Navigating Data Privacy and Utility: A Strategic Perspective

Saloni Kwatra

Doctoral Thesis, October 2024
Department of Computing Science
Umeå University
Sweden

Department of Computing Science
Umeå University
SE-901 87 Umeå, Sweden

salonik@cs.umu.se

ISBN 978-91-8070-481-6 (print)
978-91-8070-482-3 (digital)
ISSN 0348-0542
UMINF 24.10
Cover illustrated by Ida Åberg

Printed by Cityprint i Norr AB, Umeå, 2024

Abstract

Privacy in machine learning should not merely be viewed as an afterthought; rather, it must serve as the foundation upon which machine learning systems are designed. In this thesis, along with the centralized machine learning, we also consider the distributed environments for training machine learning models, particularly federated learning. Federated learning lets multiple clients or organizations train a machine learning model in a collaborative manner without moving their data. Each client participating to the federation shares the model parameters learnt by training a machine learning model on its data. Even though the setup of federated learning keeps the data local, there is still a risk of sensitive information leaking through the model updates. For instance, attackers could potentially use the updates of the model parameters to figure out details about the data held by clients. So, while federated learning is designed to protect privacy, it still faces challenges in ensuring that the data remains secure throughout the training process.

Originally, federated learning was introduced in the context of deep learning models. However, this thesis focuses on federated learning for decision trees. Decision Trees are intuitive, and interpretable models, making them popular in a wide range of applications, especially where explanability of the decisions made by the decision tree model is important. However, Decision Trees are vulnerable to inference attacks, particularly when the structure of the decision tree is exposed. To mitigate these vulnerabilities, a key contribution of this thesis is the development of novel federated learning algorithms that incorporate privacy-preserving techniques, such as k-anonymity and differential privacy, into the construction of decision trees. By doing so, we seek to ensure user privacy without significantly compromising the performance of the model.

Machine learning models learn patterns from data, and during this process, they might leak sensitive information. Each step of the machine learning pipeline presents unique vulnerabilities, making it essential to assess and quantify the privacy risks involved. One focus of this thesis is the quantification of privacy by devising a data reconstruction attack tailored to Principal Component Analysis (PCA), a widely used dimensionality reduction technique. Furthermore, various protection mechanisms are evaluated in terms of their effectiveness in preserving privacy against such reconstruction attacks while maintaining the utility of the model.

In addition to federated learning, this thesis also addresses the privacy concerns associated with synthetic datasets generated by models such as generative networks. Specifically, we perform an Attribute Inference Attack on synthetic datasets, and quantify privacy by calculating the Inference Accuracy—a metric that reflects the success of the attacker in estimating sensitive attributes of target individuals.

Overall, this thesis contributes to the development of privacy-preserving algorithms for decision trees in federated learning and introduces methods to quantify privacy in machine learning systems. Also, the findings of this thesis set a ground for further research at the intersection of privacy, and machine learning.

Sammanfattning

Integritetsskydd inom maskininlärning bör inte enbart betraktas som ett senare tillägg; det måste snarare fungera som grunden på vilken maskininlärningssystem utformas. I denna avhandling behandlar vi, förutom centraliserad maskininlärning, även distribuerade miljöer för träning av maskininlärningsmodeller, särskilt federerad inlärning. Federerad inlärning gör det möjligt för flera klienter eller organisationer att gemensamt träna en maskininlärningsmodell utan att flytta deras data. Varje klient som deltar i federationen delar modellparametrarna som lärts in genom att träna en maskininlärningsmodell på sin egen data. Även om upplägget för federerad inlärning håller data lokal, finns det fortfarande en risk att känslig information läcker genom modelluppdateringarna. Angripare skulle till exempel potentiellt kunna använda uppdateringarna av modellparametrarna för att lista ut detaljer om den data som innehas av klienterna. Så även om federerad inlärning är utformad för att skydda integritet, står den fortfarande inför utmaningar när det gäller att säkerställa att data förblir säker under hela träningsprocessen.

Ursprungligen introducerades federerad inlärning inom ramen för djupa inlärningsmodeller. Denna avhandling fokuserar dock på federerad inlärning för beslutsträd. Beslutsträd är intuitiva och tolkningsbara modeller, vilket gör dem populära inom ett brett spektrum av tillämpningar, särskilt där förklarbarheten av de beslut som fattas av beslutsträdsmodellen är viktig. Beslutsträd är emellertid mottagliga för slutledningsattacker (inference attacks), särskilt när beslutsträdets struktur exponeras. För att mildra dessa sårbarheter är ett viktigt bidrag i denna avhandling utvecklingen av nya federerade inlärningsalgoritmer som införlivar integritetsskyddande tekniker, såsom k-anonymitet och differentiell integritet (differential privacy), i konstruktionen av beslutsträd. Genom att göra detta strävar vi efter att säkerställa användarnas integritet utan att väsentligt äventyra modellens prestanda. Maskininlärningsmodeller lär sig mönster från data, och under denna process kan de läcka känslig information. Varje steg i maskininlärningsprocessen uppvisar unika sårbarheter, vilket gör det viktigt att bedöma och kvantifiera integritetsriskerna. En central del i denna avhandling är kvantifiering av integritet genom att utforma en datarekonstruktionsattack anpassad för principiell komponentanalys (Principal Component Analysis - PCA), en ofta använd teknik för dimensionsreduktion. Vidare utvärderas olika skyddsmekanismer med avseende på deras

effektivitet i att bevara integritet mot sådana rekonstruktionsattacker samtidigt som modellens användbarhet bibehålls. Förutom federerad inlärning behandlar denna avhandling även integritetsproblem förknippade med syntetiska datamängder genererade av modeller såsom generativa nätverk. Specifikt utför vi ett attributslutledningsangrepp (Attribute Inference Attack) på syntetiska datamängder och kvantifierar integritet genom att beräkna slutledningsprecisionen (inference accuracy) – ett mått som återspeglar angriparens framgång i att uppskatta känsliga attribut hos målindivider. Sammantaget bidrar denna avhandling till utvecklingen av integritetsbevarande algoritmer för beslutsträd inom federerad inlärning och introducerar metoder för att kvantifiera integritet i maskininlärningssystem. Dessutom lägger avhandlingens resultat en grund för vidare forskning i skärningspunkten mellan integritet och maskininlärning.

Preface

This thesis contains a description of different privacy models, federated learning with decision trees, evaluating the risk of machine learning models, and synthetic datasets by attacking them. The work done is based on the following papers.

Paper I S. Kwatra, and V. Torra. A Survey of Tree Aggregation. *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1-6. IEEE, 2021.

Paper II S. Kwatra, and V. Torra. A k-anonymised Federated Learning Framework with Decision Trees. *International Workshop on Data Privacy Management*, pp. 106-120. Cham: Springer International Publishing, 2021.

Paper III S. Kwatra, and V. Torra. Data Reconstruction Attack Against Principal Component Analysis. *International Symposium on Security and Privacy in Social Networks and Big Data*, pp. 79-92. Singapore: Springer Nature Singapore, 2022.

Paper IV S. Kwatra, and V. Torra. Empirical Evaluation of Synthetic Data Created by Generative Models via Attribute Inference Attack. *IFIP International Summer School on Privacy and Identity Management*, pp. 282-291. Cham: Springer Nature Switzerland, 2023.

Paper V S. Kwatra, A. Monreale, and F. Naretto. Balancing Act: Navigating the Privacy-Utility Spectrum in Principal Component Analysis. *21st International Conference on Security and Cryptography (SECRYPT)* pp. 850-857, 2024).

Paper VI S. Kwatra, and V. Torra. DISCOLEAF: Discretizing Continuous Attributes for Learning with Federated Decision Trees with Personalisation. *Privacy in Statistical Databases*, pp. 344-356. Cham: Springer International Publishing, 2024.

Paper VII S. Kwatra, and V. Torra. Rakshit: Attacking and Defending Federated Learning with Decision Trees. *Submitted.*

In addition to the papers included in this thesis, the following publication was published within the studies but is not contained in this doctoral thesis:

Paper VIII S. Kwatra, A.K. Varshney, and V. Torra. Integrally Private Model Selection for Support Vector Machine. *European Symposium on Research in Computer Security*, pp. 249-259. Cham: Springer Nature Switzerland, 2022.

This study was partially funded by the Wallenberg AI, Autonomous Systems and Software Program (WASP), funded by the Knut and Alice Wallenberg Foundation.

To my family and friends, who made this journey easier

Acknowledgements

As I write these acknowledgments, it feels as though I'm re-living my journey through the past four transformative and beautiful years. This moment feels like penning a personal diary, capturing a time I can revisit whenever I want to reflect on these experiences. My PhD journey began in December 2020, during a time of pandemic. Moving from India to Umeå, I encountered a new environment with its own set of challenges, including the long winters. I fondly remember my supervisor, **Vicenç**, encouraging us during our online scrum meetings with his cheerful reminder: "You should go out every day!" He was so right, as I realized by not going out on some days. The online meetings and attending WASP courses virtually were not the most engaging at that time, in comparison with in person classes. I clearly remember doubting myself and my decision to pursue a PhD over other job opportunities. However, the people around me made it better, and gradually, my feelings and confidence improved. I met **Sudipta** on my very first day. She is very kind with people around her. I will always remember her kindness.

This thesis is not just a reflection of academic growth but a testament to the support and encouragement from many incredible people. Moving abroad for the first time introduced me to a multitude of new experiences and challenges, all made richer by the amazing individuals who walked this path with me.

To **Vicenç**, thank you for being my supervisor. Thank you for being all ears whenever I had things to say. Your activeness is inspiring and contagious. I am beyond grateful for the opportunities you provided me, which shaped my PhD journey in a beautiful way. Attending the Nobel Prize ceremony, participating in **Paul's** Docent lecture grading committee, and the unforgettable research visit to the University of Pisa, Italy—these were experiences I had only dreamed of. The WASP study trip to the USA was another highlight of my PhD time. Thank you for choosing me as your PhD student.

Living far from my family, my colleagues at the department became my second family. Every interaction, whether big or small, added a layer of warmth and comfort to my life here. In addition to my PhD work, finding joy and balance through my passion for dance, particularly Zumba, has been a highlight of my journey. To my colleagues **Divya**, **Sourasekhar**, **Shekhar**, **Marten**, **Tanaz**, **Sabine**, **Gulçin**, **Zuzana**, **Mariam**, and **Fatemeh**, **NAUSICA** group, to the entire IT support team here, including **Tomas**, **Mattias**,

Andreas, **Bertil**, and many others in the department: It was my pleasure to work with you all.

I want to thank my friends for filling my life with joy and a spectrum of emotions. Whether it was our badminton matches at IKSU, exploring new restaurants, spontaneous walks in Umeå, or simply sharing small conversations, each moment with you brought a sense of lightness and gratitude to my days. I am thankful for **Rangu's** presence in my life here. He used to travel from Skellefteå to Umeå, and we used to gather at **Santhan's** place. I am grateful for Santhan, and the tasty food he prepared for me. Such moments were precious and will always be cherished.

A special mention to **Divya**, your arrival after ten months into my PhD was like a fresh plant in a desert (I am not talking about a cactus here:p). No words are enough to express my thankfulness for you being here. I am forever grateful for the memories we created during our time together. You taught me to extract fun out of life, even when it was hard. It was literally never a dull moment around you. May you continue spreading your liveliness around, and I wish you all the luck and strength to complete your PhD soon.

To **Prithvi**, you are an amazing soul. Meeting you was a blessing in this lifetime. Your enthusiasm, and energy to conduct anything in life is inspiring. The only regret is that you had to leave Sweden so soon, and suddenly. I miss your banters with **Divya**. I miss us. I wish we three gather in India soon.

A special mention to my apartment mates **Sourasekhar**, **Prithvi**, **Yashwant**, and **Jonas** at Klintvagen 7. I am so grateful for this home and each one of you. Sharing meals with each other is my fondest memory. Your presence made my time in Umeå easy peasy. **Sourasekhar**, and **Yashwant**, you both have provided me guidance, time to time. I always cherish the conversations I have had with you all as a family living in Klintvagen 7. I am blessed to have such apartment mates.

A special mention also goes to **Tanaz** and **Gulçin**, whom I had the pleasure of meeting in the last year. Your friendship has been refreshing. The time we spent together, filled with laughter and meaningful conversations, added a wonderful new dimension to my experience in Umeå.

A special note of gratitude also goes to **Jyoti bua**, **Gaurav jojo**, and their kids **Vedant**, and **Vamika** for their incredible company during my stay in the USA while writing the last chapter of my thesis. I felt at home. Thank you for being such an essential part of my journey and for helping me navigate this final stretch with ease and focus.

To my husband, **Saurabh**, writing you in my acknowledgments fills me with a sense of happiness. I am so grateful that you came to visit me during my time here. Your presence in my life makes every challenge seem surmountable, and I am so lucky to have you by my side. You have been a very supportive partner to me, and I look up to you in so many ways.

To my family—my **parents**, my **brothers Shibu**, and **Akash**, and my extended family—no words can fully express my gratitude for your endless love and support. Despite the miles between us, each call, message, and moment of

connection with you gave me the strength to persevere. I also want to thank my **Alka bua**, who used to teach me when I was a kid. I am grateful that my family has always provided me the freedom, belief system, and support to pursue the things. This thesis is dedicated to you, my pillars of strength. I am blessed beyond measure to have you in my life, and I love you all more than words can convey.

I would also like to express my gratitude to those whose names I may have unintentionally missed. Your presence, and contributions have been just as essential to this journey, and I am sincerely appreciative of each one of you.

Saloni Kwatra

Contents

Chapter 1

Introduction

> Arguing that you don't care
> about the right to privacy
> because you have nothing to hide
> is no different than saying you
> don't care about free speech
> because you have nothing to say
>
> Edward Snowden

1.1 Who Guards Our Digital Privacy?

The aforesaid quotation is undeniably true in real-life situations. We often hear people say, "I have nothing to hide, so why should I care?" This phrase says that there is some sort of direct correlation between desiring privacy, and having secrets to keep. This is a really old practice, which was used in earlier times to intrude privacy. Now, we are in the age of hyperactivity on internet, where the data flows freely. With the digital tracking, we not only put our privacy on stake, but also our friends, and connections. More the data, more the privacy concerns. It is high time that even the common people, who have normal looking life understand that their data is not just used for their benefits, such as, showing them relevant advertisements in a shopping application or showing a friend suggestion in a social media application. Therefore, it is important for common people to understand that their data can also be sold for money, and can be used against them by malicious adversaries. The key takeaway from this subsection is that instead of dismissing privacy concerns, it's crucial for users to take proactive steps to safeguard their very own personal data.

In Snowden's words- "Privacy is not about hiding, it is about protecting". Protecting privacy has different perspectives, as we have different stakeholders involved in the process of making data publicly available, and further processing the data. To understand the research questions, and their corresponding

solutions present in this thesis, we need to understand who are these stakeholders, and what are their roles. So, we describe different stakeholders, and their roles according to Torra [Tor22], as follows.

Data subject/participant/respondent This includes individuals, whose data is being collected, such as patients in the hospitals, customers in the banks, etc.

Data controller/holder This is the one who collects, and holds the ownership of the data, such as government agencies, service provider, hospitals, etc. They may be reliable or unreliable.

Data user/recipient This is a trustworthy or authorized party who works with the collected data for analyzing purposes, such as data analysts, data scientists, and researchers.

Attacker This is a party, who has malicious intentions to breach the privacy of data users. Any party who has authorized access to the data of users, can also be malicious, if they make inferences from the analyses, which should not be done or if they use the data for wrong purposes.

We, researchers are data users, who are analyzing the data, and quantifying the utility, and the privacy of data subjects or respondents whose data is being collected. We, researchers, also take charge as an attacker when we aim to quantify the privacy of the system. The assumptions or the knowledge we assume that the attacker knows, can be weak or strong, depending on the application.

1.2 Motivation

The motivation of this thesis is broadly about two subjects, Preserving Machine Learning (PPML), and Privacy Preserving Data Publishing (PPDP), and the ultimate goal in both of these subjects is to maintain a balanced tradeoff between the utility and privacy. Now, we will discuss why PPDP, and PPML are essential for us, and how we quantify utility, and privacy for each of them.

PPDP provides necessary techniques to make the data of individuals publicly available while preserving their privacy. Data publishing is a predominant practice, and is widely encouraged for research purposes. According to General Data Protection Regulation (GDPR) [VV17], California Consumer Privacy Act (CCPA) [Par18], and other similar regulations, data should be anonymized before making them publicly available, or analyzing them for research purpose or personal use. A data is considered anonymized if the personal identifiers are irrevocably removed. Hence, PPDP is one of the important goals of this thesis.

Now a days, Machine Learning (ML) is applied to various applications, which are being utilised in our daily lives. With the ML models having huge number of parameters, ML models always had the capability to memorize the data of

users. Because of the memorizing capability of such models, it has become really important to introduce privacy in the machine learning models. From the context of PPML, privacy can be introduced in the data (before training any machine learning model), while training, and/or after training the model, which is decided based upon the privacy-utility tradeoff requirement for the application. For analyzing the utility, we test the performance of machine learning techniques, which includes classification, and clustering performances.

For the privacy analysis, we tailor attack models, where the attacker has some background knowledge, and the goal of the attacker is to perform identity disclosure, attribute disclosure, or reconstruct the original databases.

Concerning PPDP, we incorporate privacy prior to the publishing of a database. We develop, and use methods to quantify privacy, and utility of the databases before, and after incorporating privacy. We also work with synthetic datasets, which are created using generative models. Mostly, synthetic datasets do not correspond to real users, thus synthetic data avoids re-identification/record linkage risks. But, if the attacker has some background knowledge, the attacker can infer some sensitive information by the target individual if their is a strong correlation between the attacker's background knowledge about a target individual, and the sensitive information, which the attacker aims to infer.

Concerning PPML, we incorporate privacy before training or while training a machine learning model on the databases. And then, we compare the privacy, and utility, when we train machine learning model privately with the case when we train machine learning model non-privately. For machine learning, there are also distributed frameworks for learning in a collaborative manner, such as Federated Learning (FL). When FL was proposed for the first time [McM+17], it was proposed for deep learning models. Later on, the concept of FL was extended to other numerical models, including Support Vector Machine (SVM), Logistic Regression, and many other classification, as well as regression models. When we started, there was much less work on Decision Trees-based federated models. We were interested to work with Decision Trees due to their simplistic, and explainable nature. So, our motivation in this thesis is to propose privacy-preserving solutions for performing FL with decision trees.

1.3 Research Questions

Federated Learning (FL) is a distributed machine learning framework, in which distributed clients learn a machine learning model collaboratively, without sharing their data with each other or any central server. FL offers several advantages. One of those is saving computation resources by not having to collect all the data at one location. Typically, an FL framework consists of four steps. In Step 1, the central server sends a global initialised model to the clients. In Step 2, each client trains the machine learning model on their data, and sends the updates of machine learning model to the server. In Step 3, the central server aggregates the model updates. In Step 4, the central server sends the model

updates back to the clients. Steps 2-4 goes on until the model convergence is reached. We cannot say FL is inherently privacy-preserving, as the data of each client is not shared. But, sharing of model parameters, like gradients in neural networks causes enough privacy breach in FL [ZLH19]. Hence, one of our key objectives is to incorporate privacy in the algorithms to perform FL. Initially, FL was initially proposed for deep learning models, and later on it was extended to other classical machine learning models. In this thesis, we develop algorithms to perform FL with Decision Trees. Developing FL algorithms for decision trees has its own unique challenges such as privacy. Decision trees, which are trained at local clients have different structures unlike deep learning models, in which the distributed clients receive a deep learning model with a common structure from the central server. Hence, it is different to aggregate decision trees than any deep learning model or any other classical machine learning model. So, developing privacy-preserving, and explainable FL solutions with decision trees is one of the objectives of this thesis.

Quantification of privacy is also one of the objectives of this thesis. One of the ways for privacy quantification of machine learning systems is by tailoring attack models against them. In the attack model, there are certain assumptions about the knowledge of an attacker. So, in this thesis, we devise ways to quantify privacy of a machine learning system, particularly involving one of the most popular dimensionality reduction techniques, PCA.

In this thesis, we also focus on the privacy of synthetic datasets. Synthetic data is basically a data, which do not correspond to real individuals, and at the same time, synthetic data is structurally, and statistically similar to the original data. We devise a methodology to quantify the privacy of synthetic datasets via an Attribute Inference Attack (AIA), in which the attacker is aware of some information about a target individual, and the attacker aims to infer other sensitive information about a target individual. For conducting an AIA, the attacker knows some attributes about a target individual, and a synthesized version of the dataset. If there is a strong correlation between the attributes known by the attacker, and the attributes unknown to the attacker in the synthetic dataset, then this can aid the attacker to infer the sensitive attributes of a target individual successfully.

Our main, and specific research questions are as follows.

RQ1 How to perform Federated learning with Decision Trees for the horizontally partitioned data, while preserving the privacy of the clients participating to the federation?

RQ2 How to evaluate the risk associated with sharing machine learning models' information, like Principal Component Analysis (PCA)?

RQ3 How to evaluate attribute inference risks of synthetic data?

RQ4 How to attain a sweet spot between utility, and privacy for machine learning models?

1.4 Methodology

The methodology used in this thesis is mostly experimental, and analytical. We perform experiments to provide solutions for our research questions, and then we analyse our results to provide useful trends concerning the privacy, and the utility of the data.

- To address **RQ1**, we propose an FL framework, in which distributed clients first protects their data using an anonymization technique, specifically Mondrian k-Anonymity [LDR06] in our case. Then, each client locally trains a decision tree model using Gini index [Ste09] as the splitting criteria, which is then aggregated by the central server. The central server shares the aggregated decision tree with the distributed clients. Hence, each client learns a decision tree model in a collaborative manner.

- To address **RQ2**, we propose an attack model against PCA. The aim of which is to reconstruct the database of users. To evaluate the success of our attack, we compute the proximity of reconstructed samples with the original samples.

- To address **RQ3**, we propose an attack model, which evaluates the attribute inference risks associated with synthetic data, which is an important step prior to the publishing of synthetic dataset. Hence, we quantify privacy in **RQ2**, and **RQ3**.

- To address **RQ4**, we evaluate both the utility, and the privacy of synthetic datasets, as well as the machine learning models. We aim to find a sweet spot between the utility, and privacy. As mentioned earlier, for quantifying privacy, we tailor attack models, and for quantifying utility, we apply hierarchical clustering, and other machine learning models like random forest classifier.

1.5 Main Contributions

Most of the approaches, which perform FL with decision trees were based on cryptographic approaches like Homomorphic Encryption [Che+21a], and [Che+21b], which suggested to apply homomorphic encryption on the decision paths, and gradients, respectively. Homomorphic encryption allows computations directly over the cipher text [Yi+14]. These approaches are computationally expensive. In addition to this, when we started working on federated decision tree approaches, there was more work in the setting when the data is vertically partitioned across the clients, i.e., the clients share overlapping users with different features. But, there was almost no work when the data is horizontally partitioned across the clients, where the clients share common set of features, but they have disjoint samples. This is specifically true for

decision trees. A recent survey article on decision tree-based federated approaches also mentions the same fact [WG24]. Hence in **RQ1**, our motivation is to propose an FL framework with Decision trees for the horizontally partitioned data, which is both privacy-preserving, and communication and/or computationally-efficient. From **RQ2**, our objective is to contribute to the domain of PPML, where we tailored an attack model to quantify the privacy. Basically, we quantify how much information about the user's data is encapsulated by the machine learning models. We specifically investigate PCA in our case. We tailor a data reconstruction attack, which assumes that the attacker has intercepted the eigenvectors by eavesdropping the communication channel where the eigenvectors are being shared from one client to some other client or to a central server. Eigenvectors encapsulate the information about the data. Therefore, we evaluate the proximity of reconstructed samples with the original samples to quantify the privacy. The objective of **RQ3** is to quantify the privacy of synthetic datasets. Since, synthetic datasets do not contain the data of real users, i.e., there is one to one correspondence between the real data of users, and the synthetic data. Not having one to one correspondence between the real data of users, and the synthetic data of users exclude the chance of identity disclosure. But, not having identity disclosure does not exclude the chance of having attribute disclosure. Hence, we tailored an attack model, in which the attacker has some background information about a participant or a target individual in a database, and the attacker has access to one synthesized version of the original database, and the goal of the attacker is to infer the sensitive information of the target individual. In **RQ4**, our objective is to determine a sweet spot between the utility, and the privacy, which we do by quantifying privacy, and utility in different cases with different protection mechanisms.

Therefore, our research concentrates on developing privacy-preserving FL algorithms for decision trees, devising ways to quantify privacy leakage for specific machine learning models, and quantify the privacy of synthetic dataset before making them public. We list our main contributions as follows.

- We developed a privacy-preserving framework to perform federated learning with decision trees
- We developed methods to quantify privacy of machine learning systems
- We developed a method to quantify privacy in the form of attribute inference risks associated with synthetic data

We show one to one correspondence among the concepts covered in each chapter of this thesis, and the research papers produced in Figure 1.1, which shows the outline of this thesis. Notice that, some concepts are used in more than a paper.

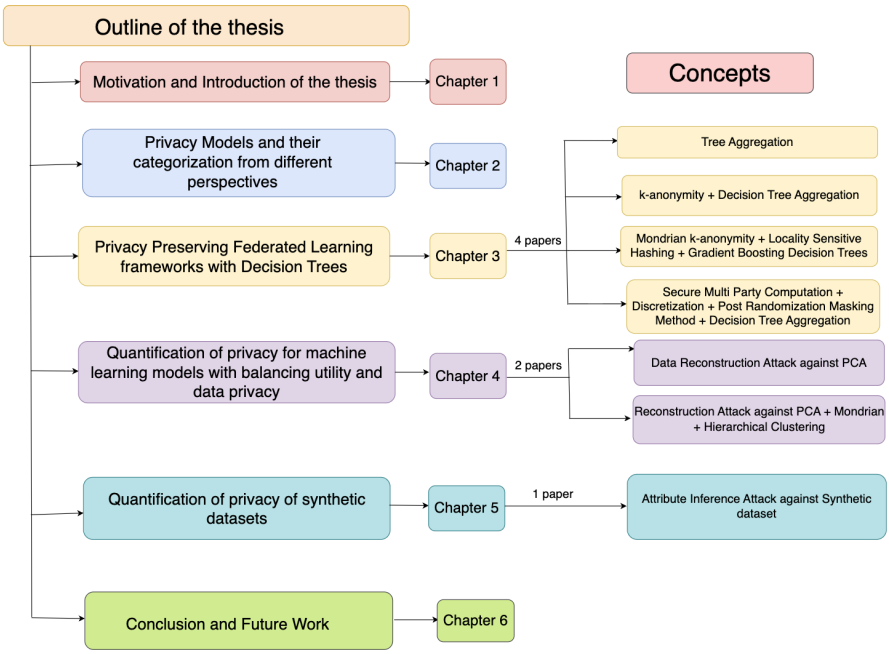


Figure 1.1: Thesis Outline

1.6 Research Method

In computer science thesis research, we mainly use a method called Design Science Research (DSR). It is way of tackling problems by the design, and development of innovative artifacts, which is commonly used in fields like information systems, computer science, and engineering [Pef+07]. We explain how it works in our context. First, we start by identifying a major problem in a our area, which lies at the the intersection of privacy, and machine learning for us. We define the objectives to solve the problem, which are mainly hitting a sweet spot between the utility, and privacy, and quantification of privacy. Then, there is Design and Development. We focus on coming up with new ways to help solve the problem we identified. This could be anything from making new software or algorithms to create new ways of doing things. We keep refining our ideas through an iterative design process. This means we make a first version, test it out, see what works and what does not, and then make it better. Once we have got something good or bad (as it is equally important to know the failure models or the things that do not work in research), we need to check how well it actually works. We do this through Evaluation. We evaluate our artifact with publicly available datasets, test how fast or accurate it runs, or compare it to other solutions out there. We also want to understand why it

works. So, we build theories to explain what we have done and why it matters. Lastly, we share what we have learned. We write about our process, what we have found, and what it means for others in research papers, or technical reports. In the end, it is all about making sure our work is understood and can be useful to others.

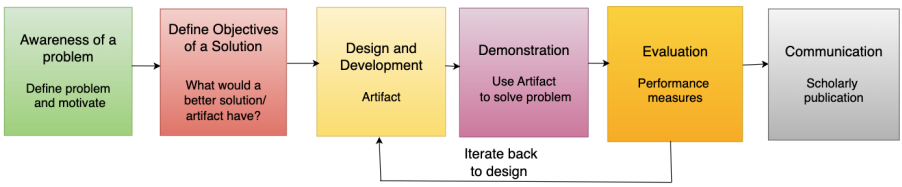


Figure 1.2: Design Science Research Methodology

1.7 Structure of the thesis

In Chapter 2, we review all the privacy models, including the methods to generate synthetic data. In Chapter 3, we introduce different algorithms to perform federated learning with decision trees, approaches of tree aggregation, privacy challenges in performing FL with decision trees and also our proposed federated learning framework with decision trees. In chapter 4, we review methods to quantify the privacy leakage of machine learning models by attacking them, which includes Membership Inference Attacks (MIA), and data reconstruction attacks, and then we explain our proposed data reconstruction attack against Principal Component Analysis, and we also discuss existing methods to quantify the privacy of synthetic data, followed by our proposed method to quantify the privacy leakage. Finally, we conclude our thesis in the last chapter with conclusions, and new directions for future research.

Chapter 2

Data Privacy Methods

> Anyone who steps back for a minute and observes our modern digital world might conclude that we have destroyed our privacy in exchange for convenience and false security.
>
> John Twelve Hawks

In an increasingly interconnected world, safeguarding personal data is more crucial than ever. Hence, we explore the domain of data privacy. In this chapter, we discuss all the concepts related with privacy models, which suggests how to govern the personal data according to the ongoing privacy regulations, which are also used in this thesis. Each privacy model has its own definition, a unique process, a unique way to quantify privacy, and have different objectives. There are different privacy models prevailing in the literature, including k-Anonymity [SS98], [Swe02], Differential Privacy(DP) [Dwo08], and Integral Privacy (IP) [TNG20]. Now, we will discuss in brief about the motives of each privacy model. k-Anonymity is applied on the data to obfuscate the data in a manner that a group of at least k data subjects have same quasi identifiers (publicly available attributes). In k-anonymous databases, a data subject is sort of "hidden" in a group of at least k-1 other individuals, thus shielding the data subjects from re-identification attacks. On the other hand, we have Differential Privacy, which is applied on the output of a query function made on a database, such that the output of a query does not change much due to the participation or non-participation of a data subject in a database. Then, we have Integral Privacy (IP), which focuses on the privacy of machine learning, and statistical models. IP is based on the principle that machine learning models recur, even when they are trained on different subsets of databases. We will focus on how these susbsets are created in Section 2.11 dedicated specifically to IP. If a ma-

chine learning model, also known as generator occurs k times, then we say that we have integral privacy à la k-Anonymity. There is also k-anonymous integral privacy, which means that each subset on which the machine learning model is trained contain at least k-elements. We also have generative networks, which learn the patterns from the original data, and the goal of generative networks is to create a data, which is structurally, and statistically similar to the original data, which is known as synthetic data [Goo+14].

We will discuss about each of the privacy models, in detail, later in this chapter. We also discuss different perspectives for the categorization of different protection methods, and methods to asses the disclosure risk for each protection method. It is to be noted that the protection methods we discuss here are for static databases. A database can change with time, and we can also have streaming data, like, data for smart grid applications. To handle dynamic, time series, and other databases, data protection methods discussed in this thesis need to be adapted to be able to apply them to different databases. We dedicate a separate chapter to synthetic datasets along with their privacy risks quantification, which is Chapter 5.

2.1 Categorization of Data Protection methods

The attributes in a dataset $\mathcal{D}$ can be classified into quasi identifiers, identifiers which are publicly available, and the others that contain sensitive information, which are called confidential attributes, and hence, not expected to be publicly available. The rest, which do not contain any sensitive information, are called non-confidential attributes. The goal of data protection methods is to protect quasi identifier. According to Domingo-Ferrer [Dom07], and Torra [Tor22], the categorization of data protection methods can be done on the basis of following factors.

- Whose privacy is being sought
- The computations to be done
- The number of data sources

The focus of the first dimension is on individuals or organizations whose data is to be protected. Data protection can be needed to a data subject or a respondent, who participated in some survey or interview, and their data is being collected. This is known as respondent's privacy. Data protection can also be needed by the owner of a collected database. The owner can be any organization, such as government agency, an insurance company, supermarket or hospitals. This is called holder's privacy. Data protection can also be needed by a user of some application. This is called user's privacy. E.g., suppose a user queries a search engine about something sensitive. In this case, there can be two types of privacy requirements by the user. The user may want to protect the query or the user may want to protect their identity. Also, a user can play

an active role in safeguarding their privacy or user can be passive. Active users make use of available privacy tools to protect their privacy themselves.

The second one is dependent on the type of computations to be done on the database. Differential Privacy (DP) is an example of computation driven approaches, as in this case, it is well-known beforehand what kind of analysis needs to be done on the database. Therefore, in the data protection method, the type of analysis to be done can be taken into consideration. There are also data-driven and result-driven approaches. In data-driven approaches, there is no prior knowledge about the type of analysis to be done on the database. Hence, the goal is to create a privacy-preserving version of the database. All the masking methods fall under data-driven approaches. In the result-driven approaches, the objective is to protect the result of the data mining techniques on the databases. E.g., when association rule mining is applied on the database to infer relevant rules consisting a set of items a user buys, and the goal is to avoid some inferences, which are sensitive.

The third dimension depends on the number of data sources, as there can be only one database corresponding to a data holder, or there can be multiple data holders, each having their own database, or a database can change over time, and we need to preserve the privacy of multiple releases of each database.

Also, the data protection methods can be categorized depending on how the original data is manipulated to build a protected dataset. A data protection method can be perturbative or non-perturbative in nature. We will now discuss various perturbative, and non-perturbative methods as shown in Figure 2.1.

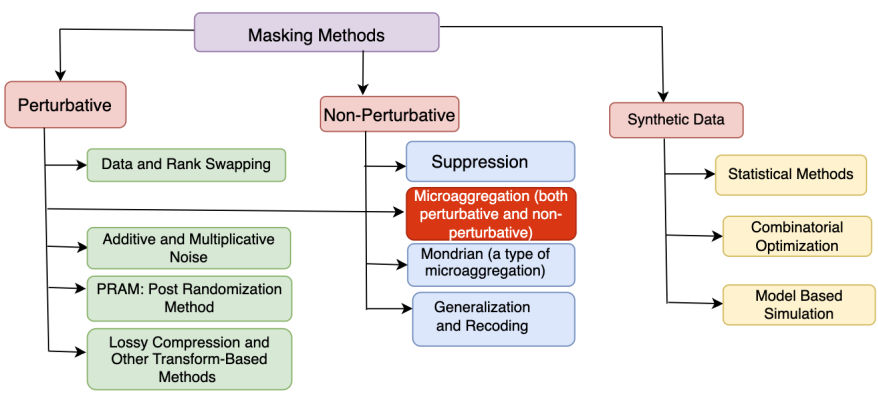


Figure 2.1: Categorization of Masking methods on the basis of the nature of transformation

2.2 Perturbative Methods

Perturbative methods involve the introduction of erroneous information in the original database. Hence, in perturbative methods, new combination of values are introduced in the protected dataset, which no longer corresponds to the ones in the original dataset. This obfuscation of data makes it difficult for intruders to learn the identity or any attribute information of a target individual. In this section, we review most of the prominent perturbative methods. Most of the perturbative methods we discuss here are implemented in the sdcMicro package in *R* [TKM15], and in the μ-Argus software [Hun+].

2.2.1 Rank Swapping

Rank swapping was originally defined for ordinal data (categorical data with a pre-arranged order) [Moo96]. But now, it can also be applied to any numerical data. In rank swapping, the data for each attribute is arranged in an ascending order, and then each element is swapped with any random element within a fixed range p. The parameter p defines the range of possible values with which to swap. Hence p controls the amount of disclosure risk in the data. Lower the value of p, larger is the privacy risk. p is usually set to a percentage of total number of records. There is no modification of values in rank swapping. Hence, the marginals of values in the original distribution is same as the marginals of values in the protected dataset. One of the main disadvantages of rank swapping method is that the correlation among the attributes is not preserved in the protected dataset, as each attribute is masked independently, and also the utility degrades with the increase in the value of p

The transparency principle states that along with the masked dataset, the information about the masking technique and the values of parameters used (such as p in the case of rank swapping) must be provided., but also provide the information that the data has been masked according to a particular method along with the value of parameters used in the masking process. According to the transparency principle, in the case of rank swapping, the disclosure of parameter p for the method rank swapping can be used by the attacker to tailor an intuitive attack model, which is basically a record linkage attack. The procedure by the attacker takes advantage of the constraint in rank swapping, where each value can be replaced with a value from a fixed set of records. To make record linkage attacks difficult for attackers, two other variants of rank swapping were proposed, which are p-distribution, and p-buckets rank swapping [NHT08]. For both of these variations of rank swapping, a value can be swapped with any record in a file, based on some probabilities. A higher probability is assigned to the values which are similar, and lower probabilities are assigned to dissimilar values. The assigned probability is never zero. The only difference between p-distribution, and p-bucket rank swapping is the definition of these probabilities.

In p-distribution rank swapping, the value of the i^{th} record is replaced with

a value of a record j, such that $j = i + r$, where r is a random value computed using the $N(0.5p, 0.5p)$ distribution.

In p-bucket rank swapping, values are ordered in ascending order, and then the ordered values are clustered into p buckets of equal size. Let B_1, B_2, ..., B_p denote these buckets. For each value v_i in a bucket B_r, we select a value a_j from a bucket B_s. This is a two-step process: first, we select the bucket B_s such that $s \geq r$. The selection of the bucket is done according to a predefined probability distribution.

$$Pr[B_s|B_r] = \frac{1}{K}\frac{1}{2^{s-r+1}}. \tag{2.1}$$

In the above equation 2.1, K is a normalization constant to make the sum of probabilities to 1. Once B_s is chosen, we select a value a_j from the bucket B_s, which is done using a uniform distribution on the elements of the bucket. In the case, $B_s = B_r, l > i$ is imposed.

2.2.2 Microaggregation

In microaggregation, clusters are formed, and a representative value is chosen for each cluster. Each value within the cluster is then replaced by this representative value [DM02]. If each cluster in the masked data contains at least k points, the data is considered k-anonymous. Microaggregation can be categorized as both perturbative and non-perturbative, depending on how the representative value is selected. If an interval is chosen as the representative, as done in Mondrian [LDR06], it is non-perturbative. This is because the replacement is with a generalized value. On the other hand, selecting the mean or median as the representative makes it perturbative, since individual values are altered. Several methods for microaggregation exist in the literature [MD98; DT05]. Some use a fixed group size, e.g., Maximum Distance to Average Vector (MDAV) [DM+06], where the micro-cluster size k is constant [DN93; DT05]. Alternatively, there are also data-oriented approaches where cluster sizes vary based on the data, e.g., Variable Maximum Distance to Average Vector (V-MDAV) [SMD06]. This flexible method generally results in lower utility loss than fixed-size approaches. We describe MDAV, and V-MDAV in Algorithm 1, and Algorithm 2, respectively.

2.2.3 Noise Addition and Multiplication

The original data is distorted by the addition or multiplication of random noise following the normal distribution or a Laplace distribution or even any other distribution. The noise following the Laplace distribution is preferred, as noise addition following the Laplace distribution satisfies Differential Privacy (DP), a privacy model, which is discussed later in detail. Refer Section 2.9. Hence, we are introducing errors in the original data to obtain masked data to make

Algorithm 1 MDAV-microaggregation with fixed micro-cluster size k

Require: Original dataset D, Micro-cluster size k
Ensure: Anonymized dataset D'

while $|D| \geq k$ **do**
    Compute the average record $\bar{d}$ from D
    Find the furthest record d_r from $\bar{d}$
    Select $k-1$ nearest points to d_r and form a micro-cluster c_i with d_r
    Remove the records in c_i from D
end while
if $|D| \geq k$ **then**
    Find the furthest record d_s from d_r
    Select $k-1$ nearest points to d_s and form a micro-cluster c_j with d_s
    Remove the records in c_j from D
end if
if $|D| > 0$ **then**
    Assign the remaining records in D to the last generated micro-cluster
end if
return Anonymized dataset D'

Algorithm 2 V-MDAV-microaggregation with variable micro-cluster sizes

Require: Original dataset D, Minimum micro-cluster size k
Ensure: Anonymized dataset D'

Compute pair-wise distance matrix on the dataset D
Compute the average record $\bar{d}$ from D
Compute the distance from each record to the average record $\bar{d}$
while $|D| > k-1$ **do**
    Find the furthest record d_r from $\bar{d}$
    Select k-1 nearest points to d_r and form a micro-cluster c_i including d_r
    Remove the records in c_i from D
    while $|c_i| \leq 2k-1$ **do**
        Get the nearest record r_{min} to c_i among the remaining records in D
        Record the distance in the previous step as d_{in}
        Compute the min distance d_{out} between r_{min} and the remaining records in D
        if $d_{in} < d_{out}$ **then**
            Add r_{min} to c_i
            Remove r_{min} from D
        end if
    end while
end while
if $|D| > 0$ **then**
    Assign the remaining records in D to the last generated micro-cluster
end if
return Anonymized dataset D'

the privacy breach difficult for the attackers. Perturbative methods make the original data inaccurate.

2.2.4 Post Randomisation Masking Method (PRAM)

PRAM was proposed for categorical data. PRAM models the randomised response mechanism via a transition matrix or a Markov matrix P [GKD98]. This process satisfies Differential Privacy (DP), which has a parameter ϵ, in which lower (0 to 1 is considered low in theory) values of ϵ corresponds to higher privacy. To understand PRAM, let us assume $C = \{c_i, c_2, c_3, \dots c_c\}$ be a set of categories, and the transition or Markov matrix P contains probabilities for changing one category into another. The transition matrix P also satisfies $\sum_{C_j \in C} P(c_i, c_j) = 1$. It is easy to conclude that we need that all the probabilities in the matrix P should be non-zero because otherwise, ϵ, the parameter in DP, will be infinite, which means no privacy at all from the perspective of differential privacy [Tor22]. There is also invariant PRAM, which ensures that the frequencies of categories do not change after applying invariant PRAM. The following definition for the transition matrix permits the user to implement PRAM. We denote $T_X(c_i)$ as frequency of category c_i, and $T_X(c_k)$ is the smallest frequency. Each value P_{ij} of the transition matrix P is filled using Equation 2.2

$$P(c_i, c_j) = \begin{cases} 1 - \frac{\theta T_X(c_k)}{T_X(c_i)} & \text{if } i = j \text{ and} \\ \frac{\theta T_X(c_k)}{(c-1)T_X(c_i)} & \text{if } i \neq j \end{cases} \tag{2.2}$$

In this definition, θ denotes the degree of perturbation. More particularly, θ equals to zero means no perturbation, and θ equals to one means full perturbation. Hence, a user can control the level of noise in the perturbed dataset using θ [Tor22].

2.3 Non-Perturbative Methods

In non-perturbative methods, there is no introduction of erroneous or false information in the original database. Hence, non-perturbative methods do not make the original data inaccurate, they make the data imprecise. There are mainly two categories of non-perturbative methods.

Generalisation and Recoding each value is replaced with a generalized value, which reduces the detail in the data.

Suppression In suppression, some values are hidden or suppressed.

As discussed earlier, Mondrian provides k-Anonymity by creating partitions/clusters of data, and in each partition, each value of the cluster is replaced by an interval. E.g., we have an age attribute, and the ages of toddlers from 0 to 3 years fall in one cluster. We can replace all the values of age attributes with the interval 0-3. Hence, Mondrian is a special case of microaggregation.

2.4 Synthetic Data

Masking methods discussed so far, which are perturbative and non-perturbative in nature modifies the original data to ensure the confidentiality. In synthetic data, the values in the original data area replaced by the artificial ones. In [Tor22], the methods to generate the synthetic data are bifurcated in the following three major classes.

- **Synthetic reconstruction** These methods rely on a dataset with a marginal distribution for the entire population and conditional probabilities for specific attributes, often derived from publicly available contingency tables. The data generation process involves multiple steps. Initially, individuals are either selected from a population or synthetically created. Subsequently, values for various attributes are iteratively assigned. Each attribute is added one at a time, ensuring that the synthetic values align with the conditional probabilities. For instance, after generating a record, the attribute *residential status* may be assigned (e.g., homeowner or tenant). In the next step, based on the *residential status*, further attributes like *property size* could be assigned. Iterative Proportional Fitting (IPF), a method developed in the 1930s, was historically used for constructing such datasets [BT13], [HW01]. More recent techniques have since been introduced for generating these datasets.

- **Combinatorial Approach** to synthetic data generation involves systematically creating new records based on combinations of different attributes in the original data. Instead of focusing on replicating statistical patterns, this method relies on generating all possible combinations of attribute values or selecting a representative sample from these combinations. This approach can be effective when the data contains categorical variables or when it is important to ensure diversity in the synthetic data. By generating combinations that may not even exist in the original dataset, this method provides privacy guarantees

- **Model based simulations** are gaining traction in research related with privacy-preserving machine learning, where generative models such as GANs (Generative Adversarial Networks) or create realistic, but fake data by training machine learning models that capture the data-generating process. Once trained, these models produce synthetic data that reflects the underlying patterns of the original data.

2.5 Information Loss

When protection methods are applied to datasets, there may be a decline in their utility. Domingo-Ferrer and Torra [DT01] proposed a formula to quantify Information Loss (IL). To evaluate the impact of masking on utility, we

compare the original dataset X with the masked dataset X'. The Information Loss can be calculated using several statistical measures- Mean Element Variation, Variation in Variable Averages, Variation in Covariances v_{ij}, Variation in Variances s_{ij}, and Variation in Correlations r_{ij}. The overall Information Loss (IL) is calculated using the following formula.

$$\text{IL} = 100 \times \left(\frac{1}{5} \left(\frac{1}{np} \sum_{i=1}^{n} \sum_{j=1}^{p} \left| \frac{x_{ij} - x'_{ij}}{x_{ij}} \right| + \frac{1}{p} \sum_{j=1}^{p} \left| \frac{\bar{x}_j - \bar{x}'_j}{\bar{x}_j} \right| + \right. \right. \\ \left. \left. \frac{1}{\frac{p(p+1)}{2}} \sum_{i \leq j} \left| \frac{v_{ij} - v'_{ij}}{v_{ij}} \right| + \frac{1}{p} \sum_{j=1}^{p} \left| \frac{s_j - s'_j}{s_j} \right| + \frac{1}{\frac{p(p-1)}{2}} \sum_{i<j} \left| \frac{r_{ij} - r'_{ij}}{r_{ij}} \right| \right) \right). \tag{2.3}$$

The above Equation 2.3 computes the average of the five components, scaled by 100, providing a comprehensive measure of how much the statistical properties of the original dataset are altered after masking or anonymization. Additionally, other methods can be applied to evaluate information loss that focus on the utility of the data from a machine learning perspective. For instance, information loss can be measured by comparing changes in accuracy, precision, recall, or F_1 score when training a machine learning model on the original dataset versus the masked dataset. Furthermore, information loss can also be assessed by comparing changes in clustering results between the original and masked datasets.

2.6 Disclosure Risk

Disclosure risk occurs, when the attacker combines the publicly available information, and the data analysis knowledge to infer the personal data of users. Disclosure risk is of mainly two kinds.

Identity Disclosure or re-identification, happens when an attacker identifies a record from a protected database. The attacker uses a publicly available database to link records and identify an individual in the protected one. Record linkage is the method/tool used for identity disclosure or re-identification.

Attribute Disclosure In attribute disclosure, the attacker has some background knowledge about a target individual, and the aim of the attacker is to infer some properties of a database.

Disclosure risk can be computed in various ways. For example, a Membership Inference Attack (MIA) reveals whether a particular data subject participated in the training of a machine learning model. Another method is reconstruction attacks, where the attacker aims to estimate the original data

points with close proximity. We discuss the procedures for both MIA and reconstruction attacks in detail in Chapter 4. In one of the early works [TD04], disclosure risk is evaluated by considering three distinct types of risks- Distance Linkage Disclosure (DLD), Probabilistic Linkage Disclosure (PLD), and Interval Disclosure (ID). To understand these risks, it is important to first grasp the concepts of distance-based and probabilistic-based record linkage.

Distance-based record linkage is used to link records that likely refer to the same entity across different data sources, such as the original and masked data files, even when there are discrepancies between the variables in these files. A distance metric—such as Euclidean distance, Gower's distance (which handles mixed data types), or Jaccard similarity—is used to measure the similarity between pairs of records. By adjusting a threshold for this similarity score, records that fall within an acceptable range are considered matches and are linked together.

Probabilistic record linkage, on the other hand, uses statistical models to estimate the likelihood that two records from the original and masked files refer to the same entity. This method relies on two key probabilities: the match probability (m-probability) and the unmatch probability (u-probability). The m-probability represents the likelihood that two records are the same given they have the same value for a particular feature, while the u-probability represents the likelihood that two records are different despite having the same value for a feature. These probabilities are combined to generate a score that determines whether the records should be linked.

The m-probability is denoted as $P(\text{agree}|\text{match})$, and the u-probability as $P(\text{agree}|\text{non-match})$.

In probabilistic record linkage, a likelihood ratio (Λ) quantifies the linkage status of record pairs.

$$\Lambda = \frac{P(\text{similarity in a feature}|\text{match})}{P(\text{similarity in a feature}|\text{non-match})}$$

Here, $P(\text{similarity in a feature}|\text{match})$ is the probability that the similarity in features occurs if the records are a match, while $P(\text{similarity in a feature}|\text{non-match})$ is the probability that the similarity occurs if the records are not a match.

With an understanding of these record linkage techniques, we can quantify disclosure risks as follows.

Distance Linkage Disclosure (DLD) This measures the average percentage of correctly matched records using distance-based record linkage.

Probabilistic Linkage Disclosure (PLD) This measures the average percentage of correctly linked records using probabilistic linkage methods.

Interval Disclosure (ID) This metric measures the percentage of original values that fall within specified intervals around their corresponding masked values, essentially assessing how close the original and masked sensitive values are.

To rank the data masking methods, an overall score is computed by combining Information Loss (IL) and the disclosure risks with specific weights:

$$\text{Overall Score} = 0.5 \times \text{IL} + 0.125 \times \text{DLD} + 0.125 \times \text{PLD} + 0.25 \times \text{ID}$$

This example gives Information Loss (IL) a weight of 0.5, emphasizing the importance of maintaining data utility. Disclosure risk receives a total weight of 0.5, divided equally between interval disclosure (ID) at 0.25 and record linkage (DLD and PLD) at a combined 0.25.

2.7 Risk-Utility Tradeoff

The risk-utility tradeoff highlights the balance between data disclosure and the utility of data for analysis and decision-making. In different contexts, weighting schemes can be adjusted to reflect specific privacy requirements. For example, in healthcare data, where patient confidentiality is crucial, a greater focus should be placed on minimizing disclosure risks. A revised weighting scheme ensures that any analysis conducted on healthcare data does not compromise patient privacy. Researchers and practitioners might assign a higher weight to DLD and PLD to capture the nuances of patient information sharing while maintaining data utility. In contrast, in financial data contexts, where accurate decision-making is essential, data utility may take precedence over strict privacy measures. Analysts might assign higher weights to Information Loss (IL) to ensure that data analysis remains effective and the insights derived are actionable. This could allow for a higher level of disclosure risk, as long as it does not hinder the generation of useful information for financial decisions. Ultimately, the weighting scheme should reflect the specific context and objectives of the analysis. Organizations must evaluate their privacy goals against analytical needs, ensuring the chosen scheme balances disclosure risks with the need for actionable insights. This dynamic process may also require ongoing adjustments as priorities evolve and new privacy challenges arise.

2.8 Privacy of Synthetic Data

Synthetic data is generated to mimic the statistical properties of real data without exposing any actual sensitive information. It can be categorized into two main types- partially synthetic data and fully synthetic data, each offering distinct privacy implications. Partially synthetic data combines original attributes with synthetic attributes. In this approach, certain sensitive attributes are replaced with synthetic values, while the remaining attributes retain their original form. In contrast, fully synthetic data is generated entirely from artificial values, with no original data points included. Both types of synthetic data are susceptible to record linkage attacks. For partially synthetic data, an attacker

may use the unchanged attributes to match synthetic records with individuals in the original dataset. If an attacker possesses external datasets that include identifiable information, they can potentially link records and re-identify individuals by exploiting the original attributes present in the dataset, as well as the patterns observed in the synthetic attributes. Even, fully synthetic data is not entirely immune to record linkage attacks. An attacker may attempt to establish connections between fully synthetic records and external datasets if the synthetic data retains certain statistical features similar to the original dataset.

While synthetic data presents a way to preserve privacy, both partially and fully synthetic data must be carefully evaluated for vulnerabilities to record linkage attacks, and other kinds of attacks. In Chapter 5 of this thesis, we discuss about the attacks against the synthetic data in more detail.

2.9 Privacy Models

2.9.1 k-anonymity

k-anonymity is one of the most popular privacy models, which is applied on the original datasets to perturb them in a manner such that, for each individual with a set of quasi-identifiers, there exists at least k-1 other individuals with the same values for quasi-identifiers. In k-anonymity, records with the same set if quasi-identifiers is called equivalence class [SS98], [Swe02]. Although, k-anonymity excludes the chance of identity disclosure, attribute disclosure is still possible without having identity disclosure. This is possible when all the values are same in sensitive attributes. This is called an homogeneity attack. If the attacker has some background knowledge about a target individual, and there is less heterogeneity in the values of sensitive attributes, then the attacker can leverage this to infer the values of sensitive attributes.

The Tables 2.1, 2.2, and 2.3 demonstrate how generalization and suppression techniques are applied to protect sensitive data in the context of k-anonymity. Table 2.1 presents the original dataset, where attributes such as Name, Age, Zip Code, and Disease are shown without any privacy-preserving transformations. This table poses a clear privacy risk, as individuals can be directly identified using quasi-identifiers like Age and Zip Code. To address this privacy risk, Table 2.2 illustrates k-anonymity with k=2 using generalization and suppression. In this anonymized table, age values are generalized into ranges like 30-35 and 35-40, ensuring that each record is indistinguishable from others in the same range. Zip Code values are partially suppressed, with the last digit replaced by a wildcard *. This masks the exact location while still retaining some geographic information. These transformations create equivalence classes of at least two individuals with the same quasi-identifiers, thus achieving $k = 2$ anonymity. This generalization and suppression also imply a trade-off between privacy and data utility, as some details are lost to ensure privacy. To achieve k=3 anonymity, further generalization is required, as shown in Table 2.3. Age

is generalized into a broader range (30-40), ensuring that at least three individuals share the same quasi-identifiers. Zip Code remains suppressed at the same level (1234*). To highlight the equivalence classes formed under each k-anonymity rule, the rows in the Tables 2.2, and 2.3 are colored.

Name	Age	Zip Code	Disease
Alice	34	12345	Flu
Bob	35	12346	Cold
Charlie	29	12345	Diabetes
David	38	12347	Flu
Emma	32	12346	Diabetes
Frank	35	12347	Cold
Grace	34	12345	Flu

Table 2.1: Original Data

Age	Zip Code	Disease
30-35	1234*	Flu
30-35	1234*	Cold
30-35	1234*	Diabetes
35-40	1234*	Flu
35-40	1234*	Cold
30-35	1234*	Flu

Table 2.2: k-Anonymous Data (k=2) Using Generalization and Suppression

Age	Zip Code	Disease
30-40	1234*	Flu
30-40	1234*	Cold
30-40	1234*	Diabetes
30-40	1234*	Flu
30-40	1234*	Diabetes
30-40	1234*	Cold

Table 2.3: k-Anonymous Data (k=3) Using Generalization and Suppression

2.9.2 k-Confusion and Probabilistic k-anonymity

In traditional k-anonymity, for any given record, there are at least $k-1$ other records with the same quasi-identifier values. However, in traditional k-anonymity, if all records in a k-anonymous group share the same sensitive

attribute, like a medical condition, an attacker can easily infer this information. Additionally, if an attacker has prior knowledge about a person, they might still be able to identify that person's record in a k-anonymous dataset.

k-confusion and probabilistic k-anonymity are a generalised version of k-anonymity. The concept of k-confusion was given by Stokes and Torra [ST12], and Probabilistic k-anonymity was given by Soria-Comas, and Domingo-Ferrer [SD12]. In k-anonymity, the probability of linking to an individual or identity disclosure risk or re-identification probability is $\frac{1}{k}$. The similarity between traditional k-anonymity, k-confusion, and probabilistic k-anonymity is that all these three models have the same probability of re-identification, which is $\frac{1}{k}$. But, at the same time, the requirement of all the values of the quasi-identifiers in each equivalence class being the same, is relaxed. This relaxation permits some variability in the records which can be used to improve the utility of the data while disclosure does not increase. For example, k-confusion ensures that properties like mean, and variance are preserved in the masked dataset. In this way, k-confusion also prevents homogeneity attack.

As we know that, introducing randomness or uncertainty in the data improves privacy. Probabilistic k-anonymity introduces uncertainty into the data via swapping of the attribute values in the equivalence class of size k. Hence, probabilistic k-anonymity and k-confusion offers better privacy protection by reducing the chances of attribute inference risks along with re-identification risks. Thus, it can balance the privacy-utility tradeoff in a better way than k-anonymity.

Now, we will discuss strict or constrained variants of k-anonymity. The focus of these variants is to avoid attribute disclosure.

2.9.3 l-Diversity

k-anonymity is prone to attribute inference attacks, when there is less heterogeneity in the values of sensitive attributes in an anonymity class. To overcome such attacks, l-Diversity was introduced. In l-Diversity, there should be diversity in the values of sensitive attributes in each group of k records [Mac+07]. This is done to confuse the attacker, and exclude the above mentioned vulnerabilities, particularly homogeneity attack or attribute inference attack. There are three existing methods in the literature, which satisfies l-Diversity, which are distinct l-Diversity, entropy l-Diversity, and recursive (c, l). Distinct l-Diversity makes sure that there is at least l distinct values in each equivalence class. Entropy l-Diversity is satisfied if for each equivalence class e, the entropy is $H(e) \geq l$, where entropy for a particular equivalence class e, and sensitive attribute with domain $\mathcal{D}$ is calculated as $H(e) = -\sum_{c \in D} \log p(e, D)$. Recursive (c, l) Diversity was introduced to control the frequency of the least frequent, and the most frequent values in the sensitive attributes. The idea is that the most frequent values should not occur too frequently, and the least frequent values should not occur too rarely.

Still, l-Diversity does not guarantee full-proof mitigation against attribute

inference attacks. Skewness attacks are possible against l-Diversity. The reason is if some values are rare in a sensitive attributes in the original dataset, the dataset protected using l-Diversity will introduce diverse values for those sensitive attributes. Hence, the attacker can leverage this information of having difference in distribution in the original file and the protected file for the sensitive attributes. If an individual links a particular individual to an equivalence class, there is a high chance the attacker can find this particular individual in the equivalence class.

2.9.4 t-Closeness

t-Closeness is another variant of k-anonymity, which was proposed to improve the privacy of k-anonymity in terms of attribute inference attacks. t-Closeness imposes a restriction on the distribution of attributes with sensitive values. According to t-Closeness, the distribution of the sensitive values in the equivalence class should be similar to the distribution of sensitive values in the complete dataset [LLV06].

Both, l-Diversity, and t-Closeness aim to reduce the chances of attribute disclosure. But, they are critiqued for impractical assumptions on the distribution of data for the sensitive attributes. Using l-Diversity, and t-Closeness also declines the utility of datasets in comparison with k-anonymity, as l-Diversity, and t-Closeness distort the correlation among the sensitive attributes and quasi identifiers in each equivalence class.

2.10 Differential Privacy

The intuition behind Differential Privacy (DP) is that the algorithm's output should be approximately the same whether or not any single individual's data is included in the input dataset. This means that the presence or absence of any one individual's data does not significantly influence the result, providing plausible deniability for individuals within the dataset [DR+14]. Hence, according to the first categorization of privacy methods, we discussed in this chapter, DP is a computation-driven approach. A randomized algorithm $\mathcal{A}$ is said to be (ϵ, δ)-differentially private if for any two datasets D and D' that differ in exactly one element (neighboring datasets), and for any subset of outputs $S \subseteq \text{Range}(\mathcal{A})$, the following condition holds:

$$\Pr[\mathcal{A}(D) \in S] \leq e^{\epsilon} \Pr[\mathcal{A}(D') \in S] + \delta. \tag{2.4}$$

In this expression, ϵ (epsilon) is a non-negative parameter that quantifies the privacy loss. Smaller values of ϵ correspond to stronger privacy guarantees. δ (delta) is a small positive parameter that accounts for a probability of failure in maintaining the privacy guarantee, often set to a very small value (e.g., 10^{-6}).

There are several mechanisms to achieve differential privacy. We discuss here Laplace mechanism, and Gaussian mechanism. The Laplace mechanism adds noise drawn from the Laplace distribution to the query result. If the query function f has sensitivity Δf (the maximum change in the output when a single input is changed), then the Laplace mechanism defines:

$$\mathcal{A}(D) = f(D) + \text{Lap}\left(\frac{\Delta f}{\epsilon}\right). \tag{2.5}$$

The gaussian mechanism adds noise drawn from the gaussian (normal) distribution to the query result. It is appropriate for achieving (ϵ, δ)-differential privacy when $\delta > 0$. The gaussian mechanism defines:

$$\mathcal{A}(D) = f(D) + \mathcal{N}(0, \sigma^2), \tag{2.6}$$

where σ is chosen based on ϵ, δ, and the sensitivity Δf.

Although DP does not make any assumptions about the data, the addition of noise in DP is dependent on the sensitivity of query function. Hence, DP provides meaningless output if the query function is highly sensitive. DP is critiqued, as it is not easily applicable everywhere like k-anonymity, and their variants. For each query function, we have to find a carefully calibrated noise.

Combining multiple differentially private mechanisms leads to the concepts of parallel and serial compositions in differential privacy. Parallel composition refers to applying differentially private mechanisms to disjoint subsets of the data. If we have a dataset divided into non-overlapping subsets, and each subset is processed using a differentially private mechanism independently, the overall privacy guarantee is governed by the maximum privacy loss among these mechanisms. For example, if we have a dataset D split into two disjoint subsets D_1 and D_2, and we apply an ϵ_1-differentially private mechanism to D_1 and an ϵ_2-differentially private mechanism to D_2, the overall privacy guarantee for the combined mechanism is $\max(\epsilon_1, \epsilon_2)$-differentially private. Serial composition refers to applying differentially private mechanisms sequentially on the same dataset or overlapping subsets. When mechanisms are applied in sequence, the overall privacy loss accumulates. E.g., if we apply an ϵ_1-differentially private mechanism and then an ϵ_2-differentially private mechanism to the same dataset, the total privacy loss is additive. Thus, the overall privacy guarantee is $(\epsilon_1+\epsilon_2)$-differentially private.

There is also Rényi Differential Privacy (RDP) in the literature, which is a relaxation of the standard differential privacy definition that uses Rényi divergence, a measure from information theory, to quantify privacy loss [Mir17]. RDP provides a more flexible framework that can lead to tighter privacy bounds in some contexts.

Rényi Differential Privacy

Rényi divergence $D_\alpha(P\|Q)$ of order $\alpha > 1$ between two probability distributions P and Q is defined as

$$D_\alpha(P\|Q) = \frac{1}{\alpha - 1} \log \left(\sum_x P(x)^\alpha Q(x)^{1-\alpha} \right). \tag{2.7}$$

For Rényi Differential Privacy, a mechanism M satisfies (α, ϵ)-RDP if for all neighboring datasets D and D'

$$D_\alpha(M(D)\|M(D')) \leq \epsilon. \tag{2.8}$$

This means that the Rényi divergence of order α between the distributions induced by M on D and D' is at most ϵ. RDP is also closed under composition, which means that if we have mechanisms satisfying RDP guarantees, their combination also satisfies an RDP guarantee.

2.11 Integral Privacy

Integral Privacy (IP) was introduced to protect the inferences drawn from machine learning and statistical models [TN16]. Introducing randomness or uncertainty enhance the privacy of the system. The objective of IP model is also to maximize the uncertainty of an attacker about any inferences, which can be deduced about the training data or the modifications done on them. Consider a dataset $\mathcal{D}$, and we apply modifications on $\mathcal{D}$ by inserting some records, and create a dataset $\mathcal{D}'$. Hence, $\mathcal{D}' = \mathcal{D} + r$. Using a machine learning model on $\mathcal{D}$, and $\mathcal{D}'$, we generate two models $\mathcal{M}$, and $\mathcal{M}'$ respectively.

An algorithm satisfies IP if the set of possible modifications, which is basically r is "large" enough so that the adversary cannot, with high confidence infer the exact set of r, which resulted in the transformation from $\mathcal{M}$ to $\mathcal{M}'$. This large was defined on the basis of how many times a model recurs, and how many are the number of sets on which the machine learning model can be trained. If a machine learning model occurs k times, then we say that we have integral privacy à la k-anonymity. The datasets that cause the machine learning model to recur are referred to as generators. There is also k-anonymous integral privacy, which means that there are at least k ways to generate the model, or equivalently there are at least k generators. The goal is to increase the uncertainty for the attacker by ensuring that the transformation from $\mathcal{M}$ to $\mathcal{M}'$ is not done by few dominating points.

IP provides machine learning model recommendation, and the recommendation is made on the basis of recurrence. If a model recurs "enough", then we say that the model is integrally private. This "enough" is decided according to the privacy requirements. The recurrence of machine learning models is driven by their ability to generalize across datasets while avoiding overfitting. Well-generalized models tend to recur across different datasets, meaning that they can learn consistent patterns without being overly influenced by any single dataset [TNG20]. This recurrence of models, even when trained on diverse

datasets, helps ensure that no single instance of the data is overly influencing the model, thus protecting the privacy of individual data points.

IP has been successfully applied to various machine learning models. For instance, Varshney *et al.* [VT23] demonstrated that neural networks can maintain Integral Privacy by recurrently learning patterns from different datasets without overfitting to any specific set of training data. Similarly, Senavirathne *et al.* [ST18] applied Integral Privacy to linear regression models, ensuring well generalized coefficients of models across different datasets without compromising the privacy of individuals within any single dataset. On the similar lines, decision trees are also shown to be made integrally private by ensuring that the same structure of splits in the tree reappeared across different datasets [ST19]. This consistency ensures that no sensitive individual data point overly influences the decision-making process. Most recently, Integral Privacy has also been applied to Support Vector Machines (SVMs), demonstrating SVMs can be trained to find optimal separating hyperplanes across different datasets while preserving privacy [KVT23].

We discussed all the privacy-preserving models prevalent in the literature, which includes k-anonymity, differential privacy, integral privacy and their variants. In our discussion, we stress upon the strengths, and shortcomings of each privacy model, and we discuss when to use which privacy model. We also discussed a variety of protection methods in this chapter along with their categorization according to different perspectives. Our discussion in this chapter leads us to the development of different privacy preserving systems, and the privacy, and utility quantification of these systems.

Chapter 3

Privacy-Preserving Federated Learning with Decision Trees

> Privacy is not something that I'm merely entitled to, it's an absolute prerequisite.
>
> Marlon Brando

Decision trees are the most interpretable machine learning models. As humans, we have explanations for our everyday decisions in casual or critical situations. Humans explain the reasoning for their decisions, especially when the decisions are taken for a critical situation. Sometimes, we also build trust on decisions of other humans too, because they provide us with understandable explanations. Similarly, to trust AI tools, and have satisfactory experiences in using them, we need explanations. So, in this chapter, we focus on decision trees. Decision trees comprises a hierarchy of nodes, where each node is a question with a split attribute and a split value [PCZ09]. In order too build a decision tree, the order of split attributes is decided on the basis of entropy-based concepts from information theory, such as Information Gain, and Gini Index. The best split is the one, which maximizes the reduction in Gini impurity, or the split that leads to the least entropy. Predictions for new test samples are obtained by traversing through the decision tree, and reaching a leaf node, which has a class label.

Bagging (random forest) and boosting techniques for decision trees improves the accuracy in comparison with decision trees, but at the expense of turning non-explainable, that is why, we focus on performing Federated Learning (FL) with decision trees. FL is a framework to learn a machine learning model in a distributed setting. Multiple clients participate in an FL framework to

learn a model in a collaborative manner. Typically, an FL framework involves four steps. In Step 1, the central server sends a global initialised model to the clients. In Step 2, each client trains the machine learning model on their data, and sends the updates of its machine learning model to the server. In Step 3, the central server aggregates the model updates, and in Step 4, the central server sends them back to the clients. Steps 2-4 goes on until the model convergence is reached. Originally, FL was introduced for deep learning models, where each client participating in the federation receive a neural network with a common structure from the server in the step 1 of FL. Each client trains a neural network on their training data, as stated in step 2. In step 3, the clients send their model information, which is basically, model weights, and biases to a central server. The central server aggregates the model updates, and sends the aggregated information back to the clients. Each client trains a neural network again corresponding to the updated model information, and sends the information to the clients. This process goes on until the model convergence is reached or a fixed number of iterations are done [McM+17].

FL is of three types, on the basis of how the data is partitioned among distributed clients. The three types are: Horizontal Federated Learning (HFL), Vertical Federated Learning (VFL), and Transfer Federated Learning (TFL). In HFL, the data is sample-partitioned across the clients, and distributed clients share common feature space. In VFL, the data is feature-partitioned, and distributed clients share the data of overlapping users. In TFL, the data is neither feature-partitioned, nor sample-partitioned. TFL combines the concept of transfer learning and FL. In TFL, the local clients leverage the pre-trained global model, which is trained on a big public dataset. The distributed clients utilise the pre-trained global model on their datasets to obtain a fine tuned model.

The aggregation step is the core of FL, where the contributions of each client participating in the federation are taken into account. Since neural networks contain numerical model information, aggregation functions such as mean, trimmed mean, median, and other numerical aggregators can be utilized in this step. On the other hand, in FL with decision trees, each distributed client trains a decision tree classifier, and the model information includes split attributes and split values, where the split attribute is the name of the feature. Before aggregation, each client should use similar criteria for splitting the dataset during the decision tree building process, so that the aggregation of split attributes and split values makes sense. Since sharing split attributes and split values of each node in a decision tree can potentially reveal a lot about the data, privacy becomes a major concern when such information is shared with other clients (in a peer-to-peer architecture) or a central server (in a master-slave architecture). Thus, the objective of this chapter is stated as follows.

> How to perform FL with decision trees while preserving the privacy of users?

Our objective is to introduce FL frameworks with decision trees, which are,

to some extent, analogous to FL with neural networks. Gambs *et al.* [GGH12] showed that reconstruction attack is possible due to sharing of classifier. So, it is crucial to incorporate privacy while sharing the information about decision trees. Therefore, we had two main challenges to address the above research question-

"The challenge one is to know how to aggregate decision trees, and the other one is to know how to preserve the privacy of users while still being able to learn a collaborative decision tree model with a balanced tradeoff between the utility and privacy." Now, we discuss how we tackle these challenges.

3.1 Construction of Decision Trees

Decision trees are a supervised learning method used for classification and regression tasks. The construction of a decision tree involves partitioning the dataset into subsets based on attribute values. This process continues recursively until a stopping criterion is reached. The primary aim is to develop a tree structure that accurately represents the decision boundaries within the data. To construct a decision tree, the first step is to select a splitting criterion. Common criteria include Gini impurity and information gain. Gini impurity, often used for classification tasks, is calculated as follows.

$$Gini(D) = 1 - \sum_{i=1}^{C} p_i^2 \tag{3.1}$$

where p_i is the probability of a randomly selected instance being classified into class i. Information gain measures how much information a feature provides about the class labels and is computed as follows.

$$IG(D, A) = H(D) - \sum_{v \in Values(A)} \frac{|D_v|}{|D|} H(D_v) \tag{3.2}$$

where $H(D)$ represents the entropy of dataset D, and D_v is the subset of D for which attribute A has value v.

After determining a suitable splitting criterion, the dataset is divided into subsets according to this criterion, creating branches in the tree. This recursive splitting continues until a stopping condition is met, such as reaching a maximum tree depth, having too few samples in a node, or achieving negligible improvement in the chosen criterion. Once the recursive process concludes, leaf nodes are formed, which either represents the predicted class for classification tasks or the average value for regression tasks. Finally, the resulting decision tree provide an interpretable model that captures the underlying patterns in the data [Bre+86].

3.2 Aggregation of Trees

Aggregation of decision trees trained by distributed clients is the core step to perform FL with decision trees. This aggregation needs to be done by the central server in a master-slave architecture of FL. None of the existing algorithms to perform FL with decision trees were doing aggregation of decision trees. Therefore, we explored the aggregation approaches for decision tree classifier, as well as other hierarchical structures in our literature survey [KT21a] to obtain the answer of our aforementioned research question. The classification of methods for tree fusion/merging can done as follows [KT21a].

- **Tree selection** To select one of the existing trees, the one that is the most similar to the given ones or the one that appears most often (plurality rule, voting) [WL19]. There are two kinds of similarity: syntactic, and semantic. When considering syntactic similarity, the tree structure that occurs the most often is selected. When considering semantic similarity, the tree structure that gives the best accuracy or any other performance metric on a validation set is selected as the representative tree. Although, the approach of tree selection is computationally efficient, it does not take into account the contribution of trees with different structures.

- **Decision Tree Aggregation** Build a new tree that combines the given ones [Put+14], [FL20]. The trees can be merged in the following ways.

  - **Step 1 Node Expansion**
    - While there are nodes to expand, start by receiving options for each node. These options include possible queries or deciding the node as a leaf.
  - **Step 2 Vote on Options**
    - Select the option with the highest number of votes. This could be either a query for further expansion or designating the node as a leaf.
  - **Step 3 Node Expansion or Leaf Assignment**
    - If the selected option is not to be a leaf, expand the node with the most voted query and proceed to further nodes.

  - Aggregation based on properties- To build a tree that satisfies some properties. E.g., minimizes a cost function. Each decision tree defines a set of regions in the space of data. Then, aggregation of decision trees is defined in terms of an aggregation of these regions. This approach is used in [FL20] considering trees for classification problems. For a given tree, each region has associated a class. Then, the regions of different trees are combined using the majority rule. The aggregated tree represents the aggregated regions. If we consider the application of all trees to a given instance, and the classification

of this instance by means of a majority voting, the aggregated tree is loss-less. That is, both approaches lead to the same result.

This approach is followed by Hall, Chawla and Bowyer [HCB97], Bursteinas and Long [BL01] (regions are called hypercubes), Andrzejak, Langner and Zabala [ALZ13] (sets of iso-parallel boxes), and Strecht, Moreira, and Soares [SMS14] (decision regions).

– The same fundamental applies to *hierarchical trees* (hierarchical structures of objects) and to decision trees. For a given dataset, any decision tree can be understood as a hierarchical structure of objects (the records in the dataset). That is, given a set of records, the tree can be used for classifying all records, and, thus, assigning each record to a leaf of the tree. Then, we can ignore the queries associated to each node and just consider the hierarchical structure.

  Therefore, fusion of decision trees can be defined in terms of the fusion of hierarchical structures [BM86],[Hul+20], [MM81]. Finally, the aggregated decision tree needs to be defined from the hierarchical structure. To do so, we need to associate a query to each node. The query should correctly classify the objects assigned to each children in the hierarchy.

  While any (consistent) arbitrary hierarchical structure can be transformed into a decision tree, selecting appropriate queries to the nodes may be non trivial. Complex queries requiring several attributes and values may be required. When nodes in decision trees are constrained to have queries of the form $A_i \in VS_j$ (for an attribute A_i and a set of values VS_j), it may be impossible to transform an aggregated hierarchy to a decision tree. In this case, approximations are required.

– Each decision tree can be understood as a set of logical rules. E.g., a decision tree with 5 internal nodes (i.e., queries) can correspond to the following rules:

  * $Q1 \wedge Q2 \rightarrow C_1$
  * $Q1 \wedge Q3 \wedge Q4 \rightarrow C_2$
  * $Q1 \wedge Q3 \wedge Q5 \rightarrow C_3$

  Then, we can consider merging of the logical expressions.

All these approaches assume that decision trees are available for inspection by the aggregation approach. This is naturally a drawback from a privacy perspective [KT21a]. Through our survey on tree aggregation approaches, we reached our goal, as we found out the decision tree aggregation approach proposed by Fan *et al.* [FL20] that we utilised in our proposed FL frameworks. Before jumping over explaining our proposed FL frameworks, we discuss the creation of heterogeneous data partitions to test the efficacy of our proposed FL frameworks in our next section.

3.3 Data Heterogeneity

Data heterogeneity is one of the challenges in FL. Data heterogeneity means that each client can own a data from entirely different distribution in comparison with some other client. Hence, the proposed FL frameworks should be tested on a non-i.i.d. To build non-i.i.d. partitions of a data (binary or multi-class), we used an optimization framework that makes sure that each client has an incomplete information about the overall data distribution of a complete dataset from which the partitions are created [Tor23]. We used this optimization framework to build non-i.i.d. partitions in our experiments. We explain the construction of non-i.i.d. partitions in the following subsection.

3.3.1 Systematic Non-i.i.d. Partitioning of a Dataset

In this section, we describe a systematic method for creating non-i.i.d. partitions of a dataset. For detailed information on this approach, refer to [Tor23]. We assume that the dataset comprises m records, with a fixed set of features, and these records are categorized into k distinct classes. Let m_i denote the number of records in the i-th class, so that $\sum_{i=1}^{k} m_i = m$.

A random allocation of records to clients would lead to i.i.d. data on each client if the original dataset is i.i.d. To ensure non-i.i.d. partitions, we propose allocating records to clients in such a way that not every client contains records from all classes. To achieve this, we define a parameter c, representing the number of classes assigned to each clients. Thus, the number of possible client types, where each client has records from exactly c different classes, is given by $T = \binom{k}{c} = \frac{k!}{c!(k-c)!}$. We call T the number of distinct client types. Additionally, let $nClients$ represent the number of clients of each type. Consequently, the total number of clients created is $d = nClients \cdot T$.

The next step involves assigning records to clients, which we model as an optimization problem. The variables to be optimized are the probabilities of assigning a record from class i to client j. Let p_{ji} denote the proportion of records from class i assigned to client j. Given that each client can only contain a limited number of classes, some probabilities must be zero, i.e., $p_{ji} = 0$ if no records from class i are assigned to client j. Define $\mathcal{Z}$ as the set of such zero probabilities. For instance, if client 1 contains only records from class 1 and class 2, then $p_{1i} \in \mathcal{Z}$ for $i \geq 3$.

Moreover, the probabilities for each client must sum to one, i.e., $\sum_{i=1}^{k} p_{ji} = 1$ for each client $j = 1, \ldots, d$. Additionally, to maintain the original class distributions, if class i has twice as many records as class h, the associated probabilities must reflect this ratio. Mathematically, this is expressed as:

$$\sum_{j=1}^{d} p_{ji} = \frac{d m_i}{m}, \quad \text{for each } i = 1, \ldots, k.$$

All probabilities must be non-negative, meaning $p_{ji} \geq 0$ for all j and i.

Since there are often many valid probability assignments that meet these constraints, we introduce an objective function to select an optimal solution. Ideally, we seek solutions where the probabilities are well-distributed across clients. For example, if a client is assigned two classes, we prefer a distribution of 0.5 for each class rather than 1.0 for one class and 0.0 for the other. This preference translates to avoiding solutions at the vertices of the feasible region. When $nClients > 1$, we also want different clients to have distinct class distributions. To achieve this, we use a quadratic objective function, as follows.

$$OF(p; \beta_{ji}) = (p_{ji} - \beta_{ji})^2 = p_{ji}^2 - 2\beta_{ji}p_{ji} + \beta_{ji}^2,$$

where β_{ji} is a random number sampled from a uniform distribution in $[0, 1]$. Recall, here j refers to the client, and i refers to the class. Setting β_{ji} to $1/c$ yields nearly equal probabilities, considering the constraints. Randomness ensures that clients of the same type have varied probabilities while remaining as close as possible to the random values and satisfying the constraints. The quadratic objective function can also be expressed as:

$$OF(p; \mathbf{B}) = p^T Q p + p^T L,$$

where $Q = I$ is the identity matrix of size $d \times k$ and $L = -2\mathbf{B}$, with $\mathbf{B} = (\beta_{11}, \beta_{12}, \ldots, \beta_{dk})$. In summary, the optimization problem is formulated as follows.

$$\begin{array}{ll} \text{Minimize} & p^T Q p + p^T L \\ \text{subject to} & \\ & \sum_{j=1}^{d} p_{ji} = \frac{dm_i}{m}, \quad \text{for each } i = 1, \ldots, k \\ & \sum_{i=1}^{k} p_{ji} = 1, \quad \text{for each } j = 1, \ldots, d \\ & p_{ji} \geq 0, \quad \text{for all } j = 1, \ldots, d \text{ and } i = 1, \ldots, k \\ & p_{ji} = 0, \quad \text{for each } p_{ji} \in \mathcal{Z} \end{array} \tag{3.3}$$

3.4 Proposed federated frameworks with decision trees for horizontally partitioned data

The idea of FL is to think locally, but act globally. This idea is fulfilled in FL, as each client shares the locally trained model on their private datasets with the central server, which the central serves aggregates, and sends it to the distributed clients for further updates or training rounds. In this way, a model is learnt in a collaborative manner without sharing the raw data.

Although the raw data of each client is not exposed, sharing of model information is enough to breach the individual privacy of each client. These privacy concerns are even high when dealing with a transparent machine learning model like decision trees. Keeping the privacy concerns associated with FL in mind, we proposed two FL frameworks with decision trees. Our proposed FL frameworks

[KT21b], [KT24] use the decision tree aggregation approach [FL20]. We will discuss each one of our proposed FL frameworks in the following subsections.

3.4.1 A k-anonymized federated framework with decision trees

To propose an FL framework with decision trees, we use a decision tree aggregation approach [FL20] for the collaboration among distributed clients. To aggregate decision trees, split attributes and split values of distributed decision trees are shared with a central server to compute an aggregated decision tree. As we discussed earlier, information about decision trees, which is basically split attribute and split values, can lead to the estimation of the original database [GGH12]. So, we should incorporate privacy models while building FL frameworks with decision trees. We show the flow diagram of our proposed approach of training decision trees in an FL setting in Figure 3.1. In our first FL framework with decision trees framework, we propose that each client first applies k-anonymity privacy model to its data prior to the training of decision trees [KT21b] [LDR06]. Mondrian provides k-anonymity to the clients participating in the federation [LDR06]. We describe the steps of Mondrian in Algorithm 3. As discussed in our previous Chapter 2, Mondrian is a special kind of microaggregation. To revise once again, we describe microaggregation, which has three steps. The step 1 involves the formation of clusters. In step 2, we select a representative value to represent each cluster. In step 3, the values in each cluster are replaced by the chosen representative value. The step 1 of microaggregation, which is formation of clusters, can be done in a variety of ways. To build clusters, Mondrian recursively splits the dataset based on the attribute with the largest range. The algorithm continues to partition the data until each cluster contains at least k records. Once the clusters are formed, the values within each cluster are replaced with an interval. Hence, Mondrian, originally is a non-perturbative method, as we replace each value with a more general value. Nevertheless, in our experiments, for the ease of use, we replace each value of the cluster with the mean of the cluster, which makes Mondrian a perturbative method in our case. After protecting the database, each client locally trains a decision tree classifier on its k-anonymous database. The nodes of the trained decision tree classifier are shared with the aggregator. The aggregator merges the decision tree level by level by computing the most frequent split attribute at each level, and then combing its split values. This process goes on until all the nodes to be merged are leaf nodes. In this way, each client learns a representative decision tree model by facilitating collaboration, and each client preserves its privacy using Mondrian, which offers k-anonymity. Each client can choose the value of k according to its own privacy requirement. By now, we already know from our Chapter 2 that higher the value of k, higher is the privacy.

Now, we will describe our experimental settings. We do not have real clients participating in the federation. So, we build partitions from an original single

Algorithm 3 Mondrian Algorithm for k-anonymity

Require: Dataset D, Privacy parameter k
Ensure: A k-anonymized dataset $\hat{D}$
Initialize $Q \leftarrow \{D\}$ (a queue containing the original dataset)
Initialize the anonymized dataset $\hat{D} \leftarrow \emptyset$
while Q is not empty **do**
    Pop a dataset D' from the front of Q
    if size of $D' < 2k$ **then**
        Add D' to $\hat{D}$ without further splitting
    else
        Choose attribute A with the widest normalized range in D'
        Split D' into two subsets, D_1 and D_2, using the median value of A
        if size of $D_1 \geq k$ and size of $D_2 \geq k$ **then**
            Push D_1 and D_2 into Q
        else
            Add D' to $\hat{D}$ without further splitting
        end if
    end if
end whilereturn $\hat{D}$

Table 3.1: Description of datasets

Dataset	**No. of Samples**	**No. of Attributes**
Diabetes	768	9
Adult	32561	14
Magic	19020	11
Bank	45211	14

database. We create 10 non-i.i.d. partitions from the original data using the approach in Section 3.3. These 10 non-i.i.d. partitions correspond to the data of 10 clients. After building the partitions of the dataset, each partition is segregated into training and testing dataset with 70% samples in the training set, and 30% samples in the testing set, which means each client has its own test dataset along with its training data. Each client anonymizes its training data, and tests the learnt federated/aggregated decision tree on its testing data. We evaluate the efficacy of our federated framework on four benchmark datasets from UCI machine learning repository, namely, Adult, Diabetes, Bank, and Magic [1]. We show the specifications of the datasets used in Table 3.1.

To start with, we decide which attributes should be the Quasi Identifiers (Q.I.D) or the publicly available attributes, and which ones are the sensitive or hidden attributes, as anonymization is applied on the Q.I.D.'s. From our results, we can infer the following points.

[1] https://archive.ics.uci.edu/

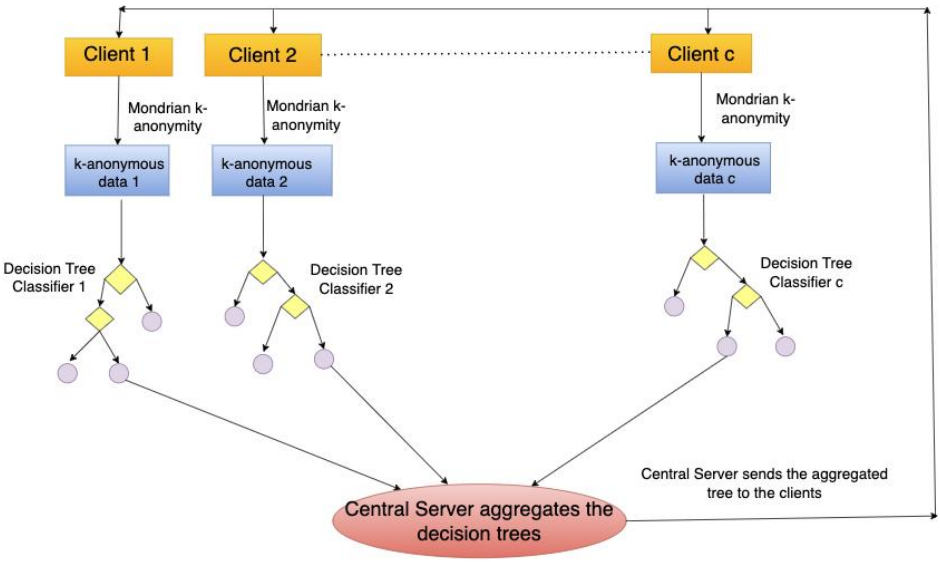


Figure 3.1: A k-anonymous FL framework with Decision Trees

- First, the predictive capability of random forest or tree aggregation is less affected by the noise injected due to k-anonymity, as the performance metrics do not drop much with increasing k. We experiment with different values of k = [2, 3, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

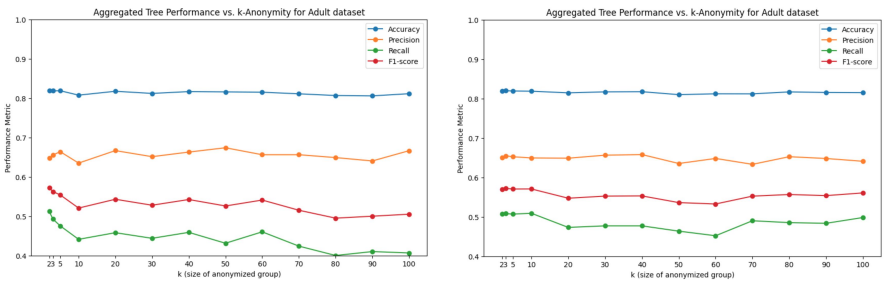
- Second, the choice of Q.I.D. also impacts the performance of model. Example- For the adult dataset, we considered Q.I.D. = [“age”, “work-class”, “education”, “occupation”, “hours-per-week”] for the results in Figure 3.2a. In Figure 3.2b, we considered Q.I.D. = [“education”, “occupation”, “hours-per-week”]. We got more stagnant results with increase in the value of k, when we have less number of Q.I.D.s in Figure 3.2b in comparison with Figure 3.2a.

- For diabetes dataset in Figure 3.3a, we used Q.I.D. =[“Pregnancies”, “Glucose”, “BloodPressure”, “SkinThickness”, “Insulin” , “BMI”, “DiabetesPedigreeFunction”, “Age”]. From k=40, the precision, recall, and F1-score touched zero. This is also because the total number of records in diabetes dataset is just 768, and k=40 to 100 introduces a lot of noise in the perturbed dataset, when all the features are considered into Q.I.D. Interestingly, when we removed “age” from the set of Q.I.D.’s, the precision,

recall, and F1-score sustained above zero even till $k = 100$. Therefore, the results are more stable when "age" is not included in the set of Q.I.D.s.

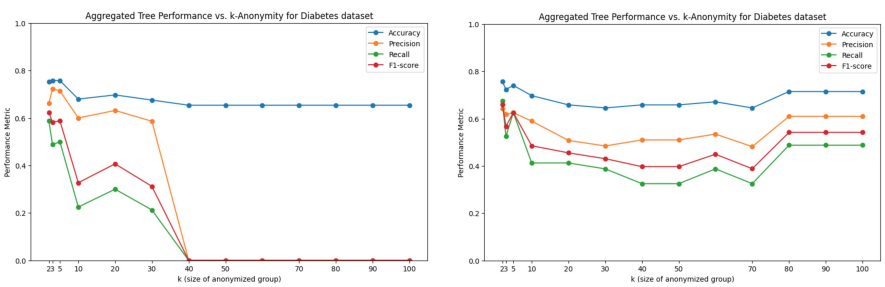
- For the bank dataset in Figure 3.4a, we used Q.I.D. = ["job", "age", "marital", "education", "housing", "duration"], and in Figure 3.4b, when we removed "age" from the set of Q.I.D.'s, the results are more stable or the performance metrics do not drop suddenly with increase in k than in Figure 3.4a, where Q.I.D. included. We got the similar trends for the magic dataset in Figures 3.5a, and 3.5b.

It is to be noted that the datasets we used for our experiments are imbalanced datasets, such as adult, bank, and diabetes. E.g. Diabetes dataset has 268 diabetic patients from the total of 768 records. So, it is important to consider other metrics than accuracy, such as precision, recall, and F1-score, as we did in our experimentation results. Now, we explain our second FL framework with decision trees.



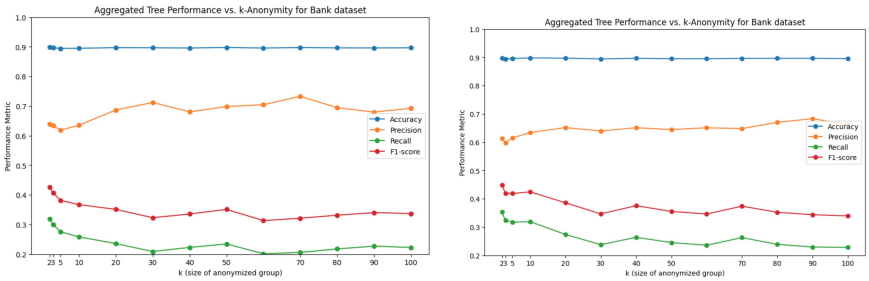
(a) More Q.I.D.'s (b) Less Q.I.D.'s

Figure 3.2: Adult dataset



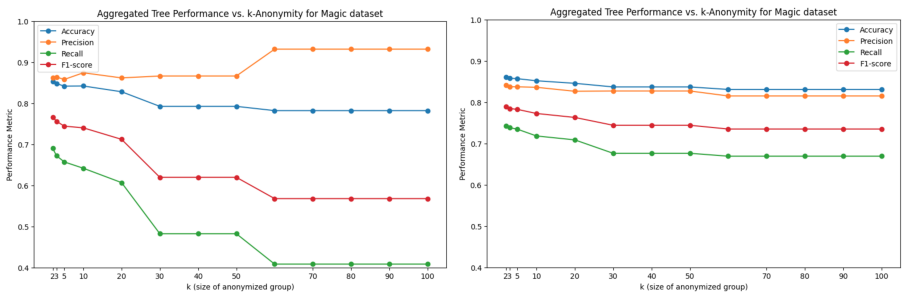
(a) More Q.I.D.'s (b) Less Q.I.D.'s

Figure 3.3: Diabetes dataset



(a) More Q.I.D.'s (b) Less Q.I.D.'s

Figure 3.4: Bank dataset



(a) More Q.I.D.'s (b) Less Q.I.D.'s

Figure 3.5: Magic dataset

3.4.2 DISCOLEAF- Personalized DIScretization of COntinuous Attributes for LEArning with Federated Decision Trees

In our second FL framework DISCOLEAF with decision trees, each client first discretizes their database, and then applies Post RAndomisation Masking Method (PRAM). After the discretization process and PRAM, each client trains a decision tree classifier, which is sent to the central server [KT24]. The central server aggregates the decision trees, and sends the merged tree to the distributed clients. It is to be noted that PRAM satisfies Differential Privacy (DP), which means that each client trains a differentially private decision tree. We show the process flow diagram of our proposed framework in Figure 3.6. The discretization approach we adopted in our proposed framework is also federated in nature, in which each client shares the maximum and minimum value of each feature to the central server, and the central server computes the global maximum, and the global minimum. This can be done in a Secure Multi Party Computation (SMPC) [Gol98], [AMP10] manner to ensure client's privacy. As, each client

creates the discretized bins according to the global minimum, and the global maximum from all the clients, it is meaningful to aggregate the machine learning model (which is decision tree in our case) trained on the dataset of each client. A client may deny to participate in the FL framework, if their minimum or maximum value is exclusive for some attribute. In this way, we offer personalized privacy, and collaborative learning to each client participating in the federation.

Discretization also reduces data complexity, leading to faster model training and inference [Gar+12]. In our work, we use Equal Width Discretization (EWD). EWD operates by calculating the data range (subtracting the minimum value from the maximum value) and then deciding on the number of bins a client wants to create. We employ the Freedman-Diaconis rule, which provides a heuristic for determining the optimal number of bins, considering the size of the data and variability. The formula for the Freedman-Diaconis rule is as follows.

$$\text{Number of bins (b)} = \frac{\text{Range}(\mathcal{D})}{2 \times \text{IQR}(\mathcal{D}) \times n^{-1/3}} \tag{3.4}$$

To perform discretization in a federated setting, we use an SMPC protocol [AMP10] to compute the largest (global maximum) and smallest (global minimum) values from the union of all clients' data. This protocol is repeated for each attribute to discretize the database in a federated manner for each client. Each client benefits from collaboration by knowing the global maximum and minimum. Once these values are known, each client can choose the number of bins (b) according to their preferences. A lower value of b implies lower privacy. After determining a suitable value of b, each client calculates the bin width by dividing the range (global maximum minus global minimum) by the number of bins, and then assigns each data point to the appropriate bin based on its value. This SMPC protocol [AMP10] is secure against malicious adversaries, as it ensures consistent inputs from parties by having a verification mechanism for secure and consistent inputs in each invocation, providing personalized privacy in the federated discretization process using the described SMPC protocol. An example for the verification mechanism is as follows. If P_1 sends an input of 60 for the age attribute and is instructed to delete inputs less than 60, P_1 cannot send a number less than 60 in future steps.

As shown in Figure 3.6, after discretization, we use PRAM. In our experiments, we use invariant PRAM, which ensures that the frequencies of categories do not change after applying invariant PRAM. See Section 2.2.4 for invariant PRAM. We denote $T_X(c_i)$ as the frequency of category c_i in the database $\mathcal{X}$, then these frequencies $T_X(c_i)$ will not be affected by masking, which is expected to produce good decision trees. Each value P_{ij} of the transition matrix P is filled using the Equation 2.2 in Chapter 2. Note that, θ equals to zero means no perturbation, and θ equals to one means full perturbation. Hence, θ permits a user to control the level of noise in the perturbed dataset [Tor22].

As we know, PRAM models the randomised response mechanism via a tran-

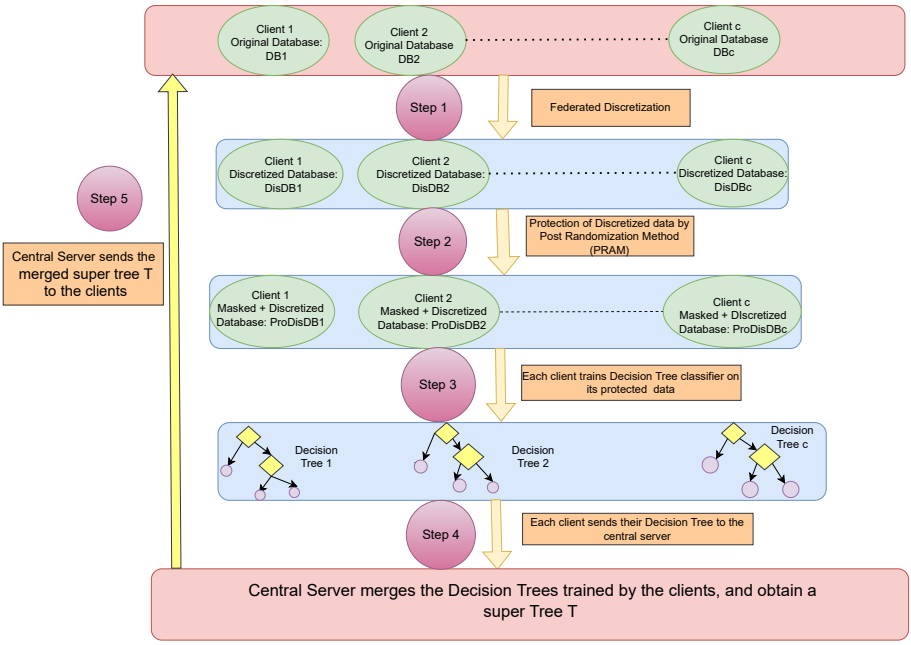


Figure 3.6: Process Flow of our proposed framework DISCOLEAF

sition/Markov matrix P. This process satisfies LDP for a given ϵ. From the θ parameter, we build a transition matrix P, which satisfies LDP. It is known (see e.g. [Tor22]) that a matrix P provides LDP with parameter ϵ when

$$\max_{i=1}^{c} \max_{j=1}^{c} \max_{l=1}^{c} \frac{P(X' = c_l|c_i)}{P(X' = c_l|c_j)} \leq e^{\epsilon}. \tag{3.5}$$

Then, given a transition matrix P, we have that the PRAM satisfies LDP for all ϵ, such that $\epsilon \geq \epsilon_0$, where ϵ_0 is defined by

$$\epsilon_0 = \log(\max_{i=1}^{c} \max_{j=1}^{c} \max_{l=1}^{c} \frac{P(X' = c_l|c_i)}{P(X' = c_l|c_j)}). \tag{3.6}$$

We perform our experiments on Adult, and Skin Segmentation datasets. Adult dataset has 32,561 records, and 14 attributes. The Skin Segmentation dataset has 24,198 records and 4 attributes. The Adult dataset consists of various demographic features such as age, education, occupation, marital status, race, sex, capital gain, capital loss, and hours worked per week. The target variable in this dataset is typically the income level, which is binary: "$>$50K" or "$\leq$50K". The "Skin Segmentation" dataset is used for image classification tasks. It consists of RGB (Red, Green, Blue) color values of pixels in images, where each pixel is labeled as either skin or non-skin. We show the plots of Kernel Density Estimation (KDE) for both the datasets in Figure 3.7, and 3.8.

There is diversity in the values of attributes for the adult, and skin segmentation dataset. Hence, discretization of attributes would make sense for these two datasets.

- Figure 3.9 illustrates how the performance of decision trees varies with their depth for the Adult and Skin Segmentation datasets. As the depth of the decision trees increases, there is a slight improvement in performance for both datasets. This indicates that the top features play a significant role in the predictive capability of the aggregated decision tree model. Consequently, this method of using decision trees in federated learning is scalable, especially if each client performs feature selection prior to training the decision trees.

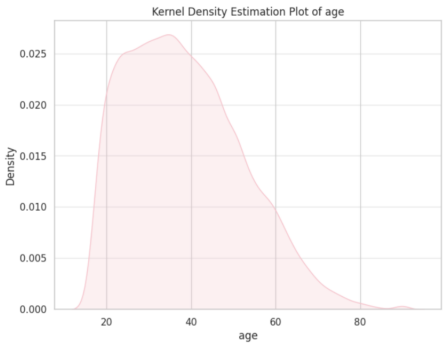
- Figure 3.10 shows the performance on each client's testing data without applying PRAM, for both i.i.d. and non-i.i.d. settings. The shaded region around each performance metric line represents the mean of performance metric $\pm$ standard deviation. There is more variance in the non-i.i.d. setting, where each client has different proportions of samples from each class. The performance of clients 2 and 8 is notably different due to the imbalanced distribution of class instances in the non-i.i.d. partitions. Specifically, client 2 has more samples from the minority class than client 8. These clients were chosen to highlight the impact of non-i.i.d. data distribution on federated learning performance.

- Figure 3.11 shows the performance on clients' testing data after applying PRAM in a non-i.i.d. setting, with varying levels of perturbation θ (0.1, 0.2, 0.3). To ensure interpretability, θ was kept the same across all clients in each experiment. However, clients can select θ based on their privacy needs. The results show that as θ increases, performance degrades, with accuracy dropping to around 0.75 at $\theta = 0.3$, which is inadequate for the imbalanced Adult dataset. Thus, the experiments stopped at $\theta = 0.3$.

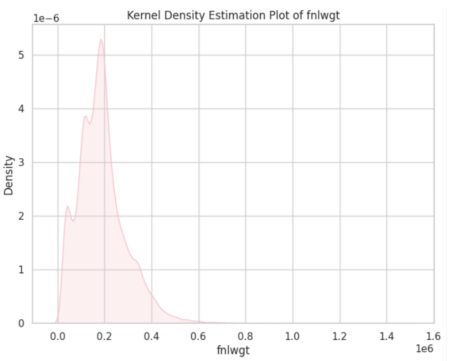
- Figure 3.12a, and 3.12b present the relationship between ϵ_0 and the number of categories in each attribute. Higher values of ϵ_0 are observed for attributes with more categories for a given θ. This trend is shown for both the Adult and Skin Segmentation datasets. Since θ indicates the level of perturbation, a higher θ results in a lower ϵ.

- Figure 3.12a illustrates the trend between ϵ_0 and the frequency of categories for the Adult dataset at different perturbation levels (θ). Attributes with the most categories, such as Education-num (16) and occupation (15), have higher ϵ_0 values compared to attributes like sex (2 categories) and race (5 categories).

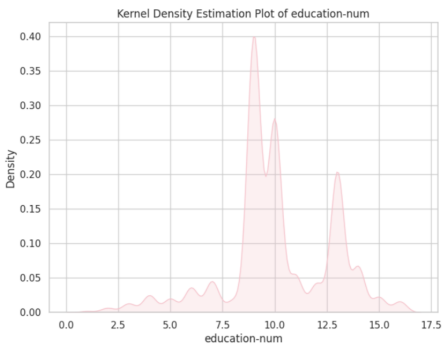
- Freedman-Diaconis rule generates many categories for different θ values, leading to high ϵ_0 values for a given θ, indicating lower privacy in Figure 3.12b for the Skin dataset.



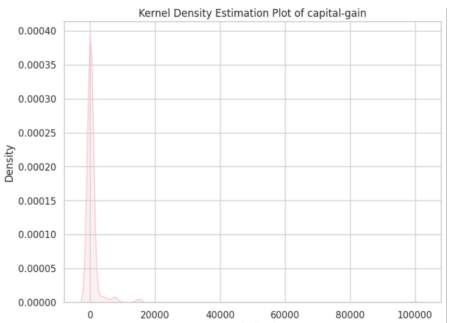
(a) Kernel Density Estimation (KDE) plot for Age attribute



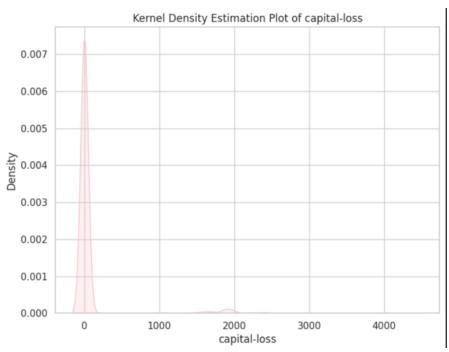
(b) KDE plot for final weight attribute



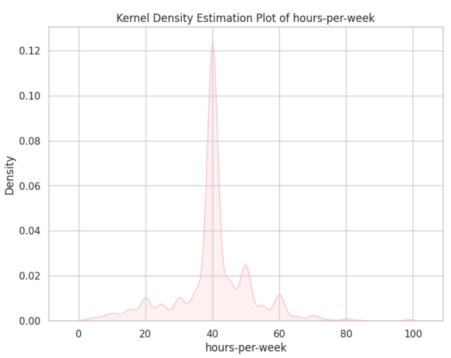
(c) KDE plot for number of years in education



(d) KDE plot for capital gains for an individual

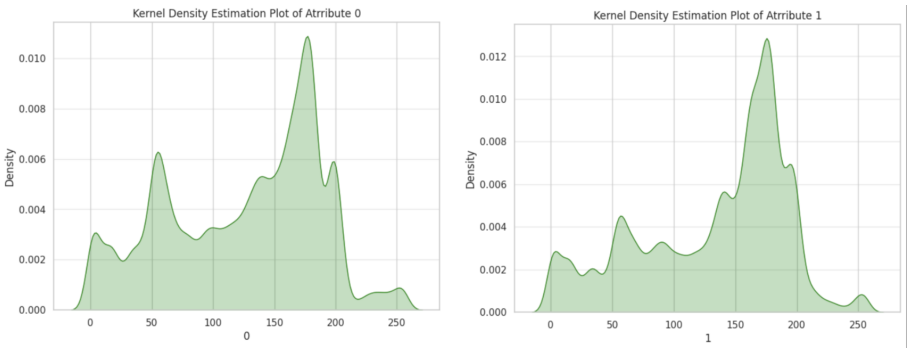


(e) KDE plot for capital loss for an individual

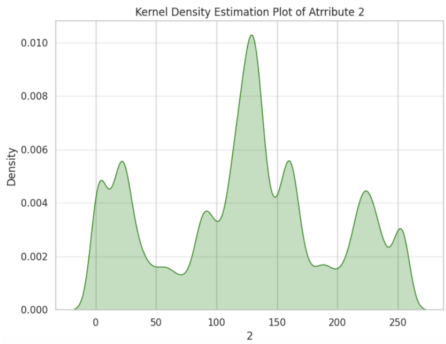


(f) KDE plot for hours per week

Figure 3.7: Kernel Density Estimation (KDE) Plots for the continuous attributes of the adult dataset

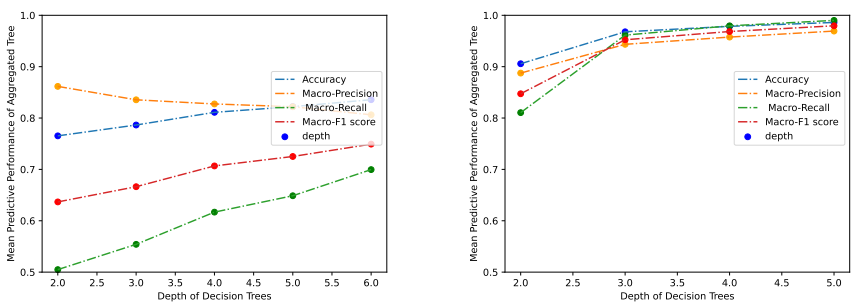


(a) KDE plot for Blue color space (b) KDE plot for Green color space



(c) KDE plot for Red color space

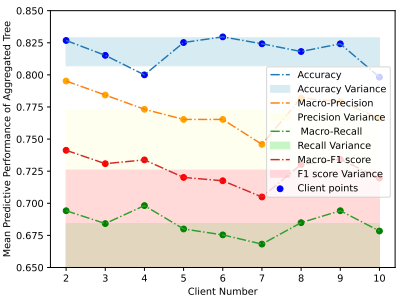
Figure 3.8: Kernel Density Estimation (KDE) Plots for the attributes of Skin Segmentation dataset



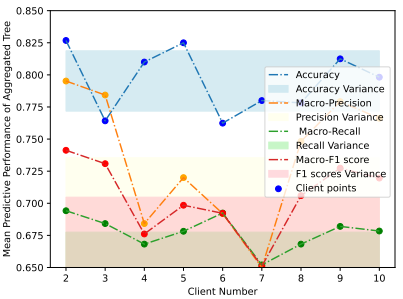
(a) Adult (b) Skin Segmentation

Figure 3.9: Performance Metrics vs. the depth of the Decision Tree

- In Figure 3.12b, the number of categories for the Skin Segmentation

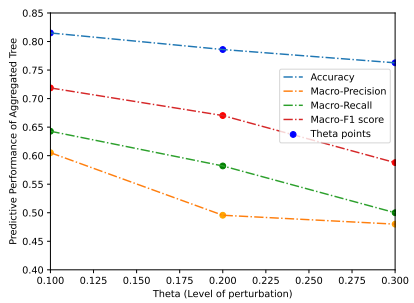


(a) Adult dataset with i.i.d. setting

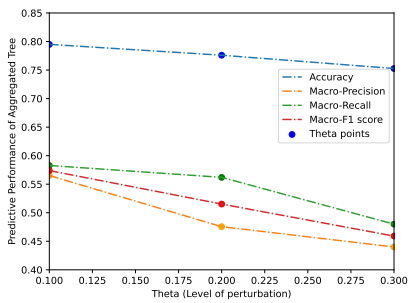


(b) Adult dataset with non-i.i.d. setting

Figure 3.10: Performance Metrics vs. Client number without applying PRAM



(a) Client 2 of Adult dataset



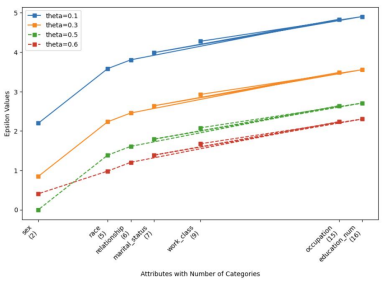
(b) Client 8 of Adult dataset

Figure 3.11: Performance Metrics vs. Theta after applying PRAM in a non-i.i.d. setting for Adult dataset

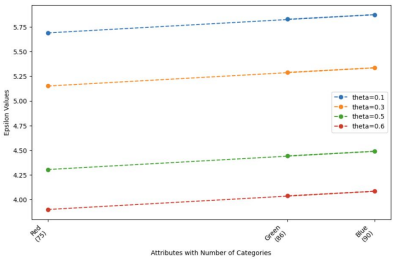
dataset was reduced. This reduction resulted in lower ϵ_0 values compared to those from the Freedman-Diaconis rule, hence higher privacy (lower ϵ_0) in Figure 3.12c.

- Figure 3.12d compares ϵ_0 values for real and fake category frequencies in each attribute. Real frequencies are evenly distributed, indicating no overly sensitive or outlier categories. For fake frequencies, a rare category with frequencies of 5, 50, and 500 was introduced, significantly increasing ϵ_0 values and thus lowering privacy. This experiment demonstrates that rarer categories lead to higher ϵ_0 values for a given θ, indicating lower privacy, while higher category frequencies result in lower ϵ_0 values, indicating higher privacy.

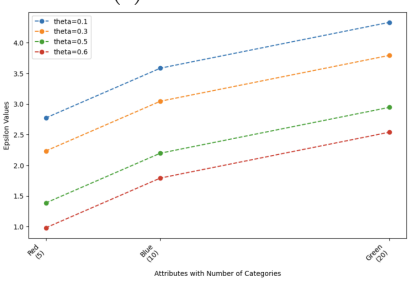
From this analysis, it is evident that for each perturbation level (θ), ϵ varies for each attribute. ϵ depends on the number of categories in each attribute



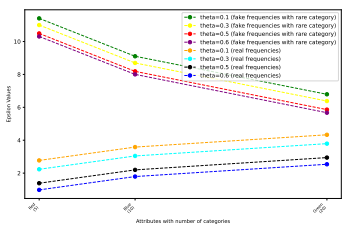
(a) Adult dataset



(b) No. of categories created using Freedman Diaconis rule for Skin dataset



(c) Without using Freedman Diaconis for Skin dataset



(d) Comparison between real and fake frequencies for Skin dataset

Figure 3.12: Epsilon Vs. Number of Categories for different perturbation levels

and the frequency of each category. Experimental results show that attributes with fewer categories have lower ϵ values, which means higher privacy. Additionally, if a category within an attribute is rare, ϵ increases for a particular θ, necessitating higher perturbation (θ) to protect such attributes with rare categories.

3.5 Privacy Analysis using Data Reconstruction Attacks

We conduct a privacy analysis on an existing Federated Learning (FL) framework known as SimFL, which performs Gradient Boosting Decision Trees (GBDT) in an FL setting, as proposed in [LWH20]. First, we explain GBDT in the following Section.

3.6 Gradient Boosting Decision Trees

Gradient Boosting Decision Trees (GBDT) are an ensemble learning technique that builds models in a serial manner. The idea is to combine multiple weak learners, typically decision trees, to create a strong predictive model. In GBDT, the final model is constructed by iteratively adding decision trees to minimize the prediction error, that is, each decision tree in GBDT improves upon the mistakes of the previous decision tree. Given a dataset with n observations (x_i, y_i) for $i = 1, \ldots, n$, the initial model $F_0(x)$ is often initialized to a constant value, such as the mean of the target variable:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma) \tag{3.7}$$

where L is a loss function that measures the difference between predicted and actual values.

In each iteration m, a new decision tree $T_m(x)$ is fitted to the residuals, which are the gradients of the loss function with respect to the predictions:

$$r_i^{(m)} = -\frac{\partial L(y_i, F_{m-1}(x_i))}{\partial F_{m-1}(x_i)} \tag{3.8}$$

The model is updated by adding the new tree, scaled by a learning rate η-

$$F_m(x) = F_{m-1}(x) + \eta T_m(x) \tag{3.9}$$

The learning rate η controls the contribution of each tree to the final model, helping to prevent overfitting. The overall objective function for Gradient Boosting Decision Trees combines a loss function with a regularization term to optimize the model's performance. This objective can be expressed as:

$$\mathcal{L}(F) = \sum_{i=1}^{n} L(y_i, F(x_i)) + \Omega(F) \tag{3.10}$$

In this equation, $L(y_i, F(x_i))$ represents the loss function measuring prediction error, with common choices including Mean Squared Error (MSE) for regression, defined as:

$$L(y_i, F(x_i)) = (y_i - F(x_i))^2 \tag{3.11}$$

For binary classification, a common choice is Log Loss, defined as:

$$L(y_i, F(x_i)) = -\left[y_i \log(F(x_i)) + (1 - y_i) \log(1 - F(x_i))\right] \tag{3.12}$$

Additionally, $\Omega(F)$ is a regularization term that helps prevent overfitting, commonly defined as:

$$\Omega(F) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} ||w_j||^2 \tag{3.13}$$

where T is the number of trees, γ penalizes the number of leaves in each tree, and λ penalizes the complexity of the leaf weights w_j. The goal of the Gradient Boosting algorithm is to minimize the overall loss function $\mathcal{L}(F)$ through the iterative addition of decision trees that correct the errors made by the existing ensemble [Fri01]. In the next section, we will discuss how to perform federated learning with GBDT.

3.7 Federated Learning with Gradient Boosting Decision Trees

Li *et al.* propsed a federated learning framework with Gradient Boosting Decision trees, which is named SimFL [LWH20], which operates through a two-step process- pre-processing and training. In the pre-processing step, each client employs Locality Sensitive Hashing (LSH) to compute hash values of its data. These hash tables are then shared among all clients, allowing each client to identify similar samples across the distributed datasets using the LSH-computed hash values. During the training step, clients integrate gradients from these similar samples, enabling collaborative learning.

To mitigate the risk of data reconstruction attacks that exploit the hashed values derived from LSH, we introduce a solution named Rakshit [KT23a]. Rakshit enhances privacy by applying the k-anonymity privacy model before the computation of hash values. This method deliberately introduces calculated errors into the hash value computation, thereby diminishing the accuracy of potential data reconstruction attacks.

Figure 3.13 illustrates the process flow diagram of our analysis, showcasing the steps and mechanisms involved in enhancing the privacy of the SimFL framework [2].

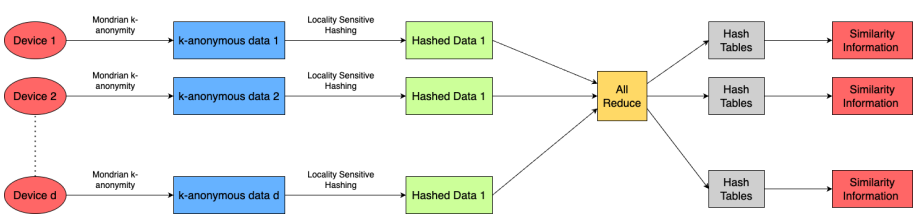


Figure 3.13: Privacy-Preserving FL with GBDT

[2]In this thesis, device(s), and client(s) have been used interchangeably in the context of federated learning

We found that sharing hashed values computed from LSH functions is not privacy-preserving [KT23a]. In the SimFL framework [LWH20], it is assumed that each client knows the hash values of the instances of all other clients. Although, on knowing the hashed value, an attacker cannot have an exact estimation of the data of users, but the approximation of the user's data is still possible. We showed two data reconstruction attacks. One is a Least Squares-based attack (LS), and the other is an Non-Linear Optimization-based (NLO) attack.

We evaluate the privacy of FL framework SimFL [LWH20] using two data reconstruction attacks. These are LS, and NLO-based attacks, which we will explain in detail in later subsections. SimFL framework utilizes LSH to find out similar samples from other clients, then adds the gradient terms of similar samples from other clients when training a GBDT model. We consider that a disadvantage of SimFL is that it assumes that every client knows the hash values of every record in other clients. We consider that knowing the hash values of the records is harmful from a privacy perspective, and we prove that this is the case using two data reconstruction attacks. The goal of the adversary in a reconstruction attack is to extract the data used in the training or inferences of a machine learning model.

3.7.1 Least Squares Attack

Motivated by [Hu+21], we show a least-squares-based data reconstruction attack on the hashed values given by a Locality Sensitive Hashing (LSH) function. The procedure is as follows.

Given a user's data u, the hashed value H is generated by hash functions. The hash function in a dot product form is computed as follows:

$$f_{a,b,r}(u) = \lfloor \frac{(a_1, a_2 \dots a_d)(u_1, u_2 \dots u_d)^T + b)}{r} \rfloor \tag{3.14}$$

where $\lfloor . \rfloor$ means the rounding operation. We use L hash functions to get the hash signature H. We write the equations of L hash functions in a matrix operation form as follows.

$$\underbrace{\begin{pmatrix} H_1 \\ \vdots \\ H_L \end{pmatrix}}_{H} = \frac{1}{r} \underbrace{\begin{pmatrix} a_{11} & a_{12} & \dots \\ \vdots & \ddots & \\ a_{L1} & & a_{Ld} \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} u_1 \\ \vdots \\ u_d \end{pmatrix}}_{u} + \frac{1}{r} \underbrace{\begin{pmatrix} b_1 \\ \vdots \\ b_L \end{pmatrix}}_{b} \tag{3.15}$$

When $L \geq d$, the problem of reconstructing the data reduces to the problem of solving an over-determined system of equations.

An unbiased estimator for a random variable x from a uniform distribution $U(a, b)$ is the expectation $E(x) = \frac{1}{2}(a + b)$. Thus, the estimation of $H_1, \dots H_L$ is given as follows:

$\hat{H} = \left(\frac{\lfloor H_1 \rfloor + \lfloor H_1 \rfloor + 1}{2}, \frac{\lfloor H_2 \rfloor + \lfloor H_2 \rfloor + 1}{2} \ldots, \frac{\lfloor H_L \rfloor + \lfloor H_L \rfloor + 1}{2}\right)$. Now, given the estimation $\hat{H}$, the projection matrix A, $Y = H$ - $\frac{1}{r}b$, then the user's data u can be reconstructed by Least Squares method by the equation:

$$\hat{u} = r \cdot (A^T A)^{-1} A^T Y \tag{3.16}$$

3.7.2 Non-Linear Optimization (NLO) Attack

When $L < d$, then Equation (3.15) corresponds to an under-determined system. The under-determined system has infinitely many solutions, which means that we cannot determine a unique estimation of a user's data. However, each entry of the random projection vector of A is chosen independently from a p-stable distribution in Equation (3.15). Consequently, using this p-stable property, we can reformulate the data reconstruction problem to a non-linear optimization problem as shown in [Hu+21]. Below is the definition of a p-stable distribution.

Definition 1 *[Zol86] A distribution $\mathcal{D}$ over $\mathcal{R}$ is called p-stable, if there exists $p \geq 0$ such that for any n real numbers $\beta_1 \ldots \beta_n$ and Independent and Identically Distributed (I.I.D.) variables $X_1 \ldots X_n$ with distribution $\mathcal{D}$, the random variable $\sum_i \beta i X_i$ also has the same distribution as the variable $(|\sum_i \beta_i|)^{\frac{1}{p}}$, where X is a random variable with distribution $\mathcal{D}$.*

It is known that when p= 2, a normal distribution $\mathcal{D}_N$, which is defined by the density function $q(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}$, is 2-stable. We use the following Observation 1 to reformulate the data reconstruction problem into an optimization problem [Hu+21].

Definition 2 *Suppose $v_1, \ldots v_n$ be random variables following normal distribution $N(0, 1)$. Let $k_1, \ldots k_n$ and p be real values. Suppose*

$$Z = \frac{1}{p}(v_1 k_1 + v_2 k_2 \ldots v_n k_n). \tag{3.17}$$

Then, the random variable Z follows a normal distribution

$$Z \sim N(0, \frac{1}{p^2} \sum_{i=1}^{n} |k_i|^2). \tag{3.18}$$

Suppose $V = \frac{1}{r} au$, where a denotes a d-dimensional projection vector with entries chosen independently from a 2-stable distribution and u denotes a user's data. Using Definition 2, we know that

$$V \sim N(0, \frac{1}{r^2} \sum_{i=1}^{d} |u_i|^2). \tag{3.19}$$

Table 3.2: Description of datasets

Dataset	Number of Samples	Dimension
cod-rna	59,535	9
Adult	32,561	14
ijcnn1	49,990	22
SUSSY	1,000,000	18
a9a	32,561	123

Recall Equation 3.15, when we have L sample values of the variable V. If we know the variance of V, we can find u, such that the difference between $\frac{1}{r^2}(\sum_{i=1}^{d}|u_i|^2)$ and the variance V is minimised. We can find u by the following Non-Linear Optimization (NLO) [Hu+21] problem.

$$\begin{aligned} \min Obj(u) = \quad & Var(V) - \frac{1}{r^2}\sum_{i=1}^{d}|u_i|^2 \\ \text{subject to,} \\ & Au + b = r\hat{H} \end{aligned} \tag{3.20}$$

Hence, we can obtain the approximated user's data $\hat{u}$ by solving the problem described by Equations (3.20).

3.8 Experiments

The datasets used in the experiments are described in Table 3.2. All the datasets are for binary classification tasks. They are: cod-rna, ijcnn1, SUSY, and adult. All these datasets are publicly available [3] [4]. The adult dataset has 14 features. We did feature selection and used five features: age, educational number, capital gain, hours per week, and final weight.

3.8.1 Parameters

In our experiments, the most important parameters are L (number of hash functions), k (size of the anonymized set). We specify the values of our parameters as follows.

- We used L = {22,23,25,30,50} for ijcnn1 dataset, L = {8,9,10,15,20,25} for cod-rna dataset, L = {18,20,22,24, 25,30,35} for SUSY dataset for LS-based attack.
- We used $L = d$-1 and $L = d$-2, where d is the dimension of the data for NLO attack.

[3] https://www.csie.ntu.edu.tw/~cjlin/libsvm/index.html
[4] https://archive.ics.uci.edu/ml/datasets.php

Table 3.3: Test Errors on a9a, SUSY and cod-rna dataset

Dataset	Size of anonymised set	Test Error
a9a	Without Mondrian	0.167
	$k = 10$	0.178
	$k = 20$	0.182
	$k = 30$	0.186
	$k = 40$	0.188
SUSY	Without Mondrian	0.2486
	$k = 10$	0.266
	$k = 20$	0.2767
	$k = 30$	0.2767
	$k = 40$	0.2767
cod-rna	Without Mondrian	0.067
	$k = 10$	0.08
	$k = 20$	0.09
	$k = 30$	0.1
	$k = 40$	0.1

- We used $k = \{10,20,30,40\}$ for obtaining the test errors of the SimFL framework on anonymized datasets.

To implement NLO attack, we used an available software [5], which minimizes a scalar function of one or more variables using Sequential Least Squares Programming (SLSQP).

3.9 Results

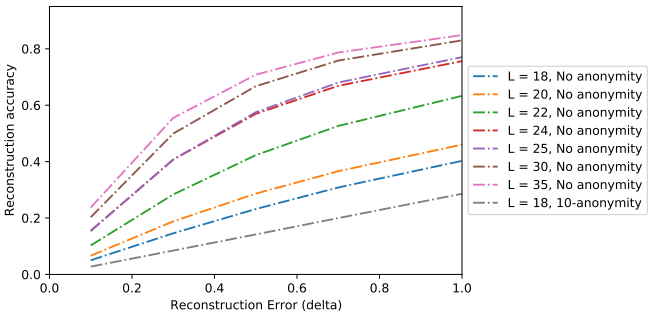
We summarize our observations as follows.

- We found that even when the number of hash functions is less than the dimension of data, the data can be reconstructed significantly by means of an NLO attack, as shown in Figure 3.15. This is in contradiction with [LWH20], where it is stated that it is impossible for a dishonest party to obtain the original data of other parties when $L < d$.

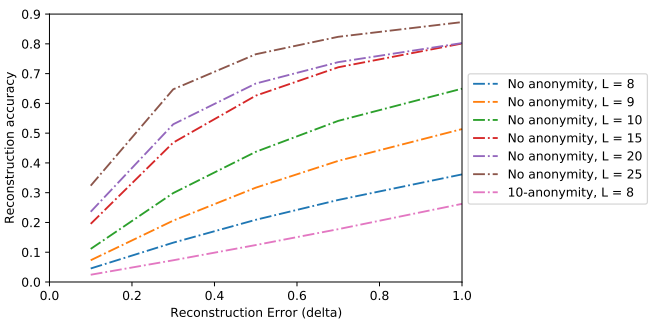
- When the number of hash functions is more than the dimension of data, the reconstruction accuracy is very high (more than 90%), as shown in Figure 3.14 using a Least Squares-based attack.

- In the case of an LS attack, we can see a trend that as the number of hash functions increases, the reconstruction accuracy also increases. For the NLO attack, the reconstruction cannot achieve the same high accuracy, when the number of hash functions increase because we are bounded by $L < d$, which means fewer constraints in the optimization method. Hence,

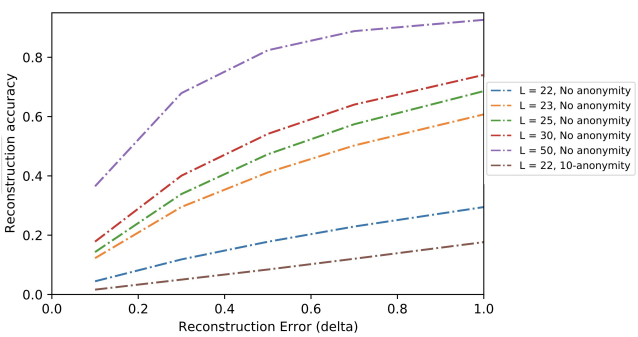
[5] `https://docs.scipy.org/doc/scipy/reference/optimize.minimize-slsqp.html`



(a) SUSY dataset

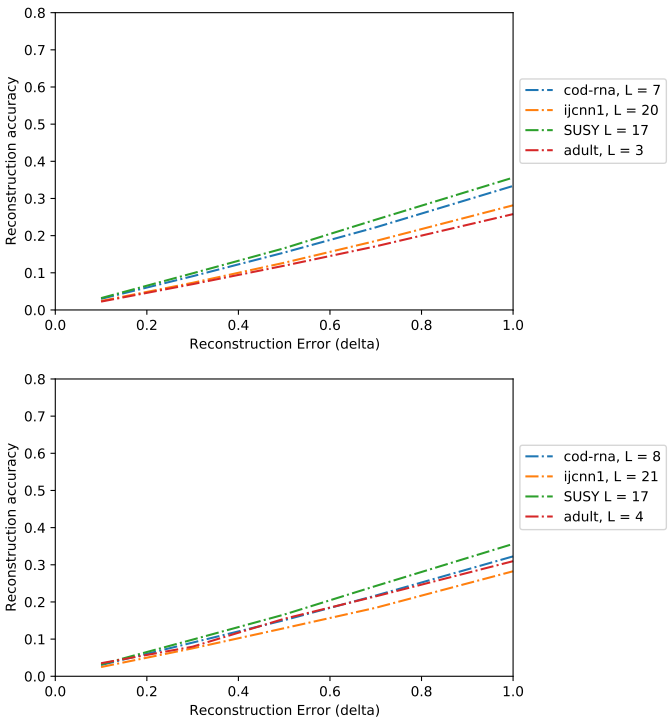


(b) cod-rna dataset

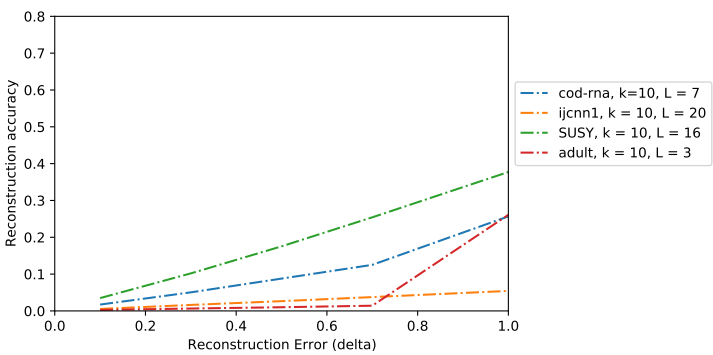


(c) ijcnn1 dataset

Figure 3.14: Reconstruction accuracy using Least Squares-based attack. L is the number of hash functions



(a) cod-rna dataset



(b) ijcnn1 dataset

Figure 3.15: Reconstruction accuracy using NLO-based attack. L is the number of hash functions, k is the parameter for k-anonymity

we agree with the authors of [Hu+21] on the performance of LS and NLO attacks in reconstructing the original data from the hash values.

- We also show the test errors of WGBDT on the anonymized datasets using

Mondrian in Table 3.3. Our results demonstrate that the test errors were almost constant for $k = \{10, 20, 30, 40\}$. Hence, our proposed approach obtains a more balanced trade-off between utility and privacy.

We introduced an additional layer of anonymization of data using Mondrian k-anonymity before applying Locality Sensitive Hashing, which helped reduce the reconstruction accuracy of the data, as shown in our results. For example, we can see that the reconstruction accuracy is less in our proposed approach Rakshit when we introduce protection using k-anonymity in Figure 3.14, and Figure 3.15.

3.10 Conclusion

This chapter contributes two novel privacy-preserving FL frameworks with decision trees. The key difference in both of the proposed FL frameworks is that one of our proposed FL framework produces k-anonymous decision trees, while the other one produces differentially private decision trees. Both frameworks share the common feature of aggregating locally trained decision trees. It is well-acknowledged that sharing these trees directly can compromise privacy in the form of reconstruction of original databases [GGH12]. To address privacy issues arising form sharing decision trees, we applied privacy models like- k-anonymity and differential privacy. From our experiments, we find out that k-anonymous decision tree offers stable results with increase in the value of k (privacy). One the other hand, differentially private decision tree leads to diminishing value of whichever performance metric utilised, with decrease in value of ϵ (increase in privacy). In addition to this, we also do a privacy-utility analysis of an existing FL framework for GBDT [LWH20], where we demonstrated that knowing the hashed values computed from LSH leads to privacy breach in the form of data reconstruction attacks.

Chapter 4

Quantification of Privacy

> Privacy is not an option, and it shouldn't be the price we accept for just getting on the Internet
>
> Gary Kovacs

Assessment or quantification of privacy is an important step in evaluating the disclosure risks of systems. As machine learning models become integral to various applications, it is essential to measure and mitigate potential privacy breaches that may arise from their use. In this chapter, our focus lies on quantifying the privacy of machine learning models. This process involves assessing how well privacy-preserving techniques perform in protecting sensitive information while maintaining the accuracy and utility of the machine learning models. There are a plethora of machine learning techniques that learn from large-scale databases. Machine learning models should be well-generalized; overfitting is an undesirable phenomenon for both the utility and the privacy of machine learning models. From the utility perspective, when a machine learning model is overfitted, it means that the model has captured the details of most of its training records, and hence the performance of the model might collapse on the unseen testing set. From the privacy perspective, when a machine learning model is overfitted, the model has learned the details of most of its training records, which leads to information leakage from the machine learning models. Attacking the machine learning systems is a way to quantify the privacy of systems. Attackers can be strong or weak, depending on their knowledge about the machine learning model parameters and/or the knowledge of the data distribution on which the machine learning model was trained. Hence, the privacy of a machine learning model should be quantified using different attack assumptions. The range of the attack assumptions lies between the maximum knowledge attacker to the minimum knowledge attacker.

Shokri *et al.* discussed Membership Inference Attacks (MIA) for neural net-

works for the first time [Sho+17]. The goal of a Membership Inference Attack (MIA) is to determine if a particular individual participated in the training of a machine learning model or not. Membership leakage leads to the privacy breach of individuals who participated in the training of machine learning model. Let us consider an example. If a patient participated in a database consists of Alzheimer's data, then the revelation of the membership of a patient confirms that a target individual has Alzheimer's, which can be misused by the attacker, as an Alzheimer's patient forget things. This means that MIA leads to additional attacks, such as attribute inference [Hay+17], [Yal+19]. Another example of privacy breach is when membership leakage occurs of an individual, who participated in a location database. This leads to the breach of location privacy.

In MIA, the attacker builds an attack model, which is basically a binary machine learning classifier. This binary machine learning classifier has the ability to differentiate between the members and non-members of the training set. This differentiation between the members, and the non-members of the training set is made on the basis of the difference in the confidence or prediction scores of the members versus non-members of the training set by the target model. Intuitively, the target model predicts the members of the training set with higher confidence in comparison with the non-members of the training set. To build such an attack model, there is a procedure, which is followed by the attacker. We show the architecture of the procedure to conduct an MIA in Figure 4.1 [1].

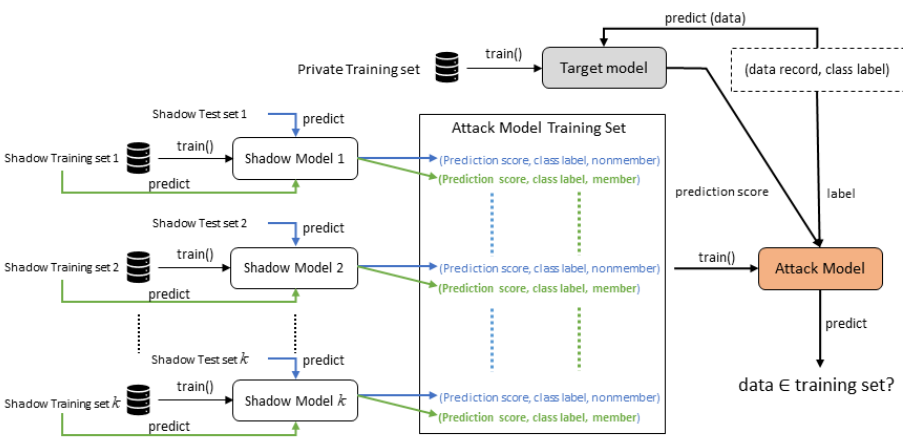


Figure 4.1: Membership Inference Attack

To build an attack model, first and foremost, the attacker creates multiple shadow datasets. A shadow dataset is the one that closely mimics the data the

[1]The figure for MIA architecture is taken from `https://github.com/sakib570/mia-synthetic-data/blob/main/mia_arch.png`

target model was trained on, which is essential for studying and testing membership inference attacks. Shokri *et al.* discussed multiple ways to construct shadow datasets. Constructing shadow dataset is a crucial step to conduct membership inference attacks on machine learning models. One method is hill climbing, which involves making small, iterative adjustments to input data to increase the likelihood that the model will classify it as part of the training set, with gradients guiding these adjustments if available. Another approach is using noisy versions of the original data, where controlled noise is added to create variations that still resemble the training data, along with techniques like data augmentation, which includes flipping or rotating images for the image data. Generating synthetic data is also an effective way to build shadow datasets. It is done either by creating data that follows the same patterns as the target training data or by using Generative Adversarial Networks (GANs) to produce new, similar data. Additionally, publicly available datasets that are similar to the target dataset can be used. These publicly available datasets can be improved with additional features or noise. Querying the target model by classifying new inputs can provide insights, allowing the construction of a relevant dataset and enhancing understanding of the model's behavior. Transfer learning can also be utilized by fine-tuning pre-trained models on similar tasks and extracting features to create new datasets that align closely with the target data. These methods collectively enable the construction of datasets that effectively mimic the original training data, which is essential for investigating and testing membership inference attacks.

Suppose, the attacker creates k shadow datasets. Each shadow dataset is divided into shadow training set, and shadow testing set. The attacker trains k machine learning models using k shadow datasets. Then, the attacker obtains the prediction scores, and class labels using the trained shadow models for each shadow test set. The attacker creates a training database on which the binary attack model can be trained. In the database, the prediction scores, and class labels serve as the feature space, and the membership (for training shadows set) or non-membership (for testing shadow set) serve as a label space. Finally, the attacker trains an attack model on this database, and utilise it to know whether some target individuals participated in the training process of a target machine learning model or not. As mentioned earlier, from this membership knowledge, the attacker can make additional attacks, such as attribute inference.

Dimensionality reduction techniques transforms a database from a high dimensional space to a low-dimensional space. There are different techniques in the literature to reduce dimension, such as PCA, Linear Discriminant Analysis (LDA) [BG98]. In Autoencoders, the encoder transforms the data to a low-dimensional latent space from a high-dimensional space [Pin+20]. In this chapter, our main research question is,

"Is there any privacy leakage/disclosure from low-dimensional space"

To answer this question, we need to quantify the privacy leakage by attacking a system that involves dimensionality reduction. We focus on the case of

Principal Component Analysis (PCA) in the following sections. We provide a detailed explanation of the following topics- PCA, Membership Inference Attack (MIA) against PCA, and data reconstruction attack against PCA.

4.1 Privacy Analysis of Principal Component Analysis

In PCA, eigenvectors capture the information about the directions of the data, where there is more variance. Let's understand PCA in more detail, for which we make a set of assumptions. Suppose, we have a set $\mathcal{D} = \{x_n \in R^d : n = 1 : N\}$ comprising N data samples corresponding to N individuals of dimension d. We subtract the mean from the data, and obtain the centered data matrix and denote it as X. The objective of PCA is to find a p dimensional subspace, which approximates each sample x_n [AW10]. The formulation of PCA is as follows:

$$\min_{\pi_p} E = \frac{1}{N}\sum_{n=1}^{N} E_n = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{N}||x_n - \pi_p x_n||_2^2 \tag{4.1}$$

where E is the average reconstruction error and π_p is an orthogonal projector, which approximates each sample x_n by $\hat{x}_n$. The solution to the PCA problem can be obtained using the Singular Value Decomposition (SVD) of a sample covariance matrix Σ_{cov}. SVD of Σ_{cov} is given by $\sum_{i=1}^{d} \lambda_i v_i v_i^T$, where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$ are the eigenvalues, and $v_1, v_2 \ldots v_d$ are the corresponding eigenvectors of Σ_{cov}, respectively. Suppose V_p depicts the matrix whose columns are the top p eigenvectors. $\pi_p = V_p V_p^T$ is the solution to the problem in (4.1). The following subsections lead us to the development of a framework, which not only quantifies the disclosure, but also quantifies the utility of the computed eigenvectors, when privacy models are incorporated into the system to protect against disclosure risks.

4.2 Membership Inference Attack against PCA

Shokri *et. al* proposed MIA for neural networks. Following a similar line of idea, Zari *et. al* came up with MIA against Principal Component (PCA) [Zar+]. The goal of MIA against PCA is to infer whether a particular data point participated in the computation of principal components or not. To conduct an MIA against PCA, the attacker must know a subset of eigenvectors of a database. The attacker computes the reconstruction error E in Equation (4.1) for a target sample, and infers whether a target sample belongs to the training set or not on the basis of a tunable threshold parameter t on the reconstruction error E.

4.3 Data Reconstruction attack Principal Component Analysis

PCA is a linear transformation technique that reduces the dimensionality of data while retaining as much variance as possible. It is a well-known fact that PCA is reversible, i.e., we can map the eigenvectors back to the original data. This leads to our development of a data reconstruction attack against PCA [KT23b]. Now, we will explain the mathematics behind the PCA process, and the reversibility of PCA through a series of equations. We start with the original data matrix X of dimensions $n \times p$ (where n is the number of samples and p is the number of features), the first step in PCA is to center the data by subtracting the mean of each feature:

$$X_{centered} = X - \bar{X}$$

where $\bar{X}$ is the mean vector of X.

Next, we compute the covariance matrix of the centered data:

$$X_{cov} = \frac{1}{n-1} X_{centered}^T X_{centered}$$

We then perform eigen decomposition on the covariance matrix to obtain the eigenvalues and eigenvectors.

$$X_{cov} = V \Lambda V^T$$

where V is the matrix of eigenvectors and Λ is the diagonal matrix of eigenvalues.

The principal components are calculated by projecting the centered data onto the eigenvectors.

$$P = X_{centered} V$$

To reconstruct the original data from the principal components, we use the equation

$$X_{reconstructed} = X_{centered} V V^T + \bar{X} \tag{4.2}$$

Here, $X_{centered} V V^T$ transforms the data back to the original feature space, and $\bar{X}$ adds the mean that was subtracted during centering. PCA is often used for dimensionality reduction by selecting a subset of the principal components that capture most of the data's variance. This means not all principal components are necessary for an approximate reconstruction of the original data. Using fewer components results in an approximation, capturing the main structure of the data while losing some details. Figure 4.2 illustrates the PCA transformation and reconstruction process for the wine quality dataset available on the UCI repository [2], showing that even with a subset of principal components, we can achieve a good approximation of the original data.

[2] https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv

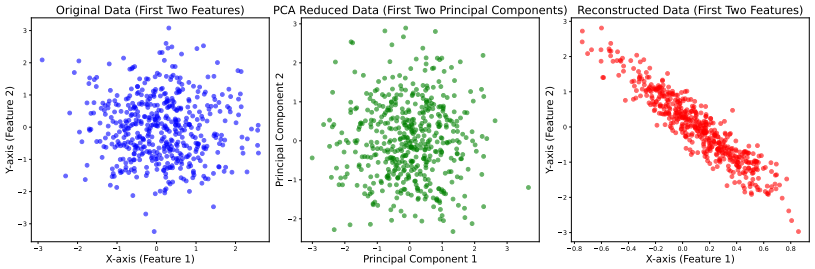


Figure 4.2: Visualization of PCA transformation and reconstruction. The original data points are projected onto the principal components and then reconstructed.

According to Zari *et al.*, leakage of some or all eigenvectors leads to MIA. When the leakage of eigenvectors is combined with the knowledge of data distribution, a data reconstruction attack is possible using Equation 4.2 [KT23b]. To reconstruct all the original variables from a subset of principal components/eigenvectors, we can map it back to p dimensions with V^T. The result is then given by $\hat{X} = PV^T$. Since we have a projection scores matrix, $P = X_{centered}V$, we obtain $\hat{X} = X_{centered}VV^T$. Obviously, we do not have access to the original centered data $X_{centered}$; we suppose that the attacker has knowledge about the distribution of $X_{centered}$. Therefore, the attacker can synthesize the data X_{syn} with a similar distribution as $X_{centered}$, and reconstruct the original data using $\hat{X} = ZV = X_{syn}V^TV$. In our experiments, we constructed synthetic data using different percentages of samples from the original data. Hence, we show that the MIA against PCA can be converted into a data reconstruction attack against PCA, if we assume a worst case scenario for the users, and a best case scenario for the attacker, where the attacker has intercepted some of the top eigenvectors, and the attacker also has a knowledge about the distribution of the data on which the principal components were computed.

The success of our proposed data reconstruction attack against PCA depends on the following two factors.

- How many top eigenvectors are known to the attacker?
- How much knowledge does the attacker has about the data distribution of a target individual?

This knowledge of data distribution owned by the attacker can be the k-anonymous version of the original database or the synthetic version of the original data, as both k-anonymous data, and synthetic data are GDPR compliant. Hence, can be made publicly available. Hence, it makes sense to utilise them in our experiments for the attacker's knowledge. We depict the membership

attack scenario, and the data reconstruction attack scenario for PCA in Figure 4.3.

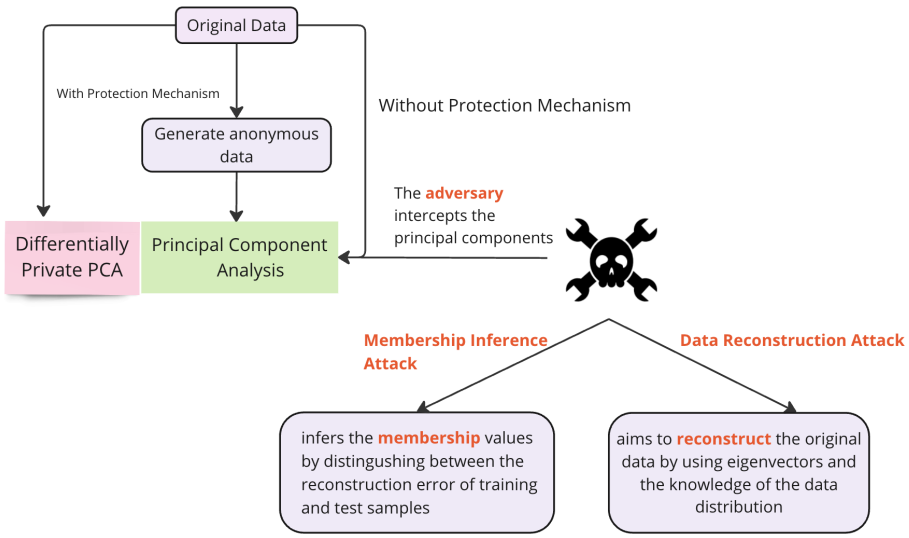


Figure 4.3: Membership Inference Attack and Data reconstruction attack against PCA

4.4 Threat Model and Attack Methodology

In our attack setting, the data curator/guardian generates synthetic data $\mathcal{D}_{syn}$ using different percentages (10%, 30%, 50%, 70%,100%) of samples from the original data $\mathcal{D}$. The synthetic data is generated using CTGAN. This will be described in Chapter 5. The curator then computes the principal components P_k of the synthetic data $\mathcal{D}_{syn}$, and sends these to a reliable party. We suppose that the attacker $\mathcal{A}$ intercepts some or all of the Principal Components (PCs) computed on the synthetic data by eavesdropping on the communication channel. According to the review in [Hu+22]; there are two kinds of knowledge useful for attackers to implement MIAs against ML models: knowledge of the **data distribution**, and knowledge of the **machine learning model/algorithm**, which learns about the patterns in training data. Knowledge of the data distribution means that the attacker has access to a shadow dataset, which has the same distribution as the original data. This is a reasonable assumption, as the attacker can obtain the shadow dataset using statistics-based synthesis when the data distribution is known and model-based synthesis when the data distribution is unknown [Sho+17]. Hence, in our attack setting, we assume that the attacker can synthesize the shadow dataset using CTGAN. By knowing some of the principal components, and the constructed shadow dataset using

CTGAN, the attacker can make a data reconstruction attack as follows.

Suppose we have an original data matrix X_{orig} of size $n \times p$. We obtain a data matrix X, after subtracting the mean vector μ from each row of X_{orig}. Let V be the $p \times k$ matrix of some k eigenvectors to reduce the dimension; these would most often be the k eigenvectors with the largest eigenvalues. Then the $n \times k$ matrix of PCA projection scores (Z) will be given by $Z = XV$. In order to be able to reconstruct all the original variables from a subset of principal components/eigenvectors, we can map it back to p dimensions with V^T. The result is then given by $\hat{X} = ZV^T$. Since we have a projection scores matrix, $Z = XV$, we obtain $\hat{X} = XVV^T$. We do not have access to the original data X; we assume that the attacker has knowledge about the distribution of X. Therefore, the attacker can synthesize the data X_{syn} with a similar distribution as X, and reconstruct the original data using $\hat{X} = ZV = X_{syn}V^TV$. We assume the attacker can access the synthetic data. In our case, we use the Conditional Tabular Generative Adversarial Network (CTGAN) to show experimental results. We generate the synthetic data using different percentages of records from the original data. To show the degree of success of the data reconstruction attack, we show the Reconstruction Accuracy (R.A.) in estimating the original data. We define R.A. as follows.

Definition 3 *[KT23b] Suppose R is the reconstructed data, which is the estimator for the original data O, where $R = \{R_1 \dots R_d\}$, and $O = \{O_1, \dots O_d\}$. Let δ be a reconstruction error, which can be tolerated to measure the level of reconstruction for a record. The reconstruction accuracy, R.A. is defined as follows .*

$$R.A. = \frac{\#\left\{\hat{R_j} : |\frac{O_j - R_j}{R_j}| \leq \delta, j = 1 \dots d\right\}}{n} \tag{4.3}$$

where # means count, and n is the number of records. Hence, R.A. is the percentage of reconstructed entries for which the relative errors are within δ.

4.5 Compared Methodologies

We compared our approach with two alternative strategies- one without any protection mechanism and another using Differentially Private Principal Component Analysis (DPPCA). Here, we describe these strategies, with the results discussed in Section 4.6.

4.5.1 No Protection Mechanism

First, we compare our proposed methodology with a scenario where no protection mechanism is employed. In this strategy, the data curator calculates the principal components directly from the original data and shares them with a reliable third party. However, if an attacker intercepts the communication

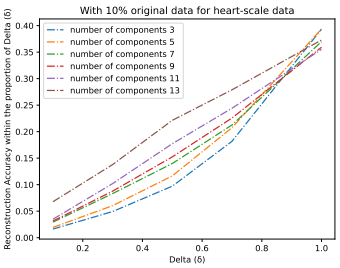
channel, they might gain access to some or all of the principal components. Using the intercepted components and their knowledge of the training data distribution, the attacker attempts to reconstruct the original data. The key difference between our proposed methodology and this strategy with no protection mechanism is that, in the proposed methodology, the leaked principal components are derived from synthetic data, whereas, in the compared strategy, they are derived from the original data.

4.5.2 Differentially Private Principal Component Analysis

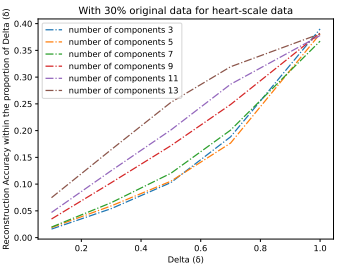
The objective of PCA is to identify the principal components of a dataset, representing the directions with the highest variance. In [IS16], a method for performing Differentially Private PCA (DPPPCA) on high-dimensional datasets was introduced. This algorithm perturbs the covariance matrix of the dataset in a differentially private manner to ensure that the PCA output remains differentially private. The algorithm takes a dataset X with n samples and d dimensions and a privacy parameter ϵ. It computes the covariance matrix S of the dataset, a $d \times d$ symmetric matrix. To preserve privacy, it adds a noise matrix N to S, where N is also symmetric and generated using the Laplace mechanism, which adds independent Laplace noise to each entry, scaled by ϵ. The perturbed covariance matrix $S + N$ is then subjected to eigen decomposition using numerical methods like the power iteration method. Finally, the algorithm outputs the top k principal components of the dataset, with k being a user-specified parameter. The added noise ensures that the output is differentially private, preventing the disclosure of information about any individual sample in the dataset. The authors also provide theoretical bounds on the privacy loss and accuracy of this method.

4.6 Experiments I- Privacy analysis via Data Reconstruction Attack

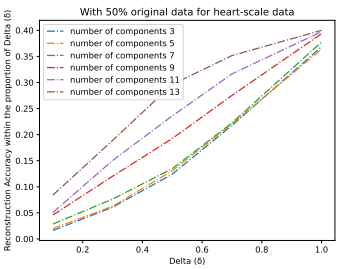
The objective of the experiments in this section is to quantify privacy via a data reconstruction attack. In our experiments, we generated synthetic datasets using different percentages of original data, including {10%, 30%, 50%, 70%, 100%}. We apply PCA to the generated synthetic datasets. Assuming that the adversary intercepted some of these principal components, we try to reconstruct the data from which the principal components were computed. We obtain the reconstruction accuracy, as shown in Figures 4.4, 4.5, and 4.6, respectively. We have an upper cap for the Reconstruction Accuracy (R.A.), as the maximum reconstruction error we can obtain is the difference between the original and synthetic data generated using CTGAN using all the original data records. We are measuring the capability of CTGAN to generate a different-looking but similar distribution of synthetic data and the privacy breach caused by the leakage



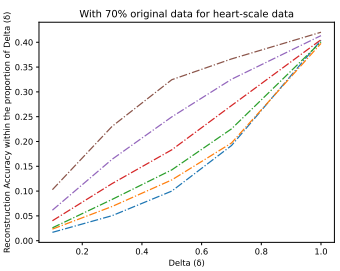
(a) 10% original data



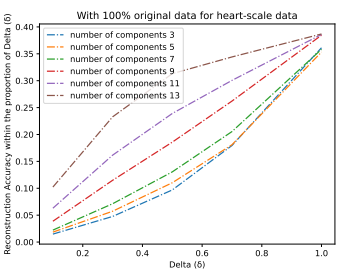
(b) 30% original data



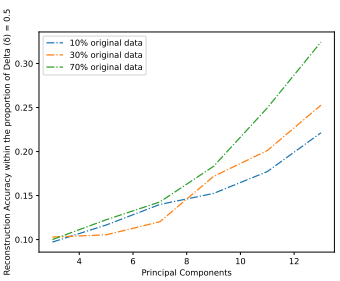
(c) 50% original data



(d) 70% original data



(e) 100% original data



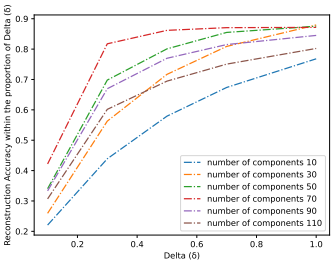
(f) R.A. vs no. of PCs with δ= 0.5

Figure 4.4: R.A. within the limit Delta (δ) for heart-scale data

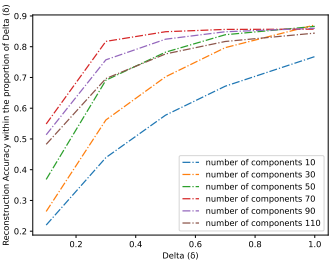
done by the principal components. We also make a comparison with DPPCA, which is described in Section 4.5.2.

Table 4.1: Description of datasets

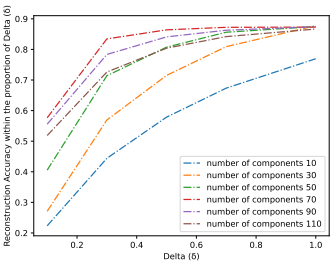
Dataset	**Number of Samples**	**Number of Attributes**
Heart-scale	270	13
a9a	32561	123
Mushrooms	8124	112



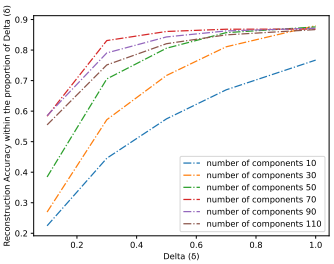
(a) 10% original data



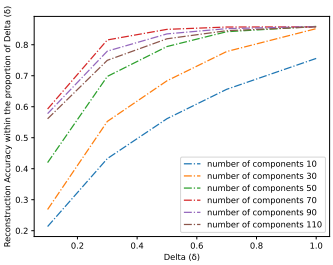
(b) 30% original data



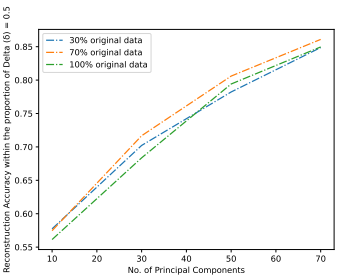
(c) 50% original data



(d) 70% original data



(e) 100% original data

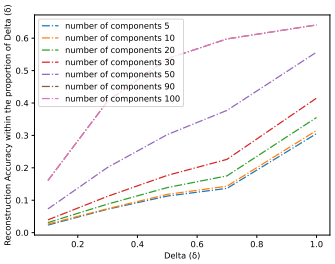


(f) R.A. vs no. of PCs with δ= 0.5

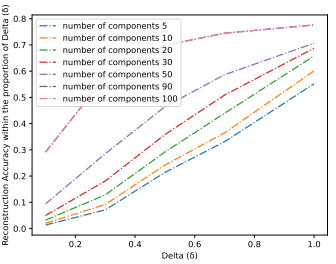
Figure 4.5: R.A. within the limit Delta (δ) for a9a data

We performed our experiments on three publicly available binary classification datasets- Heart-scale, a9a, and mushrooms. The datasets are publicly available [3]. The number of samples and the number of attributes of these datasets are described in Table 4.1. It can be seen that the range for the number of samples is from 270 to 32,561, and the number of attributes is from only 13 to 123. Each dataset has some preprocessing steps involved. The scale for the heart-scale dataset is [-1,1]. After preprocessing, the adult dataset is converted into the a9a dataset. There are 14 features in the original adult data set, eight of which are categorical and six of which are continuous. The continuous

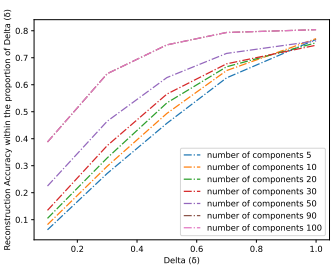
[3] `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`



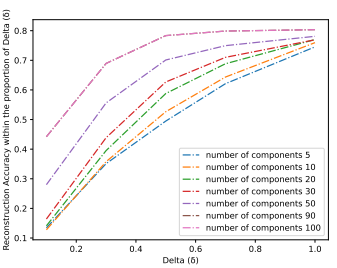
(a) 10% original data



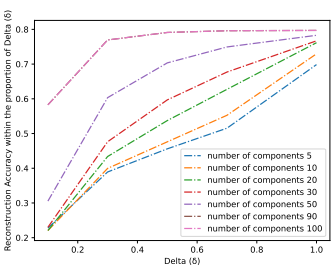
(b) 30% original data



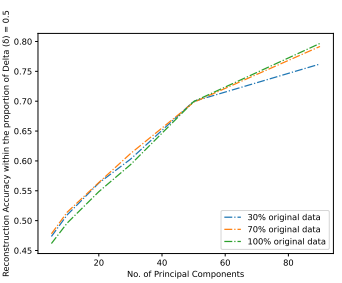
(c) 50% original data



(d) 70% original data



(e) 100% original data

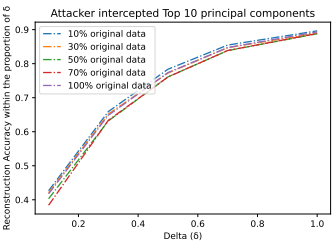


(f) R.A. vs no. of PCs with δ= 0.5

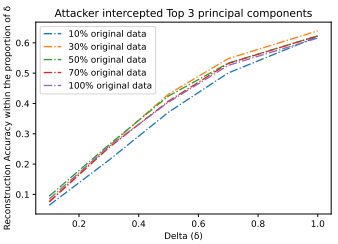
Figure 4.6: R.A. within the limit Delta(δ) for mushrooms data

features in this data set are discretized into quantiles, and a binary feature represents each quantile. In addition, a categorical feature with m categories is converted to m binary features. In the mushrooms dataset, each nominal attribute is expanded into several binary attributes. Also, the original attribute 12 has missing values and is not used. We explain our main findings from our experiments as follows.

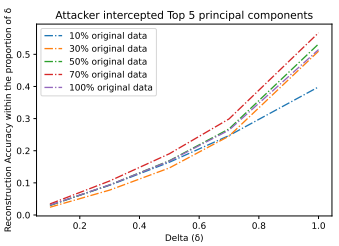
1. In Figure 4.5a, for the a9a dataset, we found that even after using just 10% samples from the original data, the R.A. is close to 90% when the attacker intercepted 110 principal components. R.A. is close to 70% when



(a) a9a data



(b) Heart-scale data



(c) Mushrooms data

Figure 4.7: R.A. without protection mechanism prior to the computation of PCs

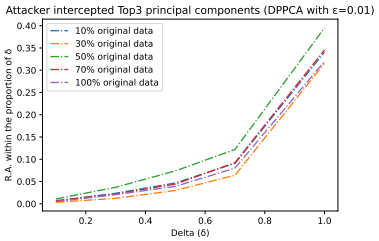
the attacker intercepted 10 principal components.

2. For the a9a dataset, the R.A. is large in comparison with the heart-scale data in Figure 4.4 and mushrooms data in Figure 4.6 dataset. The reason behind better R.A. in the case of the a9a dataset is that a9a has more categorical features. Hence, the generation of synthetic data using CTGAN could provide less protection in the case of the a9a dataset.

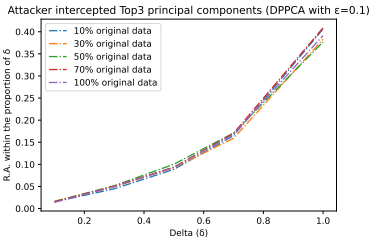
3. The maximum R.A. for heart-scale data, as shown in Figure 4.4, is close to 40%. It is less because we have a protection mechanism using synthetic data generation before the computation of principal components.

4. The minimum reconstruction in the case of mushroom data in Figure 4.6a is close to 20% when the attacker intercepted 5 or 10 principal components and only 10% of the original data was used in constructing the synthetic data.

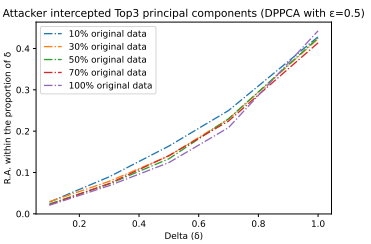
5. In Figures 4.4f, 4.5f, and 4.6f for heart-scale, a9a, and mushrooms dataset, respectively, we show a trend between R.A. and the number of principal components intercepted by the attacker. Our results show that R.A. increases as the number of principal components increases, which is also expected from theory.



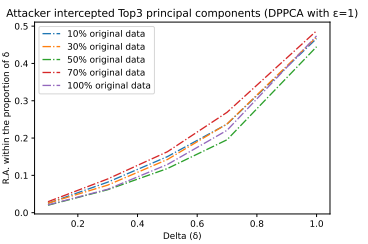
(a) ϵ= 0.01 for DPPCA



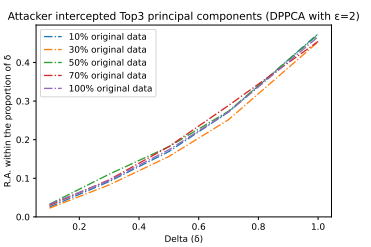
(b) ϵ= 0.1 for DPPCA



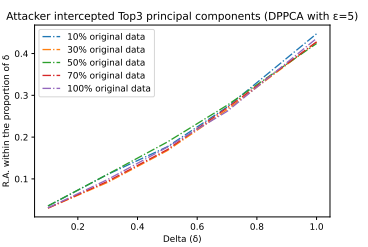
(c) ϵ= 0.5 for DPPCA



(d) ϵ= 1 for DPPCA



(e) ϵ= 2 for DPPCA



(f) ϵ= 5 for DPPCA

Figure 4.8: R.A. using DPPCA on heart-scale data when the attacker intercepted Top 3 PCs

6. We generated synthetic datasets from different percentages of original data. From Figures 4.5a to 4.5e, we observe that as we increase the percentage of samples used in generating the synthetic data, R.A. increases.

7. It is noted that there is not much difference in the R.A. when the CTGAN uses less percentage (e.g., 10%) of samples from the original data compared to using all the samples from the original data for generating the synthetic data. This shows the capability of CTGAN in successfully generating synthetic data similar to the original data using fewer samples from the original data.

8. When no protection mechanism is used, we show that the R.A. increases. E.g., in Figure 4.7b, the R.A. for the heart-scale data approaches 60%,

which is higher in comparison with the case when DPPCA is used (Refer Figure 4.8), and when the principal components were computed on the synthetic data (Refer Figure 4.4).

9. In Figure 4.8, we use DPPCA on the heart-scale data. We observe that the lesser the value of ϵ, the shallower the graph for R.A.

10. Both DPPCA and the generation of synthetic data technique output comparable R.A. The performance of DPPCA depends on the value of a privacy parameter ϵ. The lower the value of ϵ, the higher the privacy.

Therefore, from our experiments, we can conclude that generating synthetic data from the original data and then training machine learning models on the synthetic data is a good way to combat attacks against machine learning models to an extent.

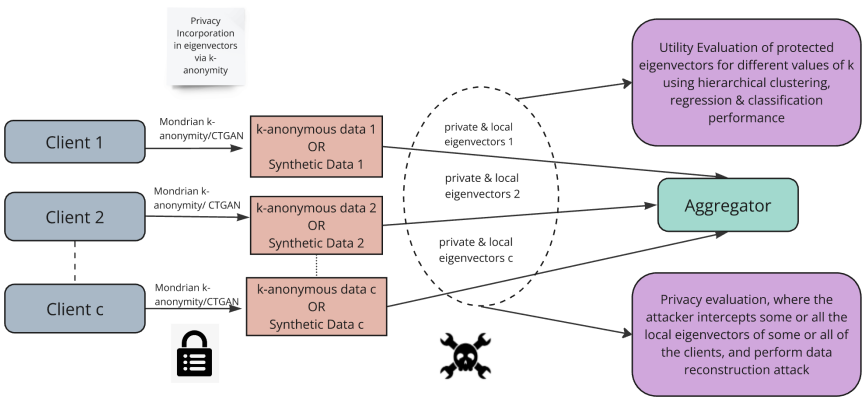


Figure 4.9: Privacy and Utility Assessment for PCA

4.7 Experiments II- Privacy and Utility Analysis for PCA

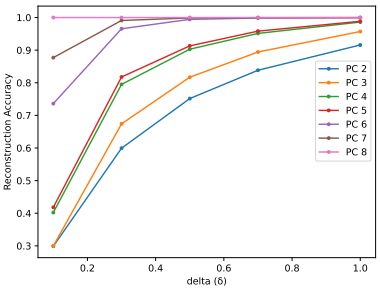
In our experiments, we assume a federated setting, where each client computes eigenvectors from their data. Since, it is federated setting, each client shares the local eigenvectors with the central server. Hartebrodt *et al.* [HR22] discussed federated learning PCA algorithms, which are based on the computation of local eigenvectors or subspaces, and local subspace aggregation to obtain a global subspace. Our attack methodology is applicable to such scenarios. So, it is important to incorporate privacy prior to the computation of eigenvectors. We use k-anonymity, and generative networks, for the private computation of eigenvectors. We show the process flow of our work using Figure 4.9. Now, we will elaborate on our experimental settings and results.

For our utility and privacy analysis, we conducted experiments on the CALIFORNIA-HOUSING and COD-RNA datasets. The CALIFORNIA-HOUSING dataset contains 20,640 records, while the COD-RNA dataset has 59,535 records. Both datasets include 9 features. The CALIFORNIA-HOUSING dataset is used for a regression task, aiming to estimate housing prices based on features like income, housing occupancy, and geographical location across various districts in California. These features contain sensitive information, making privacy protection crucial during data analysis. The COD-RNA dataset is utilized for a classification task. For the utility analysis of the CALIFORNIA-HOUSING dataset, we use the Coefficient of Determination (R^2). R^2 measures the proportion of variability in the dependent variable that can be explained by the independent variables in the model. Both the CALIFORNIA-HOUSING and COD-RNA datasets are numerical. Therefore, we implemented standardization as part of preprocessing using the `scikit-learn` library in Python.

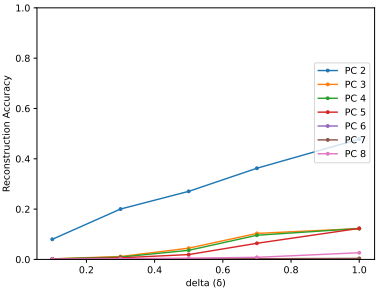
In our evaluation, we employed 10-fold cross-validation and reported the mean ± standard deviation for R^2 in Table 4.2.

PCA	# PCs	R^2
Baseline	*all*	*0.781 ± 0.019*
O-PCA	3	0.148 ± 0.034
S-PCA	3	0.134 ± 0.030
A-PCA (k=5)	3	0.135 ± 0.030
A-PCA (k=10)	3	0.147 ± 0.035
A-PCA (k=15)	3	0.134 ± 0.030
A-PCA (k=20)	3	0.134 ± 0.030
O-PCA	4	0.455 ± 0.038
S-PCA	4	0.445 ± 0.034
A-PCA (k=5)	4	0.444 ± 0.034
A-PCA (k=10)	4	0.454 ± 0.038
A-PCA (k=15)	4	0.445 ± 0.034
A-PCA (k=20)	4	0.445 ± 0.035
O-PCA	5	0.631 ± 0.034
S-PCA	5	0.624 ± 0.029
A-PCA (k=5)	5	0.624 ± 0.029
A-PCA (k=10)	5	0.629 ± 0.035
A-PCA (k=15)	5	0.624 ± 0.029
A-PCA (k=20)	5	0.623 ± 0.029
O-PCA	6	0.697 ± 0.029
S-PCA	6	0.689 ± 0.003
A-PCA (k=5)	6	0.690 ± 0.032
A-PCA (k=10)	6	0.696 ± 0.030
A-PCA (k=15)	6	0.689 ± 0.032
A-PCA (k=20)	6	0.689 ± 0.032

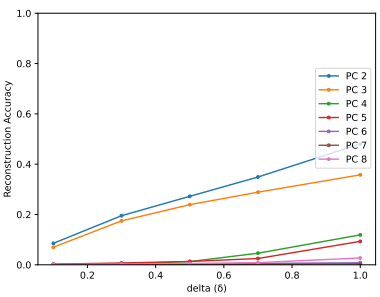
Table 4.2: Utility analysis via regression task on CALIFORNIA-HOUSING dataset. O-PCA refers to *Original* PCA, S-PCA is *Synthetic* PCA and A-PCA is *Anonymized* PCA with different values for k.



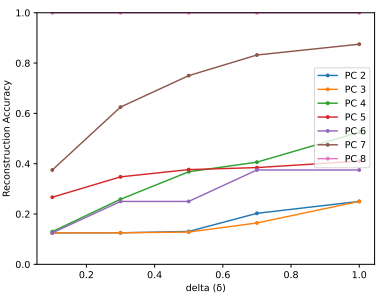
(a) R.A. in general PCA where we reach close to the original data on increasing no. of PCs



(b) R.A. b/w original and reconstructed data when 20-anonymous eigenvectors and 20-anonymous data is known



(c) R.A. b/w original and reconstructed data when 10-anonymous eigenvectors and 10% data is randomly drawn from 10-anonymous data is known

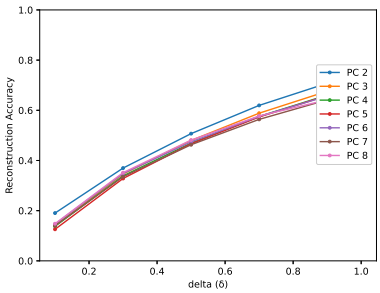


(d) R.A. b/w anonymous and reconstructed data when eigenvectors computed from the original data, and synthetic data generated using 10% samples from the original data via random sampling

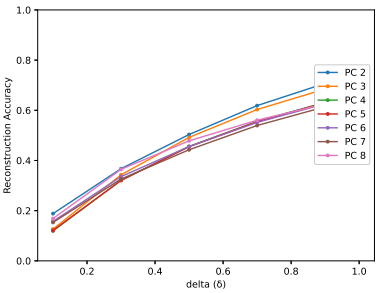
Figure 4.10: Reconstruction Accuracy (R.A.) for California housing dataset (Part 1)

We elaborate our main experimental findings as follows.

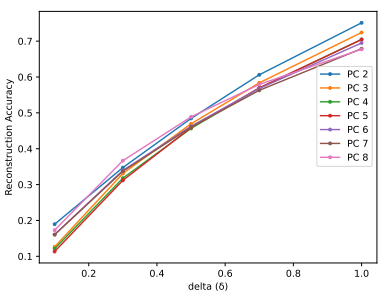
- Figure 4.10, and Figure 4.11 demonstrate the reconstruction attack results, where the reconstructed dataset is farthest from the original dataset when eigenvectors are computed on the k-anonymous data compared to the synthetic dataset. Using anonymous eigenvectors, we can get closer to the anonymous data but not to the original data, providing protection against reconstruction attacks. Therefore, we observe a decline in



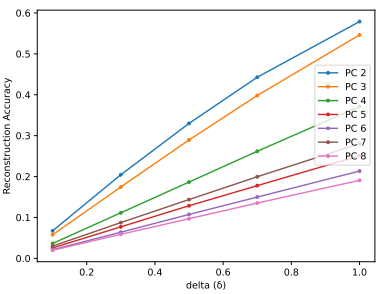
(a) Eigenvectors computed from the original data, and synthetic data generated using 10% samples from the original data via stratified sampling



(b) Eigenvectors computed from the original data, and synthetic data generated using 10% samples from the original data via random sampling



(c) Eigenvectors computed from the original data, and synthetic data generated using all the samples contained in the original data

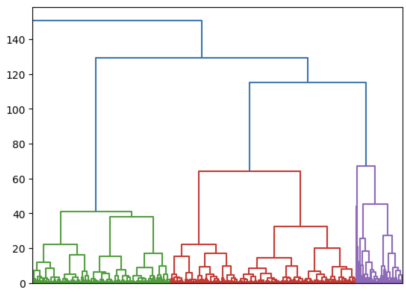


(d) Eigenvectors computed from the synthetic data, and synthetic data generated using 10% samples of the original data via stratified sampling

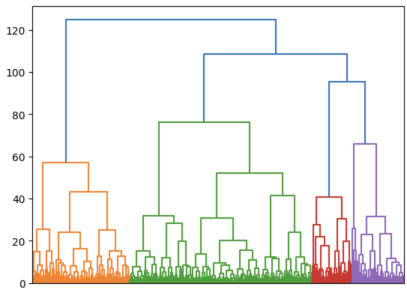
Figure 4.11: Reconstruction Accuracy between original and reconstructed data for California housing dataset (Part 2)

the attacker's efficacy in inferring user data in the CALIFORNIA-HOUSING dataset when incorporating a privacy protection mechanism before data analysis.

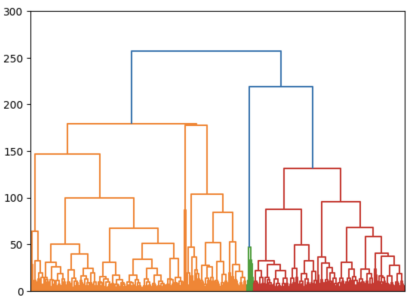
- We present dendrograms obtained after applying hierarchical clustering on the CALIFORNIA-HOUSING and COD-RNA datasets in Figure 4.12. The dendrograms for the original and anonymous data are quite similar for both datasets. However, for synthetic data, the dendrograms appear different; the Y-axis, indicating the height at which clusters are merged, is



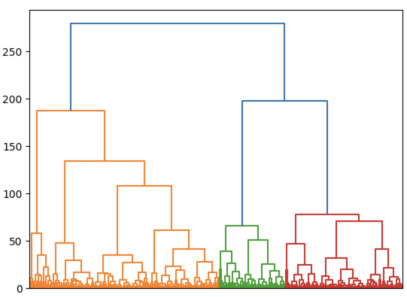
(a) Original data for California housing



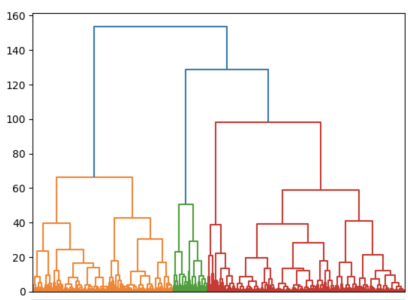
(b) Synthetic data created using CTGAN for California housing data



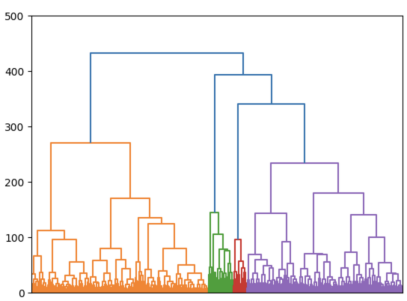
(c) 30-anonymous for California housing data



(d) Original data for cod-rna



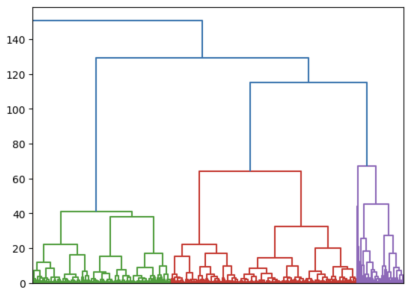
(e) Synthetic data created using CTGAN for cod-rna



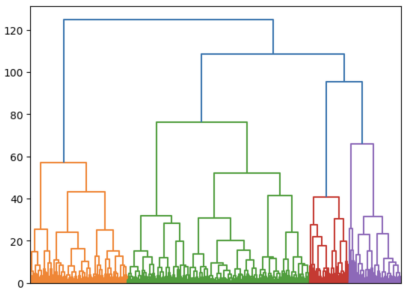
(f) 30-anonymous for cod-rna data

Figure 4.12: Dendrograms showing the HIERARCHICAL CLUSTERING for the California housing and cod-rna data

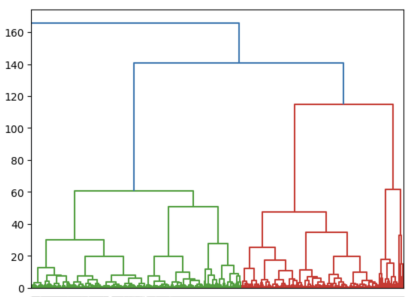
lower in the synthetic data, suggesting some loss of clustering information. In k-anonymous datasets, clusters become more compact as the value of k increases.



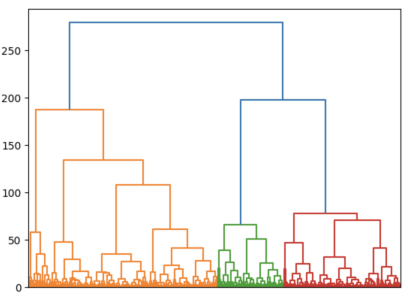
(a) Original Top 3 projection scores for CALIFORNIA-HOUSING



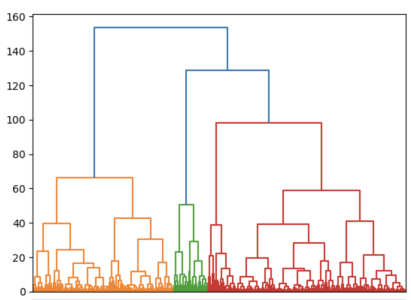
(b) Synthetic Top 3 projection scores for CALIFORNIA-HOUSING



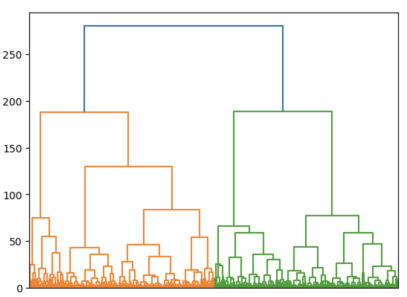
(c) 10-anonymous Top 3 projection scores for CALIFORNIA-HOUSING



(d) Original Top 3 projection scores for COD-RNA



(e) Synthetic Top 3 projection scores for COD-RNA



(f) 10-anonymous Top 3 projection scores for COD-RNA

Figure 4.13: HIERARCHICAL CLUSTERING for Top 3 projection scores of CALIFORNIA-HOUSING and COD-RNA

- Figure 4.13 shows dendrograms for the original and protected projection scores. The trends are similar to those observed with hierarchical clustering on the original, anonymous, and synthetic datasets. The differing clustering results for synthetic datasets/eigenvectors compared to the original datasets/eigenvectors are due to the synthetic data generation algorithm, CTGAN in this case, which reproduces data points within a

fixed range, leading to a loss of information regarding the actual number of clusters.

- We found that datasets and eigenvectors protected using k-anonymity produce more accurate clustering results compared to synthetic datasets created using CTGAN. For synthetic datasets, the cluster sizes are more compact than those in the original and k-anonymized datasets. Therefore, while generative algorithms can be useful for protecting outliers, they should be avoided for critical applications requiring better clustering results.

From our experiments in Section 4.6 and Section 4.7, we found out that the eigenvectors computed from k-anonymous data offer a better privacy-utility tradeoff compared to eigenvectors from synthetic data or those without any privacy protection, based on our risk and utility analysis

4.8 Conclusion

In this chapter, we focused on quantifying disclosure risk while balancing the utility. To quantify disclosure risk, we made a data reconstruction attack against PCA, and computed the proximity of reconstructed samples with the original samples. To compute the utility, we use HIERARCHICAL CLUSTERING. We not only compared the clustering results of original, and protected datasets, where we utilised k-anonymity privacy model and, generative networks for the protection; but we also compared the clustering utility of original, and protected principal components. From our analysis, we observed that k-anonymous data, and their principal components, provide a more balanced trade off between disclosure risk, and information loss over synthetic data, and synthetic projection scores. The reason is synthetic data, and projection scores led to the formation of clusters of compact sizes, resulting in the loss of information about the actual membership of data points to different clusters, as the height at which the clusters are merged declined in the case of synthetic datasets in comparison with original, and k-anonymous datasets, as shown in Figures 4.12, and 4.13. In our work, we considered the pre-aggregation scenario of FL-PCA. In the aggregation step, the central server aggregates the local eigenvectors to obtain global eigenvectors. If the eigenvectors are intercepted after the aggregation step, then it may be possible that a particular client is dominating the aggregation step, and on attacking the system using our methodology shown in Figure 4.9, the attacker infers the information of the client, who influenced the aggregation step the most. In this case, there is a possibility of individual privacy leakage. For future work, privacy and utility analysis can be done after the aggregation step in FL-PCA.

Chapter 5

Privacy Evaluation of Synthetic data

> In the digital age, privacy is not a given; it's something that must be fought for.
>
> *Tim Cook*

5.1 Synthetic Data

In the previous chapters, we discussed methods to preserve privacy, including privacy models such as k-anonymity [SS98], l-diversity [Mac+07], and t-closeness [LLV06]. While k-anonymity offers good utility, as shown in Chapter 3, it fails to protect against attribute inference attacks when the attacker has background knowledge of a target individual. This is why other variants of k-anonymity, such as l diversity, and t closeness were proposed. We also explored Differential Privacy (DP) [Dwo08]. DP, however, tends to suffer from poor utility at high privacy levels. This is especially noticeable when training decision trees on DP-protected data. In DP, the privacy level is controlled by the hyperparameter ϵ, where a lower value indicates a higher privacy level. Synthetic data has emerged as a promising alternative to preserve privacy. In this chapter, we focus on quantifying the privacy of synthetic datasets created using various generative networks. To do so, we perform an Attribute Inference Attack (AIA) against the synthetic datasets.

Within Privacy Preserving Machine Learning (PPML) and Privacy Preserving Data Publishing (PPDP), synthetic data is also a way to preserve user's privacy. Unlike real data, which is collected from actual observations, synthetic data is fabricated through algorithms designed to replicate patterns found in

real datasets. Synthetic data retains similar structural and statistical properties to the original data, enabling it to provide comparable utility. Generative networks are a common approach to creating synthetic data by closely matching the distributional properties of the original dataset. In our experiments, we focus on synthetic data generated by these networks.

We explore the privacy-utility tradeoff for synthetic data. Although its utility is often similar to the original data, the privacy of synthetic datasets remains questionable if records closely resemble or match the original data. This chapter discusses synthetic data in the context of privacy-preserving machine learning and data publishing. In PPML, machine learning models are trained on synthetic rather than original data. Research in PPML investigates whether models trained on synthetic data reduce information leakage compared to models trained on real data. For example, Khan *et al.* [KB23] examined the extent of membership leakage when models are trained on synthetic data.

When publishing data, synthetic data is released instead of the original. A proper privacy assessment of synthetic datasets must be done before publishing. Existing studies have used linkage and Attribute Inference Attacks (AIA) to assess synthetic data privacy. In a linkage attack, an adversary attempts to identify real individuals by linking records from the synthetic data to publicly available external datasets that contain Personally Identifiable Information (PII). A record in the synthetic data is linked to one in the original data based on a similarity metric, such as Euclidean distance. The synthetic record with the smallest Euclidean distance is linked to its corresponding original record.

Stadler *et al.* [SOT22] assessed the privacy of synthetic data by performing linkage attacks to evaluate the effectiveness of synthetic data and differentially private synthetic data in mitigating linkability risks. Their findings indicate that synthetic data alone does not offer strong protection against linkage attacks, especially for outliers, while differentially private synthetic data provides better protection but at the cost of reduced utility.

Our work focuses on AIA against synthetic datasets. The following sections review generative networks, privacy attacks against synthetic datasets, our attack methodology for an AIA, experimental results, and analysis.

5.2 Generative Networks

Generative models [Goo+14] play an important role in PPML by learning the underlying structure of a dataset and producing synthetic data that maintains similar statistical properties. This synthetic data can then be used instead of the original data for machine learning tasks. In this thesis, we explore the extent to which privacy is preserved in synthetic datasets, particularly against attacks like AIA.

5.2.1 Conditional Tabular Generative Adversarial Network

One of the generative models we focus on is the CTGAN, which is designed specifically for tabular data. GANs have been successful in many domains but face challenges when applied to tabular datasets, especially those containing both continuous, and categorical data types and imbalanced categories. To overcome these issues, CTGAN [Xu+19b] introduces several key modifications.

CTGAN addresses two major challenges in tabular data generation- (1) preserving the distribution of continuous data without loss of important information, and (2) handling categorical data imbalance. To tackle these, CTGAN employs "mode-specific normalization" for continuous data and introduces a "conditional vector" *cond* for handling discrete columns. By conditioning on one of the discrete columns, CTGAN can generate synthetic rows that maintain the distribution of the original data. Additionally, it uses a *training-by-sampling* technique, where *cond* and the training data are sampled based on the log frequency of each category, ensuring balanced representation across all possible discrete values. This approach enables CTGAN to generate high-quality synthetic tabular data, but the privacy implications of such data generation need to be tested through different kinds of privacy attacks against synthetic data.

The underlying mechanism of CTGAN is rooted in the classic GAN architecture, where two neural networks—a generator G and a discriminator D—compete with each other. The generator produces synthetic data while the discriminator attempts to distinguish between real and synthetic data. This dynamic is captured by the following objective function in Equation 5.1.

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \quad (5.1)$$

Here, x represents real data samples, z is the latent variable drawn from a prior distribution, $G(z)$ represents the synthetic data, and $D(x)$ is the prediction of the discriminator on whether x is real or synthetic. In our experiments, we use the data generated by CTGAN to evaluate its vulnerability to privacy attack like AIA, and assess how well it balances the privacy-utility tradeoff.

5.2.2 Differentially Private Generative Networks

To further investigate privacy, we use differentially private versions of GANs. As we have seen, Differential privacy (DP) provides a formal definition of privacy by adding noise to the data or the learning process to limit the impact of any single data point on the final output. For generative models like GANs, this involves applying differential privacy to either the gradient descent process or the output of the model itself. One such model is the Differentially Private Conditional Tabular GAN (DP-CTGAN) [Ros+20b]. In DP-CTGAN, differentially private stochastic gradient descent (DP-SGD) is applied to the discriminator, with random noise added to the gradient and a norm-clipping technique used to ensure differential privacy. This method closely follows the

principles of differentially private GANs [ZJW18]. While this provides privacy guarantees, the added noise can degrade the utility of the generated data, presenting a clear privacy-utility tradeoff.

We also experiment with another privacy-enhanced model, Private Aggregation of Teacher Ensembles PATE-CTGAN [Ros+20b], inspired by PATE-GAN [JYV18]. PATE-CTGAN partitions the original data into k subsets, each used to train a differentially private teacher model. These teacher models, acting as conditional generators, create synthetic samples conditioned on discrete columns. By aggregating the output of multiple teachers, PATE-CTGAN provides stronger privacy protection compared to a single generator model.

5.2.3 Diffusion Models for Tabular Data

Another class of generative models gaining attention for synthetic data generation is diffusion models [HJA20]. Diffusion models operate by gradually introducing noise into real data (the forward process) and then training a model to reverse this process (the reverse process) to recover or generate new synthetic data. This framework, while originally applied to image and continuous data, has recently been extended to tabular datasets.

Forward and Reverse Diffusion Processes

The forward diffusion process involves sequentially adding Gaussian noise to a real data point $\mathbf{x}_0$ over T timesteps, creating progressively noisier versions of the data. At timestep T, the data is almost completely random, resembling noise. Mathematically, this is modeled as:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I}),$$

where α_t controls the amount of noise added at each step. This process can be collapsed into a single distribution that maps from the initial data $\mathbf{x}_0$ to any noisy timestep $\mathbf{x}_t$:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}),$$

where $\bar{\alpha}_t$ represents the cumulative noise over all timesteps.

In the reverse process, the model learns to denoise the data, generating synthetic samples by gradually removing the noise and reconstructing the data. The reverse distribution is parameterized by a neural network $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, which estimates the previous timestep $\mathbf{x}_{t-1}$ from the current noisy data $\mathbf{x}_t$.

5.2.4 Tabular Diffusion Models (TAB-DDPM)

To generate synthetic tabular data, diffusion models must account for both continuous and categorical features. For continuous features, noise is added as in traditional diffusion models, but for categorical features, a discrete noise process is employed to maintain the integrity of categorical variables.

During the reverse process, the model predicts both continuous and categorical values. For continuous data, the reverse process follows the Gaussian framework, while for categorical data, softmax functions are used to map noisy representations back to valid categories. Similar to CTGAN, these tabular diffusion models incorporate conditioning mechanisms to capture dependencies between features. In our experiments, we assess the privacy-utility tradeoff of synthetic data created by tabular diffusion models, and evaluate their privacy against AIA. There is also differentially private version for diffusion models, where noise is added during both the forward and reverse diffusion processes to satisfy differential privacy. In the paper on differentially private diffusion models [Doc+22], the author demonstrates that differential privacy can be integrated with diffusion models without significantly compromising the quality of the generated data.

5.3 Privacy Attacks Against Synthetic Data

Generative models like CTGAN are tools to generate synthetic data that mimics real-world data. However, synthetic data is not immune to privacy attacks. Although synthetic data is designed to obscure sensitive information while retaining the statistical properties of the original dataset, adversaries can still exploit it for malicious purposes. In this section, we review several privacy attacks and discuss how adversaries can compromise privacy in synthetic data, focusing on linkage attacks, attribute inference attacks (AIA), and membership inference attacks.

5.3.1 Linkage Attacks

One of the most prominent privacy attacks against synthetic data is the linkage attack. In a linkage attack, an adversary attempts to identify individuals in the synthetic dataset by linking records from synthetic data to external datasets that contain Personally Identifiable Information (PII). If the synthetic data closely resembles the original data, records in the synthetic dataset may match or be very similar to records in the real dataset. By using similarity metrics, such as Euclidean distance or other distance measures, an attacker can link synthetic records back to real individuals.

For example, suppose a synthetic dataset contains a combination of demographic features such as age, gender, and occupation. If an external dataset contains the same combination of features, and some of the records are similar enough, the adversary can successfully re-identify individuals by linking the synthetic records to the external ones. Stadler *et al.* [SOT22] performed linkage attacks on synthetic datasets and demonstrated that synthetic data alone does not always provide strong protection against such attacks, especially for outliers in the original dataset. This vulnerability is heightened if differentially private mechanisms are not applied, as the generated synthetic data may retain

strong correlations with the real data.

5.3.2 Membership Inference Attacks (MIA)

Membership Inference Attacks (MIA) aim to determine whether a specific record was part of the training dataset used to generate synthetic data. These attacks exploit overfitting in machine learning models to differentiate between data points seen during training and those that were not.

In an MIA, the attacker trains shadow models that replicate the behavior of the target generative model by using known data and then analyzes the output of the target model to infer whether a particular instance was included in the training set [Sho+17]. Specifically, the adversary can leverage the fact that the model tends to provide more confident predictions for training instances compared to unseen data. By training a classifier based on the outputs of the shadow models, the adversary can estimate the likelihood of a record's inclusion in the original dataset, thereby inferring membership.

Stadler *et al.* [SOT22], and Khan *et al.* [KB23] highlights an important consideration- while synthetic data can reduce privacy risks, it does not completely mitigate membership inference attacks. Synthetic data often preserves statistical relationships present in the original dataset, which can still be exploited by MIA. Both, Stadler *et al.* [SOT22], and Khan *et al.* explore how different generative models and synthetic data generation processes affect the vulnerability to MIA, showing that generative models like GANs, and Variational Autoencoders (VAEs) are still vulnerable to these attacks. The analysis reveals that attackers can still infer membership information when the synthetic data retains too much similarity with the original data distribution.

5.3.3 Model Inversion Attacks

Model inversion attacks exploit the generative model to infer sensitive information about individuals in the original dataset. Although adversaries do not have direct access to the original data, they can query the model with specific inputs and observe the outputs to infer private attributes. Fredrikson *et al.* [FJJ15] demonstrated this form of attack, where an attacker uses the outputs of the model to recover sensitive features.

In the case of synthetic data, attackers can leverage the model to generate records that are statistically similar to those in the original dataset. By issuing targeted queries and analyzing the outputs, adversaries may be able to reconstruct sensitive information, even attributes that were not explicitly present in the dataset. Zhang *et al.* [Zha+20] showed how generative models could be exploited to perform such attacks and recover private data, which means that the privacy risks are present when synthetic data is generated by models without sufficient privacy protection mechanisms, particularly when the model overfits to the training data.

5.3.4 Reconstruction Attacks

Reconstruction Attacks go beyond inferring specific attributes by attempting to reconstruct the original dataset from synthetic data. These attacks exploit the resemblance between the synthetic data and the original dataset to regenerate individual records, or even the entire original dataset.

Reconstruction is typically carried out by solving an optimization problem that minimizes the difference between the synthetic and real data distributions [ZLH19]. An attacker may iteratively tweak the generated data until it aligns closely with the original data points. If the synthetic data generator does not sufficiently obfuscate the real data, an adversary can exploit this to reconstruct individual records, thereby compromising privacy.

5.4 Attribute Inference Attack against synthetic data

In Attribute Inference Attack (AIA), the attacker has knowledge about a subset of attributes of some specific target individuals, and at least one publicly available synthesized version of the original data. Using this knowledge, the attacker aims to deduce sensitive attributes of those specific target individuals. The attacker conducts an AIA by analyzing the patterns and correlations present in the synthetic dataset along with their background knowledge about some target individuals. Even if specific sensitive attributes are not explicitly included in the synthetic data, attackers can leverage the relationships and statistical properties of the data to uncover the hidden or confidential information about individuals, posing a significant risk to privacy of individuals. The success of attribute inference risk in synthetic datasets majorly depends on the correlation between the sensitive attributes, and the publicly available attributes. For instance, if the synthetic data maintains a strong correlation between income and medical condition, and the attacker knows the income, they can infer the medical condition of individuals in the original dataset. The attacker in an AIA is stronger than the attacker in a linkage attack, in the sense that the attacker can make use of their background knowledge about specific target individuals. Hence, the failure in a linkage attack does not exclude the possibility of an AIA [HME20].

There are different ways to conduct an AIA for synthetic datasets, and evaluate the success of an AIA. One of the ways is to utilise machine learning models to learn the patterns and correlations among attributes in the synthetic data [HME20]. The approach is to train a machine learning model like- logistic regression, decision trees, and support vector machines (SVMs) on the publicly available synthetic data. The trained models are utilised by the attacker to predict or infer unknown sensitive attributes. The evaluation of the success of an attack is done by using a metric called, Correct Attribution Probability (CAP). The CAP for a record j in the original dataset is given by

$$\text{CAP}_{s,j} = \frac{\sum_{i=1}^{n} \mathbf{1}[T_{s,i} = T_{o,j} \wedge K_{s,i} = K_{o,j}]}{\sum_{i=1}^{n} \mathbf{1}[K_{s,i} = K_{o,j}]} \tag{5.2}$$

where

- $T_{o,j}$ is the target value for the j-th record in the original dataset O.
- $T_{s,i}$ is the target value for the i-th record in the synthesized dataset S.
- $K_{o,j}$ is the key attribute value(s) for the j-th record in the original dataset O.
- $K_{s,i}$ is the key attribute value(s) for the i-th record in the synthesized dataset S.
- $\mathbf{1}[\cdot]$ is the indicator function, which equals 1 if the condition inside the brackets is true, and 0 otherwise.

Equation (5.2) calculates the empirical probability that an attacker can correctly attribute the target value $T_{o,j}$ given their knowledge of $K_{o,j}$ and access to the synthesized dataset S. We also proposed an AIA for synthetic dataset. We explain our attack methodology in the next Section 5.5.

5.5 Our Attack Methodology for AIA

Our aim is to know how successful the attackers are in inferring the information pertaining to a target individual. To execute an attribute inference attack, we assume that the attacker has access to a subset of attributes of a target individual, and a synthesized version of the original data, which is similar to [HME20]. Using the publicly available synthetic data, and a subset of original attributes, the attacker aims to infer the information of attributes unknown to the attacker, as shown in Figure 5.1. For our experimentation, we obtain the synthesized version of the original data using CTGAN, DP-CTGAN, PATE-CTGAN, and TAB-DDPM (diffusion models for tabular data).

To infer the value of a sensitive attribute for a target individual, the attacker first calculates the Euclidean distance between the subset of real attributes they already know and the corresponding subset from the synthetic data. For each target record, they find the record in the synthetic dataset that has the smallest Euclidean distance. This closest match is then used to infer the unknown attributes by taking the values from the synthetic record at that particular index. Essentially, this process aligns the synthetic data with the original data by measuring the similarity between the known real attributes and their synthetic versions using Euclidean distance [KT23c].

After obtaining the aligned synthetic data, we evaluate the success of attribute inference. To measure the success of the attack, we use the following metric called Inference Accuracy.

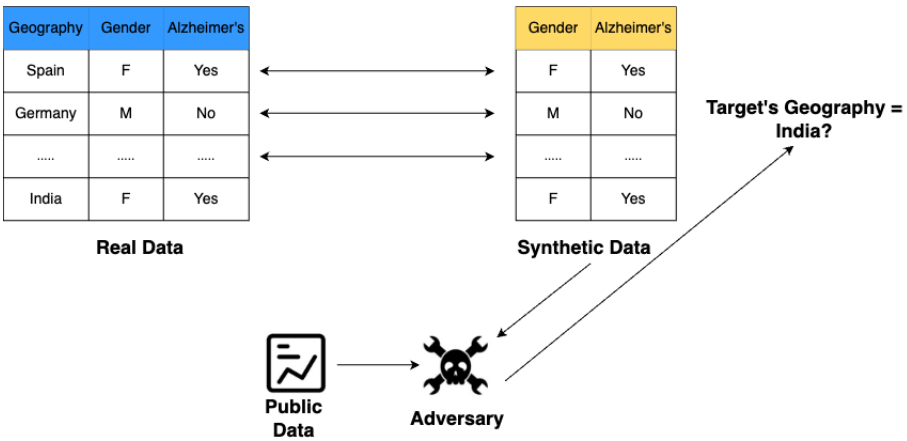


Figure 5.1: Attribute Inference Attack against Synthetic Data

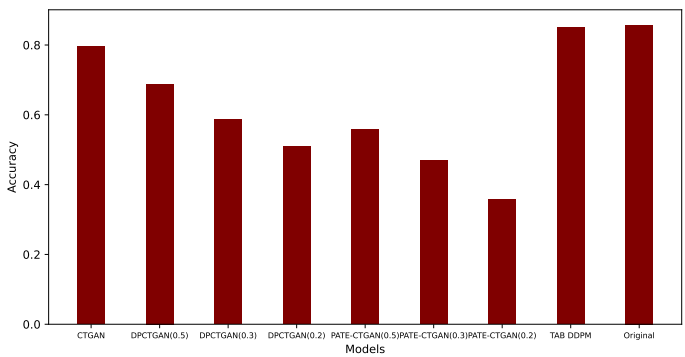


Figure 5.2: Accuracy of different kinds of GANs

Model	$\delta = 0.1$	$\delta = 0.3$	$\delta = 0.5$	$\delta = 0.7$	$\delta = 1$
CTGAN	0.191	0.575	0.871	0.977	0.995
DP-CTGAN	**0.117**	**0.352**	**0.568**	**0.706**	**0.826**
PATE-CTGAN	0.125	0.357	0.585	0.726	0.858
TAB-DDPM	0.232	0.604	0.834	0.919	0.970

Table 5.1: Attack Inference Accuracy on churn-modelling data for Age attribute

Suppose S is the synthetic record obtained after the alignment, and O is the original record. Let n be the total number of samples in the original and the synthetic data, O_j be the value of the sensitive attribute from the original

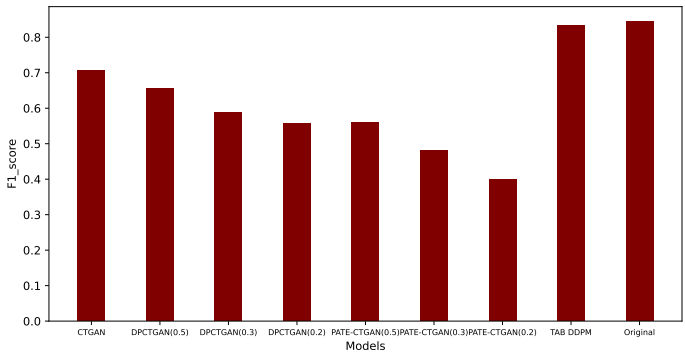


Figure 5.3: F1-score of different kinds of GANs

data, which the attacker aims to infer, and S_j is the inferred attribute in the synthetic data corresponding to the sensitive attribute O_j. Let δ be the deviation between the original and the synthetic attribute that can be tolerated to measure the level of inference for a record. The lower the δ, the closer the values of S_j and O_j must be to each other.

For the continuous attributes, the Inference Accuracy, $I.A.$ is defined as follows.

$$I.A. = \frac{\#\left\{\hat{S}_j : |\frac{O_j - S_j}{S_j}| \leq \delta, j = 1 \dots n\right\}}{n} \tag{5.3}$$

where # means count. I.A. is the percentage of inferred entries for which the relative errors are within δ.

For the categorical attributes, the above formula is more strict (as we are counting only the **exact matches**) and changes to the formula below.

$$I.A. = \frac{\#\left\{\hat{S}_j : O_j == S_j, j = 1 \dots n\right\}}{n} \tag{5.4}$$

It is important to note that the formula to compute Inference Accuracy, denoted as $I.A.$, is identical to the formula used for Reconstruction Accuracy, $R.A.$, in the previous Chapter 4. We refer to it as $I.A.$ in this chapter because the attacker aims to infer specific sensitive attributes. In contrast, in Chapter 4, the goal of the attacker was to reconstruct the entire database.

5.6 Data and Experimental Settings

We experimented with the churn modeling and diabetes datasets, as shown in Table 5.2 from the UCI Machine Learning Repository[1]. The churn modeling dataset contains details of a bank's customers, with attributes such as age, gender, and account balance. The target variable is binary, indicating whether the customer closed their account or continued as a customer.

The diabetes dataset, on the other hand, includes various health metrics such as glucose level, blood pressure, body mass index (BMI), and age. The aim is to predict the onset of diabetes. These diverse datasets allowed us to assess the performance and utility of our generative models across different contexts.

Data	#Train	#Validation	#Test	#Num	#Cat	Task Type
Churn-Modelling	6400	1600	2000	7	4	Binary Classification
Diabetes	491	123	154	8	0	Binary Classification

Table 5.2: Description of datasets

Attribute	CTGAN	DP-CTGAN, eps = 0.2	PATE-CTGAN, eps = 0.2	TAB-DDPM
Gender	0.495	0.505	0.539	0.505
Female	0.629	0.642	0.476	0.628
Male	0.383	0.390	0.505	0.403
Location	0.364	0.327	0.332	0.378
Spain	0.259	0.313	0.345	0.235
Germany	0.263	0.502	0.392	0.238
France	0.468	0.246	0.296	0.522

Table 5.3: Attack Inference Accuracy on churn-modelling data for categorical attributes

We used the publicly available code for CTGAN [2], DP-CTGAN, PATE-CTGAN [3], and TAB-DDPM [4]. We mention the value of the privacy parameter ϵ in our results section. We obtain the anticipated trend, which states that the lower the value of ϵ, the higher the privacy (shown by the results of attribute inference attack), and the lower the utility of the synthetic dataset (shown by the accuracy, precision, recall and the F1 score obtained by the random forest classifier). We provide our code [5] for executing the attribute inference attack on the synthetic datasets.

[1] https://archive.ics.uci.edu/

[2] https://pypi.org/project/ctgan/

[3] https://github.com/opendp/smartnoise-sdk

[4] https://github.com/yandex-research/tab-ddpm

[5] https://github.com/SaloniKwatra0802/Evaluation-of-Synthetic-Dataset-via-an-Attribute-Inference-Attack

Attribute	CTGAN	DP-CTGAN, eps = 0.3	PATE-CTGAN, eps = 0.3	TAB-DDPM
No. of Pregnancies (exact match)	0.057	**0.051**	0.053	0.101
$\delta = 1$	0.177	**0.163**	0.181	0.283
$\delta = 2$	0.279	**0.257**	0.287	0.446
Disease	0.505	**0.486**	0.527	0.648
Diabetic	0.678	0.475	0.663	**0.462**
Non-Diabetic	**0.412**	0.495	**0.370**	0.746

Table 5.4: Attack Inference Accuracy on diabetes data for categorical attributes

5.7 Experimental Results and Analysis

We observed that combining differential privacy with the synthetic datasets degrades the utility. We experimented with $\epsilon = \{0.2, 0.3, 0.5\}$. Our experiments favored DP-CTGAN over PATE-CTGAN in terms of utility using Random Forest classifier. This analysis is consistent with other works in this field [Ros+20a]. For categorical attributes like gender, geographical location, etc., attribute inference accuracy is low. We can say that it is better than random guessing in the case of binary categorical attributes like gender, as shown in Table 5.3, because the churn-modelling dataset was not perfectly balanced in terms of the gender and the location. We used $\epsilon = 0.2$ for DP-CTGAN and PATE-CTGAN for the results in Table 5.3, and $\epsilon = 0.3$ for DP-CTGAN and PATE-CTGAN for the results in Table 5.4.

5.8 Conclusion

The aim of this work was to evaluate the privacy of synthetic data generated by various models, particularly in the context of public release. To accomplish this, we conducted an Attribute Inference Attack on synthetic datasets generated using different generative algorithms. Our findings indicate that inferring exact attributes is not highly feasible for synthetic datasets, especially when dealing with categorical attributes. However, we discovered that approximate inference remains possible for continuous attributes, suggesting that continuous data may still pose privacy risks, even when synthetic data is used. Hence, our results emphasize that while synthetic data can mitigate some privacy risks, it does not eliminate them entirely. Overall, this work highlights the critical need for continuous evaluation of the privacy implications associated with synthetic data generation.

Chapter 6

Conclusion

> Historically, privacy was almost implicit, because it was hard to find and gather information. But in the digital world, whether it's digital cameras or satellites or just what you click on, we need to have more explicit rules - not just for governments but for private companies.
>
> Bill Gates

The work in this thesis has made significant contributions to the field of privacy-preserving machine learning, particularly in federated learning with decision trees. It also explores methods to measure privacy leakage during the training of machine learning models and the quantification of privacy in synthetic datasets. We conclude this document by summarizing our key contributions and suggesting potential directions for future research.

6.1 Key Contributions

In this section, we outline our contributions, chapter by chapter.

- In Chapter 3, we proposed two federated learning frameworks with decision trees [KT21b], [KT24], and a privacy-utility tradeoff analysis of an existing federated framework for training Gradient Boosting Decision Trees, called SimFL [LWH20]. Our proposed federated learning frameworks in Chapter 3 performs aggregation of decision trees, which are trained locally by each distributed client. We also impose privacy restrictions in our proposed federated learning frameworks following k-anonymity, and differential privacy on the data of clients. For the privacy-

utility analysis of an existing federated framework with decision trees, called SimFL, we demonstrated two data reconstruction attacks along with the utility analysis using Gradient Boosting Decision Trees. From our privacy analysis, we concluded that Locality Sensitive Hashing (LSH) is not privacy preserving alone. This means that sharing of hashed valued computed by LSH lead to privacy breaches. Therefore, an additional anonymization layer is needed prior to the computation of hash values. We proposed a framework "Rakshit" (Rakshit is a word from Hindi language, which means "protector"), in which we demonstrated that the privacy breach via a reconstruction attack is reduced, when we have an additional anonymization layer prior to the computation of hashed values computed using LSH.

- In Chapter 4, we focused on measuring the privacy risks associated with the information captured by eigenvectors in Principal Component Analysis (PCA) via a data reconstruction attack. We did a privacy-utility analysis in the case of PCA. For privacy analysis, we did a data reconstruction attack, and utilised synthetic data, k-anonymity, and differential privacy to provide protection to the data of clients. For the utility analysis, we used Hierarchical Clustering. We provide interesting insights from our privacy and utility analysis. Our experiments on privacy and utility analysis tell us when to apply which privacy protection method.

- In Chapter 5, we focused on measuring the privacy risks associated with synthetic datasets created by various generative networks. We demonstrated an Attribute Inference Attack (AIA) against synthetic datasets, in which the attacker has access to a subset of some publicly available attributes, and one (at least) synthesized version of the dataset, and the goal of the attacker is to infer the sensitive attributes of a target individual. We show the success of the attacker via a metric called Inference Accuracy, which quantifies the proximity of the sensitive values inferred by the attacker, and the actual sensitive values. From our experimental results, we show that exact attribute inference is not highly feasible. But, approximate attribute inference is possible.

6.2 Future Directions

From our work, there are several promising research directions, which could further advance the field of privacy-preserving machine learning

- **Enhancing Federated Learning Algorithms with Decision Trees** Our research introduced a one-shot federated learning algorithm for decision trees. A potential future research direction is to develop more adaptive algorithms that allow clients to update their decision trees dynamically. This could involve allowing clients to recompute their decision

trees at each iteration based on partial trees shared by a central aggregator. Additionally, clients might have different policies regarding the sharing of decision nodes. Pruning the decision paths by the clients before sharing them could provide new insights into optimally balancing privacy and model performance.

- **Explainable and Privacy-Preserving Machine Learning** In this thesis, we worked with decision trees, and proposed privacy-preserving federated learning algorithms with decision trees. We chose to work with decision trees, as a decision tree model is implicitly explainable. But, complex, and black box machine learning models like neural networks are not implicitly explainable. The decisions of neural networks need to be made explainable by applying techniques for post hoc analysis like LIME (Local Interpretable Model-Agnostic Explanations), and Shapley values [Xu+19a]. Hence, an exciting domain of future research is the intersection of explanability, privacy, and machine learning [BT24].

- **Integration of Integral Privacy and Federated Learning with Decision Trees** We utilized k-anonymity and differential privacy to enhance privacy in federated learning with decision trees. One possible future approach could be exploring integral privacy in federated learning with decision trees, where the central aggregator selects the most frequently occurring decision tree, known as the integrally private decision tree. This selection process can be implemented through string matching of decision paths shared by clients [ST19]. However, investigating decision trees at the central aggregator poses privacy risks, as it leads to the estimated reconstruction of the databases of clients [GGH12]. One potential solution is to apply k-anonymity or differential privacy at the client level and then select the integrally private decision tree at the aggregation level.

- **Privacy Quantification for Machine Learning Models** There are a plethora of machine learning models. With advancements in machine learning techniques, various types of attacks have been studied against these models. In this thesis, we have discussed and examined several attacks, such as data reconstruction attacks and attribute inference attacks. However, other attacks in the literature are also worth investigating. One of such attacks is a **Model Inversion** attack, in which the attacker analyzes the predictions and gradients provided by the model, and derive valuable insights that may reveal private data points [FJJ15]. Another attack is **Model Extraction**, which involves creating a replica of a target model through extensive querying. Attackers utilize prediction APIs to gather responses from the model and then apply machine learning techniques to approximate the target model's behavior [Che+16]. This method allows adversaries to replicate the functionality of a machine learning model and, in some cases, gain access to sensitive data. There

are also **Backdoor Attacks**, which involve embedding a hidden trigger in the training data that activates malicious behavior during inference. By subtly altering the training process, attackers can insert a backdoor into the model, which will lead to specific, unauthorized responses when the trigger is present [GDL17]. Understanding and defending against these attacks is important for ensuring the privacy of machine learning models. Each attack type, and each machine learning model requires tailored strategies to mitigate the impact of each attack for a specific machine learning model.

Our research provides a foundation for advancing privacy-preserving machine learning. By proposing privacy preserving federated learning algorithms for decision trees, and the privacy quantification strategies for synthetic data, and machine learning models, we contribute to the development of more secure machine learning applications. The proposed future directions offer opportunities to build on our findings and continue addressing emerging challenges in the field of privacy-preserving machine learning.

About the Author

Saloni Kwatra completed her Bachelors in Computer Science from the University of Delhi (2014-2017) and her Master's in Computer Science from South Asian University, New Delhi (2018-2020). Her Masters thesis was titled *Diagnosis of Alzheimers Disease Using Machine Learning Techniques*. Saloni pursued her PhD in Computer Science at Umeå University, Sweden (December 2020 - October 2024), supervised by Professor **Vicenç Torra**. Her doctoral research, titled *Navigating Data Privacy and Utility: A Strategic Perspective*, focuses on addressing privacy concerns in synthetic datasets generated by machine learning models. In this thesis, Saloni investigates the balance between privacy and utility, and propose strategies to ensure data privacy without compromising analytical performance. Her work also includes the development of novel algorithms to enhance privacy in machine learning systems, with a particular emphasis on decision trees and federated learning frameworks.

Bibliography

[ALZ13] Artur Andrzejak, Janusz Langner, and Jose-Luis Zabala. "Decision Tree Based Concept Drift Handling in Network Traffic Classification". In: *Proceedings of the 5th IEEE Conference on Network Infrastructure and Digital Content (IC-NIDC)* (2013).

[AMP10] Gagan Aggarwal, Nina Mishra, and Benny Pinkas. "Secure computation of the median (and other elements of specified ranks)". In: *Journal of cryptology* 23.3 (2010), pp. 373–401.

[AW10] Hervé Abdi and Lynne J Williams. "Principal component analysis". In: *Wiley interdisciplinary reviews: computational statistics* 2.4 (2010), pp. 433–459.

[BG98] Suresh Balakrishnama and Aravind Ganapathiraju. "Linear discriminant analysis-a brief tutorial". In: *Institute for Signal and information Processing* 18.1998 (1998), pp. 1–8.

[BL01] Audrius Bursteinas and Andrew D. Long. "Efficient Decision Tree Induction by Using Dynamic Hyper-Rectangles". In: *Proceedings of the 2001 International Conference on Computational Science*. 2001.

[BM86] Jean-Paul Barthélemy and Fred S. McMorris. "The Median Procedure for n-Trees". In: *Journal of Classification* 3 (1986), pp. 329–334. DOI: 10.1007/BF01896817.

[Bre+86] Leo Breiman et al. *Classification and Regression Trees*. Pacific Grove, CA: Wadsworth and Brooks/Cole, 1986.

[BT13] Johan Barthelemy and Philippe L Toint. "Synthetic population generation without a sample". In: *Transportation Science* 47.2 (2013), pp. 266–279.

[BT24] Aso Bozorgpanah and Vicenç Torra. "Explainable machine learning models with privacy". In: *Progress in Artificial Intelligence* 13.1 (2024), pp. 31–50.

[Che+16] Hongge Chen et al. “Stealing Machine Learning Models via Prediction APIs”. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 601–612.

[Che+21a] Xiaolin Chen et al. “Fed-eini: An efficient and interpretable inference framework for decision tree ensembles in vertical federated learning”. In: *2021 IEEE international conference on big data (big data)*. IEEE. 2021, pp. 1242–1248.

[Che+21b] Kewei Cheng et al. “Secureboost: A lossless federated learning framework”. In: *IEEE Intelligent Systems* 36.6 (2021), pp. 87–98.

[DM+06] Josep Domingo-Ferrer, Josep M. Mateo-Sanz, et al. “MDAV for microaggregation”. In: *Proceedings of International Workshop on Privacy and Anonymity in Information Society* (2006).

[DM02] Josep Domingo-Ferrer and Josep Maria Mateo-Sanz. “Practical data-oriented microaggregation for statistical disclosure control”. In: *IEEE Transactions on Knowledge and data Engineering* 14.1 (2002), pp. 189–201.

[DN93] David Defays and P. Nanopoulos. “An extension of k-means for overlapping clustering”. In: *European Journal of Operational Research* 57.1 (1993), pp. 1–12.

[Doc+22] Tim Dockhorn et al. “Differentially private diffusion models”. In: *arXiv preprint arXiv:2210.09929* (2022).

[Dom07] Josep Domingo-Ferrer. “A three-dimensional conceptual framework for database privacy”. In: *Secure Data Management: 4th VLDB Workshop, SDM 2007, Vienna, Austria, September 23-24, 2007. Proceedings 4*. Springer. 2007, pp. 193–202.

[DR+14] Cynthia Dwork, Aaron Roth, et al. “The algorithmic foundations of differential privacy”. In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407.

[DT01] Josep Domingo-Ferrer and Vicenc Torra. “A quantitative comparison of disclosure control methods for microdata”. In: *Confidentiality, disclosure and data access: theory and practical applications for statistical agencies* (2001), pp. 111–134.

[DT05] Josep Domingo-Ferrer and Vicenç Torra. “A Survey of Inference Control Methods for Privacy-Preserving Data Mining”. In: *Lecture Notes in Computer Science* (2005), pp. 53–80.

[Dwo08] Cynthia Dwork. “Differential privacy: A survey of results”. In: *International conference on theory and applications of models of computation*. Springer. 2008, pp. 1–19.

[FJJ15] Matt Fredrikson, Suman Jana, and Thorsten Joachims. "Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (2015), pp. 1322–1333.

[FL20] Chenglin Fan and Ping Li. "Classification acceleration via merging decision trees". In: *Proceedings of the 2020 ACM-IMS on foundations of data science conference*. 2020, pp. 13–22.

[Fri01] Jerome H. Friedman. "Greedy Function Approximation: A Gradient Boosting Machine". In: *The Annals of Statistics* 29.5 (2001), pp. 1189–1232. DOI: 10.1214/aos/1013203451.

[Gar+12] Salvador Garcia et al. "A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning". In: *IEEE transactions on Knowledge and Data Engineering* 25.4 (2012), pp. 734–750.

[GDL17] Tianyu Gu, Brendan Dolgov, and Sergey Levine. "BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain". In: *Proceedings of the 2017 IEEE International Conference on Computer Vision*. IEEE, 2017, pp. 3750–3758.

[GGH12] Sébastien Gambs, Ahmed Gmati, and Michel Hurfin. "Reconstruction attack through classifier analysis". In: *Data and Applications Security and Privacy XXVI: 26th Annual IFIP WG 11.3 Conference, DBSec 2012, Paris, France, July 11-13, 2012. Proceedings 26*. Springer. 2012, pp. 274–281.

[GKD98] J M˜ Gouweleeuw, Peter Kooiman, and PP De Wolf. "Post randomisation for statistical disclosure control: Theory and implementation". In: *Journal of official Statistics* 14.4 (1998), p. 463.

[Gol98] Oded Goldreich. "Secure multi-party computation". In: *Manuscript. Preliminary version* 78.110 (1998), pp. 1–108.

[Goo+14] Ian Goodfellow et al. "Generative adversarial nets". In: *Advances in neural information processing systems* (2014), pp. 2672–2680.

[Hay+17] Jamie Hayes et al. "Logan: Membership inference attacks against generative models". In: *arXiv preprint arXiv:1705.07663* (2017).

[HCB97] Lawrence O. Hall, Nitesh V. Chawla, and Kevin W. Bowyer. "Combining Decision Trees Learned in Parallel". In: *Proceedings of the Conference on Artificial Intelligence* (1997).

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[HME20] Markus Hittmeir, Rudolf Mayer, and Andreas Ekelhart. "A baseline for attribute disclosure risk in synthetic data". In: *Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy*. 2020, pp. 133–143.

[HR22] Anne Hartebrodt and Richard Röttger. "Federated horizontally partitioned principal component analysis for biomedical applications". In: *Bioinformatics Advances* 2.1 (2022), vbac026.

[Hu+21] Hongsheng Hu et al. "Differentially private locality sensitive hashing based federated recommender system". In: *Concurrency and Computation: Practice and Experience* (2021), e6233.

[Hu+22] Hongsheng Hu et al. "Membership inference attacks on machine learning: A survey". In: *ACM Computing Surveys (CSUR)* 54.11s (2022), pp. 1–37.

[Hul+20] Aline Hulot et al. "Fast Algorithms for Multiple Change-Point Detection With the Group Fused Lasso". In: *Journal of Computational and Graphical Statistics* 29.3 (2020), pp. 453–468. DOI: 10.1080/10618600.2019.1706540.

[Hun+] A Hundepool et al. "μ-argus version 3.2 software and users manual, 2005". In: *URL http://neon. vb. cbs. nl/casc* ().

[HW01] Zengyi Huang and Paul Williamson. "A comparison of synthetic reconstruction and combinatorial optimisation approaches to the creation of small-area microdata". In: *Department of Geography, University of Liverpool* (2001).

[IS16] Hafiz Imtiaz and Anand D Sarwate. "Symmetric matrix perturbation for differentially-private principal component analysis". In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2016, pp. 2339–2343.

[JYV18] James Jordon, Jinsung Yoon, and Mihaela Van Der Schaar. "PATE-GAN: Generating synthetic data with differential privacy guarantees". In: *International conference on learning representations*. 2018.

[KB23] Md Sakib Nizam Khan and Sonja Buchegger. "The Impact of Synthetic Data on Membership Inference Attacks". In: *International Symposium on Security and Privacy in Social Networks and Big Data*. Springer. 2023, pp. 93–108.

[KT21a] Saloni Kwatra and Vicenç Torra. In: *International Conference on Fuzzy Systems*. IEEE. 2021, pp. 1–6.

[KT21b] Saloni Kwatra and Vicenç Torra. "A k-anonymised federated learning framework with decision trees". In: *International Workshop on Data Privacy Management*. Springer. 2021, pp. 106–120.

[KT23a] Saloni Kwatra and Vicenç Torra. "(Rakshit): Attacking and Defending Federated Learning with Decision Trees". In: *Available at SSRN* (2023). URL: https://ssrn.com/abstract=4348221.

[KT23b] Saloni Kwatra and Vicenç Torra. "Data Reconstruction Attack Against Principal Component Analysis". In: *International Symposium on Security and Privacy in Social Networks and Big Data*. Springer. 2023, pp. 79–92.

[KT23c] Saloni Kwatra and Vicenç Torra. "Empirical Evaluation of Synthetic Data Created by Generative Models via Attribute Inference Attack". In: *IFIP Advances in Information and Communication Technology*. Springer. 2023.

[KT24] Saloni Kwatra and Vicenç Torra. "DISCOLEAF: Personalized DIScretization of COntinuous Attributes for LEArning with Federated Decision Trees". In: *International Conference on Privacy in Statistical Databases*. Springer. 2024.

[KVT23] Saloni Kwatra, Ayush K Varshney, and Vicenç Torra. "Integrally Private Model Selection for Support Vector Machine". In: *European Symposium on Research in Computer Security*. Springer. 2023, pp. 249–259.

[LDR06] Kristen LeFevre, David J DeWitt, and Raghu Ramakrishnan. "Mondrian multidimensional k-anonymity". In: *22nd International conference on data engineering (ICDE'06)*. IEEE. 2006, pp. 25–25.

[LLV06] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. "t-closeness: Privacy beyond k-anonymity and l-diversity". In: *2007 IEEE 23rd international conference on data engineering*. IEEE. 2006, pp. 106–115.

[LWH20] Qinbin Li, Zeyi Wen, and Bingsheng He. "Practical federated gradient boosting decision trees". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 04. 2020, pp. 4642–4649.

[Mac+07] Ashwin Machanavajjhala et al. "l-diversity: Privacy beyond k-anonymity". In: *Acm transactions on knowledge discovery from data (tkdd)* 1.1 (2007), 3–es.

[McM+17] Brendan McMahan et al. "Communication-efficient learning of deep networks from decentralized data". In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[MD98] Josep M Mateo-Sanz and Josep Domingo-Ferrer. "A comparative study of data protection methods". In: *Proceedings of the Joint UN/ECE-Eurostat Work Session on Statistical Data Confidentiality* (1998).

[Mir17] Ilya Mironov. "Rényi differential privacy". In: *30th computer security foundations symposium (CSF)*. IEEE. 2017, pp. 263–275.

[MM81] Timothy Margush and Fred S. McMorris. “Consensus n-Trees”. In: *Bulletin of Mathematical Biology* 43 (1981), pp. 239–244. DOI: 10.1007/BF02459517.

[Moo96] Richard Moore. *Controlled data-swapping techniques for masking public use microdata sets*. US Census Bureau [custodian], 1996.

[NHT08] Jordi Nin, Javier Herranz, and Vicenç Torra. “Rethinking rank swapping to decrease disclosure risk”. In: *Data & Knowledge Engineering* 64.1 (2008), pp. 346–364.

[Par18] Stuart L Pardau. “The california consumer privacy act: Towards a european-style privacy regime in the united states”. In: *J. Tech. L. & Pol'y* 23 (2018), p. 68.

[PCZ09] Wei Peng, Juhua Chen, and Haiping Zhou. “An implementation of ID3-decision tree learning algorithm”. In: *From web. arch. usyd. edu. au/wpeng/DecisionTree2. pdf Retrieved date: May* 13 (2009).

[Pef+07] Ken Peffers et al. “A design science research methodology for information systems research”. In: *Journal of management information systems* 24.3 (2007), pp. 45–77.

[Pin+20] Walter Hugo Lopez Pinaya et al. “Autoencoders”. In: *Machine learning*. Elsevier, 2020, pp. 193–208.

[Put+14] P. M. Putora et al. “Decision Tree Based Consensus in the Management of Oligometastatic Non-Small Cell Lung Cancer”. In: *Radiotherapy and Oncology* 110.3 (2014), pp. 303–307. DOI: 10.1016/j.radonc.2014.03.013.

[Ros+20a] L. Rosenblatt et al. “Differentially private synthetic data: Applied evaluations and enhancements”. In: *arXiv preprint* (2020). eprint: arXiv:2011.05537.

[Ros+20b] Lucas Rosenblatt et al. “Differentially private synthetic data: Applied evaluations and enhancements”. In: *arXiv preprint arXiv:2011.05537* (2020).

[SD12] Jordi Soria-Comas and Josep Domingo-Ferrer. “Probabilistic k-anonymity through microaggregation and data swapping”. In: *2012 IEEE International Conference on Fuzzy Systems*. IEEE. 2012, pp. 1–8.

[Sho+17] Reza Shokri et al. “Membership inference attacks against machine learning models”. In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18.

[SMD06] Agusti Solanas, Antoni Martínez-Ballesté, and Josep Domingo-Ferrer. “Variable MDAV for multivariate microaggregation”. In: *Proceedings of ACM SIGKDD Workshop on Privacy, Security, and Trust in KDD* (2006), pp. 17–22.

[SMS14] Pedro Strecht, Jorge Mendes-Moreira, and Carlos Soares. “Combining Multiple Trees from Random Forests: A Possible Framework”. In: *Proceedings of the 2014 IEEE International Conference on Data Mining (ICDM)* (2014).

[SOT22] T. Stadler, B. Oprisanu, and C. Troncoso. “Synthetic data–anonymisation groundhog day”. In: *31st USENIX Security Symposium (USENIX Security 22)*. 2022, pp. 1451–1468.

[SS98] Pierangela Samarati and Latanya Sweeney. “Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression”. In: (1998).

[ST12] Klara Stokes and Vicenç Torra. “n-Confusion: a generalization of k-anonymity”. In: *Proceedings of the 2012 Joint EDBT/ICDT Workshops*. 2012, pp. 211–215.

[ST18] Navoda Senavirathne and Vicenc Torra. “Approximating robust linear regression with an integral privacy guarantee”. In: *2018 16th Annual Conference on Privacy, Security and Trust (PST)*. IEEE. 2018, pp. 1–10.

[ST19] Navoda Senavirathne and Vicenç Torra. “Integrally private model selection for decision trees”. In: *computers & security* 83 (2019), pp. 167–181.

[Ste09] Dan Steinberg. “CART: classification and regression trees”. In: *The top ten algorithms in data mining*. Chapman and Hall/CRC, 2009, pp. 193–216.

[Swe02] Latanya Sweeney. “Achieving k-anonymity privacy protection using generalization and suppression”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), pp. 571–588.

[TD04] Vicenç Torra and Josep Domingo-Ferrer. “Disclosure risk measures: Discrete and continuous masking methods”. In: *Data Mining and Knowledge Discovery* 8.3 (2004), pp. 301–318.

[TKM15] Matthias Templ, Alexander Kowarik, and Bernhard Meindl. “Statistical disclosure control for micro-data using the R package sdcMicro”. In: *Journal of Statistical Software* 67 (2015), pp. 1–36.

[TN16] Vicenç Torra and Guillermo Navarro-Arribas. “Integral privacy”. In: *Cryptology and Network Security: 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings 15*. Springer. 2016, pp. 661–669.

[TNG20] Vicenç Torra, Guillermo Navarro-Arribas, and Edgar Galván. “Explaining recurrent machine learning models: integral privacy revisited”. In: *International Conference on Privacy in Statistical Databases*. Springer. 2020, pp. 62–73.

[Tor22] Vicenç Torra. *A Guide to Data Privacy*. Springer, 2022.

[Tor23] Vicenç Torra. "A systematic construction of non-iid data sets from a single data set: non-identically distributed data". In: *Knowledge and Information Systems* 65.3 (2023), pp. 991–1003.

[VT23] Ayush K Varshney and Vicenç Torra. "Integrally private model selection for deep neural networks". In: *International Conference on Database and Expert Systems Applications*. Springer. 2023, pp. 408–422.

[VV17] Paul Voigt and Axel Von dem Bussche. "The eu general data protection regulation (gdpr)". In: *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10.3152676 (2017), pp. 10–5555.

[WG24] Zijun Wang and Keke Gai. "Decision Tree-Based Federated Learning: A Survey". In: *Blockchains* 2.1 (2024), pp. 40–60.

[WL19] Gil Weinberg and Lior Las. "A Comparison of Decision Tree Selection Methods". In: *International Journal of Data Science and Analytics* 7 (2019), pp. 123–135. DOI: 10.1007/s41060-019-00125-3.

[Xu+19a] Feiyu Xu et al. "Explainable AI: A brief survey on history, research areas, approaches and challenges". In: *Natural language processing and Chinese computing: 8th cCF international conference, NLPCC 2019, dunhuang, China, proceedings, part II 8*. Springer. 2019, pp. 563–574.

[Xu+19b] Lei Xu et al. "Modeling tabular data using conditional gan". In: *Advances in neural information processing systems* 32 (2019).

[Yal+19] Andrew Yale et al. "Assessing privacy and quality of synthetic health data". In: *Proceedings of the Conference on Artificial Intelligence for Data Discovery and Reuse*. 2019, pp. 1–4.

[Yi+14] Xun Yi et al. *Homomorphic encryption*. Springer, 2014.

[Zar+] O Zari et al. *Membership inference attack against principal component analysis (2022)*.

[Zha+20] Yulong Zhang et al. "Secret revealer: Generative model-inversion attacks against deep neural networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 253–261.

[ZJW18] Xinyang Zhang, Shouling Ji, and Ting Wang. "Differentially private releasing via deep generative model (technical report)". In: *arXiv preprint arXiv:1801.01594* (2018).

[ZLH19] Ligeng Zhu, Zhijian Liu, and Song Han. "Deep leakage from gradients". In: *Advances in neural information processing systems* 32 (2019).

[Zol86] Vladimir M Zolotarev. *One-dimensional stable distributions*. Vol. 65. 1986.