Title
Estimating Privacy Leakage of Machine Learning Models

Author
O'Dell, Ryan

Publication Date
2023

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Estimating Privacy Leakage of Machine Learning Models

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Statistics

by

Ryan O'Dell

2023

ABSTRACT OF THE THESIS

Estimating Privacy Leakage
of Machine Learning Models

by

Ryan O'Dell
Master of Science in Statistics
University of California, Los Angeles, 2023
Professor Guang Cheng, Chair

A membership inference attack is a method of extracting the training data from machine learning models. Previous analysis has characterized the worst case vulnerability to membership inference by instantiating the attack algorithm as the Bayes Optimal Classifier. We extend these findings by developing practical estimators for the worst case vulnerability on a sub-class of membership inference problems that are easy to compute without resorting to computationally expensive privacy auditing techniques. Extensive simulation studies are conducted on real world data sets to show that privacy auditing techniques, such as shadow modeling, can be replaced with the proposed worst case estimators. Furthermore, we examine the notion of disparity in membership inference: that some subgroups of the population are easier to identify in the training data set than others. We use a framework to quantify the degree of disparity and demonstrate that several real world models exhibit disparity in membership inference. We advocate that average metrics of attack accuracy, commonly used in the privacy auditing literature, do not reliably convey the difference in privacy risks across different levels of the population.

The thesis of Ryan O'Dell is approved.

Guido Francisco Montúfar Cuartas

Mark S. Handcock

Guang Cheng, Committee Chair

University of California, Los Angeles

2023

To my family...

and all those who supported me

during my academic journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to extend my gratitude to all those who have contributed to the successful completion of this master's thesis. First and foremost, I would like to express my deepest appreciation to my adviser, Professor Guang Cheng, and the members of my committee for their guidance throughout the duration of my thesis research. Their expertise and insightful feedback have been instrumental in shaping the direction and quality of this work.

I am also deeply thankful to Dr. Chi-Hua Wang, whose mentorship, guidance, and meaningful conversations have played a pivotal role in the development of this paper. Dr. Wang was critical in helping me to develop the direction and core ideas explored in my research. The wealth of knowledge and expertise shared by Dr. Wang have greatly enriched my understanding of the subject matter and have undoubtedly contributed to the overall success of this research.

Furthermore, I would like to express my sincere thanks to my colleagues in the Trustworthy AI Lab at UCLA, with whom I have engaged in valuable discussions and exchanges of ideas. Their input and perspectives have been invaluable in shaping my thinking and refining the outcomes of this study. I consider myself fortunate to have been surrounded by such dedicated and talented individuals in an academic environment.

Finally, I would like to acknowledge the broader academic community and the countless scholars whose work and publications have paved the way for this research. Their contributions have laid a strong foundation upon which this thesis is built.

To all those mentioned above and to anyone else who has contributed in any way, I am truly grateful for your support. Your involvement has been instrumental in bringing this research to fruition, and I am deeply indebted to each and every one of you.

CHAPTER 1

Introduction

Machine learning is a widely popular method for statistical analysis that involves fitting a predictive model on a training data set. Most common machine learning models, such as deep neural networks, train the on the same data points multiple times, which makes these models predict these points well. However, it has been observed empirically that the machine learning models can leak information about the training data constituting privacy risks of the individual records in the training data set [HSS22]. Membership inference attacks are one type of algorithm where an adversary attempts to determine if a specific record was part of the training data [SSS16]. Furthermore, it has been shown that privacy violations are not uniform across population subgroups: individuals or groups can be more easily identified, this is known as disparity [YKT19], [HSS22]. Thus demonstrating the need for formal privacy guarantees for machine learning models.

The most popular privacy preserving framework is Differential privacy which accomplishes privacy by injecting controlled noise into a model's training procedure [Dwo08]. The noise ensures that any individual record has limited influence on the output of the model. However, the additional noise comes at a cost of reduced model utility when compared to an non-privatized model.

In the remainder of the thesis, I will examine the analytic trade-off between differential privacy and the success of membership inference, by instantiating membership inference as the Bayes Optimal classifier which corresponds to the worst case privacy violations. Furthermore, I will explore the relationship between membership inference and the phenomena

of disparity in membership inference. Lastly, I will conduct extensive simulation studies examining under what scenarios we can estimate worst case privacy risks.

CHAPTER 2

Machine Learning

The main goal of a Machine Learning is to a predictive model given samples from a distribution on $\mathcal{X} \times \mathcal{Y}$. $\mathcal{X}$ is the space of predictors or features and $\mathcal{Y}$ is the space of outcome variables we wish to predict. We want to learn some function of the form:

$$f(x; \theta) : \mathcal{X} \to \mathcal{Y}$$

that is optimal given only a finite number of data points to train on. We refer to θ as the parameters of the function, these could be the weights of a neural network or the coefficients in a regression. We estimate θ by minimizing loss function ℓ given training data $D_{train} = \{(x_i, y_i)\}_{i=1}^n$:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{n} \ell(y_i, f(x_i; \theta))$$

The notion of minimizing a loss function can be formally motivated from the principle of maximum likelihood inference. Maximum likelihood inference is a statistical method used to estimate the parameters of a statistical model based on observed data. The underlying principle is to find the values of the model's parameters that maximize the likelihood function, which quantifies the probability of obtaining the observed data given the parameter values.

In maximum likelihood inference, we assume the observed data $\{(x_i, y_i)\}_{i=1}^n$ is generated from a probability distribution that depends on the parameter θ of interest. The goal is to find the values of these parameters that make the observed data most likely to have been generated by the assumed distribution. In machine learning, our main assumption is that the conditional distribution of $\mathbb{P}(y|x)$ is contained within some family of distributions indexed

by some parameters θ:

$$\ell(\theta) = \mathbb{P}(y|x;\theta)$$

We refer to $\ell(\theta)$ as the likelihood function. Given a realization of $\{(x_i, y_i)\}_{i=1}^n$ of the data we think of the likelihood as purely a function of θ. Given that the data are generated i.i.d. from $\mathbb{P}(y|x;\theta)$, the likelihood simplifies to:

$$\ell(\theta) = \mathbb{P}(y|x;\theta) = \prod_{i=1}^{n} \mathbb{P}(y_i|x_i;\theta)$$

Typically the product is difficult to optimize so we convert the product into a sum via the logarithm and taking its negative:

$$\mathcal{L}(\theta) = -\log \ell(\theta) = \sum_{i=1}^{n} \log \mathbb{P}(y_i|x_i;\theta)$$

In the remaining sections on machine learning, we will see how maximizing the likelihood can be equivalently formulated as a loss minimizing problem.

Maximum likelihood inference offers several desirable properties, including consistency, efficiency, and asymptotic normality under certain conditions [Kee10]. It is widely used in statistics and machine learning to estimate parameters and fit models based on observed data [SB14], [Bis16].

2.1 Regression

In regression analysis, the outcome or dependent variable of interest is continuous valued: $y_i \in \mathbb{R}$. The training data consists of observations $\{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^p$. We make the assumption that our family of distributions $\mathbb{P}(y|x;\theta)$ is normally distributed:

$$\mathbb{P}(y|x;\theta) \sim N(f(x;\theta), \sigma^2)$$

Typically σ^2 is a nuisance parameter or parameter we have prior knowledge of. $f(x;\theta)$ encodes the explicit relationship between x and y. A common example is that $f(x;\theta)$ is a

linear function, e.g.

$$f(x;\theta) = x^T\theta$$

Which recovers standard linear regression. Now, the likelihood becomes:

$$\begin{aligned}\ell(\theta) = \mathbb{P}(y|x;\theta) &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-1}{2\sigma^2}\left[y_i - f(x_i;\theta)\right]^2\right) \\ &= \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(\frac{-1}{2\sigma^2}\sum_{i=1}^{n}\left[y_i - f(x_i;\theta)\right]^2\right)\end{aligned}$$

Taking the logarithm, which preserves the optimum,

$$\begin{aligned}\mathcal{L}(\theta) &= -\log \ell(\theta) \\ &= -\log\left[\frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(\frac{-1}{2\sigma^2}\sum_{i=1}^{n}\left[y_i - f(x_i;\theta)\right]^2\right)\right] \\ &= \frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[y_i - f(x_i;\theta)\right]^2 + \frac{n}{2}\log\left(2\pi\sigma^2\right)\end{aligned}$$

Disregarding the constant σ^2 we arrive at:

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{i=1}^{n}\left[y_i - f(x_i;\theta)\right]^2$$

Which is the squared error loss. Now, we have shown that:

$$\max_{\theta} \ell(\theta) \iff \min_{\theta} \mathcal{L}(\theta)$$

Thus we see that maximum likelihood estimation can equivalently be formulated as a minimizing the squared error loss for regression.

2.2 Classification

In classification, the outcome or dependent variable of interest is discrete: $y_i \in \{0, 1\}$. For simplicity, we will only cover the case of binary classification, but there are extensions to discrete sets with more than 2 values. Again, the training data consists of observations

$\{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^p$. Our main assumption is that the outcome y_i is Bernoulli conditional on a transformation of the inputs x_i:

$$\mathbb{P}(y|x;\theta) \sim \text{Bernoulli}(p)$$

Where, p is generated by taking the sigmoid function σ of f.

$$p = \sigma(f(x_i;\theta)) = \frac{\exp\left(f(x_i;\theta)\right)}{1+\exp\left(f(x_i;\theta)\right)}$$

Now we can use the maximum likelihood framework to to fit the model parameters from the data. First we note the pdf of a Bernoulli(p_i) random variable is given by:

$$p(y_i|p_i) = p_i^{y_i}(1-p_i)^{(1-y_i)}$$

By independence we have the likelihood function is given by:

$$\ell(\theta) = \prod_{i=1}^n p(y_i|x_i;\theta) = \prod_{i=1}^n p_i^{y_i}(1-p_i)^{(1-y_i)}$$

We take the negative log of the likelihood since it preserves the location of the optimum:

$$\mathcal{L}(\theta) = -\sum_{i=1}^n y_i \log(p_i) + (1-y_i)\log(1-p_i)$$

Which is commonly know as the cross entropy loss function. Thus to fit a classification model we find the θ that minimizes the cross entropy loss. Hence,

$$\max_\theta \ell(\theta) \iff \min_\theta \mathcal{L}(\theta)$$

In the next section we present a simple algorithm for minimizing loss functions.

2.3 Stochastic Gradient Descent

Now we present a simple algorithm, Stochastic Gradient Descent (SGD), for fitting a model when the loss function is differentialiable with respect to its parameters θ. SGD is a particularly useful when the data set is large, as it avoids computing the gradient of ℓ over the entire data set, but instead repeatedly computes the gradient on a much smaller sub sample of $\{(x_i, y_i)\}_{i=1}^n$ to update θ. A comprehensive analysis of SGD can be read in [SB14].

Algorithm 1 Stochastic Gradient Descent

procedure SGD($D_{\text{train}}, \theta_0, \nabla_\theta \ell, \alpha, T, m$)
  for $j = 1, \cdots, T$ **do** ▷ Number of iterations
    $\{(x_i, y_i)\}_{i=1}^m \sim D_{\text{train}}$ ▷ Sample Mini Batch of size m
    $\theta_j = \theta_{j-1} - \alpha \frac{1}{m} \sum_{i=1}^m \nabla_\theta \ell(y_i, f(x_i, \theta_j))$ ▷ Update θ
  end for
return θ_T ▷ Return last θ
end procedure

SGD requires an initial θ_0 to begin the algorithm, a learning rate α that governs how quickly θ is updated, the derivative of loss function $\nabla_\theta \ell$, a number of iterations, and size of mini batches. We essentially only need to know how to compute $\nabla_\theta \ell$ to perform SGD.

For demonstrating the derivation of the gradient $\nabla_\theta \ell$, we can assume that $f(x; \theta) = x^T \theta$ and derive the gradients required for both regression and classification. Now for regression,

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta) &= \nabla_\theta \frac{1}{2} \sum_{i=1}^n \left[y_i - \theta^T x_i \right]^2 \\
&= - \sum_{i=1}^n \left[y_i - \theta^T x_i \right] \nabla_\theta \left[y_i - \theta^T x_i \right] \\
&= \sum_{i=1}^n \left[y_i - \theta^T x_i \right] x_i
\end{aligned}$$

Thus for sample of size m from $D_{\text{train}} = \{(x_i, y_i)\}_{i=1}^n$ the update step is given by:

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \sum_{i=1}^m \left[y_i - \theta^{(t)T} x_i \right] x_i$$

In the case that $f(x; \theta)$ is linear, there is a unique solution for θ provided $(X^T X)^{-1}$ exists, where $X \in \mathbb{R}^{n \times p}$ is the matrix of observed features x. For more complicated models where $f(x; \theta)$ is nonlinear, iterative methods like SGD provides a way to learn θ.

Now, for the case of classification, even when $f(x; \theta)$ is linear there is no closed form solution of the equation:

$$\nabla_\theta \mathcal{L}(\theta) = 0$$

So to perform SGD, we just need to derive $\nabla_\theta \mathcal{L}(\theta)$, we will similarly make the assumption that $f(x;\theta)$ is linear. Noting that, the derivative of the sigmoid function is given by

$$\sigma'(b) = (1 - \sigma(b))\sigma(b)$$

We can derive the gradient of the cross entropy loss

$$\begin{aligned}
\nabla_\theta - \mathcal{L}(\theta) &= \nabla_\theta - \sum_{i=1}^{n} y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \\
&= -\sum_{i=1}^{n} y_i \nabla_\theta \log(p_i) + (1 - y_i) \nabla_\theta \log(1 - p_i) \\
&= -\sum_{i=1}^{n} y_i \frac{1}{p_i} \nabla_\theta p_i + (1 - y_i) \frac{1}{1 - p_i} \nabla_\theta (1 - p_i) \\
&= -\sum_{i=1}^{n} y_i \frac{1}{p_i} p_i(1 - p_i) x_i - (1 - y_i) \frac{1}{1 - p_i} (1 - p_i) p_i x_i \\
&= -\sum_{i=1}^{n} y_i (1 - p_i) x_i - (1 - y_i) p_i x_i \\
&= -\sum_{i=1}^{n} (y_i - p_i) x_i \\
&= \sum_{i=1}^{n} (p_i - y_i) x_i
\end{aligned}$$

Thus for sample of size m from $D_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{n}$ the update rule for SGD of classification is given by:

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \sum_{i=1}^{m} (p_i - y_i) x_i$$

CHAPTER 3

Differential Privacy

Differential privacy is a framework that addresses privacy concerns when analyzing and sharing sensitive data. It provides a mathematical guarantee of privacy protection by ensuring that the presence or absence of an individual's data does not significantly affect the results of a query or computation.

The primary goal of differential privacy is to strike a balance between the need to extract useful information from data and the obligation to protect the privacy of individuals whose data is being analyzed. It offers a rigorous and quantifiable notion of privacy, providing a privacy budget that limits the amount of information that can be leaked about any individual.

The core idea of differential privacy lies in the introduction of randomness or noise into the computation or query responses. By injecting controlled amounts of noise, the output becomes indistinguishable whether an individual's data is included or excluded, thereby preventing any inference about an individual's presence in the dataset.

To achieve differential privacy, various techniques can be employed, such as adding noise to query responses, modifying the data before analysis, or applying privacy-preserving algorithms. These techniques ensure that the statistical properties of the data are preserved while minimizing the risk of re-identification or disclosure of sensitive information.

Differential privacy has gained significant attention in recent years due to the increasing concerns surrounding privacy and data protection. It has become a crucial tool in domains such as health care, finance, social sciences, and machine learning, where data analysis is necessary but privacy is paramount.

By adopting differential privacy mechanisms, organizations and researchers can responsibly leverage sensitive data for analysis, research, and decision-making while upholding the privacy rights and protecting the confidentiality of individuals. It provides a robust framework that offers provable guarantees of privacy, promoting trust and accountability in data-driven applications.

3.1 ϵ-Differential Privacy

Definition 1. *(Neighboring Data Sets) We say that two data sets X_1 and X_2 are neighboring if they only differ in precisely one element. We denote two neighboring data sets as:*

$$X_1 \approx X_2$$

The notion of neighboring data sets is the fundamental building block of the statistical notion of privacy. Intuitively we want to know that the output of some query response is indistinguishable whether a specific record of was included or excluded.

Definition 2. *(ϵ-Differential Privacy) We say that a randomized algorithm $A : \mathcal{X} \to \mathcal{Y}$ is ϵ-Differential Private if $\forall \;\; X_1 \approx X_2$ and $\forall T \in range(A)$*

$$\mathbb{P}(A(X_1) \in T) \leq \exp(\epsilon)\mathbb{P}(A(X_2) \in T)$$

Where the probability is taken over the randomness of A. We abbreviate ϵ-Differential Privacy as ϵ-DP.

Some typical examples of randomized algorithms A are very simple methods such as reporting the group average of a data set. More complicated example is when A is a machine learning models where the outputs of A are the functions of its predicted values. When A is trained by SGD the randomness of A typically comes from two places: the selection of the mini batches in stochastic gradient descent and the random initialization of the parameters before training.

For an intuitive understanding of ϵ-DP, we can examine the dynamics of privacy based on the limits of ϵ. Now $\epsilon \to 0$, we achieve total privacy:

$$\mathbb{P}(A(X_1) \in T) = \mathbb{P}(A(X_2) \in T)$$

Meaning that the output of our algorithm A is independent of all data sets $X_1 \approx X_2$. This ensures that the output of the A is not dependent on any individual records in X but is influenced by the collective data X, meaning that the individual records in X are protected.

Differential privacy can be characterized by a special random variable called the privacy loss random variable.

Definition 3. *(Privacy Loss Random Variable) Let Y and Z be two random variables. Draw $t \sim Y$ and the privacy loss random variable, denoted $\mathcal{L}_{Y\|Z}$, is defined as:*

$$\mathcal{L}_{Y\|Z} = \log\left[\frac{\mathbb{P}(Y=t)}{\mathbb{P}(Z=t)}\right]$$

The privacy loss random variable is only defined when the support of Y and Z are equal.

Clearly, if $Y \sim A(X_1)$ and $Z \sim A(X_2)$ then we have:

$$A \text{ is } \epsilon\text{-DP} \iff \mathcal{L}_{A(X_1)\|A(X_2)} \leq \epsilon$$

Which gives a very intuitive characterization that if the privacy loss random variable is bounded above by ϵ, then we have achieved differential privacy for the algorithm A.

A natural concern at this point would be given a randomized algorithm A that may not be private how do we achieve ϵ-Differential Privacy? There are several potential ways of ensuring differential privacy, but the most common methods are the Laplace and Gaussian Mechanisms. Which privacy mechanism we choose to use primarily depends on the structure of the the algorithm A.

Definition 4. *(Laplace Random Variable). The Probability distribution function of the Laplace Random Variable with location parameter $\mu \in \mathbb{R}$ and scale parameter $b \in \mathbb{R}^+$ and is*

given by:

$$f(x;\mu,b) = \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right)$$

The centered Laplace distribution, with location $\mu = 0$, which we will denote by as $Lap(b)$ with scale parameter b.

To achieve an ϵ level of Differential privacy we will need to know how much our algorithm A changes between two neighboring records. To that end we are particularly interested in the worst case of how much A changes to know how much noise to inject into the release of A.

Definition 5. *(ℓ_1 Sensitivity). Let $f : X \to \mathbb{R}^d$, then the sensitivity of f is defined as:*

$$\Delta_1(f) = \sup_{X_1 \approx X_2} \|f(X_1) - f(X_2)\|_{\ell_1}$$

Where the supremum is taken over all data sets differing in one element.

The Laplace Mechanism achieves differential privacy by injecting Laplace noise into the output of f that is governed by the sensitivity of f.

Definition 6. *(Laplace Mechanism). $f : X \to \mathbb{R}^d$. Then we define the Laplace Mechanism to be:*

$$T_f(X) = f(X) + Y$$

Where $Y \in \mathbb{R}^d$ and $Y_i \sim Lap\left(\frac{\Delta_1(f)}{\epsilon}\right)$.

Theorem 1. *[Dwo08] The Laplace Mechanism is ϵ-DP.*

Proof. Assume $X \approx Y$. Denote the p.d.f of $T_f(X)$ evaluated at z as $p_1(z)$ and the p.d.f. of

$T_f(Y)$ evaluated at z as $p_2(z)$. Then,

$$\begin{aligned}
\frac{p_1(z)}{p_2(z)} &= \prod_i^n \frac{\exp\left(-\frac{\epsilon|f(x)_i - z_i|}{\Delta_1(f)}\right)}{\exp\left(-\frac{\epsilon|f(y)_i - z_i|}{\Delta_1(f)}\right)} \\
&= \prod_{i=1}^n \exp\left(-\frac{\epsilon\left(|f(x)_i - z_i| - |f(y)_i - z_i|\right)}{\Delta_1(f)}\right) \\
&\leq \prod_{i=1}^n \exp\left(-\frac{\epsilon\left(|f(x)_i - f(y)_i|\right)}{\Delta_1(f)}\right) \\
&= \exp\left(-\frac{\epsilon}{\Delta_1(f)} \sum_{i=1}^n |f(x)_i - f(y)_i|\right) \\
&\leq \exp\left(-\frac{\epsilon}{\Delta_1(f)} \|f(X) - f(Y)\|_{\ell_1}\right) \\
&\leq \exp(\epsilon)
\end{aligned}$$

Hence,

$$p_1(z) \leq \exp(\epsilon) p_2(z)$$

□

3.2 (ϵ, δ)-Differential Privacy

The notion of ϵ-DP can be generalized. ϵ-DP is strict in the sense that it requires every release of A to satisfy the constraints of ϵ-DP. This notation can be extended to include the probability of failing to preserve ϵ-DP constraints.

Definition 7. *(ϵ, δ)-Differential Privacy. We say that a randomized algorithm $A : \mathcal{X} \to \mathcal{Y}$ is (ϵ, δ)-Differential Private if for all data sets X_1, X_2 such that $X_1 \approx X_2$ and $\forall T \in range(A)$*

$$\mathbb{P}(A(X_1) \in T) \leq \exp(\epsilon)\mathbb{P}(A(X_2) \in T) + \delta$$

We abbreviate (ϵ, δ)-Differential Privacy as (ϵ, δ)-DP.

The δ parameter roughly corresponds to the probability that ϵ-DP condition will fail. This can be seen most clearly, when analyzing the privacy loss random variable.

Theorem 2. *[Dwo08] (ϵ, δ)-DP is equivalent to that the absolute value of the privacy loss random variable being bounded by ϵ with probability $1 - \delta$.*

$$A \text{ is } (\epsilon, \delta)\text{-DP} \iff \mathbb{P}(|\mathcal{L}_{A(X_1)\|A(X_2)}| \leq \epsilon) \leq 1 - \delta$$

Thus we see that δ governs the probability that the absolute value of privacy loss random variable is not bounded by ϵ, corresponding to a failure of the ϵ-DP conditions. This means that the ϵ-DP is more strict in that the privacy loss random variable is always bounded.

To make a non-private random algorithm A (ϵ, δ)-DP, we need a modified notion of the sensitivity of A.

Definition 8. *(ℓ_2 Sensitivity). Let $f : X \to \mathbb{R}^d$, then the sensitivity of f is defined as:*

$$\Delta_2(f) = \sup_{X_1 \approx X_2} \|f(X_1) - f(X_2)\|_{\ell_2}$$

Where the supremum is taken over all data sets differing in one element. Algorithms that are (ϵ, δ)-DP naturally involve the ℓ_2 norm. The Gaussian Mechanism can enforce (ϵ, δ)-DP guarantees.

Definition 9. *(Gaussian Mechanism). Let $f : X \to \mathbb{R}^d$. Then we define the Gaussian Mechanism to be:*

$$T_f(X) = f(X) + Y$$

Where $Y \in \mathbb{R}^d$ and $Y_i \sim \mathcal{N}(0, \sigma^2)$.

Theorem 3. *[Dwo08] Let $\epsilon \in (0, 1)$. Then for $c^2 > 2 \ln 1.25/\delta$, the Gaussian Mechanism with parameter $\sigma^2 \geq c\Delta_2(f)/\epsilon$ is (ϵ, δ)-DP.*

The Gaussian Mechanism is particularly important for machine learning because it can be applied during training of a machine learning model when trained by SGD. To enforce (ϵ, δ)-DP guarantees can only need a slight modification of SGD. We simply need to clip the gradients to a prescribed size C and apply the Gaussian Mechanism to the gradient with an appropriate variance parameter determined by C. DP-SGD was first analysed by [ACG16] and an outline of the method is below in Algorithm (2):

Algorithm 2 DP-SGD

procedure DP-SGD(D_{train} the training data set, $\ell(\theta)$ the loss function, α the learning rate, σ variance parameter, L sampling proportion, C gradient size.)

  Initialize θ_0

  for $t = 1, \cdots, T$ **do**

    Take random sample Ł$_t \sim D_{train}$ with sampling probability $\frac{L}{N}$

    For each $i \in L_t$ compute $g_t = \nabla_\theta \ell(\theta_t)$

    Clip Gradient: $\bar{g}_t(x_i, y_i) = g_t(x_i, y_i) / \max(1, \frac{\|g_t\|_{\ell_2}}{C})$

    Add Gaussian Noise: $\tilde{g}_t = \frac{1}{L}(\sum_i \bar{g}_t(x_i, y_i) + N(0, \sigma^2 CI))$

    Gradient Descent Update: $\theta_{t+1} = \theta_t - \alpha \tilde{g}_t$

  end for

return θ_T

end procedure

CHAPTER 4

Membership Inference

Membership inference attacks (MIA) in machine learning refer to the exploitation of machine learning models to determine whether a specific data point was part of the model's training dataset. These attacks aim to reveal information about the presence or absence of individual data points in the training set, which can lead to privacy breaches and potential misuse of sensitive information. The first successful membership inference attacks were reported in [SSS16].

The idea behind membership inference attacks stems from the vulnerability of machine learning models to overfitting. During training, models learn to generalize from a limited set of labeled data examples, and they often memorize some details of the training set. This memorization can inadvertently reveal information about the presence of specific data points, allowing an adversary to infer membership.

Adversaries exploit this vulnerability by leveraging the model's response or behavior. By observing the predicted values of the model on various records, the attacker aims to distinguish between records that were part of the training set and those that were not. This can be achieved by analyzing statistical properties, confidence scores, or other characteristics of the model's predictions.

Membership inference attacks can be classified into two main categories:

- White-box attacks: In these attacks, the adversary has complete knowledge of the model's architecture, parameters, and access to the data distribution on $\mathcal{X} \times \mathcal{Y}$. They can access the model's internals, such as intermediate activations or gradients, to gain

insights into the membership status of specific data points.

- Black-box attacks: In these attacks, the adversary only has partial knowledge of access to the model. In this scenario an adversary is allowed to query the model by sending features x and receive the predicted values $\hat{y}$ from the model. Additionally, this may assume the adversary has knowledge of the data distribution on $\mathcal{X} \times \mathcal{Y}$ or that the adversary has access to a reference sample from the data distribution.

Membership inference attacks have important implications for privacy, especially in scenarios where the data being modeled is sensitive, such as medical records or personal information. These attacks highlight the need for privacy-preserving techniques like differential privacy to mitigate the risk of membership inference and protect the privacy of individuals whose data is used for training machine learning models.

4.1 Formalizing Membership Inference

We will follow the conventional formulation of membership inference as an indistinguishability game [YFJ17]. We will need some notation to introduce the main concepts. We denote $\mathcal{A}$ as a membership inference attack algorithm, A the victim model training procedure, e.g. a neural network trained by Stochastic Gradient Descent,n the training sample size, $\mathcal{D}$ the data distribution, a distribution on $\mathcal{X} \times \mathcal{Y}$, and $m \in \{0, 1\}$ membership label, whether a record x was used to train A. The goal of a membership inference attack is to produce a prediction $\hat{m}$ about the membership label m for a set of records (x, y). The game of membership inference, or membership inference experiment, is described diagrammatically in Algorithm (3).

We say that the membership inference experiment is successful if the predicted membership label $\hat{m}$ is correct:

$$MIA(\mathcal{A}, A, n, D) = 1 \text{ if } \hat{m} = m$$

Algorithm 3 MIA $(\mathcal{A}, A, n, \mathcal{D})$ Experiment

$S \leftarrow \mathcal{D}^n$; $A_S = A(S)$

$m \leftarrow \{0, 1\}$ sample uniformly.

if $m = 1$ **then**

  $x \leftarrow S$

else if $m = 0$ **then**

  $x \leftarrow \mathcal{D}$

$\hat{m} = \mathcal{A}(x, A_S, n, \mathcal{D})$

return $\hat{m} = m$

In the membership inference experiment a challenger samples n records from $\mathcal{D}$ and trains the model using A on the sample S. The challenger then samples a secret bit m decide if the record is to be pulled from the training data S or sampled from $\mathcal{D}$. The adversary then uses the outputted model A_S and the query point to make a guess about the membership label m.

To assess the efficacy of membership inference we will need a metric, since the membership inference attack experiment is naturally interpreted as a classification task, we use the normalized advantage over random guessing.

Definition 10. *(Vulnerability) The vulnerability of a trained model A to a membership inference attack $\mathcal{A}$ is the normalized advantage over a random guess.*

$$V(A) = 2\mathbb{P}\left[MIA(\mathcal{A}, A, n, \mathcal{D}) = 1\right] - 1$$

Vulnerability will serve as the main theoretical and experimental benchmark for evaluating the success of membership inference attacks.

Additionally, if we assume the population of interest can be partitioned into t subgroups $z_1, z_2, \ldots, z_t$ then we can define vulnerability with respect to a subgroup. This will be useful for analyzing when some groups in the training data set are easier to identify than others.

Definition 11. *(Subgroup Vulnerability) The vulnerability of a trained model A with respect to a subgroup z to a membership inference attack $\mathcal{A}$ is the normalized advantage over a random guess restricted to $Z = z$.*

$$V_z(A) = 2\mathbb{P}\left[MIA(\mathcal{A}, A, n, \mathcal{D}) = 1 | Z = z\right] - 1$$

The accuracy of membership inference can be directly related to the ϵ parameter if the model satisfies the ϵ-DP conditions.

Theorem 4. *[YFJ17] Let A be an ϵ-DP learning algorithm and $\mathcal{A}$ be a membership adversary. Then we have:*

$$V(A) \leq \exp(\epsilon) - 1$$

This inequality further expands our understanding of differential privacy in the following sense. We know that if A is ϵ-DP with $\epsilon \to 0$, then A is perfectly private. In this case, the accuracy of membership inference is bounded above by 0, meaning that membership inference is essentially impossible. Additionally, if ϵ is large, then $V(A)$ is upper bounded by a large constant indicating that membership inference attacks can be very accurate.

4.2 Methods of Membership Inference

There are several variants of membership inference methods that depend on the underlying victim model. The two methods we will focus on in this thesis are metric based attacks and shadow modeling attacks. Metric based attacks apply a simple threshold to the output of a target model to predict whether a query point was used to train the target model. Shadow modeling attempts to learn a classifier that given the predictions of the target model will distinguish between the distribution of the predictions on the training and testing set [SSS16].

Definition 12. *(Adversarial Features) We refer to the random variable $\mathbb{W}$ generated by the mapping $w \leftarrow \phi_W(A_S, x)$ as the Adversarial features. Where ϕ_W is a function of the predicted values of A_S.*

In general, membership inference will proceed in 3 stages outlined below.

Algorithm 4 Membership Inference Strategy

Phase 1: Adversary will prepare an attack algorithm $\mathcal{A}(x, A_S, n, \mathcal{D})$. Which depends on the victim model A_S, sample size n, and data distribution $\mathcal{D}$.

Phase 2: Adversary extracts features from A_S: $w \leftarrow \phi_W(A_S, x)$.

Phase 3: Adversary applies $\mathcal{A}$ on extracted features w to produce membership guess $\hat{m}$.

The most common examples of adversarial features are when $\mathbb{W}$ is the predicted probabilities, the predicted class labels, the values of the loss function used to train the model A_S, or more complicated features such as the models gradients in some white-box attacks. For the remainder of the paper we will focus on black box attacks as they assume the adversary has the most limited set of knowledge.

4.2.1 Metric Based Attack

In a metric based attack, we use a decision rule by simply applying a threshold τ to some function of a target model's predictions. In practice, τ is either prescribed a priori by the adversary or can be estimated in a simulation setting. The most common metric based attack is when a threshold is applied to the loss function used to train the maching learning model.

Definition 13. *(Loss Based Attack) In the loss based attack a prescribed threshold τ is used determine the decision boundary where $\phi_W = \ell(y, A_S(x))$, thus the attack algorithm $\mathcal{A}$ is governed by:*

$$\mathbb{1}\left[\ell(y, A_S(x))\right) \leq \tau]$$

This idea seeks to exploit that machine learning models can achieve very small values of the training loss. For example, when trained by SGD, the same data points are used to train the model repeatedly and these records will have small values of the loss function. The

intuition behind why this attack can work is that the data points the model is trained on will have small values of the loss function while data on a testing data set will have higher values. Thus we use the loss function ℓ to determine which (x, y) that were used to train the model A.

4.2.2 Shadow Modeling Attack

Shadow modeling is a more complicated attack that attempts to mimic the behavior of the target model and train a classifier to distinguish between the predictions on the training and testing data sets. The shadow modeling procedure was first described in [SSS16]. The full procedure for shadow modeling is described in Algorithm (5).

In a shadow modeling attack an adversary repeatedly samples training and testing sets from the distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$, these are called shadow data sets. The adversary then trains a shadow model on the training set and produces predictions for both the training and test sets. A shadow model is a model that is as close to the victim model A_S as possible. The adversary appends a set of membership labels , m^{out}, m^{in} to the predictions on both the shadow training and testing sets. This step is repeated t times to produce a large "attacking" data set, produced by unioning all the shadow data sets with membership labels. Lastly, a classifier is trained to distinguish the assigned membership labels using only the predictions y_{pred}.

The shadow modeling technique is not free from assumptions and assumes that the adversary has some background knowledge of the victim model. In shadow modeling we are assuming that an adversary $\mathcal{A}$ has knowledge of the specifications of the model $A_{\mathcal{S}}$. For example, if A_S is a neural network, the adversary may know approximately the training set size n, the architecture of A_S, and the specifications of the optimization used during training.

Shadow modeling is computationally expensive since it requires fitting t models with the same specifications, or as close as possible, to the target model A_S. However, shadow mod-

Algorithm 5 Shadow Modeling

procedure SHADOW MODELING($\mathcal{A}, A, n, \mathcal{D}, t$)
  $\mathcal{S} \sim \mathcal{D}$ ▷ Sample training set size n
  Train: $A_{\mathcal{S}} = A(\mathcal{S})$ ▷ Train Victim Model on $\mathcal{S}$
  for $j = 1, \cdots, t$ **do** ▷ Build Shadow Data Sets
    $\mathcal{S}_j^{in} \sim \mathcal{D}$ ▷ Sample training set size n
    $\mathcal{S}_j^{out} \sim \mathcal{D}$ ▷ Sample testing set size n
    Train: $A_{\mathcal{S}_j^{in}} = A(\mathcal{S}_j^{in})$ ▷ Train Shadow Model j
    Predict: $\hat{y}_j^{in} = A_{\mathcal{S}_j^{in}}(\mathcal{S}_j^{in})$ ▷ Get Shadow Training predictions
    Predict: $\hat{y}_j^{out} = A_{\mathcal{S}_j^{in}}(\mathcal{S}_j^{out})$ ▷ Get Shadow Testing predictions
    Create: $m_j^{in} = 1$ ▷ Vector of $1's$ size n
    Create: $m_j^{out} = 0$ ▷ Vector of $0's$ size n
    Create Shadow Data Set j: $Shadow_j = (\hat{y}_j^{in}, m_j^{in}) \cup (\hat{y}_j^{out}, m_j^{out})$
  end for
  Create Attack Data Set: $\cup_{j=1}^{t} Shadow_j$
  Train Attack Classifier $\mathcal{A}$ with features $\hat{y}$ to predict mem label on the Attack Data.
  return $\mathcal{A}$ ▷ return Attack Model
end procedure

eling is the de-facto standard benchmark method in evaluating the accuracy of membership inference attacks.

CHAPTER 5

Characterizing Membership Inference

The next section examines a characterization of membership inference following the work developed by [YKT19]. There are a few key ideas, the first is to analyze membership inference as a classification problem, where the adversarial attack algorithm with features $\mathbb{W}$ is the Bayes optimal classifier, this will correspond to the theoretical worst case membership inference attack. An extended notion of model generalization is developed to characterize the vulnerability of a model with respect to the Bayes optimal attack.

5.1 Worst Case Vulnerability

Our main interest is to quantify the maximum vulnerability of a trained model A to any attack that uses a given set of adversarial features $\mathbb{W}$:

$$\max_{\mathcal{A}:\mathbb{W}\to\{0,1\}} V(\mathcal{A}\circ\phi_w)$$

This is particularly import because this will correspond to the theoretical worst case privacy infringement of a model A to membership inference attacks. Since vulnerability is a linear function of the accuracy of a membership inference attack, we know that minimizing the probability of misclassification will maximize vulnerability. It is well known that the Bayes Classifier minimizes the probability of misclassification [DLG14].

Definition 14. *(Bayes Classifier). The Bayes classifier is given by:*

$$C^{bayes}(x) = \arg\max_{y\in\{0,1\}} \mathbb{P}[Y = y|X = x]$$

Thus, defining the attack algorithm as the Bayes optimal classifier will maximize $V(A)$.

Definition 15. *(The Bayes Adversary) The Bayes Adversary, denoted $\mathcal{A}^*_{\mathbb{W}}$, with given adversarial features $\mathbb{W}$ is given by:*

$$\mathcal{A}^*_W = \arg\max_{m \in \{0,1\}} \mathbb{P}[M = m | W = w]$$

The Bayes optimal classifier trivially achieves the worst case vulnerability, since the Bayes optimal classifier minimizes the probability of misclassification.

We will consider two types of adversaries $\mathcal{A}$, sub-group aware adversaries: ($Z \in \mathbb{W}$), where the an indicator variable for the subgroup Z is included in the set of adversarial features. And regular adversaries: ($Z \notin \mathbb{W}$), where $\mathbb{W}$ is a function of the predicted values of A.

Theorem 5. *[YKT19] The worst-case vulnerability to a subgroup-aware (Bayes) adversary is equal or higher compared to a regular (Bayes) adversary.*

$$V(A^*_{\mathbb{W},Z}) \geq V(A^*_{\mathbb{W}})$$

The main take away is that in the worst cast scenario, the accuracy of membership inference is as good or better when we included sub-group indicators Z in our adversarial features $\mathbb{W}$. This has particularly important implications for privacy auditing, mainly that we should always include sensitive indicators Z in our feature set $\mathbb{W}$ to get more realistic estimates of the worst possible case. Indeed this is always possible, since we have access to the vector of features $\mathcal{X}$ in the data we can just pull the sensitive variables of interest Z from them.

5.2 Distributional Generalization

Several studies have empirically demonstrated there is a strong relationship between over fitting and the accuracy of membership inference. [YKT19] define a more general notion

of predictive model generalization that subsumes the standard definition of comparing the average loss on a training and testing set. This is used to characterize the the worst case vulnerability of A.

Definition 16. *(Property Function). A property function is any function that takes as input a model A_S and an example x and returns a numeric vector. We denote a property function as $\pi(A_S, x)$. For any set T in the range of π, we define the corresponding probability measures as generated by π:*

$$\mu_1^\pi(T) = \mathop{\mathbb{P}}_{\substack{S\sim\mathcal{D}^n \\ x\sim S}}[\pi(A_S, x) \in T] \quad \textit{training}$$

$$\mu_0^\pi(T) = \mathop{\mathbb{P}}_{\substack{S\sim\mathcal{D}^n \\ x\sim \mathcal{D}}}[\pi(A_S, x) \in T] \quad \textit{testing}$$

The key difference between μ_1^π and μ_0^π is that the randomness of μ_1^π is defined over the randomness of the training examples $x \sim S$ and μ_0^π is defined over the randomness of data from the data distribution $x \sim \mathcal{D}$. Some common examples of property functions are the predicted values of the model itself $\pi(A_S, x) = A_S(x)$. In this case, μ_1^π and μ_0^π are probability measures generated by the predicted values of the model A_S. We are particularly interested in the case that $\pi = \phi_W$ is the adversarial feature extraction map and how different these two measures are for a particular set of adversarial features $\mathbb{W}$.

Definition 17. *(Distributional Generalization Gap). The Distributional Generalization Gap associated to property function $\pi(A_S, x)$ is given by:*

$$R(\pi, d) = d(\mu_1^\pi, \mu_0^\pi)$$

Where $d(\cdot, \cdot)$ is any valid distance between probability measures.

This notion of generalization characterizes the worst case vulnerability.

Theorem 6. *[YKT19] The worst-case vulnerability to membership inference attack with a given set of adversarial features W is equal to the distributional generalization gap under the total-variation distance:*

$$V(A_W^*) = R(\phi_W, d_{TV})$$

Where

$$d_{TV}(\mu, \mu') = \sup_{T \subseteq W} |\mu(T) - \mu'(T)|$$

Proof. Let us denote the Bayes error L^*, the $0-1$ classification error of the Bayes classifier $\mathcal{A}^*_W$. Thus,

$$L^* = \mathbb{P}[\mathcal{A}^*_W(w) \neq m]$$

So,

$$\begin{aligned} V(\mathcal{A}^*_W) &= 2(1 - \mathbb{P}[\mathcal{A}^*_W(w) \neq m]) - 1 \\ &= 1 - 2L^* \end{aligned}$$

Using Le Cam's Method:

$$\begin{aligned} L^* &= \frac{1}{2} - \frac{1}{2} d_{TV}\left(\mathbb{P}[W|M=1], \mathbb{P}[W|M=0]\right) \\ &= \frac{1}{2} - \frac{1}{2} d_{TV}\left(\mathop{\mathbb{P}}_{\substack{S \sim \mathcal{D}^n \\ x \sim \mathcal{S}}}[\phi_W(A_S, x)], \mathop{\mathbb{P}}_{\substack{S \sim \mathcal{D}^n \\ x \sim \mathcal{D}}}[\phi_W(A_S, x)]\right) \\ &= \frac{1}{2} - \frac{1}{2} d_{TV}\left(\mu_1^{\phi_w}, \mu_0^{\phi_w}\right) \end{aligned}$$

Thus,

$$\begin{aligned} V(\mathcal{A}^*_W) &= 1 - 2L^* \\ &= 1 - 2(\frac{1}{2} - \frac{1}{2} d_{TV}\left(\mu_1^{\phi_w}, \mu_0^{\phi_w}\right) \\ &= d_{TV}\left(\mu_1^{\phi_w}, \mu_0^{\phi_w}\right) \end{aligned}$$

□

Corollary 1. *[YKT19] Let $\ell(A_S, x) = \mathbb{1}\left[A_S(x) \neq y(x)\right]$ be the $0-1$ loss, and the adversary's features be the loss values $W = \ell(A_S, x)$. Then, the standard generalization gap equals worst case vulnerability:*

$$V(A^*_{\ell(A_S,x)}) = |R(\ell, d_{MD})|$$

Where d_{MD} is the mean discrepancy distance:

$$d_{MD}(\mu, \mu') = \int \omega d\mu(\omega) - \int \omega d\mu'(\omega)$$

Proof.

$$\begin{aligned} R(\ell, d_{TV}) &= |\mathbb{P}[\ell(A_S, x) = 1 | M = 1] - \mathbb{P}[\ell(A_S, x) = 1 | M = 0]| \\ &= |\mathbb{E}[\ell(A_S, x) | M = 1] - \mathbb{E}[\ell(A_S, x) | M = 0]| \\ &= |R(\ell, d_{MD})| \end{aligned}$$

□

In the experimental section, we will empirically verify this will result which was not experimentally verified in [YKT19].

5.3 Estimating Vulnerability

Estimating vulnerability of a given model A_S is important for experimental verification of bounds that ϵ-DP provides, as well as giving practitioners a framework to estimate the inherit privacy risks when deploying models with highly sensitive features. Using simulation to estimate vulnerability to membership inference attacks is known as privacy auditing. The current field of privacy auditing is highly empirical, we note some of the contemporary simulation methods for producing estimates of vulnerability. First we can start with a formal notion of privacy auditing.

Definition 18. *(Privacy Auditing). Given a model A_S (private or non-private) and its set of adversarial features $\mathbb{W}$, we define privacy auditing to be the any simulation technique that produces an estimate of the model A_S vulnerability to membership inference.*

In practice, auditors will select one or a few techniques, such as shadow modeling, and perform membership inference against A_S in a simulation and provide an estimate of $V(A)$.

Additionally, the modeler does have some influence on the set of adversarial features $\mathbb{W}$. In particular, modelers can restrict the outputs of the model, thus restricting the possible forms of the adversarial feature extraction mechanism ϕ_W. The main cases commonly assessed in the privacy auditing literature are the following [HSS22]:

1. The adversarial features $\mathbb{W}$ are predicted values from the model A_S, i.e. $\phi_W(A_S, x) = A_S(x)$. In the case of classification, these are the predicted probabilities.

2. The adversarial features $\mathbb{W}$ are the loss function used during training of the model, i.e. $\phi_W(A_S, x) = \ell(y, A_S(x))$. In the case of classification, there are a few functions ℓ, but in our cases we will use the $0-1$ loss.

3. The two previous cases can be extended by including subgroup indicator variables Z in addition to the predicted values or loss function values to make the adversarial attack algorithm sub-group aware (W, Z).

The main idea of using simulation to estimate $V(A)$ is to essentially play the MIA game repeatedly.

Definition 19. *(Model Specific Estimate of Vulnerability). Given a model A_S, its set of adversarial features $\mathbb{W}$, and a membership inference algorithm $\mathcal{A}$. We sample r training data sets of size n: $\{S_i\}_{i=1,\ldots,r}$ and testing data sets of size n: $\{\bar{S}_i\}_{i=1,\ldots,r}$. We define the model-specific estimate of vulnerability to be the unbiased estimator:*

$$\hat{V}(\mathcal{A}) = \frac{1}{r}\sum_{i=1}^{r} v_i$$

Where we use an unbiased estimator of v_i :

$$v_i = \frac{2}{n}\sum_{j=1}^{n} \mathbb{1}\left[\mathcal{A}(S_i^{(j)}, A_{S_i}, n, \mathcal{D}) = 1\right] + \frac{2}{n}\sum_{j=1}^{n} \mathbb{1}\left[\mathcal{A}(\bar{S}_i^{(j)}, A_{S_i}, n, \mathcal{D}) = 0\right] - 1$$

Theorem 7. *The estimator v_i, in Definition (19), is an unbiased estimator of vulnerability $V(A)$.*

Proof. Note we can express the accuracy of a membership inference attack using a union as follows:

$$\mathbb{P}\left[\text{MIA}(\mathcal{A}, A, n, \mathcal{D}) = 1\right] = \mathbb{P}(\mathcal{A} = 1, m = 1) + \mathbb{P}(\mathcal{A} = 0, m = 0)$$

Now taking expectations:

$$\begin{aligned}
\mathbb{E}[v_i] &= \mathbb{E}\left[\frac{2}{n}\sum_{j=1}^{n} \mathbb{1}\left[\mathcal{A}(S_i^{(j)}, A_{S_i}, n, \mathcal{D}) = 1\right] + \frac{2}{n}\sum_{j=1}^{n} \mathbb{1}\left[\mathcal{A}(\bar{S}_i^{(j)}, A_{S_i}, n, \mathcal{D}) = 0\right] - 1\right] \\
&= \frac{2}{n}\sum_{j=1}^{n} \mathbb{E}\left[\mathbb{1}\left[\mathcal{A}(S_i^{(j)}, A_{S_i}, n, \mathcal{D}) = 1\right]\right] + \frac{2}{n}\sum_{j=1}^{n} \mathbb{E}\left[\mathbb{1}\left[\mathcal{A}(\bar{S}_i^{(j)}, A_{S_i}, n, \mathcal{D}) = 0\right]\right] - 1 \\
&= \frac{2}{n}\sum_{j=1}^{n} \mathbb{P}\left[\mathcal{A}(S_i^{(j)}, A_{S_i}, n, \mathcal{D}) = 1\right] + \frac{2}{n}\sum_{j=1}^{n} \mathbb{P}\left[\mathcal{A}(\bar{S}_i^{(j)}, A_{S_i}, n, \mathcal{D}) = 0\right] - 1 \\
&= \frac{2}{n}\sum_{j=1}^{n} \mathbb{P}\left[\mathcal{A} = 1, m = 1\right] + \frac{2}{n}\sum_{j=1}^{n} \mathbb{P}\left[\mathcal{A} = 0, m = 0\right] - 1 \\
&= 2\mathbb{P}\left[\mathcal{A} = 1, m = 1\right] + 2\mathbb{P}\left[\mathcal{A} = 0, m = 0\right] - 1 \\
&= 2\mathbb{P}\left[\text{MIA}(\mathcal{A}, A, n, \mathcal{D}) = 1\right] - 1 \\
&= V(A)
\end{aligned}$$

Hence v_i is unbiased for $V(A)$. □

There are several reasons why estimating vulnerability in this inherently model specific to the attack algorithm $\mathcal{A}$. In practice an adversary could use a different attack algorithm that could result in a different vulnerability. For practitioners, there are several things to consider when reporting a model specific estimate of vulnerability: (1) they are specific to one particular set of adversarial features, (2) they are specific to one particular attack algorithm, (3) they are specific to what knowledge the adversary has about the data distribution $\mathcal{D}$. In practice, we do not know what attack algorithm, features, or knowledge a potential adversary has. In that sense these estimates provide a limited window into the potential privacy leakage of a model, they are specific to our assumptions about what an adversary's knowledge may be.

5.4 Estimating Worst Case Vulnerability

The discussion in the previous section regarding model specific estimates of vulnerability illustrate an important short coming in the privacy auditing literature: that there is no universally accepted way to estimate vulnerability without making strong assumptions on the adversary's knowledge or method.

The characterization of worst case vulnerability in terms of the total variation distance suggest potential techniques for estimating strategies of the worst case vulnerability as a way to potentially address this shortcoming. In this section, we will attempt to close the gap by proposing estimators of the worst case vulnerability by leveraging their characterization in terms of the total variation distance.

For the experiments we will focus on the case when the adversarial features are discrete valued. There are two main cases when the set of adversarial features $\mathbb{W}$ is a discrete random variable. We will focus on the case that the model A_S is a classification model.

1. Let $\{\hat{y}_0, \hat{y}_1\}$ denote the predicted values for the classes $\{0, 1\}$ respectively for the model A_S. The model A_S only releases the predicted class labels, thus the feature extraction mechanism is given by:
$$\phi_W(A_S, x) := \max\{\hat{y}_0, \hat{y}_1\}$$

2. The model A_S only releases the training loss function, in the case of classification, $0-1$ loss, thus the feature extraction mechanism is given by:
$$\phi_W(A_S, x) := \mathbb{1}[y \neq A_S(x)]$$

Both of these scenarios have been explored in the literature as an attempt to reduce the set of information released by a model when compared to releasing the predicted probabilities. However, several studies have demonstrated that membership inference is still successful when the features are discrete [HSS22].

Moreover, the scenarios when the adversarial features are discrete can always be reduced to the case of the $0-1$ loss. Whenever, an adversary has access to the predicted class from A_S, as long as they have the entire vector of data points (x, y), they can apply the loss function $\mathbb{1}[y \neq A_S(x)]$ to that example. Thus these examples reduce to only one use case: when the adversarial features are the $0-1$ loss function.

Given the characterization of worst case disparity for the $0-1$ loss from Corollary (1), suggest potential estimation strategies. We are required to come up with an estimator for $|R(\ell, d_{MD})|$. We simply average the training loss and testing loss as our estimator as plug in estimator for $|R(\ell, d_{MD})|$. In the experimental section, we will verify Corollary (1) using real world data sets that was not examined in [YKT19].

CHAPTER 6

Disparity

6.1 Disparity

Disparity in machine learning refers to the existence of unfair or biased outcomes and treatment among different groups of individuals when utilizing machine learning algorithms for decision making. These disparities can arise from various sources, including biased training data, biased features, or biased decision-making processes within the algorithms themselves. We are particularly interested in the case of discriminatory outcomes: Machine learning algorithms can lead to discriminatory outcomes if they disproportionately favor or disadvantage certain groups. For instance, an algorithm used in the criminal justice system for predicting recidivism rates may have higher error rates or make more false positive predictions for certain racial or socioeconomic groups, leading to unfair treatment. This classical notion of disparity in outcomes can be extended to the notion of vulnerability to membership inference attacks.

Definition 20. *(Disparity). Assume that $\mathcal{A}$ is a membership inference attack algorithm. Then disparity in vulnerability, or disparity, $\Delta V_{z,z'}(\mathcal{A})$ between two subgroups z and z' is the difference in vulnerability of the subgroups z and z':*

$$\Delta V_{z,z'}(\mathcal{A}) = V_z(\mathcal{A}) - V_{z'}(\mathcal{A})$$

Detecting the presence of disparity is particularly important because metrics such as average vulnerability may disguise that there are some subgroups z in the data set that are easier to identify in a membership inference attack.

Theorem 8. *[YKT19] Assume that the model A satisfies $\epsilon-DP$, then the magnitude of worst case disparity between any subgroups z and z' is uniformly bounded for any adversary with features W:*

$$\left|\Delta V_{z,z'}(\mathcal{A}^*_{W,Z})\right| \leq \exp(\epsilon) - 1$$

Theorem(6) is particularly important because it extends the protection that differential privacy provides for vulnerability to the worst case of disparity. Thus in practice, by privatizing models, the risk of disparity can be reduced.

In the experimental section, we will demonstrate that models such as neural networks can exhibit disparity to membership inference attacks. This will demonstrate that for practitioners conducting privacy auditing to report estimates of disparity to get more detailed measures of privacy risks.

6.2 Estimating Disparity

In the next section we will follow a framework developed by [YKT19] to detect model specific estimates of vulnerability to membership inference attacks.

To get model specific estimates of disparity we can follow the same estimation strategy for model specific estimates of vulnerability. We sample r training data sets of size n : $\{S_i\}_{i=1,\ldots,r}$ and testing data sets of size n :$\{\bar{S}_i\}_{i=1,\ldots,r}$. We get model specific estimates of subgroup vulnerability:

$$\hat{V}_z(\mathcal{A}) = \frac{1}{r}\sum_{i=1}^{r} v_{i,z}$$

Where we use an unbiased estimator of subgroup vulnerability $v_{i,z}$:

$$v_{i,z} = \frac{2}{n_z}\sum_{j=1}^{n_z} \mathbb{1}\left[\mathcal{A}(S_i^{(j)}, A_{S_i}, n_z, \mathcal{D}) = 1\right] + \frac{2}{\bar{n}_z}\sum_{j=1}^{\bar{n}_z} \mathbb{1}\left[\mathcal{A}(\bar{S}_i^{(j)}, A_{S_i}, \bar{n}_z, \mathcal{D}) = 0\right] - 1$$

Where n_z is the size of subgroup z in S_i and $\bar{n}_z$ is the size of the subgroup z in $\bar{S}$. Essentially, for $V_z(A)$, we use the same estimator for vulnerability, but only use examples that belong to the subgroup of interest z when subgroup vulnerability $v_{i,z}$.

Assuming we have t subgroups $z_1, \ldots, z_t$. We are interested in the following problem: given estimates $\{v_{i,z}\}_{i=1\ldots r}$ across different subgroups, we want to find evidence that the actual subgroup vulnerabilities differ:

$$V_{z_1} \neq V_{z_2} \neq \ldots \neq V_{z_t}$$

We have multiple measurements, the model-specific vulnerability estimates, for different subgroups $v_{i,z_1}, v_{i,z_2}, \ldots, v_{i,z_t}$ for the same victim model A_{S_i}. The standard approach is to use a repeated measure one way ANOVA. We can follow up the F test with pairwise dependent t-tests with correction for multiple comparisons, e.g. Bonferroni Method. In the experimental section, we will use this framework to demonstrate real world evidence of disparity.

CHAPTER 7

Experiments

In this chapter we will analyze our proposed estimators for privacy leakage and model specific disparity by conducting membership inference attacks on popular privacy benchmark data sets. These will help determine to what extent that empirical privacy auditing techniques, like shadow modeling, can be substituted with worst case estimators. Additionally, model specific estimates of disparity will serve to demonstrate that average privacy violation metrics may tell an incomplete story: that there are subgroups or individuals in data sets that are easier for adversaries to identify.

7.1 Law School Data

One case study will be based on the Law School Admissions Data Set. The data consists of a survey among law students attending school in the U.S. in 1991 [San04], [Scu23]. The primary purpose of the survey was to analyze what factors influenced whether students were passing the bar examination. The total number of students in the survey was $20,800$. A summary of the variables of interest is below in Table 7.1.

Variable	Description
age	the student's age in years
decile1	the student's decile in the school given their grades in Year 1
decile3	the student's decile in the school given their grades in Year 3
fam inc	the student's family income bracket (from 1 to 5)
lsat	the student's LSAT score
ugpa	the student's undergraduate GPA
gender	the student's gender
race1	a category specifying: Asian, Black, Hispanic, Other, or White
cluster	a category encoding the tiers of law school prestige
fulltime	whether the student will work full-time or part-time
bar	whether the student passed the bar exam on the first try

Table 7.1: Law School Admissions Data.

7.2 Adult Data

Another case study will be conducted on the popular privacy benchmark data set the Adult Census Data set. The data consists of records from the 1994 census where the dependent variable of interest is whether an individual was earning over $50,000$ in yearly income [BK96]. The total number of individuals in the study is $48,842$. A description of the variables are listed below in Table (7.2).

Variable	Description
age	the individual's age in years
workclass	category Private, Self Employed, Federal Government, ...
fnlwgt	survey weights in the census
education	category of Level of Education
marital status	category of individuals marital status
occupation	occupation category
relationship	Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried
race	category White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black
sex	category Female, Male
capital-gain	continuous
capital-loss	continuous
hours-per-week	continuous, number of working hours per week
native-country	category, country of origin
outcome	True or False if annual income is greater than $50K$

Table 7.2: Adult Census Data.

The census survey weight variable, "fnlwgt", is removed in the experiments it is not relevant to the study besides the design of the survey.

7.3 Experimental Design

Since the dependent variable of interest for both data sets is a binary variable, the modeling associated with each data set is naturally formulated as a classification task. For the Adult Data set the variable "outcome" is the dependent variable and for the Law School data the variable "bar" is the dependent variable. We will study neural network models as the victim models for our experiments since they were some of the first models that demonstrated vulnerability to membership inference attacks [SSS16].

For detection of disparity, we will use the race indicator variable as a partition of the population into subgroups for both of the data sets. All the experiments will focus discrete adversarial features:$\phi_W = \mathbb{1}[y \neq A_S(x)]$, when the model releases whether the data point (x, y) was predicted correctly. To implement the worst case estimators we will simply used the plug in estimators of the $0-1$ loss on the training and testing data sets, and taking their absolute difference to be our estimate of worst case vulnerability.

7.3.1 Victim Model Implementation

All experiments are conducted in Python (version 3.10) and were seeded to be reproducible.

Both victim neural networks are custom implemented for tabular data sets using the popular framework PyTorch [PGM19]. The neural network that I explored were simple: 1 hidden layer with 8 hidden units. The weights w were all randomly initialized using by the uniform distribution: $w \sim \text{Unif}\left[-1/\sqrt{10}, 1/\sqrt{10}\right]$. The neural network fitted for 200 epochs by Stochastic Gradient descent with a learning rate of $\alpha = 0.01$. No form of regularization on the weights w is used during training.

7.3.2 Attack Model Implementation

For all the experiments we will use shadow modeling attacks. For each model, we will use 5 shadow models, and the the attack classifier is an XGBoost classifier with 100 decision trees [CG16]. There is one nuance of shadow modeling that needs to be addressed. In the original paper on shadow modeling, [SSS16], there is an assumption that the adversary has access to a the data distribution $\mathcal{D}$. To simulate having access to $\mathcal{D}$, we will split our data set in three separate data sets: training, testing, and reference. The reference data set will be randomly sampled to build the 5 shadow models on that will have the same hyper-parameters and training implementations as the victim models. The victim model will be trained on the training data set and the accuracy of the attack model will be assessed only on the training and testing data set.

The training and testing data set are kept to the same size to be as close to the membership inference game. Arbitrarily increasing or decreasing the sizes of the training and testing data sets will have a substantial effect on estimated vulnerability. For all experiments, we set the training and testing data set sizes to be approximately 40% of the entire data with the remaining 20% used for the reference data set. In some experiments, authors allow the reference and training data sets to share the same data, but as a worst case analysis we do not allow the training and reference data sets to have any overlap.

Furthermore, the shadow modeling attacks are repeated 200 times for each victim model. Each time, the split of the training, testing, and reference data sets is changed.

7.4 Law School Data Results

The next section contains the results of the worst case vulnerability estimators, model specific estimates of vulnerability, and model specific estimates of disparity for the neural network.

Figure (7.3) depicts the empirical distribution of the shadow model specific estimates of

vulnerability broken down across the different race attributes for the neural network victim model. The distributions for non-white individuals are approximately centered around 2.5% vulnerability while the distribution for white individuals are centered around 0% vulnerability. In addition, the variability of the estimates for subgroup of White individuals is significantly smaller than all other races. Graphically there appears to evidence of disparity in vulnerability among the different races.

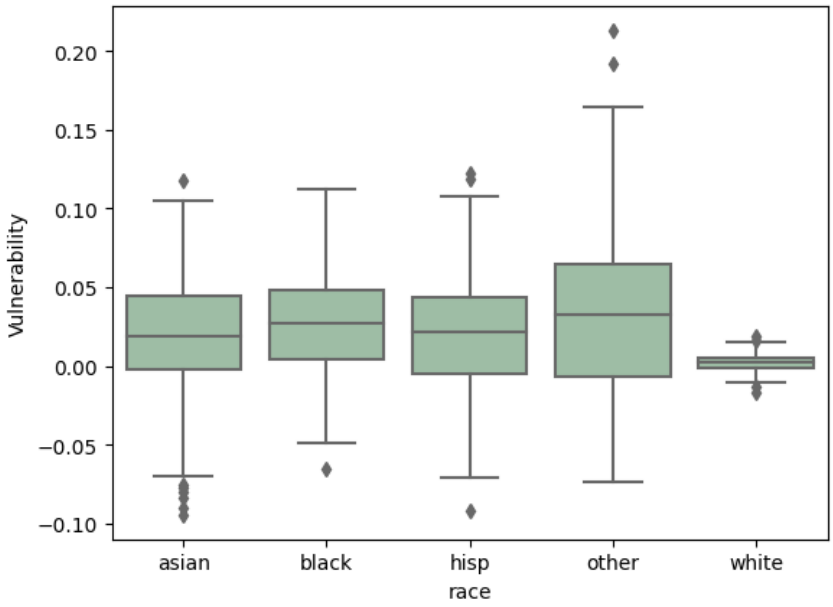

Figure 7.1: Shadow Model Estimates of Vulnerability broken down by Race for the Law School Data set.

Table (7.3) depicts the results of the repeated measure ANOVA of the shadow model estimates of vulnerability for the neural network. The p-value of the F-test is approximately $4.4e^{-15}$, so the null hypothesis of disparity is rejected at the signficance level of $\alpha = 0.01$. Indicating that for the neural network there is evidence of disparity in vulnerability, confirming the visual results seen in Figure (7.1)

Since the ANOVA F-test rejects the null hypothesis, we perform Bonferroni adjusted pairwise t-tests to determine which groups exhibit statistically significant disparity, with

Table 7.3: ANOVA Results for the Law School Data set.

	F Value	DF	$Pr(> F)$
race	9.185115	796	$4.440025e^{-15}$

Table 7.4: Bonferroni Adjusted Pairwise t-tests for the Law School Data set.

z	z'	t	p	p-corr.	Reject Null
asian	black	-2.1616	0.0318	0.0455	False
asian	hisp	-0.3078	0.7586	0.7586	False
asian	other	-3.0455	0.0026	0.0053	True
asian	white	5.7856	0.0000	0.0000	True
black	hisp	1.9998	0.0469	0.0586	False
black	other	-1.4794	0.1406	0.1562	False
black	white	10.4066	0.0000	0.0000	True
hisp	other	-2.855	0.0048	0.0079	True
hisp	white	6.6791	0.0000	0.0000	True
other	white	8.158	0.0000	0.0000	True

significance level $\alpha = 0.01$. The results are contained below in Table (7.4). The table contains the t-values of the test, the p-values, the adjusted p-values, and whether the t-test is rejected at the $\alpha = 0.01$ level based on the adjusted p-value.

The particular striking observation is that there is evidence of disparity between white individuals and all other races. Moreover, the direction of the disparity is positive indicating that the other races are easier to identify in the training data set. Additionally, the other category exhibits statistically significant disparity between all other races except for African American.

Lastly, Figure (7.4) depicts the results of the worst case estimate of vulnerability (x-axis)

in comparison to the estimated shadow model estimates of vulnerability (y-axis). For the neural network model there is evidence that the worst case estimates of vulnerability are correlated with shadow modeling. The correlation coefficient below is 0.66.

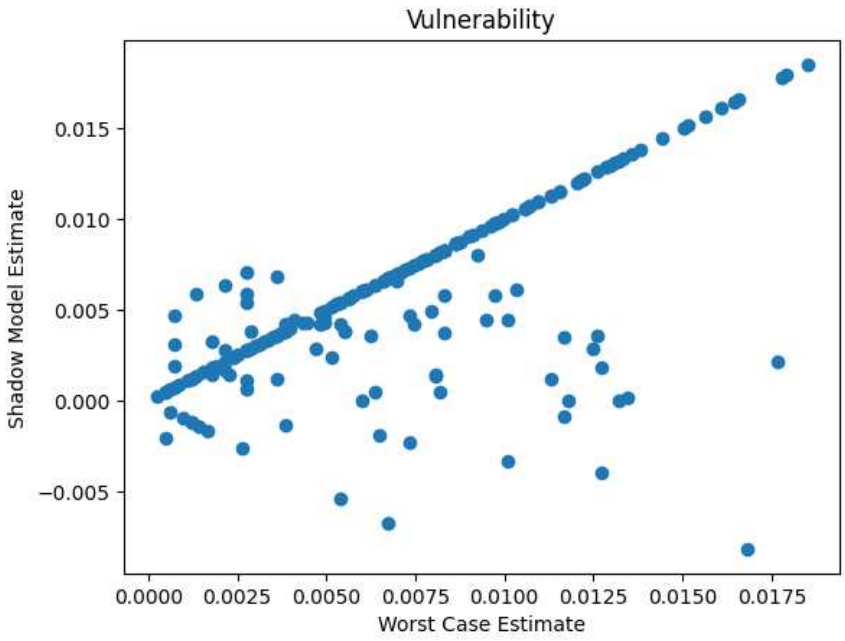

Figure 7.2: Worst Case Vulnerability and Shadow Model Vulnerability for the Law School Data set.

7.5 Adult Census Data Results

The next section contains the results of the worst case vulnerability estimators, model specific estimates of vulnerability, and model specific estimates of disparity for the Adult Census Data set.

Figure (7.3) depicts the distributions of the shadow model specific estimates of vulnerability broken down across the different race attributes for the neural network victim model. Graphically there appears to evidence of disparity in vulnerability among the different races as the distributions have different medians and variance across all racial categories.

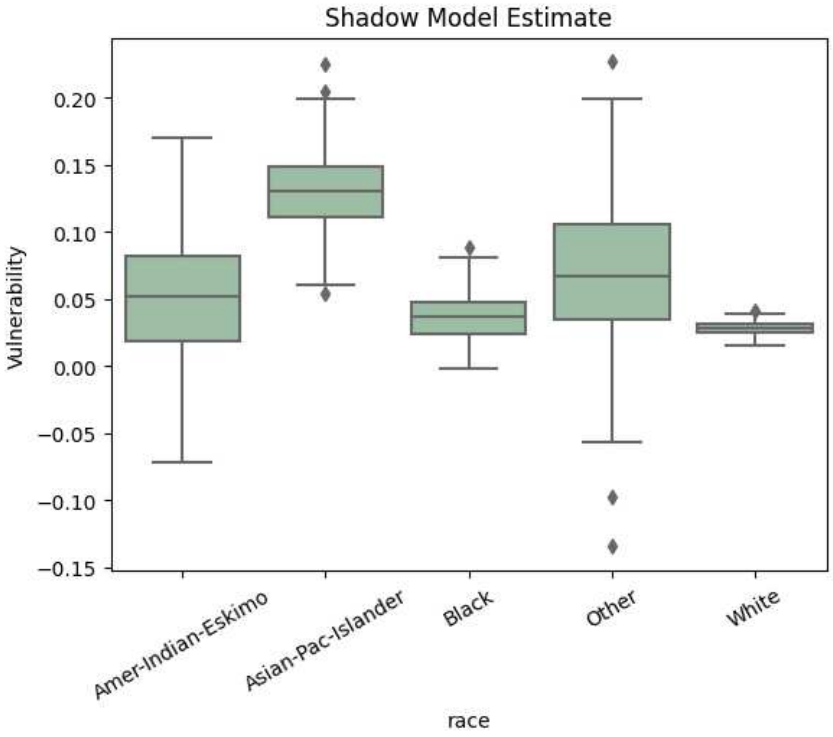

Figure 7.3: Shadow Model Estimates of Vulnerability broken down by Race for the Adult Data set.

Table (7.3) depicts the results of the repeated measure ANOVA of the shadow model estimates of vulnerability for the neural network. The p-value of the F-test is approximately $4.4e^{-133}$, so the null hypothesis of disparity is soundly rejected at the significance level of $\alpha = 0.01$. Indicating that for the neural network on the Adult data set there is strong evidence of disparity in vulnerability, confirming the visual results seen in Figure (7.3).

Table 7.5: ANOVA Results for the Adult Data set.

	F Value	DF	$Pr(> F)$
race	234.553506	796	$5.440307e^{-133}$

Since the ANOVA F-test rejects the null hypothesis so we perform Bonferroni adjusted pairwise t-tests to determine which groups exhibit statistically significant disparity, with significance level $\alpha = 0.01$. The results are contained below in Table (7.4). The table contains the t-values of the test, the p-values, the adjusted p-values, and whether the t-test is rejected at the $\alpha = 0.01$ level based on the adjusted p-value.

The Bonferroni Adjusted t-tests confirm that every group exhibits disparity among all other groups. This result is not particularly surprising given the Figure (7.3) and the p-value of the F-test being so small. One particularly concerning take away from these experiments is that the white individuals appear to be the most protected, i.e. least the vulnerable to membership inference attacks.

Lastly, Figure (7.4) depicts the results of the worst case estimate of vulnerability (x-axis) in comparison to the estimated shadow model estimates of vulnerability (y-axis). For the Adult Census data set the neural network model there is evidence that the worst case estimates of vulnerability are highly correlated with shadow modeling. The correlation coefficient below is 0.998.

Table 7.6: Bonferroni Adjusted Pairwise t-tests for the Adult Data set.

z	z'	t	p	p-corr.	Reject Null
'Amer-Indian-Eskimo'	'Asian-Pac-Islander'	-18.4981	0.	0.	True
'Amer-Indian-Eskimo'	'Black'	3.4963	0.0006	0.0006	True
'Amer-Indian-Eskimo'	'Other'	-3.352	0.001	0.001	True
'Amer-Indian-Eskimo'	'White'	6.148	0.	0.	True
'Asian-Pac-Islander'	'Black'	40.6659	0.	0.	True
'Asian-Pac-Islander'	'Other'	13.5096	0.	0.	True
'Asian-Pac-Islander'	'White'	51.7445	0.	0.	True
'Black'	'Other'	-7.5032	0.	0.	True
'Black'	'White'	8.1134	0.	0.	True
'Other'	'White'	10.2026	0.	0.	True

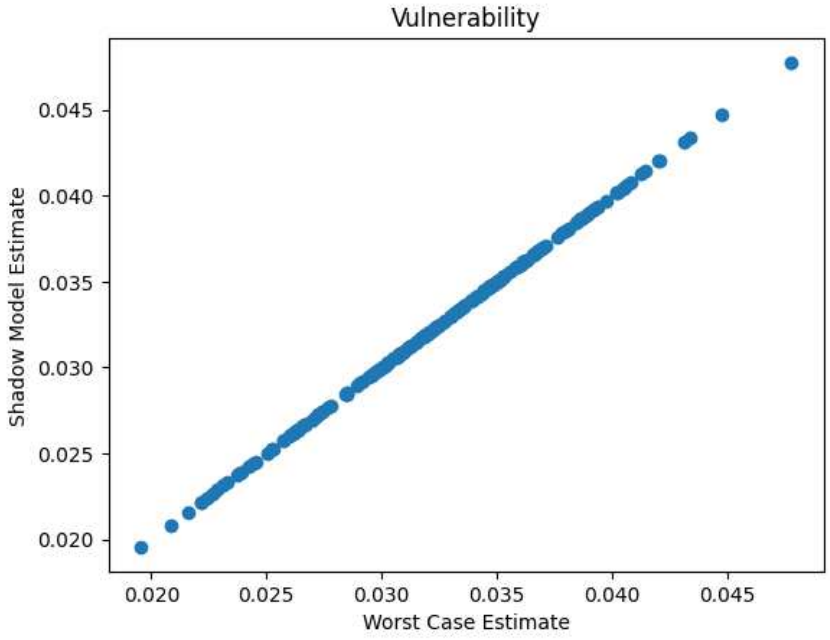

Figure 7.4: Worst Case Vulnerability and Shadow Model Vulnerability for the Adult Data set.

CHAPTER 8

Discussion

In this thesis we explored the notion of worst case vulnerability and the notion of disparity in vulnerability to membership inference attacks. We proposed estimators for the a subcase of discrete adversarial features and implemented shadow model attacks on real world data sets. We demonstrated there there is evidence of statistically significant disparity when viewing the population partition by race on two popular privacy benchmark data sets. In the final section, we will address some guidelines for practitioners conducting privacy auditing and address some possible future directions for the general case of continuous adversarial features.

8.1 Practitioners Considerations

The real world experiments demonstrated several nuances for practitioners conducting privacy auditing to consider.

First, a simple one hidden layer neural network can exhibit statistically significant evidence of disparity among subgroups. Adversaries attempting to infer the training data set can seek to exploit this by using subgroup aware adversaries to improve the accuracy of attacks. Thus practitioners should estimate disparity among sensitive subgroups to get more realistic estimates of privacy leakage.

Second, in the case of classifiers like neural networks with discrete adversarial features, practical estimators of worst case privacy violations can help to compute an estimate of

vulnerability without resorting to computationally extensive simulation techniques. Differentially privatizing models bounds even the theoretical worst case of privacy leakage and should always be used when making models trained on highly sensitive data available to the public. One ethical drawback of computed estimators for privacy leakage is that they can be exploited by adversaries. An adversary can use the same estimator to identify which subgroups will be easiest to identify and add those subgroup indicators into the adversarial feature set. However, I strongly believe the usefulness of these estimators for practitioners seeking to mitigate privacy risks is more beneficial than an adverse actor seeking to exploit them.

Lastly, the main nuance for modelers seeking to deploy models to the public is that they will not know in advance what set of adversarial features (Z, W) will be used by the attacker. Especially since any variable or combination of variables in the set of $\mathcal{X}$ can simply be appended to the output of the model W to form (Z, W). Thus the only sound defense strategy, with provable guarantees against both disparity and vulnerability to membership inference attacks is to make machine learning models differentially privatized.

8.2 Future Directions

One particularly interesting question, that is not addressed in this thesis, is how membership inference attacks such as shadow modeling are related to the Bayes Optimal Attacker. We can compute estimates of worst case privacy violations and examine how correlated they are to existing attack strategies, but the question of how to develop a method that accurately approximates the Bayes Optimal Attacker is still unknown. A fruitful direction to explore in the future would be to determine attack strategies that produce attack models that are consistent with the Bayes Optimal Attacker. Some papers have attempted to characterize which attacks are the Bayes Optimal Attacker, but are not analytically tractable (they are left as high dimensional integrals) [SDO19]. Furthermore, [SDO19] have suggested that

shadow modeling may be a Monte Carlo approximation to one of the terms derived in their expression of the Bayes Optimal Attack without providing formal analysis. This avenue merits further investigation.

One major use case for estimating worst case vulnerability not addressed in this thesis when the set of features released by the model is continuous $\phi_w(A_s, x) : \mathcal{X} \to \mathbb{R}^d$. This is would be the case where a classification model releases the predicted class probabilities. This is an important issue that needs to be addressed in future studies. The problem of estimating $V(\mathcal{A}^*_W)$ reduces to estimating the expression:

$$d_{TV}(\mu_1^\pi, \mu_0^\pi) = \sup_A |\mu_1^\pi(A) - \mu_0^\pi(A)|$$

If we assume μ_1^π and μ_0^π are absolutely continuous measures with respect to the Lebesgue measure and have density functions f_1 and f_0 respectively then the above simplifies to:

$$d_{TV}(\mu_1^\pi, \mu_0^\pi) = \frac{1}{2} \int |f_1(x) - f_0(x)| \, dx$$

In general, the problem becomes estimating the above integral based on samples from the measures μ_1^π, μ_0^π. There are well known results from kernel density estimation that allow for consistent estimation of total variation distance between two probability measures from samples.

Let $f_1^n(x)$ and $f_0^m(x)$ be the empirical distribution functions estimated via Kernel Density Estimation for densities f_1 and f_0 respectively. We are interested in bounding:

$$|d_{TV}(f_1^n, f_0^m) - d_{TV}(f_1, f_0)|$$

Using the fact that $d_{TV}(\cdot, \cdot)$ is a norm we end up with:

$$|d_{TV}(f_1^n, f_0^m) - d_{TV}(f_1, f_0)| \leq d_{TV}(f_1^n, f_1) + d_{TV}(f_0^m, f_0)$$

See [SFG12] for an in depth discussion of consistent estimation of integral probability metrics.

The main result from Kernel Density Estimation (KDE) for absolutely continuous probability measures f in $\mathbb{R}^d$ with KDE f^n [Tay18]:

$$\lim_{n\to\infty} \int_\Omega |f^n - f| \, d\Omega \to 0 \;\; a.s.$$

Note the convergence of the above quantity is exactly $1/2$ $d_{TV}(f^n, f)$, i.e. the total variation distance between kernel density estimate and the underlying density f. Thus the result we get is the following:

$$\lim_{\substack{n\to\infty \\ m\to\infty}} |d_{TV}(f_1^n, f_0^m) - d_{TV}(f_1, f_0)| \to 0 \;\; a.s.$$

In essence we can get consistent estimator of $d_{TV}(f_1, f_0)$ by using the KDE of f_1 and f_0 and numerically computed an estimate of the integral by any numerical estimation of integrals, e.g. Simpson's rule or trapezoidal rule.

There are two issues with this approach that need to be addressed before worst case estimates can be made available to practitioners. First, this method would rely on kernel density estimates which is difficult in practice due to selection of the bandwidth and dimension of the problem. Some general rules of thumb or guidance would be needed for privacy auditing guidelines. The second is that we need to correctly characterizing the source of randomness in the measures μ_1^π and μ_0^π which is taken over the sampling of the data sets $S \sim \mathcal{D}^n$ and $x \sim \mathcal{D}$ or $x \sim S$. Analysis on a theoretically tractable case would be highly beneficial to understand the feasibility of this approach.

REFERENCES

[ACG16] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. "Deep Learning with Differential Privacy." In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, oct 2016.

[Bis16] Christopher M. Bishop. *Pattern recognition and machine learning*. Information Science and Statistics. Springer, 1 edition, 2016.

[BK96] Barry Becker and Ronny Kohavi. "Adult." UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

[CG16] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." *CoRR*, **abs/1603.02754**, 2016.

[DLG14] Luc Devroye, Gábor Lugosi, and László Györfi. *Probabilistic theory of pattern recognition*. Stochastic Modelling and Applied Probability. Springer, 1 edition, 2014.

[Dwo08] Cynthia Dwork. "Differential Privacy: A Survey of Results." In *Theory and Applications of Models of Computation*, 2008.

[HSS22] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S. Yu, and Xuyun Zhang. "Membership Inference Attacks on Machine Learning: A Survey." *ACM Comput. Surv.*, **54**(11s), sep 2022.

[Kee10] Robert W. Keener. *Theoretical statistics*. Springer Texts in Statistics. Springer, 1 edition, 2010.

[PGM19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. "PyTorch: An Imperative Style, High-Performance Deep Learning Library." *CoRR*, **abs/1912.01703**, 2019.

[San04] Richard Sander. "A Systemic Analysis of Affirmative Action in American Law Schools." *Stanford Law Review*, **57**:367–483, 11 2004.

[SB14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms.* Cambridge University Press, 2014.

[Scu23] Marco Scutari. "fairml: A Statistician's Take on Fair Machine Learning Modelling.", 2023.

[SDO19] Alexandre Sablayrolles, Matthijs Douze, Yann Ollivier, Cordelia Schmid, and Hervé Jégou. "White-box vs Black-box: Bayes Optimal Strategies for Membership Inference.", 2019.

[SFG12] Bharath K. Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, and Gert R. G. Lanckriet. "On the empirical estimation of integral probability metrics." *Electronic Journal of Statistics*, **6**(none):1550 – 1599, 2012.

[SSS16] R. Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. "Membership Inference Attacks Against Machine Learning Models." *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, 2016.

[Tay18] C. C. Taylor. "Nonparametric Density Estimation: The L1 View." *Royal Statistical Society. Journal. Series A: General*, **148**(4):392–393, 12 2018.

[YFJ17] Samuel Yeom, Matt Fredrikson, and Somesh Jha. "The Unintended Consequences of Overfitting: Training Data Inference Attacks." *CoRR*, **abs/1709.01604**, 2017.

[YKT19] Mohammad Yaghini, Bogdan Kulynych, and Carmela Troncoso. "Disparate Vulnerability: on the Unfairness of Privacy Attacks Against Machine Learning." *CoRR*, **abs/1906.00389**, 2019.