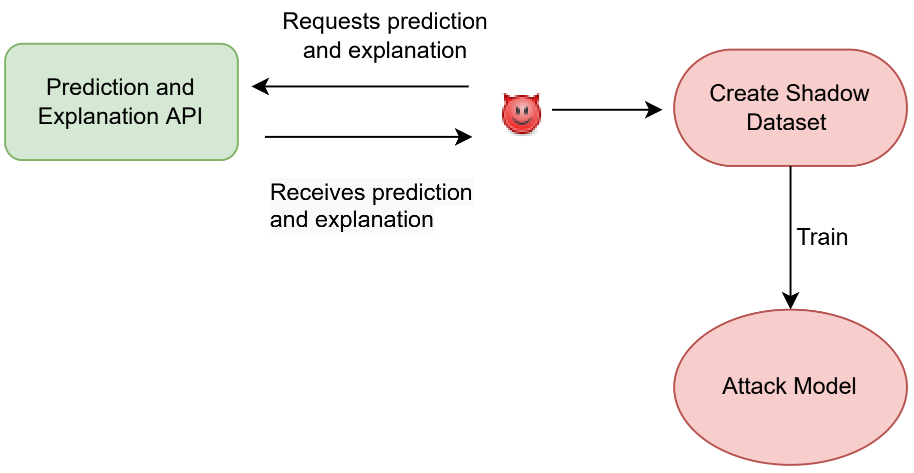

Requests prediction and explanation
Prediction and Explanation API
Create Shadow Dataset
Receives prediction and explanation
Train
Attack Model

**Figure 7.4:** Membership Inference Attack Model

### 7.2.3 Synthetic Data Model

Three algorithms were benchmarked for the synthetic data generation using Differential Privacy (differentially private stochastic gradient descent). The explanation for generating data can be read in section 6.4. The general GAN architecture for all the models can be found in Appendix A.2.5. The synthetic Data models are built using the Keras library and differentially private stochastic gradient descent (DP-SGD) in the TensorFlow privacy package [65].

#### Differentially Private GAN (DP-GAN)

Algorithm 2 explains the process of training a GAN using differentially private stochastic gradient descent. For a given number of iterations ($T$), the discriminator is trained with a batch of original data ($E_{x \sim P_{real(x)}}[\log(D(x))]$) and the fake data from the generator ($E_{z \sim P_{fake(z)}}[\log(1 - D(G(z)))]$). After completing the discriminator's training, the weights are fixed, and the generator will next undergo training by utilizing the discriminator. After $T$ iterations, the generator attempts to deceive the discriminator by producing samples comparable to the original data.

#### Differentially Private Wasserstein GAN (DP-WGAN)

Algorithm 3 explains the process of training a Wasserstein GAN using differentially private stochastic gradient descent [56]. Algorithm 3 differs from Algorithm 2 in the concept of clipping, using Wasserstein distance as the loss function and the training iteration of the discriminator. Instead of clipping the gradients, weights involved in each neuron are clipped [56]. Training iterations for the discriminator are greater than the generator and are denoted by $n_d$.

#### Differentially Private Conditional Tabular GAN (DP-CTGAN)

Algorithm 4 explains the process of training a Conditional GAN using differentially private stochastic gradient descent [66]. The loss function 6.9 is translated into a conditional probability. The discriminator is now trained with a batch of original data given a label ($E_{x \sim P_{real(x)}}[\log(D(x|y))]$) and the fake data points from the generator given a label ($E_{z \sim P_{fake(z)}}[\log(1 - D(G(z|y)))]$) [67]. The label is the output class which is either *Stroke* denoted by $1$ or *No Stroke* denoted by $0$.

**Algorithm 2** An algorithm for differentially private GAN

```
Require: $\eta_g$, learning rate for generator. $\eta_d$, learning rate for discriminator. $c_g$, gradient clipping
    constant. $T$, number of epochs for training both the discriminator and generator. $b$ batch size for
    training the discriminator. $l$, input dimension size for the generator. $\sigma^2$, noise scale.
Ensure: Differentially private discriminator.
 1: Initialize descriminator parameter $\theta_0$ randomly.
 2: for i = 1, . . . ,$T$ do
 3:     Sample $\{x^{(i)}\}_{i=1}^{b}$ a batch of original data points.
 4:     Sample $\{z^{(i)}\}_{i=1}^{l}$ a batch of fake data points from the generator
 5:     $x_{real}, y_{real} = p_{real}(x)$, where $y_{real}$ is labeled as $1$ to indicate original.
 6:     $z_{fake}, y_{fake} = p_{fake}(z)$, where $y_{fake}$ is labeled as $0$ to indicate fake.
 7:     $d_{train}$, Merge the above samples .
 8:     For each ($x$,$y$) in ($d_{train}$), Compute Gradient, $g_i$(x) $\leftarrow \nabla_i L(\theta_i, y)$
 9:     Clip Gradient, $\overline{g}_i(x) \leftarrow g_i$(x) / $\max(1, \frac{\|g_i(x)\|_2}{c_g})$
10:     Add Noise, $\tilde{g}_i(x) \leftarrow \overline{g}_i(x) + \mathcal{N}(0, \sigma^2 c_g^2 I)$
11:     Gradient Descent, $\theta_{i+1} \leftarrow \theta_i - \eta_d \tilde{g}_i(x)$
12:     Make the weights of the discriminator not trainable after the end of batch.
13:     Train the generator by using the discriminator.
14: end for
```

**Algorithm 3** An algorithm for differentially private WGAN

```
Require: $\eta_g$, learning rate for generator. $\eta_d$, learning rate for discriminator/ critic. $c_p$, weight clipping
    constant. $T$, number of epochs for training the generator. $n_d$, number of epochs for training the
    discriminator. $b$ batch size for training the discriminator. $l$, input dimension size for the generator.
    $\sigma^2$, noise scale.
Ensure: Differentially private discriminator.
 1: Initialize descriminator parameter $\theta_0$ randomly.
 2: for i = 1, . . . ,$T$ do
 3:     for i = 1, . . . ,$n_d$ do
 4:         Sample $\{x^{(i)}\}_{i=1}^{b}$ a batch of original data points.
 5:         Sample $\{z^{(i)}\}_{i=1}^{l}$ a batch of fake data points from the generator
 6:         $x_{real}, y_{real} = p_{real}(x)$, where $y_{real}$ is labeled as $-1$.
 7:         $x_{fake}, y_{fake} = p_{fake}(x)$, where $y_{fake}$ is labeled as $1$.
 8:         $d_{train}$, Merge the above samples .
 9:         For each ($x$,$y$) in ($d_{train}$), Compute Gradient, $g_i$(x) $\leftarrow \nabla_i L(\theta_i, y)$
10:         Add Noise, $\tilde{g}_i(x) \leftarrow \overline{g}_i(x) + \mathcal{N}(0, \sigma^2 \mathbf{I})$
11:         Gradient Descent, $\theta_{i+1} \leftarrow \theta_i - \eta_d \tilde{g}_i(x)$
12:         Clip Weights, $\theta_{i+1} \leftarrow$ clip($\theta_{i+1}$,$-c_p$,$c_p$)
13:     end for
14:     Make the weights of the discriminator not trainable after the end of batch.
15:     Train the generator by using the discriminator.
16: end for
```

**Algorithm 4** An algorithm for differentially private CTGAN

```
Require: $\eta_g$, learning rate for generator. $\eta_d$, learning rate for discriminator. $T$, number of epochs
    for training the generator and discriminator. $b$ batch size for training the discriminator. $l$, input
    dimension size for the generator. $\sigma^2$, noise scale. $c_g$, gradient clipping.
Ensure: Differentially private discriminator.
 1: Initialize descriminator parameter $\theta_0$ randomly.
 2: for i = 1,...,$T$ do
 3:     Sample $\{x^{(i)}, label^{(i)}\}_{i=1}^{b}$ a batch of original data points.
 4:     Sample $\{z^{(i)}, label^{(i)}\}_{i=1}^{l}$ a batch of fake data points from the generator
 5:     $[x_{real}, label], y_{real} = p_{real}(x)$, where $y_{real}$ is labeled as $1$.
 6:     $[x_{fake}, label], y_{fake} = p_{fake}(x)$, where $y_{fake}$ is labeled as $0$.
 7:     $d_{train}$, Merge the above samples .
 8:     For each ($x$,$y$) in ($d_{train}$), Compute Gradient, $g_i$(x) $\leftarrow \nabla_i L(\theta_i, y)$
 9:     Clip Gradient, $\overline{g}_i(x) \leftarrow g_i$(x) / $\max(1, \frac{\|g_i(x)\|_2}{c_g})$
10:     Add Noise, $\tilde{g}_i(x) \leftarrow \overline{g}_i(x) + \mathcal{N}(0, \sigma^2 c_g^2 I)$
11:     Gradient Descent, $\theta_{i+1} \leftarrow \theta_i - \eta_d \tilde{g}_i(x)$
12:     Make the weights of the discriminator not trainable after the end of batch.
13:     Train the generator by using the discriminator.
14: end for
```

### Considerations for Synthetic Data Model

*Stroke Prediction Dataset* was processed using min-max normalization to get feature values with a fixed range between $-1$ and $1$ [59]. It is a necessary process to improve the stability and performance of the synthetic data models. Consequently, the explanations would also be on a similar scale. Therefore, using the correct scaling factor to do an inverse transformation on the findings would be one technique to get the original values. This is plausible, but it adds a degree of ambiguity and complication. Thus it is neglected in this study. A disadvantage of synthetic data models is that the discrete features were transformed into continuous features; therefore, a threshold to convert from continuous to the discrete domain is needed to resolve this issue [56]. In this work, the threshold was set as $0.5$, similar to the work by *Xie et al.* [56].

### Privacy Parameters for Synthetic Data Model

The quantity of noise and the privacy budget are the two most essential criteria to fine-tune while making adjustments. Unfortunately, no simple answer can be provided to this problem. In order to accomplish this goal, one would need to conduct an exhaustive examination of the domain and the level of compliance possessed by the data, which is beyond the ambit of this work. The privacy budget was calculated using the TensorFlow Responsible AI toolkit [65]. The overall privacy budget ($\varepsilon$) was computed from the noise, training iterations, sampling ratio, and failure probability ($\delta$).

The clipping threshold for this study is set based on the advice from *Abadi et al.* [52]. The more significant the clipping value, the less the exposure of the gradients to the generator, and the data quality deteriorates. For the failure probability ($\delta$), the rule of thumb is to set $\delta$ as the inverse of the dataset size [68].

### 7.2.4 Data Compression Model

A data compression model was used to compress the original and synthetic data, followed by a principal component analysis for dimensionality reduction to visualize the quality of the synthetic data generated [69]. Figure 7.5 is the pictorial representation of the data compression model.

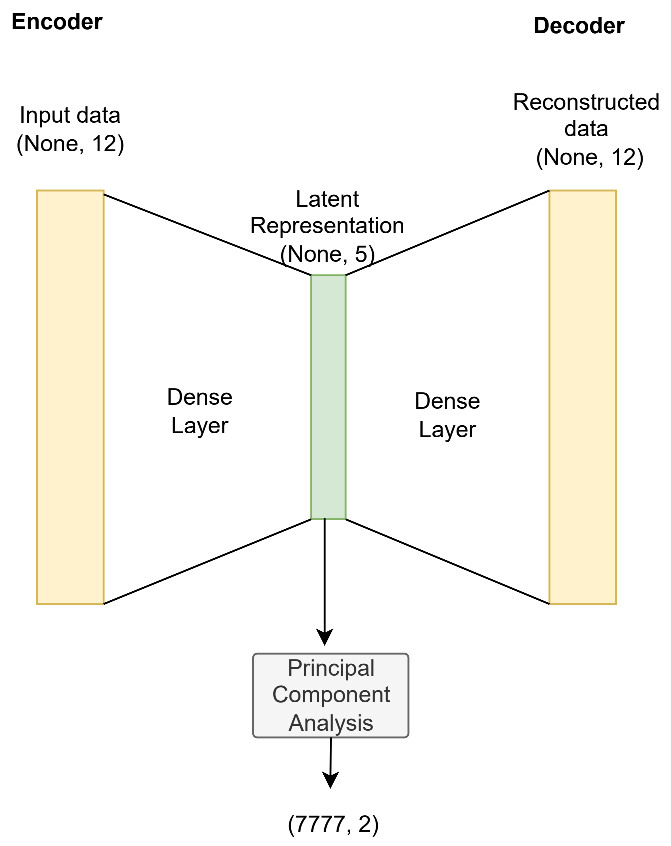

Encoder
Decoder
Input data (None, 12)
Reconstructed data (None, 12)
Latent Representation (None, 5)
Dense Layer
Dense Layer
Principal Component Analysis
(7777, 2)

**Figure 7.5:** Data Compression Model

The architecture of the data compression model are as follows,

- The encoder is a single dense layer with ReLU as the activation function. The dense layer takes the number of features as the input (None, $12$) and compresses data to give $($None, $5)$ dimension as the latent representation.
- The decoder is a single dense layer with ReLU as the activation function that reconstructs the data from the dimension $($None, $5)$ back to $($None, $12)$.
- Encoder and decoder are trained together such that the output of the encoder is the input to the decoder [70]. The combined model of the encoder and decoder is trained using the stochastic gradient descent as the optimizer, with a learning rate of $0.0001$ and reconstruction loss (Mean Squared Error) [71]. Refer to section A.2.1 to understand reconstruction loss.
- The latent representation from the encoder for all the records in the original data $(7777, 5)$ is fed as the input for the principal component analysis (PCA) to reduce further the dimension of the latent representation to $(7777, 2)$.
- Synthetic data generated in the study were first transformed using the built encoder and further compressed using the built PCA model.

The goal of the data compression model was to train the encoder to reduce the dimension of the data and then perform PCA on the latent representation; therefore, the reconstruction loss was not effectively minimized. After applying the PCA, the first and the second principal components are used to visualize the quality of the synthetic data from the synthetic data model. Details about the principal component analysis and also how to interpret the plot with principal components can be found in Appendix A.2.7.

## 7.3 Privacy-Preserving Framework

The centralized architecture for this study consists of a data owner, model owner, explanation provider, and explanation recipient who requests prediction and explanation through the API.

#### Data Owner

The data owner is responsible for three roles: possession, responsibility, and execution [72]. The idea of possession refers to having legal and physical custody of the data, whether stored on the cloud or locally. Accountability means the data owner is held accountable and responsible for the quality of the data and the upkeep of privacy. Last, execution denotes that the data owner authorizes or extends ownership rights to parties inside the trusted environment so that those parties may act on the data. Figure 7.6 illustrates the framework followed in this research. The two roles of the data owner are as follows,

1. The data owner extends the ownership rights to the model owner to build a model (black-box) on the *Stroke Prediction Dataset*. The black-box model predicts the probability of having a *Stroke* or *No Stroke* for the given modifiable and nonmodifiable features.
2. The data owner is also responsible for creating differentially private synthetic data to generate counterfactual explanations.

#### Model Owner

The model owner trains a black-box model on the data provided by the data owner. The *Stroke Prediction Dataset* consists of input features and output class, either *Stroke* or *No Stroke*. The objective of the model owner is to find a function that maps the input and the output class label [73].

#### Explanation Provider

The roles of the explanations provider are as follows,

3. The explanation provider creates a counterfactual explanation model using the prediction model provided by the model owner.
4. The synthetic data provided by the data owner is also used to generate counterfactual explanations.

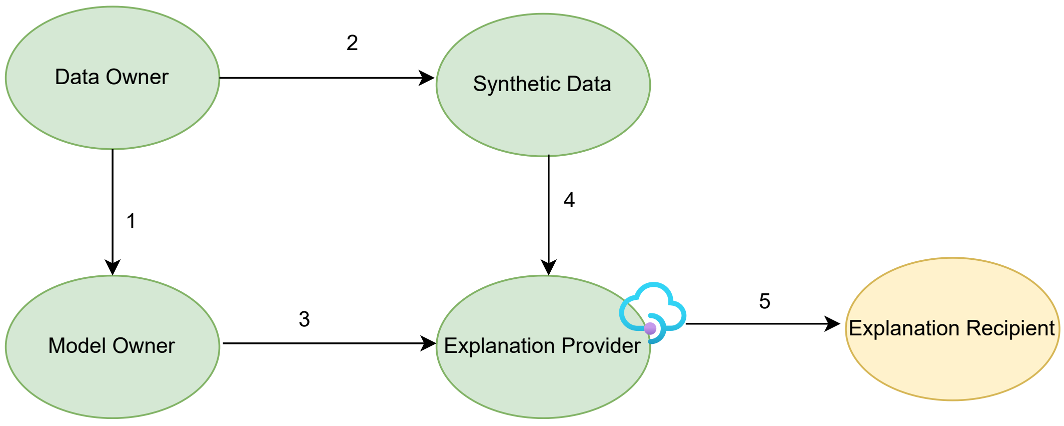


**Figure 7.6:** Framework for Privacy-Preserving Counterfactual Explanations

## 7.4 Metrics

In this section, the utility and the privacy metrics are discussed. Utility metrics will assist in differentiating the quality of the counterfactual explanation derived from the synthetic data and the original data. Privacy metrics will assist in analyzing the quantity of information that has been disclosed to the adversary due to the excessive querying of counterfactual explanations.

### 7.4.1 Utility Metrics

The utility measures have been extracted from the DiCE research paper [17]. The metrics are similar to the objective function in Equation 3.3 except for sparsity.

#### Proximity

Proximity measures how close the query instance (x) is to the generated counterfactuals. $k$ denotes the number of counterfactuals generated. dist$(c_i, x)$ denotes a distance function which can be either a $L_1$ (Manhattan) or $L_2$ (Euclidean) distance metric [74].

$$\text{Proximity} = \sum_{i=1}^{k} \text{dist}(c_i, x) \tag{7.1}$$

#### Sparsity

Sparsity counts the total number of changes done to the features. $d$ denotes the number of features. Sparsity metrics compare the feature value of the query instance (x) and counterfactual instance (c); if they are not equal, the distance score is added by $1$.

$$\text{Sparsity} = \sum_{i=1}^{k} \sum_{j=1}^{d} 1_{[c_i^j \neq x_i^j]} \tag{7.2}$$

#### Diversity

Diversity measures how different are the generated counterfactuals when compared with each other. $k$ denotes the number of counterfactuals.

$$\text{Diversity} = \sum_{i=1}^{k-1} \sum_{j=1}^{k} \text{dist}(c_i, c_j) \tag{7.3}$$

### 7.4.2 Privacy Metrics

#### Relative Entropy

When two probability distributions are compared, relative entropy is used to determine how far apart they are from each other [75]. Information gain is another name for this statistic. $X$ denotes the original (true) distribution, and $X^*$ denotes the adversary's estimate. Relative entropy measures in bits because of the use of $\log_2$. A smaller value means more information is leaked and vice-versa. For example, a password known by all participants results in zero entropy.

$$priv_{RLE} \equiv D_{\text{KL}}(X||X^*) = \sum_{x,x^*} p(x^*) \log_2 \frac{p(x*)}{q(x)} \tag{7.4}$$

#### Accuracy of the Membership Inference Attack

The accuracy of the membership inference attack model explained in section 7.2.2 was benchmarked before and after applying Differential Privacy. The total ratio of accurate predictions to misclassifications on the test data is used to measure the accuracy. True positive ($TP$) and True Negative ($TN$) denote accurate predictions. False positive ($FP$) and False negative ($FN$) denote misclassification.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.5}$$

## 7.5 Experimental Setup

This section discusses how the experiments were setup. Refer to section A.2 to understand the used neural network layer, optimizer, and loss function.

### 7.5.1 Effect of Low Proximity and High Diversity

#### Problem

*To study the impact of proximity low and diversity high and multiple counterfactual explanations.*

*Aïvodji et al.* [1] developed a counterfactual explanation-based model extraction attack (Refer to section 4.4) by constructing shadow datasets using counterfactual explanations. The model extraction attack has a recommended setting that recommends making the *proximity* and *diversity* parameters high while generating counterfactual explanations. In contrast to what was suggested by *Aïvodji et al.* [1], *proximity* and *diversity* were kept at low and high levels, respectively, throughout the investigation. This decision is based on the intuition that the shadow dataset generated by simply querying with *proximity* low and *diversity* high will provide a significant approximation of the original data distribution.

#### Setup

A black-box model (Stroke Prediction) trained on the original data predicts the probability of *Stroke* or *No Stroke* given the modifiable and nonmodifiable features. The black-box model is a Multi-layer perceptron [63] with $6$ hidden dense layers having ReLU as the activation function and the final output layer with a Softmax activation function trained for $100$ epochs with Negative log-likelihood as the loss function, and stochastic gradient descent with a learning rate of $0.0001$ as the optimizer. The architecture of the black-box model can be found in Appendix A.2.2. The existing DiCE framework was used to generate explanations for the outcomes predicted (decision) by the black-box model. The default value of *proximity* and *diversity* in the DiCE framework is $0.5$ and $1.0$, respectively; however, how far the values for *proximity* and *diversity* can be set from the default value is not explained in the research of DiCE, Hence, in this study, an iterator for the values of *proximity* $(1$ to $10)$ and *diversity* (*proximity value* $+2)$ was set, and for the *proximity* and *diversity* values of $3$ and $5$, respectively, the optimization algorithm for the DiCE worked well.

The query approach for generating multiple counterfactuals, which serve as the foundation for the shadow dataset, is shown in Table 7.1. For simplicity, $40$ instances from the test data are used to query.

| **Surrogate Model** | **Queries** | **CF generated in 1 query** | **Size of Shadow Dataset** |
|---|---|---|---|
| 1 | 20 | 100 | 2000 |
| 2 | 20 | 50 | 1000 |
| 3 | 20 | 20 | 400 |
| 4 | 40 | 100 | 4000 |
| 5 | 40 | 50 | 2000 |
| 6 | 40 | 20 | 800 |

**Table 7.1:** Query Strategy used to generate counterfactual explanations. The accuracy of the surrogate models is used to study the effect of low *proximity* and high *diversity*.

On each of the shadow datasets generated, models were fitted similarly to the task of the black-box model. The models fitted are called as ”Surrogate model” because these models closely approximate the black-box model. The surrogate model consists of $3$ dense layers with ReLU as the activation function and the final output layer with Softmax as the activation function. The models were trained for $100$ epochs using stochastic gradient descent as the optimizer [76], with a learning rate of $0.001$ and Negative log-likelihood as the loss function. The architecture of the surrogate model was built when the architecture of the black-box model was unknown. The architecture of the surrogate model can be found in Appendix A.2.8. The accuracy (Refer to equation 7.5) of surrogate model is used to understand the effect of setting *proximity* low and *diversity* high.

## 7.6 Comparison of Results between Synthetic Data Models

**Problem**

*To study the effectiveness of the three synthetic data models in generating synthetic data.*

Three algorithms were used to generate synthetic data. More details on the algorithm can be found in section 7.2.3. Unlike the existing framework of DiCE that generates counterfactual explanations from the original data, synthetic data is used in this study.

**Setup**

Table 7.2 describes the three algorithms and the various parameters. The overall privacy budget ($\varepsilon$) was computed from the noise, the training iterations ($T$), the sampling ratio $= \frac{250}{7777}$ ($\frac{\text{Batch Size}}{\text{Size of Dataset}}$), and failure probability ($\delta$). Details on the privacy parameters can be read in sections 6.1 and 7.2.3.

| Parameters | DP-GAN | DP-WGAN | DP-CTGAN |
|---|---|---|---|
| Generator (optimizer) | Adam | Adam | Adam |
| Discriminator/ Critic (optimizer) | DP-SGD | DP-SGD | DP-SGD |
| Learning rate($\eta_g, \eta_d$) | 0.0001 | 0.0001 | 0.0001 |
| Loss | Min-Max [6.9] | Wasserstein | Min-Max |
| Noise ($\sigma^2$) | 0.5 | 0.5 | 0.5 |
| Privacy Budget ($\varepsilon$) | 476 | 476 | 476 |
| Clipping ($c_g$) | 3 | $-$ | 3 |
| Weight Clipping ($c_p$) | $-$ | $0.1$ | $-$ |
| Training iteration ($T$) | 1000 | 1000 | 1000 |
| Discriminator training iteration ($n_d$) | $-$ | 2 | $-$ |
| Sampling ratio ($q$) | 0.03 | 0.03 | 0.03 |
| Failure probability ($\delta$) | 0.0001 | 0.0001 | 0.0001 |
| Input dimension for generator ($l$) | 21 | 21 | 21 |

**Table 7.2:** Training Parameters for the Synthetic Data Models. Three synthetic data models are used in the experiments.

### 7.6.1 Effect of noise on the Accuracy of Model and Synthetic Data

#### Problem

*To study the effect of varying noise values on the accuracy of the synthetic data model and the quality of data generated.*

The stable synthetic data model from the previous experiment is used in this experiment. In addition, different noise values are introduced into the synthetic data model to study the synthetic data model's behavior and the data's quality.

#### Setup

Table 7.3 denotes the different noise values used to generate synthetic data. The quality of the data generated for the varying noise is visualized using the data compression model (Refer to section 7.2.4).

| **Synthetic Data** | **Noise ($\sigma^2$)** | **Privacy Budget ($\varepsilon$)** | **Clipping threshold ($c_g$)** | **Failure Probability ($\delta$)** | **Training iterations ($T$)** | **Sampling ratio ($q$)** |
|---|---|---|---|---|---|---|
| Data 1 | 0.5 | 476 | 3 | 0.0001 | 1000 | 0.03 |
| Data 2 | 0.7 | 143 | 3 | 0.0001 | 1000 | 0.03 |
| Data 3 | 1 | 56.5 | 3 | 0.0001 | 1000 | 0.03 |
| Data 4 | 10 | 2.18 | 3 | 0.0001 | 1000 | 0.03 |
| Data 5 | 150 | 0.1 | 3 | 0.0001 | 1000 | 0.03 |

**Table 7.3:** Synthetic data generated for varying noise values. The noise ($\sigma^2$) values chosen are used to generate $5$ synthetic data using the stable synthetic data model from previous experiment.

### 7.6.2 Utility and Privacy Trade-off

#### Problem

*To study the trade-off between the utility of counterfactual explanations and the privacy loss caused by multiple counterfactual explanations in private and non-private settings.*

Counterfactual explanations generated using synthetic data are considered private (Refer Table 7.3), and the counterfactual explanations generated using the original data are considered non-private settings.

#### Setup

A set of counterfactual explanations ($2, 4, 6, 8, 10$) for a given query instance is generated using the synthetic and original data. The generated counterfactual explanations in both settings (private and non-private) are quantified using the utility metrics (Refer to section 7.4.1).

Privacy metrics explained in section 7.4.2 are quantified in the following way, counterfactual explanations are generated in private and non-private settings with $10$ queries, and $100$ counterfactuals are generated in each query. Generated explanations are the foundation for the shadow dataset. Surrogate models were built on the shadow dataset of private and non-private settings, similar to the task of the black-box model. The architecture can be found in Appendix A.2.8.

Finally, for the test data, relative entropy measures the closeness of the distribution between the predicted outcome ($X^*$) from the surrogate models, that is, the adversary's estimate versus the true outcome ($X$) for both private and non-private settings; the outcome is *Stroke* or *No Stroke*. The accuracy of the membership inference attack (Refer to section 7.2.2 to read about the attack) and relative entropy scores are used to study the effect of using synthetic data in contrast to original data.

Chapter 8

# Experiments and Results

## 8.1 Effect of Proximity Low and Diversity High

The choice of low proximity and high diversity as opposed to the proposed setting by *Aïvodji et al.* [1] is investigated in this section. Readers can find the problem, how the experiment was setup and the choice of the value of proximity and diversity in section 7.5.1.

$6$ surrogate models are built on the shadow dataset generated by counterfactual explanations. In Figure 8.1, each surrogate models' accuracy is plotted. The accuracy of the surrogate models $4, 5, 6$ was higher than models $1, 2, 3$, this is because the shadow dataset for the surrogate models $4, 5, 6$ are built with the same number of queries, and models $1, 2, 3$ are built with the same number of queries (Refer to Table 8.1). In short, the number of queries to generate counterfactual explanations are the differentiating factor for the accuracy of the surrogate models.

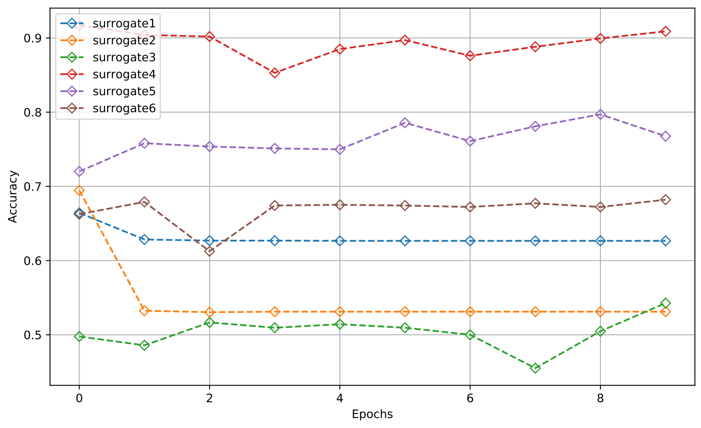


**Figure 8.1:** Accuracy of surrogate models averaged after every $10$ epochs.

A substantial approximation of the original data distribution could be found in the shadow dataset $4$, which was used to train the surrogate model $4$. A visualization of the shadow dataset $4$ is included in the Appendix A.2.9. Compared to the other shadow datasets, the more significant query budget and the significant number of counterfactuals generated made the shadow dataset $4$ to accurately approximate the original data, which resulted in the accuracy of the surrogate model $4$ being high compared to the other models.

| **Surrogate Model** | **Training Iteration** | **Number of Queries** | **Size of dataset** | **Accuracy** |
|---|---|---|---|---|
| 1 | 100 | 20 | 2000 | 63.05 |
| 2 | 100 | 20 | 1000 | 54.74 |
| 3 | 100 | 20 | 400 | 50.35 |
| 4 | 100 | 40 | 4000 | 89.30 |
| 5 | 100 | 40 | 2000 | 76.24 |
| 6 | 100 | 40 | 800 | 66.81 |

**Table 8.1:** Comparing the accuracy of the $6$ surrogate models.

In the context of a theft-motivated model extraction attack (Refer to section 4.3), the attacker aims to construct surrogate models that are as accurate as or more precise than the target model (black-box model). The averaged accuracy of the surrogate model $4$ is $89.30$, and the accuracy of the black-box model (Stroke Prediction model) is $87.43$. With a small dataset, the surrogate model $4$ performed better than the black-box model. The accuracy plot of the black-box model can be found in Appendix A.2.10. Therefore, using proximity low, diversity as high, and multiple queries, the accuracy of the surrogate model is better than the black-box model. An adversary can exploit the trade-off between proximity and diversity to query and generate counterfactual explanations and use these explanations to construct strong theft-motivated model extraction attacks.

## 8.2 Comparison of Results between Synthetic Models

The effectiveness of the three synthetic data models is studied using the generated data. Readers can find the problem and how the experiments were setup in section 7.6.

### DP-GAN

Algorithm 2 (DP-GAN) failed to produce samples based on the original data distribution. In Figure 8.2 two features from the generated synthetic data are plotted. The two discrete features are *Private job* and *formerly smoked*. The generator attempts to repeatedly create just one output for the features that seems credible to the discriminator; this phenomenon is known as mode collapse.

Readers may examine the complete results of the generated synthetic data of Algorithm 2 in Appendix A.2.11.

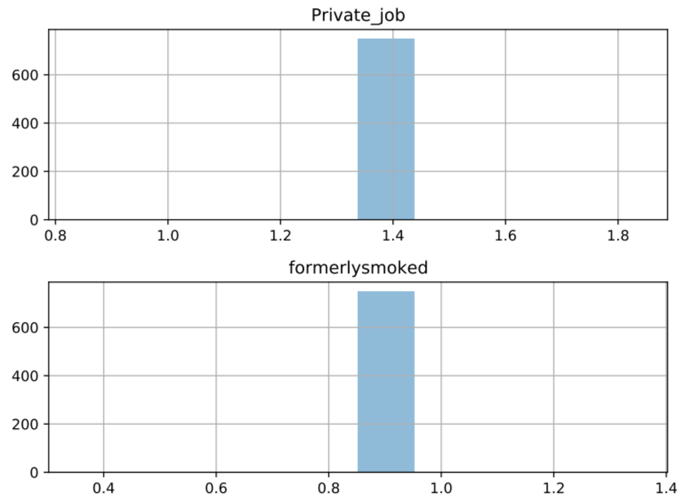


**Figure 8.2:** Histogram of the two discrete features from the synthetic data generated from Algorithm 2. The generator generates a single output for all the instance (Mode collapse).

### DP-WGAN

Compared to algorithm 2, Algorithm 3 did not suffer from mode collapse. However, the generated synthetic data is still not good enough. The data generated for some features is in the shape of a bell curve because of the inclusion of Gaussian noise in the discriminator's training. It is evident that the inclusion of Gaussian noise in the discriminator's training has affected the generator. In Figure 8.3 two features from the generated synthetic data are plotted to understand the effect of the inclusion of Gaussian noise in the generated synthetic data.

Readers may examine the complete results of the generated synthetic data of Algorithm 3 in Appendix A.2.12.

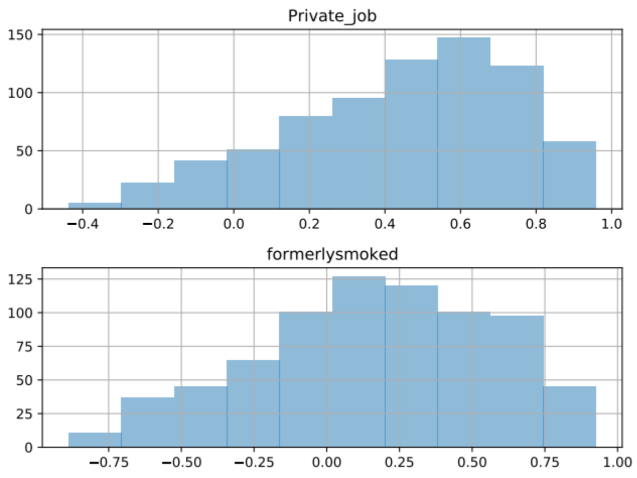


**Figure 8.3:** Histogram of the two discrete features from the synthetic data generated from Algorithm 3. The generator generates synthetic data in a bell shaped curve (Effect of Gaussian Noise).

### DP-CTGAN

Algorithm 4 (DP-CTGAN) is trained given the label *Stroke* or *No Stroke* hence the synthetic data model did not suffer from mode collapse. In Figure 8.4 two discrete features from the generated synthetic data are plotted. Inclusion of the Gaussian noise in the discriminator's training seemed to have less impact on the generator. Furthermore, algorithm 4 enables the generation of synthetic data based on the labels (outcome); thus, it reduces the overhead of getting the labels (outcome) for the generated synthetic data from the black-box model. Readers may examine the complete results of the generated synthetic data of Algorithm 4 in Appendix A.2.13. Algorithm 4 (DP-CTGAN) is very stable compared to the other algorithms; hence it is used in the following experiments to understand the effect of accuracy and the quality of data for varying noise values.

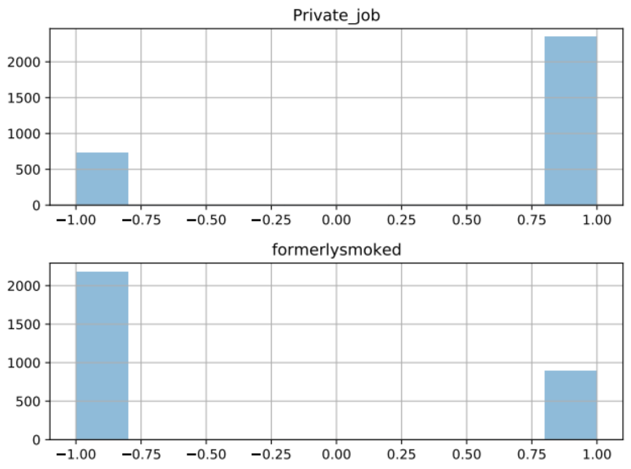


**Figure 8.4:** Histogram of the two discrete features from the synthetic data generated from Algorithm 4.

## 8.3 Effect of Noise on the Accuracy of Model and Synthetic Data

This section will study the effect of varying noise values on the accuracy of Algorithm 4 (DP-CTGAN) and the quality of the generated data. Readers can find the problem and how the experiments were setup in section 7.6.1.

The trade-off between noise and privacy budget would help understand the results in this section. Smaller the noise value, the larger the privacy budget ($\varepsilon$); on the contrary, the larger the noise value, the smaller the privacy budget. Different noise values used in the experiment can be read in Table 8.2.

**Accuracy of the model for varying noise**

The noise impacts the accuracy of the model (Algorithm 4). The model's accuracy is high when the Noise value is $0.5$, and the accuracy for the model is $94.1$. Since the accuracy was high, the data generated is very similar to the original data. The model with Noise $0.5$ is also very stable (see Figure 8.5). A Noise value of $0.7$, slightly higher than the previous noise value, shows a drop in the model's accuracy. The model's accuracy with a Noise of $0.7$ is $91.7$, and the model is less stable than the model with a Noise of $0.5$ (see Figure 8.5). A Noise of $1$ further resulted in a drop in accuracy. The accuracy for the model with Noise $1$ is $75.2$. A Noise value of $10$ showed a very unstable training compared to previous noise values, and the accuracy for the model with Noise $10$ is $53.7$. Finally, for Noise $150$, the model's accuracy is $34.7$. The results indicate that adding more noise makes the model more unstable and results in low accuracy.

From the perspective of privacy, a smaller Noise of $0.5$ results in a larger privacy budget of $476$ (see Table 8.2). According to differential privacy principles, the smaller the privacy budget, the better the privacy protection. This means less statistical information is hidden by the generated synthetic data with Noise $0.5$. On the contrary, significant Noise values $10$ and $150$ have a privacy budget of $2.81$ and $0.1$, respectively, meaning the synthetic data hides more statistical information of the original data.

In short, achieving a smaller privacy budget for better privacy protection is simply degrading the accuracy of the model by using significant noise values. Readers can examine the model's accuracy for varying noise values in Figure 8.5 and Table 8.2.

| **Noise value ($\sigma^2$)** | **Privacy Budget ($\varepsilon$)** | **Training Iteration** | **Accuracy** |
|---|---|---|---|
| 0.5 | 476 | 1000 | 94.1 |
| 0.7 | 143 | 1000 | 91.7 |
| 1 | 56.5 | 1000 | 75.2 |
| 10 | 2.18 | 1000 | 53.7 |
| 150 | 0.1 | 1000 | 34.7 |

**Table 8.2:** Averaged training accuracy to understand the effect of noise on the model's accuracy.

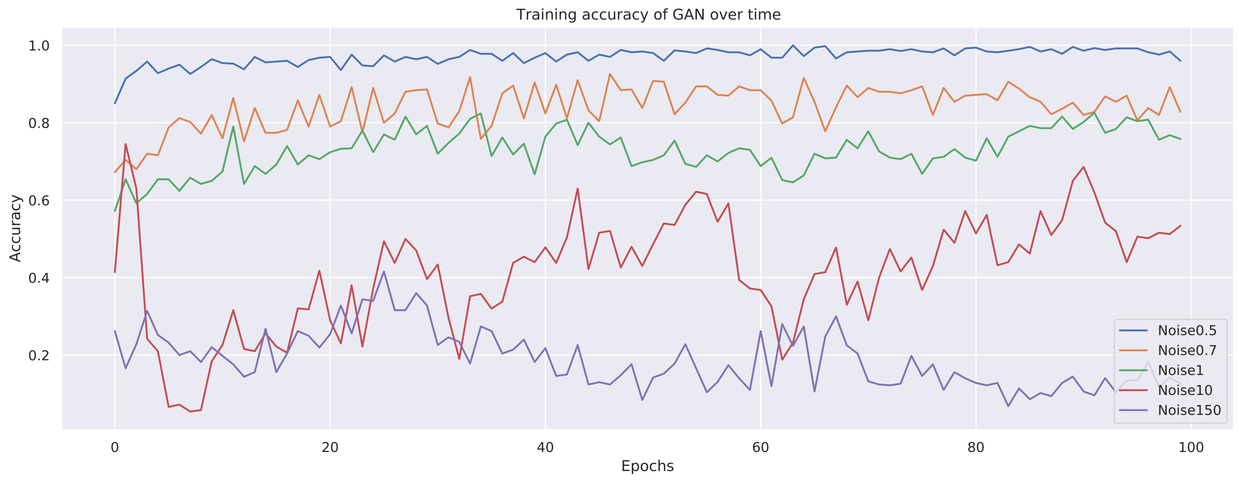


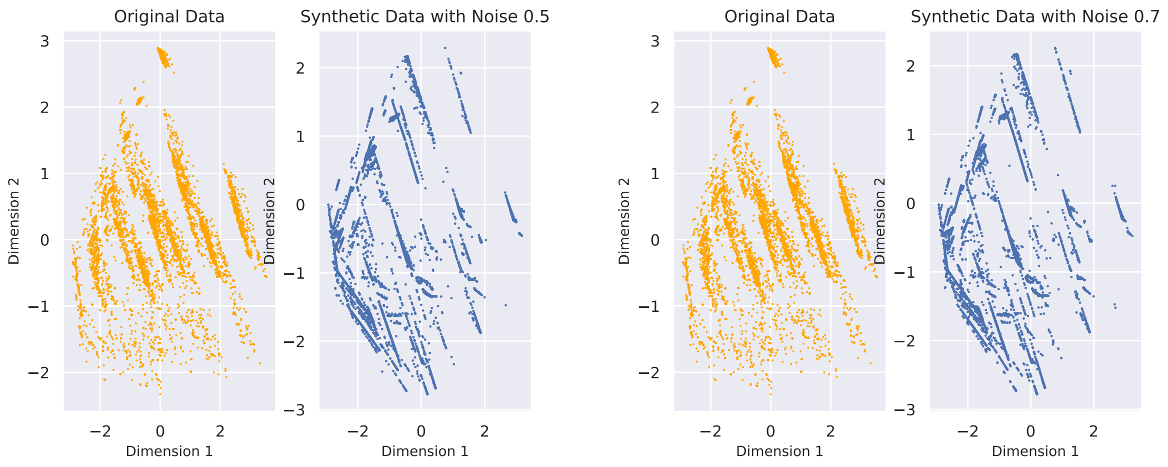
**Figure 8.5:** Accuracy of the Algorithm 4 (DP-CTGAN Model) for varying noise values averaged after every $100$ epochs.

### Quality of the synthetic data for varying noise

The quality of the data generated for the varying noise values is visualized after compressing the features into a smaller dimension. Readers can find the details about the data compression model in section 7.2.4. Only the first two principal components (Dimension) are plotted in Figure 8.6 to understand better how much variation the data showed for the noise values. Dimension $1$ and Dimension $2$ are the eigenvectors after applying Principal component analysis. More details about the eigenvectors can be found in section A.2.7.

A small noise value $(0.5)$ introduced into the model results in high accuracy, and the data is of high quality, closely resembling the original data. However, as the noise increases, the data quality deteriorates; this can be seen through the diminishing principal components, Dimension 1 and Dimension 2.

A large noise value of $150$ results in low accuracy and poor data quality. To conclude, data quality depends on the noise introduced since noise affects the model's accuracy.



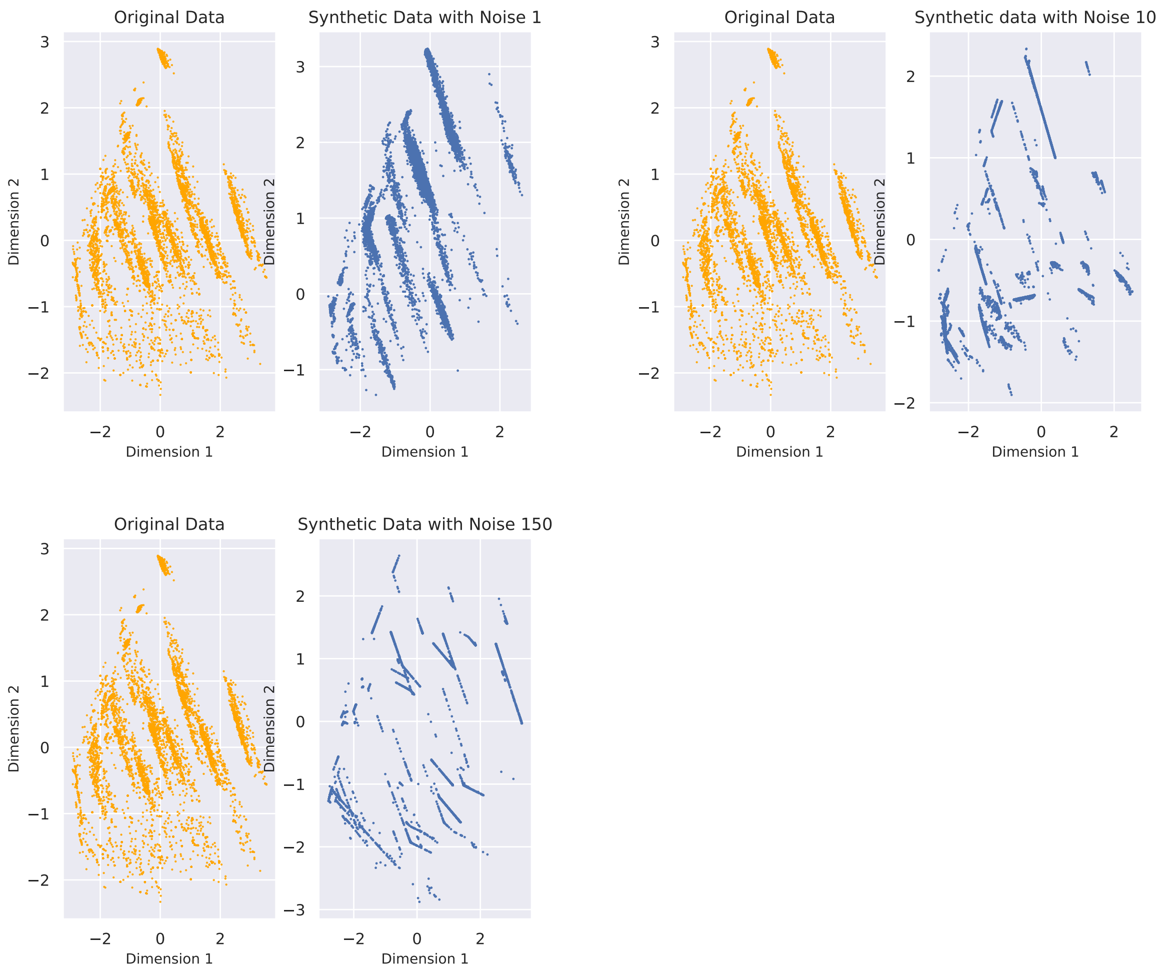


**Figure 8.6:** Visualizing the quality of data generated for varying noise.

## 8.4 Utility and Privacy Trade-off

This section will study the trade-off between utility and privacy when synthetic data is used to generate counterfactual explanations. Readers can find the problem and how the experiments were setup in section 7.6.2.

#### Utility of Counterfactual Explanations

The utility metrics discussed in section 7.4.1 are utilized to evaluate the effectiveness of the counterfactuals generated on the synthetic and original data. The utility metrics are proximity, sparsity, and diversity. DiCE framework aims to minimize *proximity*, that is, the distance between the query instance and the generated counterfactuals, and maximize *diversity*, that is, the distance between each generated counterfactuals. The DiCE framework does not optimize *sparsity*. Readers can examine the optimization function of DiCE in section 3.2.2.

Figure 8.7 demonstrates the utility on both private and non-private setting. For the non-private setting (Baseline), the counterfactuals were generated using the original data, and for the private setting, the counterfactuals were generated using synthetic datasets from the previous experiment.

Synthetic Data generated with Noise $0.5$ closely approximated the original data because the model's accuracy was high (see Figure 8.6, 8.5), and the counterfactuals generated were very similar to those generated on the original data. In addition, the metric scores, proximity, sparsity, and diversity were close to the non-private setting (Baseline). Similar results were noticed for the synthetic data with Noise $0.7$.

Synthetic data generated with Noise $1$ shows a degradation in the quality of data because of the degradation of the model's accuracy (see Table 8.2). The metric scores were different from the baseline. There is a gap between the proximity line plot of baseline and synthetic data with Noise $1$. This indicates that the generated counterfactuals are far away from the query instance. There is also a gap between the baseline sparsity line plot and the synthetic data with Noise $1$, which indicates that the feature changes between the query instance and counterfactual explanations are significant. Finally, the diversity line plot for the baseline and synthetic data with Noise $1$ shows a significant gap showing that diversity between the counterfactuals has decreased.

Synthetic data generated with Noise $10$ and $150$ failed to approximate the original data; hence the counterfactuals generated were very different from those generated on the original data. The counterfactuals generated in each query were similar and repeated more often because of the data scarcity and because only a few explanations were generated with contrastive results (outcome). The proximity line plot showed an increasing growth; the sparsity line plot showed an increasing growth, and the gap between baseline and synthetic data's diversity line plot increased in comparison to other synthetic data with Noise $0.5, 0.7, 1$.

In short, smaller noise values result in counterfactual explanations similar to the non-private setting. As the noise value increases, the counterfactuals generated are ineffective, and also the proximity with the query instance will increase as the amount of noise in the data increases, and the sparsity with the query instance will increase as the amount of noise in the data increases and the diversity in the generated counterfactuals will decrease as the amount of noise in the data increases.

Readers can examine the utility metrics for the baseline and synthetic data generated with varying noise in Figure 8.7.

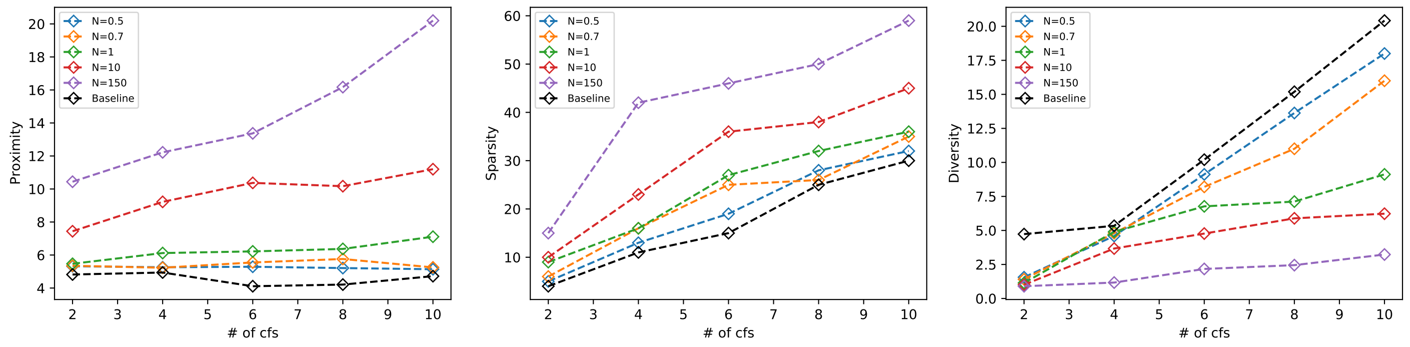
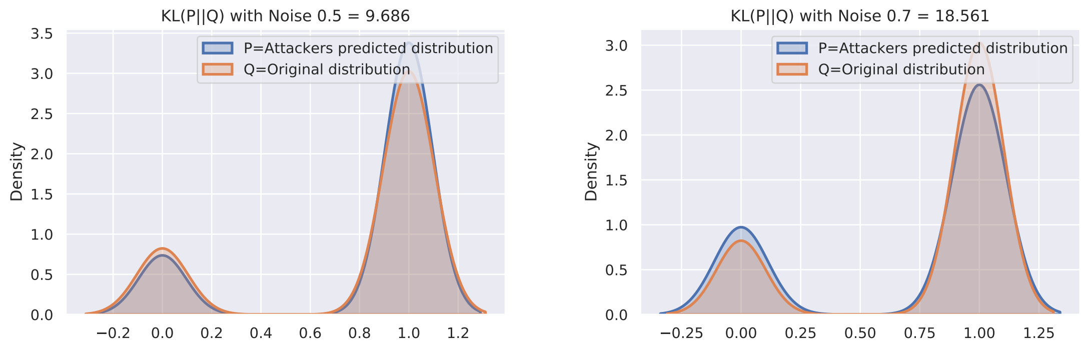


**Figure 8.7:** Comparing the utility of counterfactuals for varying noise values

### Privacy loss due to Counterfactual Explanations

The privacy metrics discussed in section 7.4.2 are utilized to evaluate the loss of privacy due to the counterfactuals generated on the synthetic and original data. In addition, readers can find the membership inference attack (MIA) process in section 7.2.2. The accuracy of MIA is used to understand the privacy loss in private and non-private settings. Readers can find the setup of this experiment in section 7.6.2.

Figure 8.8 depicts the miss classification made by the surrogate models trained on the shadow datasets generated by the adversary when synthetic data is used to generate counterfactual explanations. The distribution of the discrete labels *Stroke* $(1)$ and *No Stroke* $(0)$ for both the predicted estimation by the adversary and actual estimation are smoothed to better visualize the gap between the predicted and actual. The miss classification of the labels is minimal for the synthetic dataset with Noise $0.5$, $0.7$, and a small miss classification gap can be seen when synthetic data with Noise $1$ is used to generate counterfactual explanations. As the noise to create synthetic datasets increases, the miss classification also increases, proving that the amount of information gained by the adversary decreases.



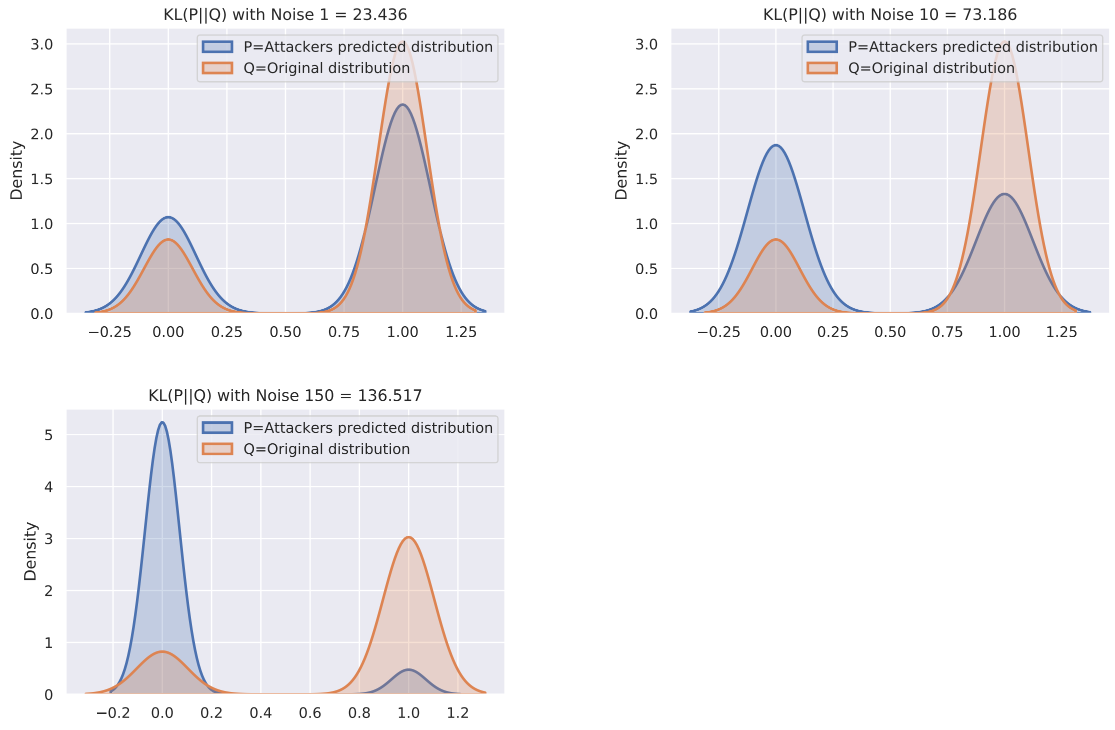


**Figure 8.8:** Relative Entropy and Miss Classification between the adversary's estimate (predicted) and actual estimate.

Table 8.3 depicts the relative entropy score. Relative entropy measures how far apart the adversary's estimation of the outcome (*Stroke, No Stroke*) from the original (true) distribution. Data generated with smaller noise values result in smaller entropy values which means much information is released because of the counterfactual explanations. On the contrary, data generated with a more considerable noise value results in a more significant entropy value resulting in less information exposure. When counterfactual explanations are generated using the original data, the accuracy of membership inference attacks is high (see Table 8.3). On the contrary, using synthetic data with increasing noise reduces the accuracy of the membership inference attack.

| **Data** | **Relative Entropy** | **MIA Accuracy** |
|---|---|---|
| original data (Baseline) | 4.321 | 89.05 |
| Noise 0.5 | 9.685 | 81.7 |
| Noise 0.7 | 18.561 | 80.2 |
| Noise 1 | 23.436 | 77.4 |
| Noise 10 | 73.186 | 42.1 |
| Noise 150 | 136.517 | 20.7 |

**Table 8.3:** Entropy Scores and Accuracy of MIA

The interpretation of the utility and privacy trade-off is that when smaller noise values are included to generate synthetic data, the effectiveness of the counterfactual explanations is comparable to the non-private setting; however, much information is leaked to an adversary due to multiple queries. Conversely, if more noise is added to generate synthetic data, the counterfactuals' effectiveness reduces; however, the adversary gains less information through the explanations.

Chapter 9

# Conclusion

This section discusses the results and the limitations of the work, and the conclusion followed by future work.

## 9.1 Discussions

In this section, some discussion points are covered based on the the performed experiments.

#### Effect of Proximity Low and Diversity High

The existing counterfactual explanations (DiCE) framework relies on using the original data and setting the proximity and diversity constraint. An adversary can use the trade-off between proximity and diversity, as demonstrated in the experiments, to query and generate many counterfactual explanations; these explanations tend to provide much information to the adversary. The possible countermeasure to mitigate this risk is to use a default value to set the proximity and diversity by studying the information exposure through the explanations and performing query auditing to differentiate between a legitimate user and an adversary [1].

#### Synthetic Data Model

In this study, synthetic data were generated using Differential Privacy principles to limit the danger of privacy loss and to generate counterfactual explanations in contrast to the original data. Three algorithms were benchmarked, out of which DP-CTGAN (Conditional GAN) generated data with a high approximation to the original data. An added advantage of DP-CTGAN is generating data by specifying the labels/ outcome; this reduces the overhead of communicating with the model owner to generate the labels for the synthetic data. Synthetic data from the DP-CTGAN can be sent to the explanation provider to build the counterfactual explanation model.

#### Effect of Noise on the Accuracy of Model and Quality of Data

Noise is a significant parameter in determining privacy. According to Differential privacy principles, larger Noise yields a smaller privacy budget and vice versa. An increase in Noise affects the model's accuracy [77]; therefore, to study the impact of varying Noise on the synthetic data model, different noise values were used to generate synthetic data in the experiments. An increase in Noise impacted the accuracy of the synthetic data model, and the quality of the data deteriorates but provides higher privacy guarantees.

#### Trade-off Between Utility and Privacy

The study's findings demonstrated a trade-off between utility and privacy:

- Practical counterfactual explanations may be drawn from synthetic data with small noise levels. However, they expose statistical information of the underlying data, enabling an adversary to undertake attacks.
- Synthetic data generated with large noise preserves the privacy of individuals by exposing less information through the counterfactual explanations, but the explanations provided are ineffective.

To achieve some degree of anonymity and privacy while generating multi-objective counterfactual explanations, considerable amounts of noise should be introduced to make synthetic data, but also the effect of noise on the counterfactual explanations should also be studied. The ideal situation of preserving privacy and providing effective explanations seems very challenging.

### 9.1.1 Limitations

The limitations and unknown factors are covered in this section.

#### Model

DiCE framework used to generate counterfactual explanations depends on two properties *proximity* and *diversity*. The research of DiCE does not explain to what extent the values of the two properties can be set; hence in this study, an iterator was used to set the values. Due to the ambiguity of the extent to which the values can lie, the results of the experiments could vary.

The synthetic data model (Algorithm 4) failed to generate discrete data for the discrete features. Instead, the synthetic data model converted the discrete variables into the continuous domain. In order to turn the continuous domain into the discrete domain, thresholding was employed. The thresholding value used could vary depending on the generated synthetic data.

#### Data

In this study, a single dataset (*Stroke Prediction Dataset*) was used to perform the experiments. The inference of this study will remain relevant in all use cases; However, the hyperparameters used in the experiment are data-dependent and could vary based on the criticality of the domain. For example, the privacy budget and the noise level, depends on how sensitive the data is and how much information can be exposed.

#### User Studies

The effectiveness of the counterfactual explanations was studied using metrics; however, user involvement and validation were not in the scope of this study. User studies could have helped more in understanding the effectiveness of the counterfactual explanations when generated using synthetic data.

## 9.2 Conclusion

This work seems exceptionally challenging because merging these domains, Explainable AI and Privacy-Enhancing Technologies is not simple. Multi-objective counterfactual explanations (DiCE) contribute to a better comprehension of the inference reached by the black-box model; Nevertheless, doing so will make the system more transparent, resulting in the loss of individuals' privacy. On the other hand, it is not always simple to provide explanations while protecting individuals' privacy. Therefore, it is necessary to strike a balance between the explainability of the outcome and the individual's privacy.

Sub questions for the mentioned research question are answered based on the study. Readers can find the research questions in section 1.4.

1.1 How vulnerable are multi-objective counterfactual explanations to membership inference attack?

In the experiments, the generation of multiple counterfactuals and the deliberate selection of low *proximity* and high *diversity* made multi-objective counterfactual explanations vulnerable to attacks. The *diversity* between the counterfactual explanations helps the attacker to build shadow datasets from the counterfactual explanations that closely approximated the original data, thus making it easy to perform a membership inference attack. The attacker's hypothesis that the shadow dataset corresponded to the original data proved accurate. It was shown in the results of MIA that utilizing original data to develop counterfactual explanations enhanced the accuracy of the membership inference attack, indicating that multi-objective counterfactual explanations are very vulnerable.

1.2 Given the counterfactual explanation method *DiCE*, how effective is Differential Privacy in preserving the privacy of individuals?

Differential Privacy adds noise scaled to meet the desired level of Privacy, compromising the model's (system) accuracy. There are many techniques to apply when it comes to noise, but this research focuses on generating synthetic data using differentially private algorithms. The reason behind using synthetic data is that since the existing architecture of DiCE depends on original data for generating explanations, the explanations are simply an individual's instance and therefore leak information about the individual. In the case of multiple explanations, the explanations leak information about a group of individuals.

In contrast to original data, synthetic data will preserve an individual's or group's data privacy because the synthetic data are fake data that approximates the original data, and the Privacy of individuals can be ensured. Furthermore, the post-processing property of Differential Privacy enables generating explanations from the synthetic data that are also differentially private; hence it is safe to perform any processing on the synthetic data. Hence, using synthetic data with privacy guarantees ensured by Differential privacy is very effective in preserving the Privacy of individuals when used to generate counterfactual explanations.

1.3 How effective are the generated counterfactuals after applying Differential Privacy?

The synthetic data were generated in the experiments using varying Noise to study the effectiveness of the generated counterfactual explanations. The effectiveness of the counterfactual explanations depends on the quality of the data generated. The quality of the data depends on the use of Noise to generate the synthetic data. Lower the Noise better the quality of data, and the generated counterfactual explanations are very effective and can help the end-user understand the feature's relationship with the prediction made. On the contrary, using a large

Noise value, the quality of the synthetic data degrades, affecting the effectiveness of the counterfactual explanations. The reason behind ineffective counterfactual explanations is the data scarcity and failing to provide explanations with the contrastive outcome.

RQ 1 : *To what extent can we preserve privacy when generating multi-objective counterfactual explanations?*

Multi-objective Counterfactual Explanations will revolutionize the AI field by expanding the use of AI in critical domains; however, this comes with the cost of compromising the privacy of individuals. Preserving privacy when generating multi-objective counterfactual explanations is not a deterministic outcome; however, it depends on the context, criticality of the domain, amount of information that can be exposed due to the explanations, and how effective the counterfactual explanations should be. From this study, we can infer that we can ultimately preserve the privacy of individuals by compromising the effectiveness of the counterfactual explanations; however, it will eventually lead to a loss of understandability of the explanations and will not help the end-user to contest the AI decision if considered biased. The second inference is that it is also possible to compromise privacy and generate effective explanations, and this could be ideal in a secured environment when the explanation model is not exposed to the public. The ideal situation of preserving privacy and providing effective explanations is not possible, but it is possible to strike a balance between privacy and explainability.

To conclude, the extent to which we can preserve privacy is domain-specific and needs a thorough study of the domain in collaboration with using methods explained in this study to test the approach and feasibility.

## 9.3 Future Work

This study utilized synthetic data in contrast to original data to generate counterfactual explanations; the other possible way would be to use the original data to develop counterfactual explanations, after which the counterfactual explanations would be perturbed by using noise in such a way that the privacy of individuals can be ensured and minimum information loss can be guaranteed.

The centralized architecture was the primary emphasis of this study. In this design, the data owner, model owner, and explanation provider are members of the same organization. However, the architecture can be decentralized by subdividing the roles such that the data owner, model owner, and explanation provider can all be separate parties that want to collaborate in a privacy-preserving way to build the prediction and the counterfactual explanation model.

An additional difference to the decentralized architecture would be made when there are several data owners, and each owner is interested in computing the prediction model and the counterfactual explanation model without sharing the data between parties; the possible direction would be to use secure multi-party computation.

The experiments performed in this study could also be further tested on multiple datasets with user studies to measure the effectiveness of the counterfactual explanations when the Differential Privacy mechanism is utilized.

Data generated from the synthetic data model claims to hide statistical information of the data, thus proving safe for further processing. In the experiments, it was noticed that, the synthetic data that provides high privacy guarantees were generated from synthetic data model with low accuracy and vice versa. The possible future work is to study if it is possible to hide statistical information of the data without degrading the accuracy of the synthetic data model.

# Bibliography

[1] U. Aïvodji, A. Bolot, and S. Gambs, "Model extraction from counterfactual explanations," *CoRR*, vol. abs/2009.01884, 2020. [Online]. Available: https://arxiv.org/abs/2009.01884

[2] C. Warlow, "Epidemiology of stroke," *The Lancet*, vol. 352, pp. S1–S4, 1998. [Online]. Available: https://www.thelancet.com/pdfs/journals/lancet/PIIS0140673698900861.pdf

[3] M. J. O'donnell, D. Xavier, L. Liu, H. Zhang, S. L. Chin, P. Rao-Melacini, S. Rangarajan, S. Islam, P. Pais, M. J. McQueen *et al.*, "Risk factors for ischaemic and intracerebral haemorrhagic stroke in 22 countries (the interstroke study): a case-control study," *The Lancet*, vol. 376, no. 9735, pp. 112–123, 2010. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/20561675/

[4] A. D. Lopez, C. D. Mathers, M. Ezzati, D. T. Jamison, and C. J. Murray, "Global and regional burden of disease and risk factors, 2001: systematic analysis of population health data," *The lancet*, vol. 367, no. 9524, pp. 1747–1757, 2006. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/16731270/

[5] V. L. Feigin, B. Norrving, M. G. George, J. L. Foltz, G. A. Roth, and G. A. Mensah, "Prevention of stroke: a strategic global imperative," *Nature Reviews Neurology*, vol. 12, no. 9, pp. 501–512, 2016. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/27448185/

[6] T. Hirsch, K. Merced, S. Narayanan, Z. E. Imel, and D. C. Atkins, "Designing contestability: Interaction design, machine learning, and mental health," in *Proceedings of the 2017 Conference on Designing Interactive Systems*, ser. DIS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 95–99. [Online]. Available: https://doi.org/10.1145/3064663.3064703

[7] W. Eschenbach, "Transparency and the black box problem: Why we cannot trust ai," pp. 3–8, 02 2021. [Online]. Available: https://www.researchgate.net/publication/349412290_Transparency_and_the_Black_Box_Problem_Why_We_Cannot_Trust_AI

[8] J. M. Durán and K. R. Jongsma, "Who is afraid of black box algorithms? on the epistemological and ethical basis of trust in medical ai," *Journal of Medical Ethics*, vol. 47, no. 5, pp. 329–335, 2021. [Online]. Available: https://jme.bmj.com/content/47/5/329

[9] D. Gunning, "Darpa's explainable artificial intelligence (xai) program," 03 2019, pp. 2–2. [Online]. Available: https://doi.org/10.1002/ail2.61

[10] C. Molnar, "Interpretable machine learning," 2022. [Online]. Available: https://christophm.github.io/interpretable-ml-book/

[11] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning: data mining, inference and prediction," 2009. [Online]. Available: http://www-stat.stanford.edu/~tibs/ElemStatLearn/

[12] B. Letham, C. Rudin, T. H. McCormick, and D. Madigan, "Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model," *The Annals of Applied Statistics*, vol. 9, no. 3, Sep 2015. [Online]. Available: http://dx.doi.org/10.1214/15-AOAS848

[13] J. H. Friedman and B. E. Popescu, "Predictive learning via rule ensembles," *The Annals of Applied Statistics*, vol. 2, no. 3, Sep 2008. [Online]. Available: http://dx.doi.org/10.1214/07-AOAS148

[14] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 8–8, 2018. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8466590

[15] U. C. for Medicare Medicaid, "E-Health General Information | CMS," dec 1 2021. [Online]. Available: https://www.cms.gov/Medicare/E-Health/EHealthGenInfo

[16] T. Miller, "Contrastive explanation: a structural-model approach," *The Knowledge Engineering Review*, vol. 36, p. e14, 2021. [Online]. Available: https://www.cambridge.org/core/journals/knowledge-engineering-review/article/contrastive-explanation-a-structuralmodel-approach/69A2E32B160C2C7FB65BC88670D7AEA7

[17] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," *CoRR*, vol. abs/1905.07697, 2019. [Online]. Available: http://arxiv.org/abs/1905.07697

[18] S. Nass, L. Levit, and L. Gostin, *Beyond the HIPAA Privacy Rule: Enhancing Privacy, Improving Health Through Research*, 02 2009. [Online]. Available: https://www.researchgate.net/publication/254429594_Beyond_the_HIPAA_Privacy_Rule_Enhancing_Privacy_Improving_Health_Through_Research

[19] R. Shokri, M. Stronati, and V. Shmatikov, "Membership inference attacks against machine learning models," *CoRR*, vol. abs/1610.05820, 2016. [Online]. Available: http://arxiv.org/abs/1610.05820

[20] J. Chen, W. Wang, and X. Shi, "Differential privacy protection against membership inference attack on machine learning for genomic data," 08 2020. [Online]. Available: https://www.researchgate.net/publication/343448976_Differential_Privacy_Protection_Against_Membership_Inference_Attack_on_Machine_Learning_for_Genomic_Data

[21] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014. [Online]. Available: http://dx.doi.org/10.1561/0400000042

[22] M. H. Jensen, "Cracking the Black Box of Healthcare," aug 23 2018. [Online]. Available: https://omhu.com/publications/cracking-the-black-box-of-healthcare/

[23] I. Sample, "Computer says no: why making AIs fair, accountable and transparent is crucial | Artificial intelligence (AI) | The Guardian," nov 5 2017. [Online]. Available: https://www.theguardian.com/science/2017/nov/05/computer-says-no-why-making-ais-fair-accountable-and-transparent-is-crucial

[24] S. Wachter, B. Mittelstadt, and L. Floridi, "Why a Right to Explanation of Automated Decision-Making Does Not Exist in the General Data Protection Regulation," *International Data Privacy Law*, vol. 7, no. 2, pp. 76–99, 06 2017. [Online]. Available: https://doi.org/10.1093/idpl/ipx005

[25] N. Vollmer, “Recital 71 EU General Data Protection Regulation (EU-GDPR). Privacy/Privazy according to plan.” jul 2 2021. [Online]. Available: https://www.privacy-regulation.eu/en/r71.htm

[26] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, “A survey of methods for explaining black box models,” *ACM Comput. Surv.*, vol. 51, no. 5, aug 2018. [Online]. Available: https://doi.org/10.1145/3236009

[27] M. T. Ribeiro, S. Singh, and C. Guestrin, “”why should I trust you?”: Explaining the predictions of any classifier,” *CoRR*, vol. abs/1602.04938, 2016. [Online]. Available: http://arxiv.org/abs/1602.04938

[28] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” 2013. [Online]. Available: https://arxiv.org/abs/1312.6034

[29] S. Wachter, B. D. Mittelstadt, and C. Russell, “Counterfactual explanations without opening the black box: Automated decisions and the GDPR,” *CoRR*, vol. abs/1711.00399, 2017. [Online]. Available: http://arxiv.org/abs/1711.00399

[30] S. Verma, J. P. Dickerson, and K. Hines, “Counterfactual explanations for machine learning: A review,” *CoRR*, vol. abs/2010.10596, 2020. [Online]. Available: https://arxiv.org/abs/2010.10596

[31] T. Miller, “Explanation in artificial intelligence: Insights from the social sciences,” *Artificial Intelligence*, vol. 267, pp. 1–38, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370218305988

[32] S. Dandl, C. Molnar, M. Binder, and B. Bischl, “Multi-objective counterfactual explanations,” *Lecture Notes in Computer Science*, p. 448–469, 2020. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-58112-1_31

[33] T. Wang, “Property inference attacks on neural networks using dimension reduction representations,” *semanticscholar*, 2018. [Online]. Available: https://www.semanticscholar.org/paper/Property-Inference-Attacks-on-Neural-Networks-using-Wang/54232c6de4614862d3c8b0258d44209e70383c36

[34] M. Jagielski, N. Carlini, D. Berthelot, A. Kurakin, and N. Papernot, “High-fidelity extraction of neural network models,” *CoRR*, vol. abs/1909.01838, 2019. [Online]. Available: http://arxiv.org/abs/1909.01838

[35] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, “A survey on homomorphic encryption schemes: Theory and implementation,” 2017. [Online]. Available: https://arxiv.org/abs/1704.03578

[36] Y. Lindell, “Secure multiparty computation (mpc),” 2020, https://eprint.iacr.org/2020/300. [Online]. Available: https://eprint.iacr.org/2020/300

[37] T. ElGamal, “A public key cryptosystem and a signature scheme based on discrete logarithms,” in *Advances in Cryptology*, G. R. Blakley and D. Chaum, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985, pp. 10–18. [Online]. Available: https://link.springer.com/chapter/10.1007/3-540-39568-7_2#citeas

[38] M. Nassar, A. Erradi, and Q. Malluhi, “Paillier’s encryption: Implementation and cloud applications,” pp. 1–5, 10 2015. [Online]. Available: https://ieeexplore.ieee.org/document/7338149

[39] Q. Zhu and X. Lv, "2p-dnn : Privacy-preserving deep neural networks based on homomorphic cryptosystem," *ArXiv*, vol. abs/1807.08459, 2018. [Online]. Available: https://arxiv.org/abs/1807.08459

[40] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology — EUROCRYPT '99*, J. Stern, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 223–238. [Online]. Available: https://link.springer.com/content/pdf/10.1007%2F3-540-48910-X_16.pdf

[41] A. C.-C. Yao, "How to generate and exchange secrets," *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pp. 162–167, 1986. [Online]. Available: https://www.semanticscholar.org/paper/How-to-generate-and-exchange-secrets-Yao/88645f0b1c344181a7ff705bd6c639bf515a23cf

[42] J. Choi and K. Butler, "Secure multiparty computation and trusted hardware: Examining adoption challenges and opportunities," *Security and Communication Networks*, vol. 2019, pp. 1–28, 04 2019. [Online]. Available: https://www.researchgate.net/publication/332179471_Secure_Multiparty_Computation_and_Trusted_Hardware_Examining_Adoption_Challenges_and_Opportunities

[43] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016. [Online]. Available: https://arxiv.org/abs/1610.02527

[44] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016. [Online]. Available: https://arxiv.org/abs/1602.05629

[45] R. Wilson and P. Rosen, "Protecting data through perturbation techniques: The impact on knowledge discovery in databases." *J. Database Manag.*, vol. 14, pp. 14–26, 01 2003. [Online]. Available: https://www.researchgate.net/publication/220373844_Protecting_Data_through_Perturbation_Techniques_The_Impact_on_Knowledge_Discovery_in_Databases

[46] J. P. Near and C. Abuah, *Programming Differential Privacy*, 2021, vol. 1. [Online]. Available: https://uvm-plaid.github.io/programming-dp/

[47] C. Dwork and A. Roth, *The Algorithmic Foundations of Differential Privacy*, ser. Foundations and trends in theoretical computer science. Now, 2014. [Online]. Available: https://books.google.nl/books?id=Z3p8swEACAAJ

[48] S. W. Kwak, J. Ahn, J. Lee, and C. Park, "Differentially private goodness-of-fit tests for continuous variables," *Econometrics and Statistics*, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2452306221001143

[49] I. Mironov, "Renyi differential privacy," *CoRR*, vol. abs/1702.07476, 2017. [Online]. Available: http://arxiv.org/abs/1702.07476

[50] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," vol. Vol. 3876, 01 2006, pp. 265–284. [Online]. Available: https://www.researchgate.net/publication/225124717_Calibrating_Noise_to_Sensitivity_in_Private_Data_Analysis

[51] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," vol. 4004, 05 2006, pp. 486–503. [Online].

Available: https://www.researchgate.net/publication/221348113_Our_Data_Ourselves_Privacy_Via_Distributed_Noise_Generation

[52] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, Oct 2016. [Online]. Available: http://dx.doi.org/10.1145/2976749.2978318

[53] D. Enthoven and Z. Al-Ars, "Fidel: Reconstructing private training samples from weight updates in federated learning," 2021. [Online]. Available: https://arxiv.org/abs/2101.00159

[54] Andrés de Loera-Brust, Caitlin Brandt, and Abigail Durak, Paul Ginsburg,, "The opportunities and challenges of data analytics in health care," nov 1 2018. [Online]. Available: https://www.brookings.edu/research/the-opportunities-and-challenges-of-data-analytics-in-health-care/

[55] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014. [Online]. Available: https://arxiv.org/abs/1406.2661

[56] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou, "Differentially private generative adversarial network," 2018. [Online]. Available: https://arxiv.org/abs/1802.06739

[57] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014. [Online]. Available: https://arxiv.org/abs/1406.2661

[58] f. , "Stroke Prediction Dataset." [Online]. Available: https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

[59] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. [Online]. Available: https://scikit-learn.org/stable/

[60] J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," pp. 1–4, 2009. [Online]. Available: https://link.springer.com/content/pdf/10.1007/978-3-642-00296-0_5.pdf

[61] R. K. Mothilal, A. Sharma, and C. Tan, "Github - interpretml/DiCE: Generate Diverse Counterfactual Explanations for any machine learning model." jun 1 2022. [Online]. Available: https://github.com/interpretml/DiCE

[62] F. Chollet *et al.* (2015) Keras. [Online]. Available: https://github.com/fchollet/keras

[63] M.-C. Popescu, V. Balas, L. Perescu-Popescu, and N. Mastorakis, "Multilayer perceptron and neural networks," *WSEAS Transactions on Circuits and Systems*, vol. 8, 07 2009. [Online]. Available: https://www.researchgate.net/publication/228340819_Multilayer_perceptron_and_neural_networks

[64] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[65] G. Andrew, S. Chien, and N. Papernot, "Github - tensorflow/privacy: Library for training machine learning models with privacy for training data," jun 7 2022. [Online]. Available: https://github.com/tensorflow/privacy

[66] L. Rosenblatt, X. Liu, S. Pouyanfar, E. de Leon, A. Desai, and J. Allen, "Differentially private synthetic data: Applied evaluations and enhancements," 2020. [Online]. Available: https://arxiv.org/abs/2011.05537

[67] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014. [Online]. Available: https://arxiv.org/abs/1411.1784

[68] C. Canonne, "What is $\delta$, and what $\delta$ifference does it make?" DifferentialPrivacy.org, 2021, https://differentialprivacy.org/flavoursofdelta/.

[69] S. Mishra, U. Sarkar, S. Taraphder, S. Datta, D. Swain, R. Saikhom, S. Panda, and M. Laishram, "Principal component analysis," *International Journal of Livestock Research*, p. 1, 01 2017. [Online]. Available: https://www.researchgate.net/publication/316652806_Principal_Component_Analysis

[70] J. Zhai, S. Zhang, J. Chen, and Q. He, "Autoencoder and its various variants," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8616075

[71] S. Vastrad, "Don't Stop at Ensembles — Unconventional Deep Learning Techniques for Tabular Data," jun 19 2020. [Online]. Available: https://towardsdatascience.com/dont-stop-at-ensembles-unconventional-deep-learning-techniques-for-tabular-data-8d4e154f1053

[72] B. Herzberg, "The Datamasters: Data Owners vs. Data Stewards vs. Data Custodians," nov 11 2021. [Online]. Available: https://blog.satoricyber.com/the-datamasters-data-owners-vs-data-stewards-vs-data-custodians/

[73] J. Brownlee, "Neural Networks are Function Approximation Algorithms - Machine Learning Mastery," mar 17 2020. [Online]. Available: https://machinelearningmastery.com/neural-networks-are-function-approximators/

[74] M. Mercioni and H. Stefan, "A survey of distance metrics in clustering data mining techniques," 06 2019. [Online]. Available: https://www.researchgate.net/publication/334816241_A_Survey_of_Distance_Metrics_in_Clustering_Data_Mining_Techniques

[75] I. Wagner and D. Eckhoff, "Technical privacy metrics," *ACM Computing Surveys*, vol. 51, no. 3, pp. 1–38, may 2019. [Online]. Available: https://doi.org/10.1145%2F3168389

[76] S. ichi Amari, "Backpropagation and stochastic gradient descent method," *Neurocomputing*, vol. 5, no. 4, 1993. [Online]. Available: https://www.sciencedirect.com/science/article/pii/092523129390006O

[77] E. Bagdasaryan and V. Shmatikov, "Differential privacy has disparate impact on model accuracy," 2019. [Online]. Available: https://arxiv.org/abs/1905.12101

[78] b. i. National heart, lung, "What is a stroke?" 2022. [Online]. Available: https://www.nhlbi.nih.gov/health/stroke

[79] M. Clinic, "Elevated blood pressure - Diagnosis and treatment - Mayo Clinic," jan 14 2021. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/prehypertension/diagnosis-treatment/drc-20376708#:~:text=Stage%201%20hypertension%20is%20a,90%20mm%20Hg%20or%20higher.

[80] W. H. Organization and I. D. Federation, “Definition and diagnosis of diabetes mellitus and intermediate hyperglycaemia : report of a who/idf consultation,” pp. 5–5, 2006. [Online]. Available: https://apps.who.int/iris/handle/10665/43588

[81] W. H. Organization, “Obesity and overweight,” 2016. [Online]. Available: https://www.who.int/en/news-room/fact-sheets/detail/obesity-and-overweight

[82] N. Labs, “Cerebrovascular accident (stroke),” 2021. [Online]. Available: https://nurseslabs.com/cerebrovascular-accident-stroke/

[83] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML’10. Madison, WI, USA: Omnipress, 2010, p. 807–814. [Online]. Available: https://dl.acm.org/doi/10.5555/3104322.3104425

[84] T. Szandała, “Review and comparison of commonly used activation functions for deep neural networks,” 10 2020. [Online]. Available: https://www.researchgate.net/publication/344757203_Review_and_Comparison_of_Commonly_Used_Activation_Functions_for_Deep_Neural_Networks

[85] J. Howard and S. Gugger, “Deep Learning for Coders with fastai and PyTorch [Book],” jul 1 2020. [Online]. Available: https://www.oreilly.com/library/view/deep-learning-for/9781492045519/

[86] S. Ruder, “An overview of gradient descent optimization algorithms,” 2016. [Online]. Available: https://arxiv.org/abs/1609.04747

[87] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

Appendix A

# Appendix

## A.1 Data Description

The collected data (Refer to section 7.1) for this study takes into account both modifiable and non-modifiable factors that impact cerebrovascular accident.

Cerebrovascular Accident (CVA) is defined as, ”an ischemic stroke or brain attack, is an abrupt loss of brain function caused by an interruption in the blood flow to a specific area of the brain” [78]. Risk factors for stroke are often put into two groups: modifiable and nonmodifiable factors. *High blood pressure*, *obesity*, *heart disease*, and *diabetes* are among the factors that can be modified by making healthy lifestyle choices such as quitting *smoking*, changing one’s *living situation*, and creating an improved *work-life* balance. *Age* and *gender* are examples of nonmodifiable factors [3].

Diastolic blood pressure of more than 90 millimeters of mercury (mmHg) and systolic pressure of more than 140 millimeters of mercury (mmHg) constitute hypertension (high blood pressure) [79]. The pancreatic beta cells release insulin, which is an anabolic hormone. Upon ingestion of food, insulin transports glucose from the blood to muscle, liver, and fat cells. As the insulin level rises, abnormalities in insulin production, insulin action, or both result in diabetes mellitus [80]. Obesity is a condition in which excess body fat has accumulated to the degree that it may have adverse health effects. When a person’s body mass index (BMI), calculated by dividing their weight by the square of their height, is more than 30 kg/m2, they are categorized as obese [81].

Stroke is caused because the neurons in the brain can no longer maintain aerobic respiration because of the decreased cerebral blood flow caused by the above mentioned modifiable factors, so the mitochondria switch to anaerobic respiration, which generates large amounts of lactic acid, changing the pH and preventing neurons from producing enough ATP for the brain cells to continue functioning (plaque) [82] (see Figure A.1).

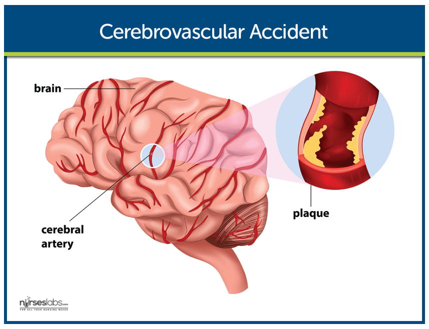


**Figure A.1:** Cerebrovascular accident [82]

## A.2 Models, Algorithms, Results

### A.2.1 Details about Dense layer, loss, activation function, gradient descent

#### Dense Layer

The basic operation in a node of a dense layer is given by activation(input $*$ weight $+$ bias). This operation is known as summation [63]. Figure A.2 is a pictorial representation of neural network architecture. The summation operation in each layer is given below, and the activation function is denoted by $f$.

**Input Layer**

$$o1 = f(w1 * x1 + b1)$$
$$o2 = f(w2 * x2 + b2)$$
$$o3 = f(w3 * x3 + b3)$$

**Hidden Layer**

$$o4 = f(wo1 * o1 + bo1) + f(wo2 * o2 + bo2) + f(wo3 * o3 + bo3)$$
$$o5 = f(wo1 * o1 + bo1) + f(wo2 * o2 + bo2) + f(wo3 * o3 + bo3)$$

**Output Layer**

$$\text{output} = f(wo4 * o4 + bo4) + f(wo5 * o5 + bo5)$$

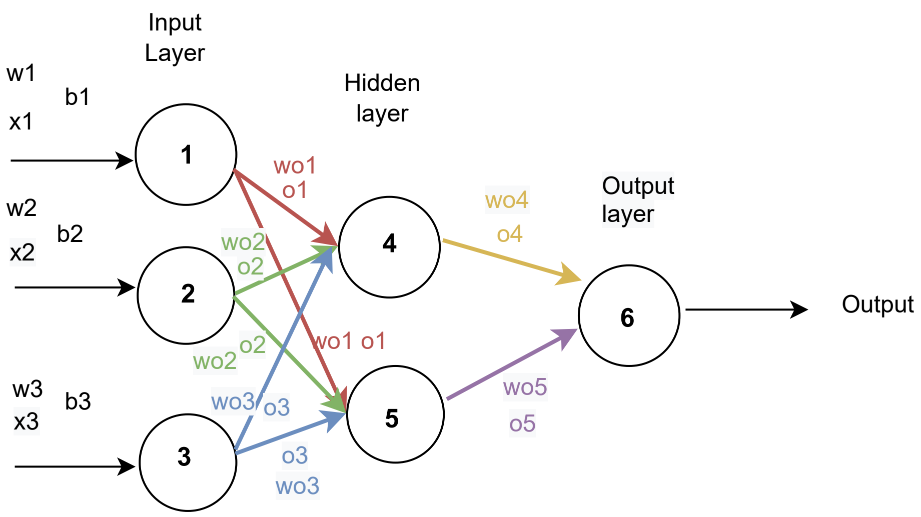


**Figure A.2:** Dense Layer

#### ReLU activation function

The output of the summation of inputs, weights, and bias are squashed using the activation function ($f$). The rectified linear unit will squash the output of the summation to zero if it is less than zero; otherwise, return the same value as the output [83]. Figure A.3 represents the plot of the RelU function.

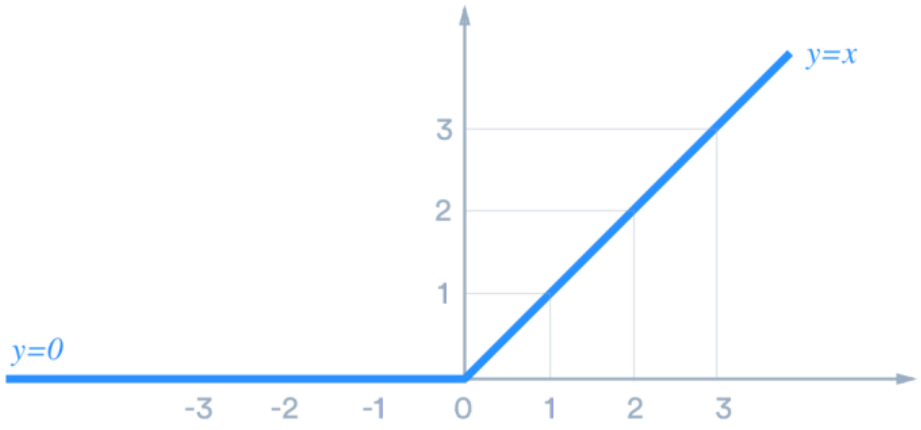


**Figure A.3:** ReLU [84]

### Softmax activation function

The output of the summation operation is squashed into probabilities using the Softmax activation function [64]. The Softmax activation function is usually used in classification problems, especially in the output layer. In Figure A.4 given some input features $(x1, x2, x3)$ of an instance the output layer predicts the probabilities of the instance either having *Stroke* or *No Stroke*.

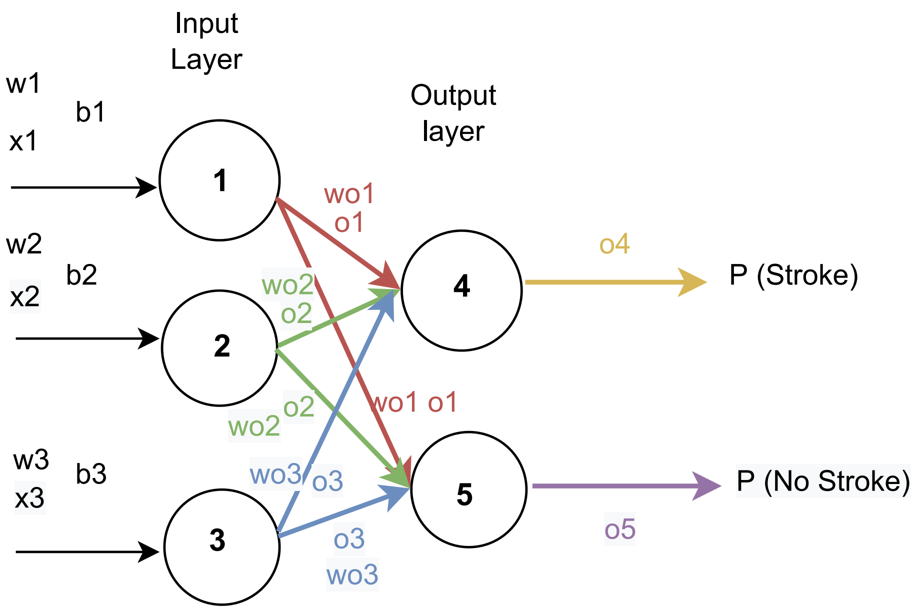


**Figure A.4:** Classification problem using Softmax as the activation function

For example, $o4 = 2.6$ and $o5 = -1.2$, the probability of *Stroke* $= \frac{e^{(2.6)}}{e^{(2.6)}+e^{(-1.2)}} \approx 0.97$ and *No Stroke* $= \frac{e^{(-1.2)}}{e^{(2.6)}+e^{(-1.2)}} \approx 0.03$. Given the features $(x1, x2, x3)$ probability of having Stroke is $97$ percent.

### Negative log-likelihood loss

Loss functions measure the error between the predicted and actual outcome. Negative log-likelihood loss function is often used when the Softmax activation function is used in the output layer. Minimizing the Negative log-likelihood loss function maximizes the likelihood of predicting the correct class [85]. Equation A.1 is the Negative log-likelihood loss function, $y$ denotes true label, $\hat{y}$ denotes predicted probabilities of the class.

$$l = -\sum_{i=1}^{n}(y_i \log \hat{y} + (1 - y_i)\log(1 - \hat{y})) \tag{A.1}$$

### Reconstruction loss

The mean squared error (MSE) gives reconstruction loss for the data compression model (Autoencoder). Mean squared error measures the average error between the input and reconstructed data. Equation A.2 is the reconstruction loss, where $Y_i$ is the input data to the encoder and $\hat{Y}_i$ is the data reconstructed by the decoder.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i) \tag{A.2}$$

### Stochastic gradient descent

Stochastic gradient descent is an optimization algorithm that updates the parameters (weights) for each instance [86]. For example, if an instance is predicted with a higher probability of *Stroke* and the actual prediction is *No Stroke*, the error between the predicted and actual outcome is calculated using a loss function. Stochastic gradient descent updates the parameters (weights) by computing the gradient ($\nabla$) of the loss function ($L$) with respect to the parameters ($w$) for each instance ($x_i$) given a learning rate ($\eta$) (see Equation A.3).

$$w_{i+1} \leftarrow w_i - \eta \nabla_i L(w_i; x_i, y) \tag{A.3}$$

### RMSProp

RMSProp makes use of the adaptive learning rate. If the learning rate ($\eta$) as specified in Equation A.3 is a large value, the weights ($w$) are not effectively updated, and for a small learning rate, the convergence to the global optimal is slow. The idea of an adaptive learning rate is that a larger learning rate is used in the beginning, and the learning rate is decayed after every iteration using the root mean square of the previous gradients.

The parameters are updated using Equation A.4, where $i$ denotes the iteration, $w$ denotes the parameters (weights), $g$ denotes the gradient, $\eta$ denotes the learning rate, $\sigma$ denotes the average of the previous gradients and $\alpha$ denotes a smoothing parameter.

$$w_{i+1} \leftarrow w_i - \frac{\eta}{\sigma_i} g_i; \quad \sigma_i = \sqrt{\alpha(\sigma_{i-1})^2 + (1 - \alpha)(g_i)^2} \tag{A.4}$$

### Adam

Adam optimizer is similar to RMSProp (Refer to section A.2.1) the only difference is, a momentum is added [87]. Adam optimizer, in short, is RMSProp + Momentum. In Figure A.5, the movement of the parameters (weights) is in the negative direction of the gradient to reach the global minimum (minima). However, often, the optimization gets stuck in the local minimum (minima); to avoid this, momentum (velocity) is used to push the gradients.

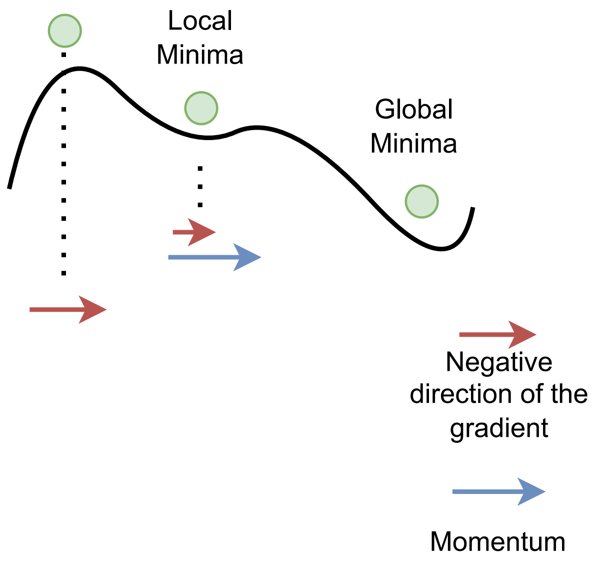


**Figure A.5:** Momentum to prevent getting stuck in local minima

## A.2.2 Stroke Prediction Black-box model architecture

```
Model: "Stroke Prediciton Model"
_________________________________________________________________
Layer (type) Output Shape Param #
=================================================================
dense_7 (Dense) (None, 40) 520

re_lu_6 (ReLU) (None, 40) 0

dense_8 (Dense) (None, 80) 3280

re_lu_7 (ReLU) (None, 80) 0

dense_9 (Dense) (None, 100) 8100

re_lu_8 (ReLU) (None, 100) 0

dense_10 (Dense) (None, 50) 5050

re_lu_9 (ReLU) (None, 50) 0

dense_11 (Dense) (None, 21) 1071

re_lu_10 (ReLU) (None, 21) 0

dense_12 (Dense) (None, 5) 110

re_lu_11 (ReLU) (None, 5) 0

dense_13 (Dense) (None, 2) 12

softmax_1 (Softmax) (None, 2) 0

=================================================================
Total params: 18,143
Trainable params: 18,143
Non-trainable params: 0
_________________________________________________________________
```

**Figure A.6:** Stroke Prediction Black-box model architecture

## A.2.3 DiCE Algorithm

**Algorithm 5** An algorithm for generating counterfactual explanations, DiCE

```
Require: $\lambda_1$, parameter for proximity. $\lambda_2$, parameter for diversity. $f$, trained black-box model. $x$, query instance. $T$, epochs for generating counterfactuals.
Ensure: DiCE generates diverse and multiple counterfactuals.
1: Select the query instance $x$
2: for e = 1,...,$T$ do
3:     Select a random instance from the dataset to be the initial counterfactual.
4:     Optimize the loss function using Equation 3.3.
5:     Store the set of counterfactuals that minimize the loss function.
6: end for
7: Display the set of counterfactuals.
```

#### Example for DiCE

Counterfactuals are generated using random sampling from the original data, and the loss function is optimized using gradient descent. For this example, the learning rate is set to $0.005$, and the initial hyperparameters ($\lambda_1, \lambda_2$) is set to $0.5$ and $1.0$ respectively. Distance function is denoted by $d$. Let diversity also be a distance function that measures the distance between the counterfactuals.

1. Query instance is predicted with high probability of having a *Stroke*, end-user requests explanations for *No-Stroke* ($y^{'} = 0$).
   ”Male”:1, ”Age”: 50, ”Smoking”: 0, ”Hypertension”: 0, ”Glucose”: 165.97, ”BMI”: 31, ”Stroke”: 1

2. Select random instances from the data to be the initial counterfactuals (2)
   $c_1$ = ”Male”: 1, ”Age”: 50, ”Smoking”: 0, ”Hypertension”: 0, ”Glucose”: 110.67, ”BMI”: 20, ”Stroke”: 0
   $c_2$ = ”Male”: 1, ”Age”: 50, ”Smoking”: 0, ”Hypertension”: 0, ”Glucose”: 97.97, ”BMI”: 25, ”Stroke”: 0

3. While $|f(c_i) - y^{'}| < 0.5$; moving to the opposite boundary, *No-Stroke*

   • Optimize the loss function

$$L = \frac{\lambda_1}{k}\sum_{i=1}^{k} d(c_i, x) - \lambda_2 \,\mathrm{dpp\dot{}diversity}(c_i....c_k)$$
$$L = \frac{0.5}{2}(70.15) - 1.5(\frac{1}{18.7})$$
$$L = 17.45$$

   • Decrease $\lambda_1$ and $\lambda_2$

$$\lambda_1 = \lambda_1 - \eta\frac{\partial L}{\partial\lambda_1} \approx 0.33$$
$$\lambda_2 = \lambda_2 - \eta\frac{\partial L}{\partial\lambda_2} \approx 0.99$$

4. Repeat the Steps 2-3

5. Return the counterfactuals that minimize the loss function

### A.2.4 Membership Inference Attack

```
Model: "Membership Inference Attack"
_________________________________________________________________
Layer (type) Output Shape Param #
=================================================================
dense_64 (Dense) (None, 30) 390

re_lu_28 (ReLU) (None, 30) 0

dropout_40 (Dropout) (None, 30) 0

dense_65 (Dense) (None, 50) 1550

re_lu_29 (ReLU) (None, 50) 0

dropout_41 (Dropout) (None, 50) 0

dense_66 (Dense) (None, 30) 1530

re_lu_30 (ReLU) (None, 30) 0

dropout_42 (Dropout) (None, 30) 0

dense_67 (Dense) (None, 20) 620

re_lu_31 (ReLU) (None, 20) 0

dropout_43 (Dropout) (None, 20) 0

dense_68 (Dense) (None, 10) 210

re_lu_32 (ReLU) (None, 10) 0

dropout_44 (Dropout) (None, 10) 0

dense_69 (Dense) (None, 5) 55

re_lu_33 (ReLU) (None, 5) 0

dropout_45 (Dropout) (None, 5) 0

dense_70 (Dense) (None, 2) 12

softmax_4 (Softmax) (None, 2) 0

=================================================================
Total params: 4,367
Trainable params: 4,367
Non-trainable params: 0
_________________________________________________________________
```

**Figure A.7:** MIA Architecture

### A.2.5 General GAN Architecture

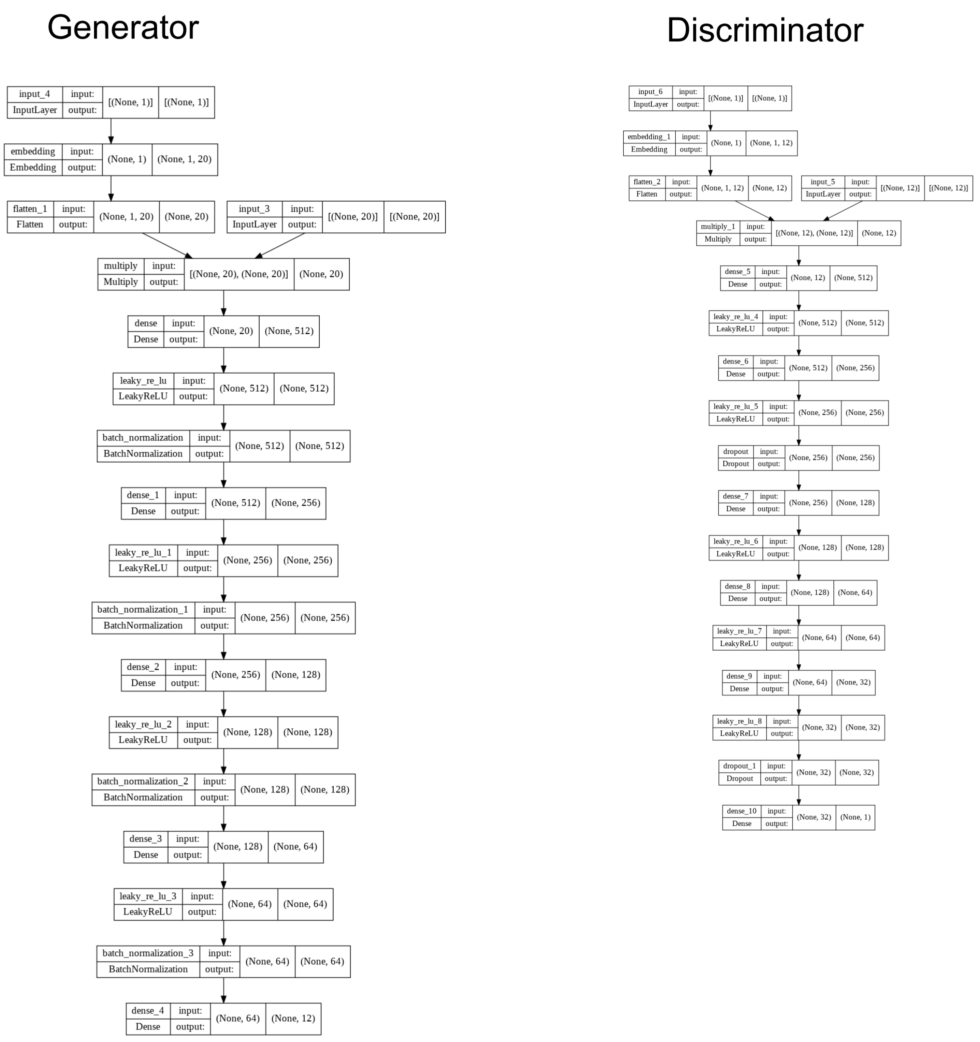

Generator
input_4 InputLayer input: output: [(None, 1)] [(None, 1)]
embedding Embedding input: output: (None, 1) (None, 1, 20)
flatten_1 Flatten input: output: (None, 1, 20) (None, 20)
input_3 InputLayer input: output: [(None, 20)] [(None, 20)]
multiply Multiply input: output: [(None, 20), (None, 20)] (None, 20)
dense Dense input: output: (None, 20) (None, 512)
leaky_re_lu LeakyReLU input: output: (None, 512) (None, 512)
batch_normalization BatchNormalization input: output: (None, 512) (None, 512)
dense_1 Dense input: output: (None, 512) (None, 256)
leaky_re_lu_1 LeakyReLU input: output: (None, 256) (None, 256)
batch_normalization_1 BatchNormalization input: output: (None, 256) (None, 256)
dense_2 Dense input: output: (None, 256) (None, 128)
leaky_re_lu_2 LeakyReLU input: output: (None, 128) (None, 128)
batch_normalization_2 BatchNormalization input: output: (None, 128) (None, 128)
dense_3 Dense input: output: (None, 128) (None, 64)
leaky_re_lu_3 LeakyReLU input: output: (None, 64) (None, 64)
batch_normalization_3 BatchNormalization input: output: (None, 64) (None, 64)
dense_4 Dense input: output: (None, 64) (None, 12)
Discriminator
input_6 InputLayer input: output: [(None, 1)] [(None, 1)]
embedding_1 Embedding input: output: (None, 1) (None, 1, 12)
flatten_2 Flatten input: output: (None, 1, 12) (None, 12)
input_5 InputLayer input: output: [(None, 12)] [(None, 12)]
multiply_1 Multiply input: output: [(None, 12), (None, 12)] (None, 12)
dense_5 Dense input: output: (None, 12) (None, 512)
leaky_re_lu_4 LeakyReLU input: output: (None, 512) (None, 512)
dense_6 Dense input: output: (None, 512) (None, 256)
leaky_re_lu_5 LeakyReLU input: output: (None, 256) (None, 256)
dropout Dropout input: output: (None, 256) (None, 256)
dense_7 Dense input: output: (None, 256) (None, 128)
leaky_re_lu_6 LeakyReLU input: output: (None, 128) (None, 128)
dense_8 Dense input: output: (None, 128) (None, 64)
leaky_re_lu_7 LeakyReLU input: output: (None, 64) (None, 64)
dense_9 Dense input: output: (None, 64) (None, 32)
leaky_re_lu_8 LeakyReLU input: output: (None, 32) (None, 32)
dropout_1 Dropout input: output: (None, 32) (None, 32)
dense_10 Dense input: output: (None, 32) (None, 1)

**Figure A.8:** GAN Architecture

### A.2.6 Architecture of Data Compression

```
Model: "model_1"
_________________________________________________________________
Layer (type) Output Shape Param #
=================================================================
input_1 (InputLayer) [(None, 12)] 0

dense (Dense) (None, 5) 65

=================================================================
Total params: 65
Trainable params: 65
Non-trainable params: 0
_________________________________________________________________
```

**Figure A.9:** Architecture of the Encoder

### A.2.7 Principal Component Analysis

The principal component analysis is used for dimensionality reduction. In short, PCA finds the projection of the entire data space; these projections are known as principal components. These principal components are the eigenvectors. In Figure A.10 , $u$ and $v$ denotes the eigenvector $1$ and $2$ respectively.

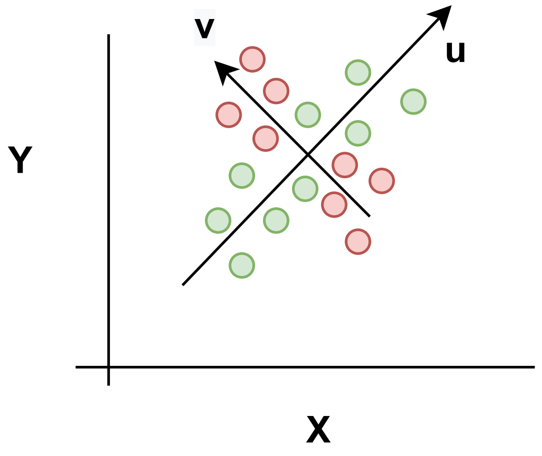


**Figure A.10:** Principal Components

An example is given to understand the process of PCA,

- Suppose this is a data, $Y = \begin{bmatrix} 0 & 17 \\ 2 & 19 \\ 3 & 21 \\ 4 & 23 \\ 6 & 20 \end{bmatrix}$

- The data is mean centred and normalized using the variance, $Y = \frac{1}{\sqrt{5}} \begin{bmatrix} -3 & -3 \\ -1 & -1 \\ 0 & 1 \\ 1 & -3 \\ 3 & 0 \end{bmatrix}$

- Covariance of the matrix is calculated using $cov(Y) = \frac{1}{N-1}Y^TY$, $\begin{bmatrix} 1 & \frac{13}{20} \\ \frac{13}{20} & 1 \end{bmatrix}$
- The eigenvalues of the covariance matrix are calculated, $\lambda_1 = \frac{33}{20}$, $\lambda_2 = \frac{7}{20}$
- Given $\lambda_1$, eigenvector is calculated using, $cov(Y) - \lambda_1$I; where I denotes identity matrix, $u$=$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$
- Given $\lambda_2$, eigenvector is calculated using, $cov(Y) - \lambda_2$I; where I denotes identity matrix, $v$=$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$
- Since the covariance matrix is symmetric, the eigenvector $u$ and $v$ are orthogonal.
- These eigenvectors correspond to the reduced dimension of the data. Figure A.10 shows how eigenvectors when plotted look. Data visualization of the synthetic data will also look the same but for $7777$ data points.
- Given these eigenvectors it is also possible to reconstruct back the data $Y$.

### A.2.8 Architecture of the Surrogate Models

```
----------------------------------------------------------------
Layer (type) Output Shape Param #
================================================================
Linear-1 [-1, 0, 15] 195
ReLU-2 [-1, 0, 15] 0
Linear-3 [-1, 0, 10] 160
ReLU-4 [-1, 0, 10] 0
Linear-5 [-1, 0, 5] 55
ReLU-6 [-1, 0, 5] 0
Linear-7 [-1, 0, 2] 12
LogSoftmax-8 [-1, 0, 2] 0
================================================================
```

**Figure A.11:** Surrogate model architecture

### A.2.9 Shadow Dataset 4

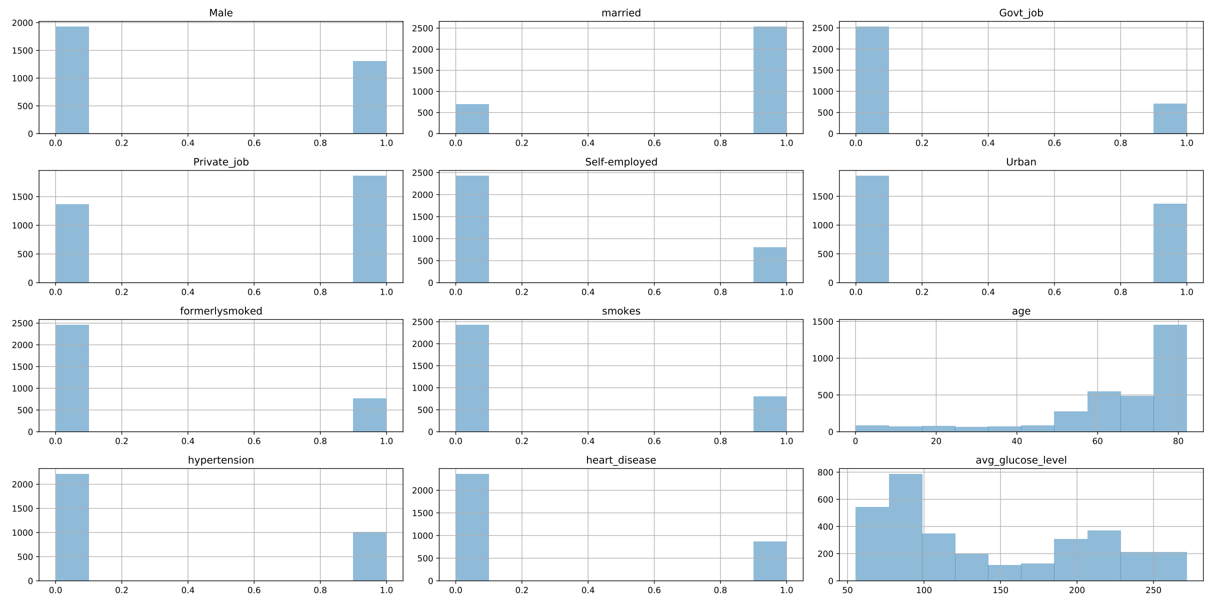


**Figure A.12:** Shadow Dataset 4 generated by querying

### A.2.10 Accuracy of Stroke Prediction Model (Target model)

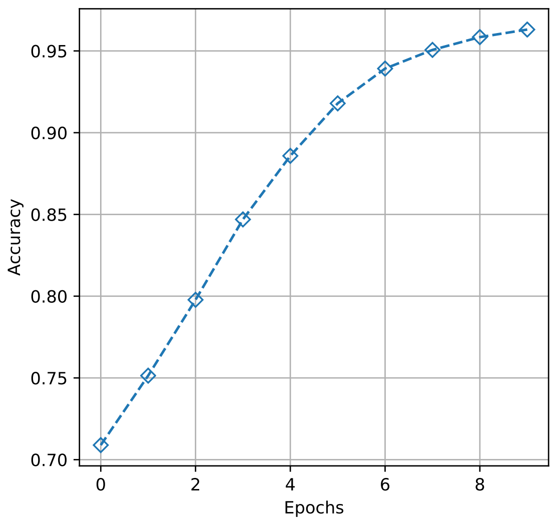


**Figure A.13:** Accuracy of the Stroke prediction model averaged after every 10 epochs

### A.2.11 Algorithm 2 results

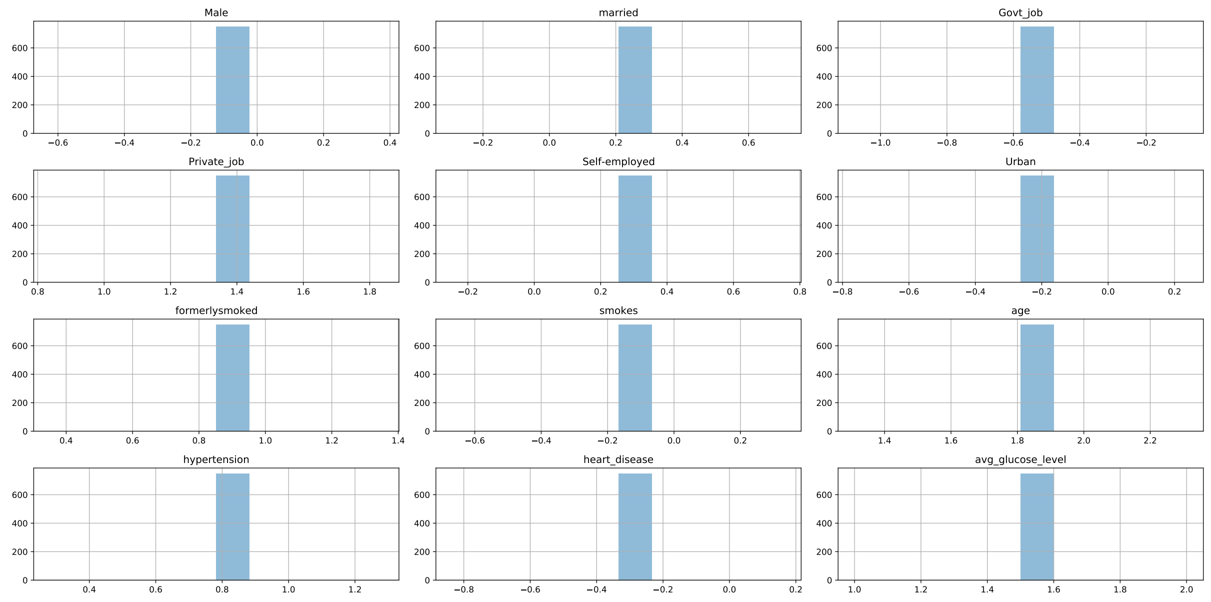


**Figure A.14:** Distributions generated by Algorithm 2

### A.2.12 Algorithm 3 results

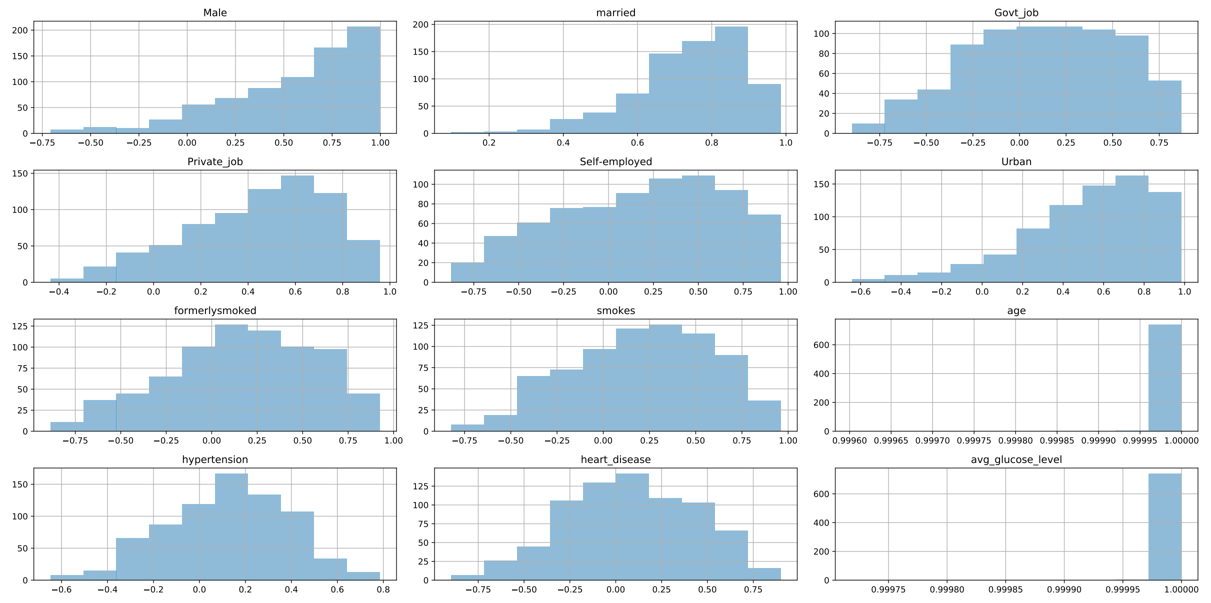


**Figure A.15:** Distributions generated by Algorithm 3

### A.2.13 Algorithm 4 results

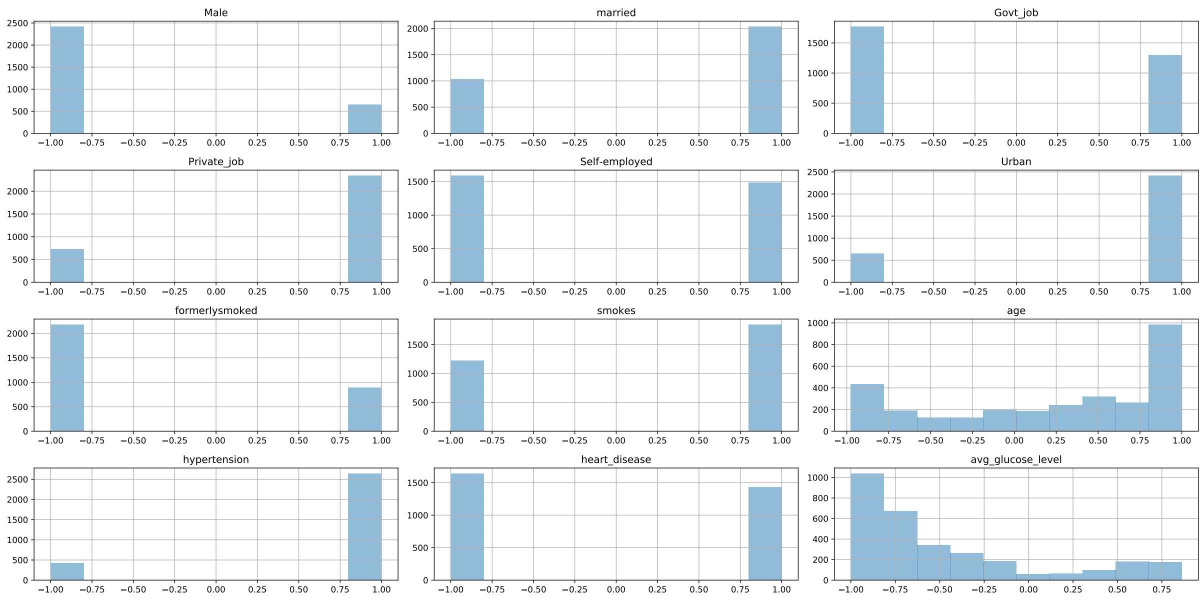


**Figure A.16:** Distributions generated by Algorithm 4