UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR IT-SICHERHEIT

# Extending the Likelihood Ratio Attack to Random Forest Models

*Ausweitung des Likelihood-Ratio-Angriffs auf Random Forest Modelle*

**Bachelorarbeit**

im Rahmen des Studiengangs
**IT-Sicherheit**
der Universität zu Lübeck

vorgelegt von
**Julius Benedict Niehoff**

ausgegeben und betreut von
**Prof. Dr. Esfandiar Mohammadi**

mit Unterstützung von
**M.Sc. Marven Kummerfeldt**
**M.Sc. Johannes Liebenow**

Lübeck, den 21. Juli 2025

## Abstract

Machine learning is an essential tool in modern data science. The ever-growing amounts of data gathered from websites, users, customers, patients, and more are being used as databases to train machine learning models. Even though this data is generally anonymized before its use in the training of machine learning models, the model can still leak information. More privacy attacks targeting these machine learning models have been developed and improved in recent years. Analyzing these attacks is crucial for reducing the privacy risk associated with machine learning models. Determining the reasons for an attack's success will enable developers to implement more effective countermeasures.

Membership inference attacks are used to determine whether a specific data point was used to train a model. These attacks can reveal sensitive information from the model's training data. Depending on the model, this information may be personal, medical, or financial in nature.

The Likelihood Ratio Attack is a membership inference attack that uses a hypothesis test to attack neural networks. Given the mathematical nature of this test, we hypothesize that this attack can be applied to other machine learning models, independent of their architecture. Additionally, we aim to link overfitting to the success of the Likelihood Ratio Attack. Overfitting occurs in many machine learning models during the training process. Furthermore, we utilize random forest models to link specific model parameters to the success of the Likelihood Ratio Attack.

By extending the Likelihood Ratio Attack from neural networks to random forests, we achieve a direct comparison between the attacks' behavior when applied to two different model types. We can generate various models with varying degrees of overfitting by tuning the parameters of the random forest models. This enables us to successfully link overfitting to an increase in the Likelihood Ratio Attacks' success. Additionally, we develop a new metric to measure a model's overfitting. This method enables us to predict the success of the Likelihood Ratio Attack before conducting it.

## Zusammenfassung

Machine Learning stellt ein zentrales Werkzeug innerhalb der modernen Data Science dar. Die kontinuierlich wachsenden Datenmengen, die aus Quellen wie Webseiten, Nutzern, Kunden oder Patienten stammen, bilden die Grundlage für das Training von Machine Learning Modellen. Obwohl diese Daten vor dem Training in der Regel anonymisiert werden, besteht weiterhin das Risiko, dass Modelle vertrauliche Informationen preisgeben. In den letzten Jahren sind zunehmend sogenannte Privacy Attacks auf Machine Learning Modelle entwickelt und weiter verbessert worden. Die Analyse dieser Angriffe ist essenziell, um die mit dem Einsatz solcher Modelle verbundenen Risiken zu minimieren. Ein tiefgehendes Verständnis der Ursachen für den Erfolg dieser Angriffe ermöglicht die Entwicklung wirksamer Gegenmaßnahmen.

Membership Inference Attacks zielen darauf ab, festzustellen, ob ein bestimmter Datenpunkt im Trainingsdatensatz eines Modells enthalten ist. Solche Angriffe können sensible Informationen über die Trainingsdaten offenlegen, wobei je nach Modell persönliche, medizinische oder finanzielle Inhalte betroffen sein können. Der Likelihood Ratio Attack stellt eine spezifische Form des Membership Inference Attacks dar, bei der ein Hypothesentest zur Anwendung kommt, um neuronale Netze zu kompromittieren. Aufgrund des mathematischen Charakters des Tests lässt sich vermuten, dass dieser Angriff auch auf andere Machine Learning Modelle übertragbar ist; unabhängig von deren Architektur. Ziel dieser Arbeit ist es, einen Zusammenhang zwischen Overfitting und dem Erfolg des Likelihood Ratio Attacks zu untersuchen. Overfitting tritt bei vielen Machine Learning Modellen im Verlauf des Trainingsprozesses auf. Zusätzlich kommen Random Forest Modelle zum Einsatz, um spezifische Modellparameter mit dem Erfolg des Likelihood Ratio Attacks in Beziehung zu setzen.

Durch die Erweiterung des Likelihood Ratio Attacks von neuronalen Netzen auf Random Forests wird ein Vergleich des Angriffsverhaltens zwischen zwei verschiedenen Modellen ermöglicht. Die gezielte Anpassung der Modellparameter erlaubt es, Random Forest Modelle mit unterschiedlich starkem Overfitting zu erzeugen. Auf diese Weise lässt sich ein klarer Zusammenhang zwischen Overfitting und einem erhöhten Angriffserfolg herstellen. Darüber hinaus wird eine neue Metrik zur Messung des Overfitting eingeführt, mit der sich der Erfolg des Likelihood Ratio Attacks bereits im Vorfeld vorhersagen lässt.

## Erklärung

Ich versichere an Eides statt, die vorliegende Arbeit selbstständig und nur unter Benutzung der angegebenen Hilfsmittel angefertigt zu haben.

Lübeck, 21. Juli 2025

## Acknowledgements

I would like to thank Marven Kummerfeldt and Johannes Liebenow for their extensive support throughout the work and for answering all of my questions. Thanks also to Esfandiar Mohammadi, who made this thesis possible. Also, special thanks to Elena Waelsch, David Waelsch, Emma Niggemann, and Maximilian Schmitz for proofreading this thesis.

# Contents

# 1 Introduction

Machine learning is a rapidly growing and increasingly important area of research [MJ18]. However, as with all new digital and technological advances, machine learning is subject to serious security concerns in its application and development. With the growing and widespread adoption of machine learning models across multiple digital sectors, these concerns develop into serious risks. The mitigation and prevention of risk factors and security incidents are important areas of research.

Membership inference attacks are one of the risk factors developers of machine learning models need to account for. These attacks aim to gain information about a machine learning model's training data. If successful, membership inference attacks, such as the Likelihood Ratio Attack (LiRA) [CCN+22], can leak information about the training data of their targeted model. Leaked data can include personal, financial, or medical information. Most membership inference attacks are developed to target neural networks [HSS+22]. In this work, however, we leverage the statistical nature of LiRA and apply it to random forest models. LiRA employs a Hypothesis test using the Neyman-Pearson Lemma [NPP33]. In the LiRA approach, two distinct distributions are constructed, which serve as the hypotheses for the test. The distributions are obtained by training shadow models using subsets of the training data, where a target point is carefully excluded from one of the distributions and included in the other. By leveraging the mathematical nature of LiRA, we can assess whether the underlying function of LiRA is independent of its target model.

Additionally, we create multiple experiment runs with varying random forest model parameters for each run. By carefully selecting the values for these parameters, we produce models with different degrees of overfitting. Overfitting is a phenomenon that occurs in machine learning models during the training process. It can also significantly contribute to the success of membership inference attacks because overfitting is often an indicator of a poorly trained model. A membership inference attack can utilize this knowledge to infer membership, as the model's behavior changes depending on whether the point was included in the training data. By creating models with different degrees of overfitting, we can isolate overfitting as a contributing factor to LiRA and measure its influence on the attack's success.

Our goal is to adapt the existing LiRA to random forest models. We theorize that this is possible because the mathematical nature of the hypothesis test used by the attack to infer membership is independent of the model's architecture. Following this, we aim to link overfitting to the success of LiRA. Overfitting has already been identified as a major contributing factor for other membership inference attacks; we thus theorize that it can also significantly impact LiRA. Building on this theory, we also try to connect a successful LiRA to the model parameters chosen during training. We hypothesize that this connection may exist because the parameters can directly influence the degree of overfitting observed in the final model, thereby impacting the subsequent attack.

In summary, our goals are:

- Adapting the existing LiRA to random Forest models.
- Determine the impact of specific model parameters, maximum depth, and number of estimators, of random forest models on the success of LiRA.
- Determine if the success of LiRA is dependent on its target model being overfitted.

This thesis is structured into three main parts. The first part contains the chapters Preliminary, Background, and Related Work. The Preliminary chapter will provide the reader with a basic understanding of the fields of machine learning and membership inference. Then, LiRA is explained in detail in the Background chapter. The Related Work chapter offers insights into the broader field of membership inference on random forests, with a focus on work closely related to this thesis.
Following these chapters are Our Approach and Experiments. The chapter Our Approach provides a detailed breakdown of your approach. Here we include both theoretical considerations and our final implementation. In the chapter Experiments, our results are displayed.
The last part contains our Discussion, Conclusion, and Outlook. In this part, we interpret and analyze our results. Lastly, we provide a conclusion and an outlook toward possible future work.

# 2 Preliminary

In this chapter, we cover the theoretical basics that will be built upon later in this thesis. This includes machine learning, membership inference attacks, and adversarial knowledge. As LiRA, which is the basis of this thesis, is a membership inference attack on neural networks that we extended to random forests, all of this information is crucial to understanding our work.

## 2.1 Machine Learning

Machine Learning (ML) is a core component of this thesis, as it provides the framework for the target models and the associated attack. ML models are algorithms that learn patterns from data and make decisions or predictions without the need for explicit programming [HTF09, GBC16a]. In this section, an overview of neural networks and random forests is provided to enhance understanding of the LiRA, which is used to attack these models in Our Approach.

Since extensive amounts of data are required to train accurate models, standard databases exist for various tasks, such as image classification, recognizing handwritten digits, or identifying languages. This includes the CIFAR-10 [Kri12] dataset. A ML model is initially trained using the training images and their corresponding classifications, also known as labels, which represent the predictions that the model aims to achieve. After training, the model is then tested with the test images to determine its actual performance on previously unseen data, as these images are not part of the training dataset. The exact process during training and the number of iterations performed during the training and testing cycle are highly dependent on the model architecture and parameters [HTF09, GBC16a].

### 2.1.1 Neural Networks

A neural network can be represented as a function $f(x, w) = y$ [MP43]. The network can then return an output $y$, the predictions for a given input $x$. During training, the goal is to adjust the parameters $w$ to achieve better predictions by minimizing a loss function $\ell(f(x, w), y)$.

Achieving $100\%$ accuracy during training is possible, but not required [ZBH+17]. The reason for this is that perfect accuracy during training does not necessarily translate to a better test accuracy. The overall goal is for the model to make accurate predictions using test data, which is not used in training, sampled from the same distribution as the training data. Additional techniques, such as weight regularization, are necessary to enhance the model's performance on tests.

### 2.1.2 Random Forest

A random forest is an ensemble learning method that constructs a collection of decision trees to improve predictive accuracy and robustness compared to individual classifiers [Bre01]. During training, each decision tree in the forest is built using a random subset of the training data. Decision trees start with a single decision as the root node [Qui86]. The following internal roots also represent decisions. Each decision selects a feature and a threshold for this feature, which is used to split the data. Finally, once a leaf is reached, all the data that remains for this leaf is classified as one class, so that each leaf represents a class.

The number of estimators, which represents the number of decision trees in the forest, and the maximum depth of each tree are important parameters that control the model's complexity and performance. Increasing the number of estimators generally improves stability and accuracy, but comes with increased computational cost. The maximum depth limits how deep each tree can grow, preventing overfitting by reducing model complexity [Qui86, Bre01].

Once all trees are trained, the random forest makes predictions by aggregating the outputs of each tree. For classification tasks, this is achieved through majority voting, where each tree casts a vote for a class label, and the most common label is returned as the final prediction [Bre01]. This ensemble approach produces more stable and accurate results than any single tree and is particularly effective in handling noisy or high-dimensional datasets. Random Forests are typically more interpretable and better suited to structured, tabular data.

## 2.2 Membership Inference Attacks

Choosing an attack according to the goal is crucial for the attack's success. A poisoning attack aims to manipulate a model during training, while a membership inference attack exploits fully trained models [BNL12, SSSS17].

Membership inference attacks typically aim to determine if a specific data point was used to train a model [CCN+22]. A successful attack is a privacy breach. The success

of the membership inference attack is dependent on the model being attacked. This leads to certain amplifications that increase the success rate of a membership inference attack that can occur during the training process [HSS+22]. One of these amplifications is called memorization and describes a model memorizing single data points during training. Memorization is important because it influences the loss of the memorized sample, thereby affecting its loss. A loss-based membership inference attack can then exploit this change.
When a model achieves a significantly better score during training than in testing, it is considered overfitted [YGFJ18]. Overfitting has been identified as a crucial factor for the success of membership inference attacks. Overfitting often results in models behaving differently when given training or test data. An attacker can construct an attack that recognizes these differences and detects if their chosen inputs were part of the training or test data. Thus, overfitting plays an essential role in membership inference. Other works have previously identified overfitting as a significant contributing factor to the success of membership inference attacks [HSS+22, YGFJ18].
Binary classifier-based and metric-based approaches are the two primary methods used to construct membership inference attacks.

### 2.2.1 Binary Classifier-based

Binary classifier-based membership inference attacks reduce the problem of inferring membership to a binary classification task [HSS+22]. In such attacks, the adversary first trains multiple shadow models that replicate the target model's behavior. The training data for these shadow models is carefully selected so that the membership status of each data point, whether it is part of the shadow models' training datasets, is explicitly known. By observing the outputs of the shadow models on both member and non-member data points, the attacker constructs a labeled dataset. This dataset is then used to train a binary classifier, which functions as the attack model. By building a known dataset, the attacker can simplify the problem's complexity. Instead of considering the membership of the target sample by analyzing only the behavior of the target model, the attacker can use multiple shadow models split into two classes: one where the models contain the target sample and one where it is excluded.

**Advantage of Shadow Models** The advantage of using shadow models to simulate the target model's behavior is that the attacker does not need access beyond basic queries to the target model [SSSS17]. Additionally, even if the attacker were to know the target model type, they might not have access to the parameters used for training, leaving them to make an educated guess to construct their attack. To effectively utilize shadow models, they do not need to be of the same model type as the target model. They need to mimic their behavior. Using multiple shadow models helps avoid overfitting the attack to a single instance of the target model, thereby increasing the attack's ability to generalize [SSSS17].

### 2.2.2 Metric-based

Metric-based attacks exploit the way machine learning models generate predictions by analyzing specific metrics derived from those predictions [HSS$^{+}$22]. These attacks assume that the model behaves differently on data it has seen during training compared to data it has not seen. For example, in prediction correctness-based attacks, the attacker assumes that the model is more likely to classify an input correctly if it was part of the training data. Other attack strategies rely on metrics such as prediction loss, where the attacker observes the error made by the model on an input, or prediction confidence, where the attacker examines the model's confidence in its output. By comparing these metrics across different inputs, the attacker attempts to infer whether a particular input was used during training, effectively inferring membership.

### 2.2.3 Loss-based Membership Inference Attacks

Loss-based membership inference attacks focus on the model's loss, as models typically have a lower loss on training samples when compared to samples from outside their training data [SSSS17]. This type of attack is more effective against overfitted models than against well-generalized models [YGFJ18]. For attacks of this type, the attacker computes the loss of the model for their target point. The loss is typically lower if the model already knows the sample, meaning it is part of the training data. Attacks often use a threshold to determine whether the loss is low or high, and therefore whether the target point is a member or non-member. While many loss functions exist, this thesis focuses on the commonly used cross-entropy loss function. The cross-entropy loss measures the difference between two probability distributions, typically comparing the predicted probabilities and the true labels, penalizing confident but wrong predictions more heavily.

## 2.3 Adversarial Knowledge

When conducting membership inference attacks, several assumptions can be made about the attacker's knowledge regarding the model and training data [HSS$^{+}$22]. The training data distribution can be known, partially known, or unknown to the attacker. Black-box and white-box are the classifications used to describe the attacker's level of knowledge about the targeted model.

White-Box attacks enable the attacker to access all the information of the model, including learned parameters, architecture, and training methods. Additionally, the attacker may also know the training data distribution. This allows the attacker access to shadow datasets with the same distribution as the training data.
Black-Box limits the adversary to information regarding the training data distribution and queries, allowing them to gain only the model's predictions for the input. The predictions generated by black-box models can be further categorized into three categories [HSS$^+$22].
**Full confidence scores** grant the attacker all confidence scores. Enables the attacker to determine the predicted output class.
**Top-K confidence scores** allow the attacker only a selection of the top-K confidence scores.
**Prediction labels only** restrict the attacker to only the predicted output class. A model's confidence score is a metric that indicates the model's level of confidence in its predictions. An attacker can use confidence scores to craft a better attack [CW17].

# 3 Background

This chapter focuses on LiRA by Carlini et al. [CCN+22]. The attack serves as the basis for this thesis and is an integral part of our approach. In addition to their proposed attack, Carlini et al. made another contribution with LiRA. They propose a new metric to evaluate membership inference attacks. Since we will use this metric to evaluate LiRA on random forests, it is also highlighted in this chapter.

## 3.1 Evaluating Membership Inference Attacks based on their True-Positive Rate at a low False-Positive Rate

Carlini et al. [CCN+22] argue that instead of using balanced accuracy to evaluate a membership inference attack, their proposed True-Positive Rate (TPR) at low False-Positive Rate (FPR) should be used as a metric instead. A TPR describes the positive examples that the model correctly identified. The FPR describes the number of negative examples the model incorrectly identifies as positive. They propose this new metric because the balanced accuracy does not accurately reflect the attack's ability to identify samples from the training data. Because the balanced accuracy only considers the average of the TPR, it can be misleading when evaluating an attack.
Carlini et al. give an example using a loss-based attack. While the attack has a balanced accuracy of $60\%$, this accuracy falls to just $48\%$ for the $1\%$ of samples with the lowest losses. They argue that a high TPR at a low FPR is a more accurate metric, as it also considers false positives. With the addition of a low FPR to the evaluation, the attack can much more confidently predict which samples were part of the original training data.

## 3.2 Likelihood Ratio Attack

Carlini et al. [CCN+22] also introduce LiRA. This membership inference attack uses the Neyman-Pearson Lemma [NPP33] for statistical hypothesis testing. The Neyman-Pearson Lemma provides an optimal method for deciding between two hypotheses. For LiRA, two hypotheses are constructed. One where a specific sample $(x, y)$ was used to train the targeted model $f$, and one where the sample was not part of the model's training data. These Distributions are formalized as $\mathbb{Q}_{in} = \{f \leftarrow \mathcal{T}(D \cup \{(x, y)\}) | D \leftarrow \mathbb{D}\}$ and

$\mathbb{Q}_{out} = \{f \leftarrow \mathcal{T}(D \setminus \{(x,y)\}) | D \leftarrow \mathbb{D}\}$, with $\mathbb{Q}_{in}$ containing $(x,y)$. $\mathcal{T}$ represents the training algorithm used. The hypothesis test is performed as described in the Neyman-Pearson Lemma:

$$\Lambda(f; x, y) = \frac{p(f|\mathbb{Q}_{in}(x,y)}{p(f|\mathbb{Q}_{out}(x,y)} \tag{3.1}$$

with $p(f \mid \mathbb{Q}_b(x,y))$ defined as the probability density function. However, because the two above described distributions are unknown, Carlini et al. define two new distributions, $\tilde{\mathbb{Q}}_{in}$ and $\tilde{\mathbb{Q}}_{out}$ describing the distribution of the loss of model trained on or not trained on the target sample. Then they replace the probabilities in Equation (3.1) with an easier version to calculate:

$$p(\ell(f(x)_y) \mid \tilde{\mathbb{Q}}_{in/out}(x,y)) \tag{3.2}$$

Here $\ell(f(x)_y)$ denotes the loss function $\ell$ applied to the models confidence $f(x)_y$.
To ensure that the cross-entropy loss is approximately normal, Carlini et al. apply a logit scaling to the model's confidence:

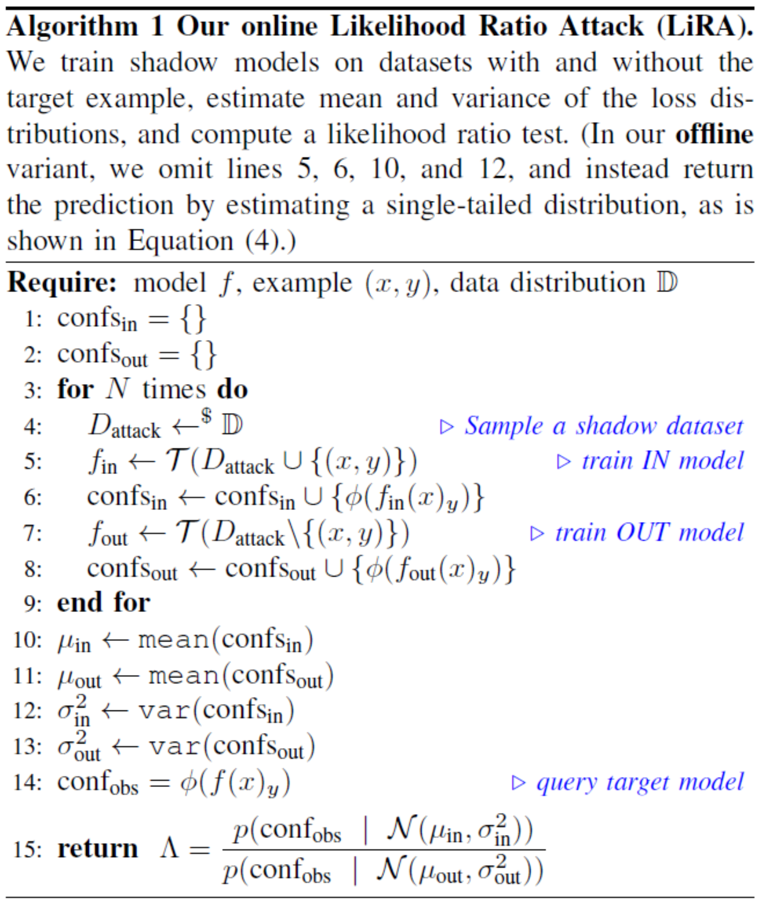
$$\phi(f(x)_y) = \log\left(\frac{f(x)_y}{1 - f(x)_y}\right) \tag{3.3}$$

With query access to the model $f$, Carlini et al. now use Equation (3.2) as a likelihood test for a one-dimensional statistic. They further reduce the number of necessary shadow models by assuming $\tilde{\mathbb{Q}}_{in/out}(x,y)$ as a Gaussian distribution.

To conduct this attack, shadow models are trained to determine the confidence distribution of the original models. Confidence distributions represent the model's confidence scores, which are the predicted probabilities of the correct class, over many samples. Carlini et al. use the algorithm in Figure 3.1 for their attack.

Carlini et al. [CCN+22] describe two possible setups for LiRA, online and offline.

**Online LiRA** For the online setting, the attacker trains shadow models where one half of the models contains a specific sample $(x,y)$ and the other half does not. Now, the logit-scaled confidence scores of the shadow models are calculated. The distribution of the scores is estimated to be Gaussian, resulting in two Gaussian distributions when modeled parametrically: one for models that contain $(x,y)$ and one for models that do not include $(x,y)$. Then the target model is queried with the previously selected sample. Lastly, the attacker uses the Neyman-Pearson Lemma to perform a likelihood ratio test and obtain the final score for the Likelihood Ratio Attack. The final score represents the likelihood



**Algorithm 1 Our online Likelihood Ratio Attack (LiRA).** We train shadow models on datasets with and without the target example, estimate mean and variance of the loss distributions, and compute a likelihood ratio test. (In our **offline** variant, we omit lines 5, 6, 10, and 12, and instead return the prediction by estimating a single-tailed distribution, as is shown in Equation (4).)

```
Require: model $f$, example $(x, y)$, data distribution $\mathbb{D}$
 1: $\text{confs}_{\text{in}} = \{\}$
 2: $\text{confs}_{\text{out}} = \{\}$
 3: for $N$ times do
 4:     $D_{\text{attack}} \xleftarrow{\$} \mathbb{D}$                                   ▷ Sample a shadow dataset
 5:     $f_{\text{in}} \leftarrow \mathcal{T}(D_{\text{attack}} \cup \{(x, y)\})$                 ▷ train IN model
 6:     $\text{confs}_{\text{in}} \leftarrow \text{confs}_{\text{in}} \cup \{\phi(f_{\text{in}}(x)_y)\}$
 7:     $f_{\text{out}} \leftarrow \mathcal{T}(D_{\text{attack}} \backslash \{(x, y)\})$                ▷ train OUT model
 8:     $\text{confs}_{\text{out}} \leftarrow \text{confs}_{\text{out}} \cup \{\phi(f_{\text{out}}(x)_y)\}$
 9: end for
10: $\mu_{\text{in}} \leftarrow \texttt{mean}(\text{confs}_{\text{in}})$
11: $\mu_{\text{out}} \leftarrow \texttt{mean}(\text{confs}_{\text{out}})$
12: $\sigma^2_{\text{in}} \leftarrow \texttt{var}(\text{confs}_{\text{in}})$
13: $\sigma^2_{\text{out}} \leftarrow \texttt{var}(\text{confs}_{\text{out}})$
14: $\text{conf}_{\text{obs}} = \phi(f(x)_y)$                                  ▷ query target model
15: return $\Lambda = \dfrac{p(\text{conf}_{\text{obs}} \mid \mathcal{N}(\mu_{\text{in}}, \sigma^2_{\text{in}}))}{p(\text{conf}_{\text{obs}} \mid \mathcal{N}(\mu_{\text{out}}, \sigma^2_{\text{out}}))}$
```

Figure 3.1: Algorithm of the Likelihood Ratio Attack by Carlini et al. [CCN+22]. The equation referenced in the description is found here: Equation (3.4)

that the sample is a member. A higher score indicates a higher likelihood. This version of the attack requires the attacker to train new shadow models for each targeted point.

**Offline LiRA** For the offline setting, the attacker uses only shadow models that do not contain the targeted sample $(x, y)$. This reduces the required computations, as the attacker does not need to train models for each new target point. In this version, a one-sided hypothesis test, as shown in Equation (3.2), is used to calculate the final score for LiRA. To use this version of the attack, lines 5, 6, 10, and 12 are omitted from the attack described in Figure 3.1. Additionally, line 15 is changed to the one-sided hypothesis test:

$$\Lambda = 1 - \Pr[Z > \phi(f(x)_y)], \text{ where } Z \sim \mathcal{N}(\mu_{out}, \sigma^2 out) \tag{3.4}$$

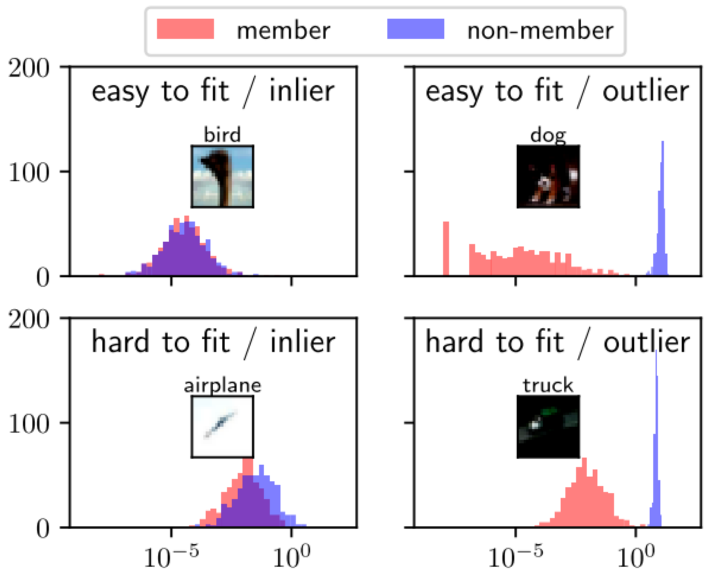


Figure 3.2: Visualization of member (red) non-member (blue) distributions by Carlini et al. [CCN+22] on CIFAR-10.

## 3.3 Member and Non-Member Distributions

When analyzing LiRA, member-non-member distributions can be used to visualize the statistical contrast between the model's behavior on member versus non-member data points. This contrast is a key feature of LiRA, enabling one to determine whether a data point is a member or a non-member of the training data. Carlini et al. [CCN+22] include Figure 3.2, describing multiple possible cases when analyzing a specific data point. The figure showcases different scenarios and distributions. The columns compare "inliers" and "outliers". The rows display the difference in the sample loss. The top row includes samples with a low loss, and the bottom row includes samples with a high loss.

Figure 3.2 showcases the crucial difference in the loss of members and non-members. While in some cases the loss overlaps to create nearly identical distributions, in other cases, two very distinct distributions are visual. Later, we will use this as an essential part of our analysis. Being able to distinguish between member and non-member samples by their loss is the basis for loss-based membership inference attacks. These distributions illustrate how training shadow models with or without a specific sample affects the loss of that sample when viewed as a member and a non-member.

Additionally, Carlinin et al. [CCN+22] apply a logit scaling to the cross-entropy loss to fit the distributions to an approximately normal curve. This process is displayed in Figure 3.3

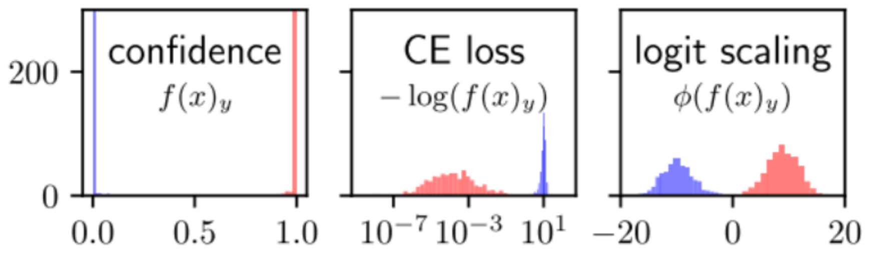


Figure 3.3: Visualization of the logit transformation when applying the logit transformation (right) and cross-entropy loss (middle) to the model's confidence (left) [CCN+22]

and described in Equation (3.3). The left diagram displays the model's confidence. In the middle, the Cross-entropy loss is displayed. On the right, logit scaling is applied to the cross-entropy loss, creating the logit-scaled confidence.

## 3.4 Implementation Details

Carlini et al. [CCN+22] provide the source code for their attack on their GitHub[1]. They use the PyTorch [PGM+19] library to train, test, and query neural networks. PyTorch is an open-source deep learning library that provides flexible and efficient tools for building and training neural networks. The code they provide is written in the Python [VRD09] programming language. Python is a high-level, interpreted programming language commonly used in fields ranging from web development to data science and artificial intelligence. We use this existing code to adapt LiRA to random forest models.

In their existing code, they separate the attack workflow into four parts. First, the training script trains several shadow models using a pre-trained neural network, specifically a ResNet [HZRS16]. Pretrained models are classification models that have been trained on a general dataset and are then fine-tuned by the end-user for their specific use case. This reduces the time and resources needed to train a model. Because LiRA requires several shadow models for the attack, using a pretrained model significantly reduces the time needed to train each shadow model. The training script randomly splits the training data. This is done because the training script is run multiple times for each attack. By splitting the training data, each shadow model contains a partially different set of points.

[1]`https://github.com/tensorflow/privacy/tree/master/research/mi_lira_2021`

This is crucial for the attack, as the goal is to differentiate between the loss of a member versus a non-member. By creating shadow models with or without a point, the shadow models can be grouped into the distributions $\tilde{\mathbb{Q}}_{in/out}$ for each point. The exact points used for each model are saved in a separate file with the corresponding shadow model. Then the inference script queries the models and saves their output features for the entire training data. The output features are converted into confidence scores, which are then logit-scaled. Next, the logit-scaled inference scores are created using the previously saved logit-scaled confidence scores. Figure 3.3 showcases that for this step, only the logit-scaled confidence is well approximated by a normal distribution, prompting the attack to use this metric over the cross-entropy loss. Lastly, the results are plotted.

# 4 Related Work

In this chapter, we aim to provide an overview of similar works in the field of membership inference attacks.
The metric for evaluating membership inference attacks, proposed by Carlini et al. [CCN$^+$22] as part of their LiRA, has been adopted by other authors to evaluate their membership inference attacks.
Shachor et al. [SRG24] propose an improved method when performing membership inference attacks. They show that dividing the data into small subsets and training adversary models for each subset can lead to a significantly improved TPR at a low FPR, an evaluation metric proposed by Carlini et al., of the attacks.

To gain a better understanding of current research, a collection of other works that summarize their contributions to the field has been compiled.
Hu et al. [HSS$^+$22] provide one of the first comprehensive surveys on membership inference attacks across a broad range of machine learning models and settings. They categorize attacks based on adversarial knowledge, target model types, and attack strategies. The survey also reviews defense approaches, such as differential privacy, regularization, and output obfuscation, highlighting their strengths and weaknesses. While defensive strategies are important, they are not the topic of this thesis. Additionally, three main factors contributing to the success of membership inference attacks have been identified by gathering data from the sources listed in the survey:

- Overfitting
- Model type
- Diversity of training data

Of these three factors, we have already highlighted overfitting in the Preliminary chapter and in our Introduction. Overfitting is a vital part of our thesis and has been formalized by Yeom et al. [YGFJ18].
Yeom et al. [YGFJ18] investigate the connection between overfitting and privacy risk in machine learning models, particularly focusing on membership and attribute inference attacks. They demonstrate that the generalization gap (training accuracy minus test accuracy) provides a strong indicator of risk for membership inference: larger gaps correlate

with greater success in attacks. Additionally, they show that feature influence, a metric measuring how changes in input features affect model output, plays a crucial role in attribute inference. Furthermore, they prove that while overfitting is sufficient to enable these attacks, it's not necessary. A model may still leak private information via structure or learned patterns, even when generalization is strong.

The second factor that is important to our work is the model type. Many state-of-the-art membership inference attacks, like LiRA, are conducted on neural networks [HSS$^+$22]. However, recently, more work has been published on membership inference attacks on models other than neural networks. While writing our thesis, two of these papers were published that also include random forests as their targeted models.
Akbarian and Aminifar [AA25] focus specifically on membership inference attacks against random forests. They propose an attack that identifies training samples by analyzing how consistently they fall within certain decision boundaries across the forest's trees. Their method is effective even in federated learning scenarios, where an attacker has limited access to the training data. Federated learning is a specific method for training machine learning models; however, this is not relevant to our thesis, as we do not use this method. Their results show that random forests can be vulnerable to this type of structural attack, and they offer a simple metric to help measure this risk in practice.

Preen and Smith [PS25] present a two-stage method for assessing the vulnerability of tree-based models, including random forests, to membership inference attacks. First, their approach utilizes model parameters, such as maximum depth, number of estimators, and learning rate, to screen for high-risk configurations before training. Then they apply structural metrics after training for further risk assessment. They demonstrate that these risk indicators generalize well across datasets and model types and validate their method using advanced attacks, such as the LiRA. Additionally, they find that strong model performance does not necessarily imply higher privacy risk, suggesting that privacy and accuracy can be balanced.
The work of Preen and Smith [PS25] is particularly interesting, as it establishes a correlation between the model's parameters and the success of the attack. Furthermore, they also utilized the LiRA in their evaluation. Because these two points present considerable overlap with our work, we will consider their results when analyzing our results in our Discussion.

# 5 Our Approach

In this chapter, we present the methodology applied in this work. We describe how LiRA, initially designed for neural networks, was adapted and applied to random forest models. Additionally, we provide a high-level outline of our approach and changes to the implementations. Then, we conclude this chapter by detailing our final implementation and experimental setup. We also provide a detailed explanation of our data generation process, which supports our results. Our final implementation is also available on our institute GitLab[2].

Using the implementation described in the Background chapter, we successfully implemented LiRA, Figure 5.1, developed by Carlini et al. [CCN$^+$22], and reproduced their results using neural networks trained on the CIFAR-10 dataset. This reproduction serves as the first step for our following implementation of LiRA on random forests.

## 5.1 Reproducing the Likelihood Ratio Attacks results

Figure 5.1 shows the final attack score. Both the online (blue and orange) and offline (green and red) versions are represented in this graph. For each version, a variation with fixed variance exists. The fixed variance version assumes that the variance of the member and non-member distributions is the same for all inputs. This calculation is performed for each distribution of each input, assuming the variance is not fixed. Additionally, we can see a curve representing a loss-based membership inference attack using a global threshold. Because Carlini et al. argue that LiRA outperforms attacks of this type, this attack is included for comparison. The dotted gray diagonal line represents the baseline for randomly guessing. Any attack scoring above this line is successful. Because LiRA aims to reduce the FPR, represented on the x-axis, while maintaining a high TPR, represented on the y-axis, an ideal attack would begin in the top left corner. This indicates a high TPR at a low FPR. The ideal attack would then move horizontally to the top right corner, as the restriction of a low FPR would be relaxed gradually as the graph moves along the x-axis. The Figure 5.1 shows that most attacks start with a moderate TPR that increases

[2]`https://git.uni-luebeck.de/its/theses/bachelor-julius_benedict_niehoff-dp_membership_output_distribution/-/tree/main`

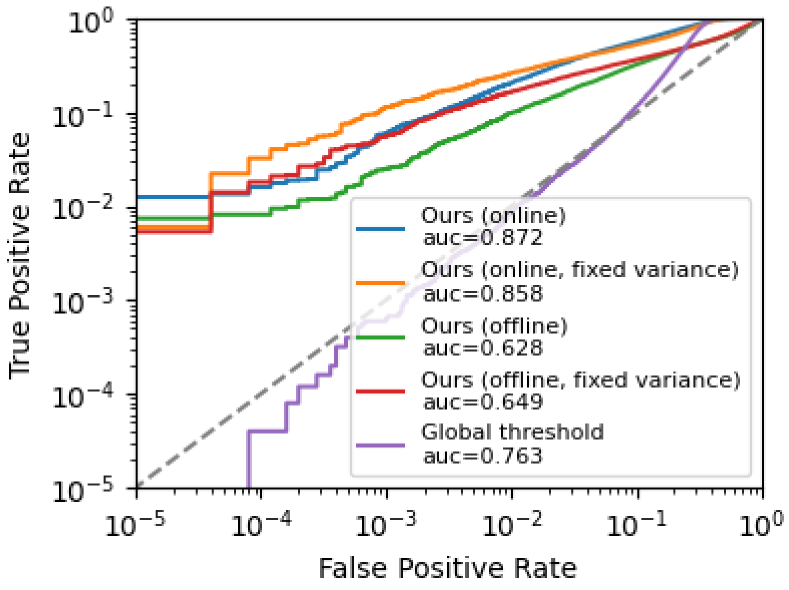


Figure 5.1: Our results for recreating the likelihood ratio attack by Carlini et al. [CCN+22] on CIFAR-10

with the FPR. The global threshold version performs worse than the depicted versions of LiRA, initially falling below the global threshold for a low FPR. Finally, the area under the curve (AUC) for each attack is displayed in the bottom right corner. The AUC measures the likelihood that the attack ranks a randomly chosen member higher than a randomly chosen non-member. A higher score indicates a better attack.

We aim to apply LiRA to random forest models. Carlinin et al. [CCN+22] conduct the attack on a neural network for classification. In our approach, we will utilize random forest classification models.

## 5.2 Implementation Overview

Before tackling the technical implementation, we first discuss the steps necessary to adapt the LiRA to random forests. Additionally, we will also highlight the challenges that we faced during the implementation.

### 5.2.1 Training Random Forest Classifiers

The first step in our approach involves training random forest classifiers. During this and the following steps, we use the scikit-learn library [PVG+11]. Scikit-learn is an open-source Python library that provides simple and efficient tools for machine learning. We

use this library for training and interacting with random forest models.
While we can train random forests on CIFAR-10, for reasons that will be detailed later in the Final Implementation, we switched to a different dataset, MNIST, for our implementation. However, this did not impact our general implementation of random forests.
Initially, we replace all uses of neural networks with random forests. We then ensure that all functions are correctly adapted to expect random forest models, instead of neural networks. This includes replacing queries and other operations with the appropriate version used for random forests.

### 5.2.2 Adapting the Likelihood Ratio Attack to Random Forest Classifiers

When adapting LiRA from neural networks to random forests, we need to make a minor theoretical adjustment in addition to the above-mentioned practical adjustments. While this attack is based on a statistical hypothesis test that is independent of the technical implementation of the target model, the specific values used for these tests need to be adjusted slightly to account for the technical implementation of the model. Random forests do not return a raw logit when queried, unlike a neural network. They instead return the predicted label. This means we can not use the traditional logits for the hypothesis test. To substitute for the lack of raw logits, we instead use the probabilities of each class provided by the random forest. While raw logits are not the same as the probabilities, they serve a similar purpose. The raw logits of neural networks are passed through a softmax function to create the probabilities. The softmax function converts the raw logits into probabilities. Before passing through the softmax function, the raw logits are not bound like probabilities. Although probabilities are not a direct replacement for raw logits, they can still be utilized in our context. This is because the raw logits are converted into probabilities before any LiRA-specific calculations are made. The application of the logit-scaling occurs after the conversion into probabilities by the softmax function. Additionally, the probabilities of random forest are returned in the same shape as the raw logits of neural networks. Retaining the same shape is crucial, as it allows us to pass these values directly to the cross-entropy loss function and other functions that require probabilities, without any further adjustments. This is important as we use the cross-entropy loss for plotting the member-non-member distributions. The ability of random forests to already supply class-based probabilities also means that we do not need to calculate them ourselves from the raw logits using a softmax function.

## 5.3 Final Implementation

This section will provide details for our final implementation. Here we explain how we implement the steps mentioned above. Given the scores of our random forests trained on CIFAR-10, we must consider additional changes to the source code by Carlini et al. [CCN+22], specifically switching to a different dataset.

### 5.3.1 Changing the Dataset from CIFAR-10 to MNIST

To gain a better understanding of the results that led to the switch of datasets, we first provide some background information on the datasets used, namely CIFAR-10 and MNIST.

**CIFAR-10** The CIFAR-10 [Kri12] dataset is used for image classification and consists of 60,000 images grouped into 10 classes. These images are split into a training dataset (50,000 images) and a test dataset (10,000 images). Each image has a shape of $32 \times 32$ pixels and has three color channels. The 10 classes are:

- airplane
- automobile
- bird
- cat
- deer
- dog
- frog
- horse
- ship
- truck

**MNIST** The MNIST [Den12] dataset is also used for image classification and consists of 70,000 images grouped into 10 classes. The images are split into a training dataset (60,000 images) and a test dataset (10,000 images). Each image has a shape of $28 \times 28$ and is a grayscale image. The 10 classes are the digits from 0 to 9. This dataset contains handwritten digits as images.

**Training Random Forest Classifiers on the MNIST Dataset** The solution to improve the performance of our random forest models is to select a new dataset. We decided to use the MNIST dataset [Den12]. We chose this dataset because we know from other works that random forests can achieve good scores on this dataset without the need for extensive model parameter adjustments [ZF18].

### 5.3.2 Implementation Details

In our final implementation, we modify the source code from Carlinin et al. [CCN+22] to align with our desired approach. As previously stated, we use the scikit-learn library [PVG+11] for random forests. To fit the source code to random forests, we replace all usage of neural networks with random forests. While this approach is straightforward for most instances, a few adjustments are necessary when switching to a different machine learning model. While the neural networks used by Carlini et al. are trained by building on a pre-trained model, random forests are usually not pre-trained. This means that we train our random forests without any pre-training. To split the training data into subsets for each shadow model, we keep the original function. However, we adjust the function slightly to work with the reduced dimensions of the MNIST dataset. Following this adjustment, we also adjust all other functions that directly use the training or test data to account for the reduced dimensions. Because the MNIST dataset provides only one dimension of data, whereas the CIFAR-10 dataset provides three, we need to adjust all functions that use the dataset accordingly.

After completing all minor adjustments to fit the MNIST dataset to the code and training random forests, we now focus on making specific adaptations to LiRA. As stated, we use the probabilities for each class as a replacement for raw logits. This replacement works as intended, as demonstrated in our Experiments. Changing to probabilities from raw logits was a seamless transition, as it mostly allowed us to skip the softmax transformation before applying the logit-scaling.
In addition to these changes to the implementation of LiRA, we add more functions to generate some of our results. These additions aim to create the member-non-member distributions that we highlighted in the Background chapter. To create these distributions for a target point in the training dataset, we use the following Algorithm 1.
To create the member-non-member distribution, we query each model for its probabilities of the target point. The probabilities are then passed to the loss function to obtain the loss. Then the loss is saved as a member or non-member, depending on whether the point was part of the model's training data. Using the saved losses of all models, the distributions of member and non-member losses are plotted.
Our Algorithm 1 creates the member-non-member distributions previously described in the Background chapter. However, for our results, we added two additional elements to these plots. For each distribution, member and non-member, handed to the plot function, the function also adds the mean of each distribution to the plot. Additionally, the distance between the means is also calculated and displayed. We will use this metric later in our analysis.

**Algorithm 1:** Creating Member-Non-Member Distributions for a Target Point $x$ Using 250 Shadow Models. This algorithm first calculates the loss of the target point for each model and saves it to one of the two distributions. Then, the distributions are plotted.

```
Input: Target point $x$
Output: Member and non-member distributions
1 member ← [ ]
2 non_member ← [ ]
3 for each $i$ in 1 to 250 do
4     Compute loss $\ell(f(x)_y)$
5     if $x$ is in the training data of model $i$ then
6         member.append($x$)
7     else
8         non_member.append($x$)
9 plot_distribution(member, non_member)
```

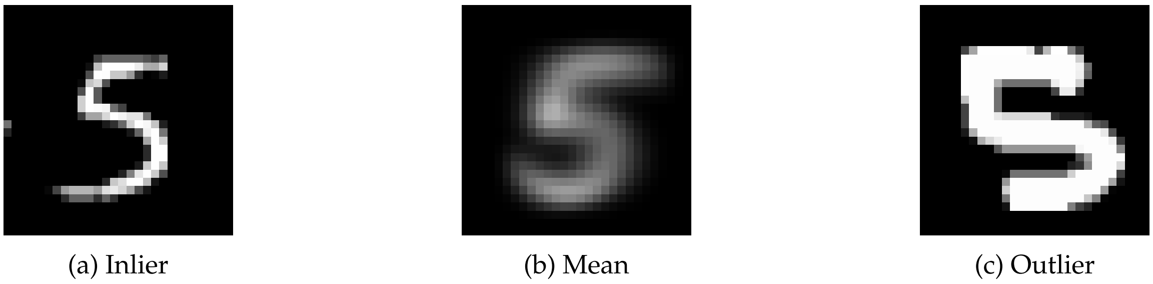


Figure 5.2: The Inlier, Outlier, and mean for Label 5 of the MNIST dataset.

Lastly, we determined specific target points for our analysis. Since MNIST has 60,000 training images, selecting any image at random provides only limited insights, as we do not know whether this image represents an average case for its label or an extreme case. Therefore, we focus on specific points for our analysis. We use the most distinct inlier and outlier for each label.

To determine these points, we first calculate the mean of each label. We can then calculate the points with the greatest or lowest distance to this mean. Those points are the inliers and outliers for their respective labels. While other outliers, points that are far outside the typical values for their class, may exist in the dataset, we focus on these specific furthest outliers from their mean. We have visualized the inlier, outlier, and mean for label 5 in Figure 5.2

# 6 Experiments

In this chapter, we will present our results. We use the approach and implementation described in Our Approach to obtain these results.
First, we will show the training and test scores of our experiments. Then, we present the results of our implementation of LiRA. After that, we showcase the member-non-member distributions we highlighted as an essential part of our analysis. Additionally, we also list the distance between the means of the member and non-member distributions. Finally, we conclude the chapter with a summary.

## 6.1 Results

### 6.1.1 Reproducing the Likelihood Ratio Attack

The reproduction of LiRA proposed by Carlini et al. [CCN+22] was the basis for our approach. Thus, we already displayed our results for this step in Figure 5.1 along with a detailed explanation of the information displayed in the figure.

### 6.1.2 Training and Test Scores

We have conducted multiple experiment runs. Each run consists of an initial training step, where $250$ random forests are trained and tested. Then, the member-non-member distributions, which we defined in the Background chapter, are calculated for the inliers and outliers of each label. Finally, LiRA is conducted, and the results are displayed in a figure. Due to the poor performance of random forest classifiers on the CIFAR-10 dataset, we used the MNIST Dataset for our experiments.
Our initial random forest model achieved training and test scores below $40\%$ on CIFAR-10. After adjusting the model parameters, maximum depth, and the number of estimators, our best model achieved a training score of $73\%$ and a test score of $47\%$. While this represents a significant improvement from our starting point, the model's performance remains too inconsistent. With these scores, a membership inference attack would not yield accurate results.

Table 6.1: Training and Test scores. The training and test scores displayed are for various experiment runs with different parameter values. The rows display the training and test scores at varying numbers of estimators, while the columns show the scores for different maximum depths.

| | Number of Estimators per Forest | | |
|---|---|---|---|
| | 10-50 | 250-300 | 400-500 |
| Maximum depth 7 | 0.9049/0.9068 | - | 0.9157/0.9183 |
| Maximum depth 25 | - | 0.9815/0.9647 | - |
| Maximum depth 50 | 0.9779/0.9574 | - | 0.9818/0.9652 |
| Maximum depth 200 | 0.9805/0.9596 | - | 0.9842/0.9669 |

In Table 6.1, the training and test scores for the most essential experiments are shown. Using the MNIST dataset, we achieve scores ranging from 90% to 98% in training and 90% to 96% in testing. Lower scores are also possible when adjusting the parameters to hinder model performance. We can observe that the scores increase slightly as the number of estimators increases. The maximum depth has a much larger influence on the test scores. An increased maximum depth leads to a significant increase in scores. However, this increase is not linear. As shown in the table, the increase from maximum depth 7 to maximum depth 25 is much larger than the increase from 25 to 50. Additionally, we add an experiment to the table using a maximum depth and number of estimators in the middle of our selected values to showcase an additional data point for Table 6.1.
These experiments will be discussed more thoroughly later in the chapter.
For our analysis, we decided to use the following values for maximum depth and number of estimators:

- **Maximum Depth** 7 and 50; The values for maximum depth were selected by analyzing the results of multiple experiment runs. We selected 7 as our lower bound because we want to highlight the difference in the model's behavior for a low and high maximum depth. 7 was chosen to provide a sufficient distance from our higher bound, without compromising the model's performance by further decreasing the maximum depth. Our selection of the higher value of 50 was made because values higher than that had minimal effect on the training and test scores. There was also no significant impact on the member-non-member distributions for values beyond 50. Thus, we opted to use 50 to preserve resources.

- **Number of Estimator ranges** $10 - 50$ and $400 - 500$; To determine the ranges for the number of estimators per random forest classifiers, we observed that while there is some impact on the training and test scores, as well as the member-non-member distributions, this impact was much smaller than that of the maximum depth. Thus, we selected values that represent similar extremes to the maximum depth. A lower bound that does not have an overwhelming negative impact on the model's performance. The higher bound discourages further increases due to the minimal return on invested computational resources and time.

### 6.1.3 Likelihood Ratio Attack on Random Forest

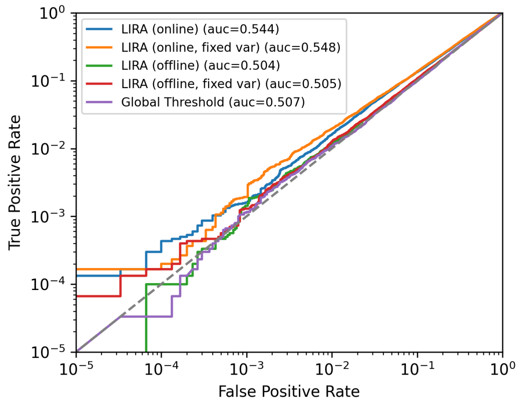
Figure 6.1 shows the results of the LiRA on random forests. The four experiments use the values for the parameters mentioned above, namely maximum depth of $7$ and $50$, and the number of estimators ranging from $10$ to $50$ and $400$ to $500$, for their random forest models. Figure 6.1 (a) shows random forests with a maximum depth of $7$ and $10-50$ estimators. For Figure 6.1 (b), the estimators are increased to $400-500$. In Figure 6.1 (c), we increase the maximum depth to $50$ and keep the estimators at $10-50$. Additionally, Figure 6.1 (d) shows random forests with a maximum depth of $50$ and $400-500$ estimators.

In Figure 6.1 (a), we can see that the attack is not successful. For this experiment, we do not achieve a low FPR at a high TPR. All the different attack scenarios shown propagate toward the dotted diagonal. As previously explained in Background, this line indicates a success rate of $50\%$, equivalent to random guessing. Additionally, we can see that the AUC in the figures does not go above $55\%$ for all variants of the attack. The AUC also shows that both offline versions of the attack have a lower average success rate, $50.4\%$ and $50.5\%$, than the global threshold attack with $50.7\%$. When analysing all of the attack success rates at a low FPR of $0.001\%$, all the attacks have a very low TPR around $0.001\%$.

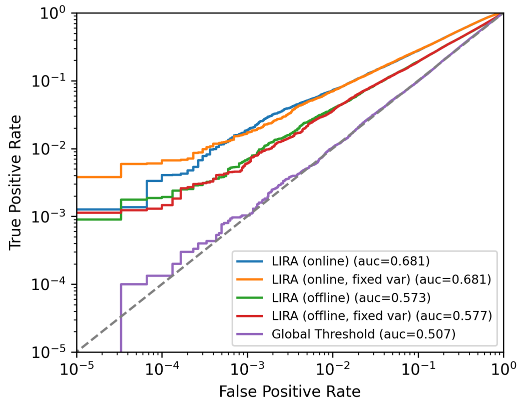
We achieve slight improvements to our results in Figure 6.1 (b) and (c). For Figure 6.1 (b), we can see that all versions of the attack have an improved average success rate that does not fall below the global threshold version. The attacks also achieve a TPR around $0.1\%$ at a FPR of $0.001\%$.

In Figure 6.1 (c), we can see that our attacks are improved when directly compared to (b). For this scenario, the AUC for all attacks is improved by $13.2\%$ to $20.7\%$. This includes the global threshold version's success rate, which is improved by $17.2\%$. Most of the attacks also have a slightly better TPR at a low FPR of $0.001\%$. However, the online version with a fixed variance (orange) performs worse than the other version of the attack. It has a lower initial TPR than all other variants and closely follows the global threshold from a TPR of $0.01\%$ to $10\%$. After that, this version displays a small jump and is again in line with the other versions.

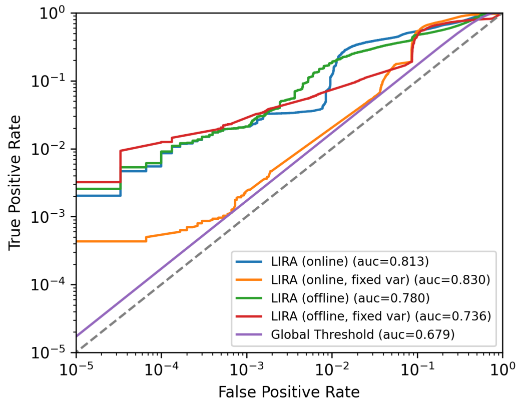
The most considerable improvement compared to our initial Figure 6.1 (a) is in Figure 6.1 (d). While the initial TPR at a low FPR of $0.001\%$ is slightly worse than in (b) or (c), there is a significant jump at a FPR of $0.01\%$. This jump only occurs for versions of the attack that do not use a fixed variance. It also brings this attack scenario ahead of the original LiRA conducted on neural networks, as both versions now achieve a TPR of $10\%$ or above at a very low FPR of $0.01\%$.



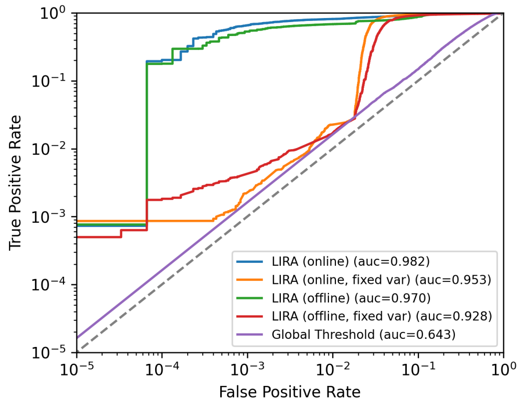
(a) LiRA on random forest with parameters maximum depth 7, estimators 10-50



(b) LiRA on random forest with parameters maximum depth 7, estimators 400-500



(c) LiRA on random forest with parameters maximum depth 50, estimators 10-50



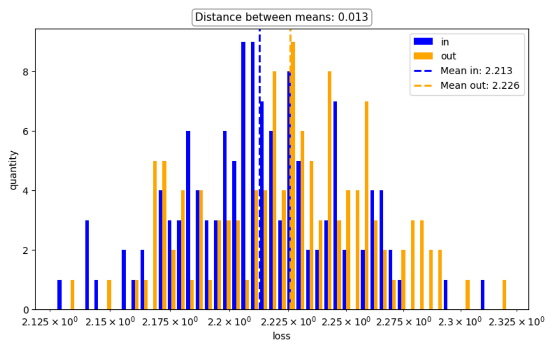
(d) LiRA on random forest with parameters maximum depth 50, estimators 400-500

Figure 6.1: Multiple LiRAs on random forest classifiers with different parameters. The pairings of the values for the parameters are selected from a maximum depth of $7$ and $50$, and the number of estimators ranging from $10$ to $50$ and $400$ to $500$.

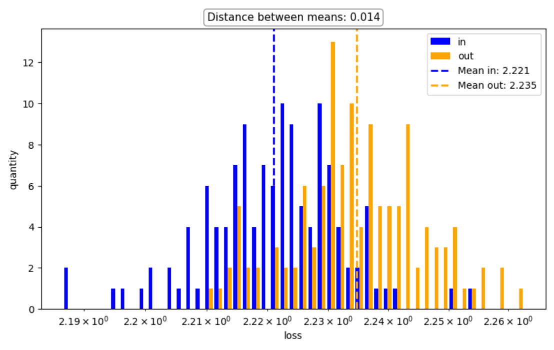
### 6.1.4 Member-Non-Member Distributions

The member-non-member distributions shown in this section will focus on the specific inliers and outliers defined in Our Approach. Selecting these inliers and outliers for each label allowed us to assess the attack's success more precisely. We determine these points by calculating the mean value for each of the ten labels contained in the MNIST dataset. We then calculate the closest and farthest points from the mean for each label.
Additionally, we also add the mean for each distribution and the distance between those means. The mean for each distribution is calculated by adding all the points together and then dividing the sum by the number of points. This enables us to measure the distance between the means of the distributions. The mean for each distribution is not added as a point to the distributions. We add the mean by displaying a dotted line to represent it.
To create the distributions, we use the same approach detailed in Our Approach.
Figure 6.2 shows the loss-based member-non-member distributions for the outlier of label 5. The bottom row shows the distributions separated, matching the example in Figure 3.2 from Carlini et al. [CCN+22], which we discussed in the Background chapter. Despite this sample being an outlier for its label, the top row shows the sample as an inlier with both distributions, member and non-member, overlapping. We can also see a difference in the distance between the means of the two rows. For the top row, the distances are $0.013$ and $0.014$. The distances in the bottom row are $0.379$ and $0.313$.

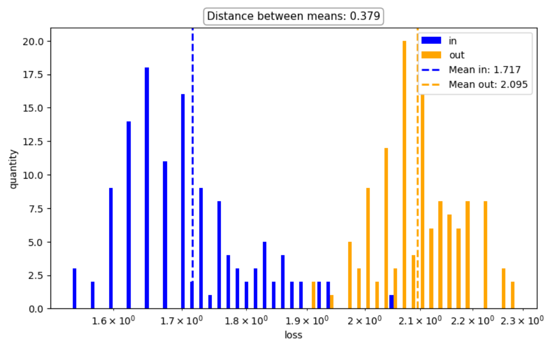
The reverse can be seen in Figure 6.3. This figure displays the same data as Figure 6.2, however it does not display the same target point. Instead, the inlier of the same label is the target point for Figure 6.3. Here, the top row matches the example in Figure 3.2. In the top row, the distances are $0.005$ and $0.007$. The distances in the bottom row are $0.105$ and $0.094$. However, in the bottom row of Figure 6.3, the inlier displayed visually matches the outlier, rather than an inlier.
To determine whether a point is an inlier or outlier, the addition of the mean of each distribution and the distance between those means is necessary. If we view the distributions without the context of the displayed points and try to interpret whether the point is an inlier or outlier by visually comparing it to the example from Carlini et al. [CCN+22], we would not always be correct. Because the points can appear as inliers or outliers in the visual graph, independent of whether they are actual inliers or outliers, we cannot determine this without an additional metric.
In Table 6.2, the distance between means showcases the difference between outliers and inliers for the different values of the model parameters. Table 6.2 shows the distance between means of the member and non-member distributions. The values in the table are in line with the previous observation from Figure 6.2 and Figure 6.3. An increase



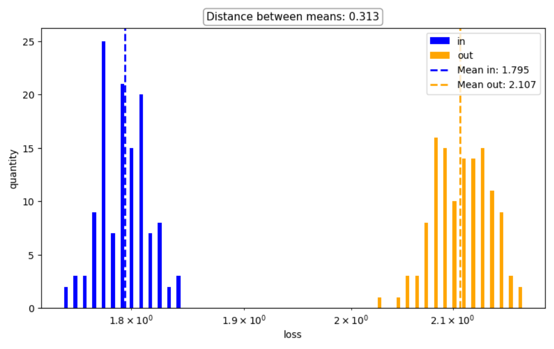
(a) Member-Non-Member Distribution of the loss for the outlier of label five on a random forest with parameters maximum depth 7, estimators 10-50



(b) Member-Non-Member Distribution of the loss for the outlier of label five on a random forest with parameters maximum depth 7, estimators 400-500



(c) Member-Non-Member Distribution of the loss for the outlier of label five on a random forest with parameters maximum depth 50, estimators 10-50



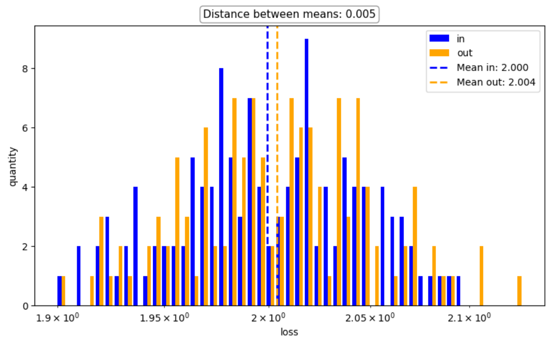
(d) Member-Non-Member Distribution of the loss for the outlier of label five on a random forest with parameters maximum depth 50, estimators 400-500

Figure 6.2: Member-Non-Member Distribution of the loss for the outlier of label five on a random forest. Each subplot displays the results for a different random forest, trained with varying parameter values.

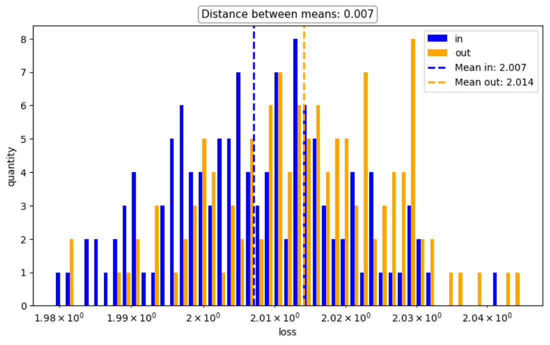
in the values of the model parameters also directly increases the distance between means. Additionally, when viewing the means isolated from the visualization of the distributions, the inliers and outliers are still separated, further highlighting that the figures alone cannot provide accurate data.

## 6.2 Summary

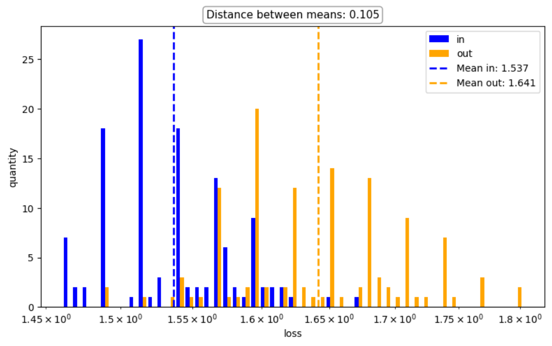
We successfully reproduced LiRA, as shown in Figure 6.1. The results in that figure show that increasing the model parameters, maximum depth, and number of estimators directly enhances the success of LiRA.



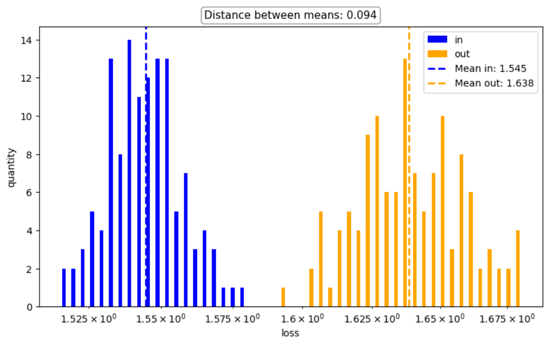
(a) Member-Non-Member Distribution of the loss for the inlier of label five on a random forest with parameters maximum depth 7, estimators 10-50



(b) Member-Non-Member Distribution of the loss for the inlier of label five on a random forest with parameters maximum depth 7, estimators 400-500



(c) Member-Non-Member Distribution of the loss for the inlier of label five on a random forest with parameters maximum depth 50, estimators 10-50



(d) Member-Non-Member Distribution of the loss for the inlier of label five on a random forest with parameters maximum depth 50, estimators 400-500

Figure 6.3: Member-Non-Member Distribution of the loss for the inlier of label five on a random forest. Each subplot displays the results for a different random forest, trained with varying parameter values.

After that, we showcased our training and test scores in Table 6.1. We included scores for several experiments in the table, focusing on those most essential for our subsequent analysis. The most important scores are those for experiments with the parameter pairings of 7 and 50 for the maximum depth and $10 - 50$ and $400 - 500$ for the Number of estimators. Next, we presented the member-non-member distributions we created. This included both an inlier sample, Figure 6.3, and an outlier sample, Figure 6.2. For each sample, we created four distributions, one for each of the selected model parameter pairings. In the member-non-member distributions, we added the mean of each distribution and the difference between those means. Both figures clearly show that an increase in model parameters also directly increases the distance between the means of the member

Table 6.2: Distance between the means of member-non-member distributions. The table displays multiple experiments with varying parameters, including Maximum Depth (MD) and Estimators (E).

| | MD 7, E 10-50 | MD 7, E 400-500 | MD 50, E 10-50 | MD 50, E 400-500 |
|---|---|---|---|---|
| Label 0 Outlier | 0.018 | 0.026 | 0.366 | 0.305 |
| Label 0 Inlier | 0.004 | 0.000 | 0.001 | 0.002 |
| Label 1 Outlier | 0.017 | 0.017 | 0.485 | 0.350 |
| Label 1 Inlier | 0.001 | 0.001 | 0.001 | 0.000 |
| Label 2 Outlier | 0.038 | 0.038 | 0.472 | 0.338 |
| Label 2 Inlier | 0.001 | 0.006 | 0.012 | 0.013 |
| Label 3 Outlier | 0.004 | 0.009 | 0.442 | 0.348 |
| Label 3 Inlier | 0.001 | 0.002 | 0.005 | 0.003 |
| Label 4 Outlier | 0.012 | 0.018 | 0.246 | 0.202 |
| Label 4 Inlier | 0.001 | 0.000 | 0.007 | 0.004 |
| Label 5 Outlier | 0.013 | 0.014 | 0.379 | 0.313 |
| Label 5 Inlier | 0.005 | 0.007 | 0.105 | 0.094 |

and non-member distributions. This gap widens as the model's parameters and the success rate of the attack increase. These findings are further underlined in Table 6.2 by the increasing distances between means for other labels that do not have their distributions displayed.

# 7 Discussion

In this chapter, we discuss the results of applying LiRA, as proposed by Carlini et al. [CCN+22], to random forest models. The chapter aims to interpret and analyze the results presented in Experiments. For this, we are focusing on the influence of the distance between the means of the member and non-member distributions, as presented in our Experiments. Additionally, we compare our results to those of Preen and Smith [PS25].

## 7.1 Application of the Likelihood Ratio Attack to Random Forest Classifiers

We successfully adapted the LiRA to random forests. This shows that the statistical basis for LiRA can be safely transferred between different model architectures. The hypothesis test using the Neyman-Pearson Lemma [NPP33] delivers the same results for neural networks and random forests.

Although we did need to change the dataset, this change is not due to LiRA. LiRA can be applied to random forest classifiers trained on the CIFAR-10 dataset. Due to the poor training and test results of random forests on this dataset, the attack's success would have been significantly impacted. This shows a shortcoming of LiRA. To effectively attack a model with LiRA, the model must be sufficiently trained to produce usable results for the attack.

The performance of our trained random forest classifiers on the CIFAR-10 and MNIST datasets aligns with current research [XKL+21],[ZF18]. They further state that random forests take in considerably less information compared to neural networks. This is due to random forest only considering the value of a data point, whereas neural networks can also include the relative position, given their internal reasoning structure. Choosing the MNIST dataset enabled us not only to apply the attack to random forests but also to consider a range of attack scenarios against models with various degrees of overfitting. This helped us to analyze the attacks' behavior for different scenarios further.

## 7.2 Relationship between Maximum Depth and Likelihood Ratio Attack Success

When analyzing the different experiments at multiple maximum depths, it is clear that an increased maximum depth leads to an increase in the attack's success. We observed that this is not a linear increase. When considering the training and test scores, as displayed in Table 6.1, we can assume that there are one or more categories of models along this increase. Our chosen maximum depth of 7 represents a category for a model that is well-generalized and achieves a test score higher than the training score, thereby displaying no overfitting, as determined by the generalization gap established by Yeom et al. [YGFJ18]. The maximum depth of 50 best fits a severely overfitted model.

**Overfitting** Overfitting has already been identified as a significant contributing factor to the success of membership inference attacks. Combining this knowledge with our experiments enables us to conclude that the increased success we observe in the attack with an increased maximum depth or number of estimators per random forest is likely linked to the random forest classifier models being overfitted. Increasing the parameters beyond a certain point enables a single tree to memorize one or, with a sufficiently large increase, multiple data points. We can observe this trend in our distributions of members and non-members. For trees with a low maximum depth, the distributions of members and non-members overlap. For trees with a greater maximum depth, the reverse holds. While this would be consistent with samples being outliers or inliers, we observed that this phenomenon occurs independently of whether the samples are outliers or inliers.
Figure 7.1 shows a direct comparison between outliers and inliers at different maximum depths. The top row of the figure shows the distributions for the same data point, an inlier for label 5. We can see that Figure 7.1 (a) aligns with our expectations for inliers, as described by Carlini et al. [CCN+22].
In contrast, this does not hold for the increased model parameters in Figure 7.1 (b). We can see that the distributions do not overlap for the increased parameters, presenting as a potential outlier. Additionally, we measured this by using the means of the distributions and calculating the distance between them.
For the outliers, the reverse holds. The outlier fits the expectations at higher model parameters, Figure 7.1 (d), but not at lower model parameters, Figure 7.1 (c).

## 7.3 Member and Non-Member Distributions

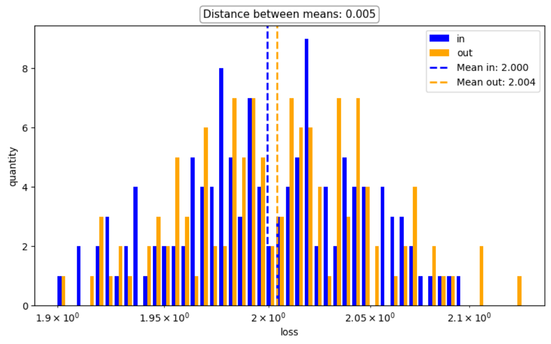
Because the member and non-member distributions can overlap or be completely apart, independent of whether a data point is an inlier or outlier, but rather determined by the model parameters used to train the classifier, a more accurate measure for outliers and inliers is the distance between means. We have added this distance to our diagrams and provided an overview in Table 6.2, which showcases the distances for both outliers and inliers of multiple labels. However, we still need to consider that the points shown in Figure 6.2 and Figure 6.3 are the most extreme outliers and inliers for their respective labels.
When directly comparing these figures to Figure 6.1, we can see that despite the member-non-member distributions for low maximum depth looking nearly identical with very similar distances between means, we achieve a much more successful attack on the model with more estimators per random forest. This means that while the data points shown in Figure 6.2 and Figure 6.3 can give an initial indication regarding the attack's success, they alone cannot fully predict it.
If we were to use the values of distances between means for a single label from Figure 7.1, we can create another metric for our observations. By matching the distances between means for identical model parameters, maximum depth, and number of estimators, we can develop a range of distances. The ranges for the lower model parameter are from $0.005$ to $0.013$, and for the higher model parameter, they are from $0.094$ to $0.313$. Here we can now see that the data points in between these two extremes can have a much wider range of distances between their member and non-member distributions for higher model parameters. Potentially allowing an attacker to use these ranges as an indicator for overfitting. A more accurate measure would be to calculate the distances between means for all data points for each label and create an average. This average can then be used in conjunction with the model parameters to determine a threshold at which the model is likely to overfit, thereby granting an advantage to an attack using loss-based membership inference.

## 7.4 Comparing our Results to A Hierarchical Approach for Assessing the Vulnerability of Tree-Based Classification Models to Membership Inference Attack

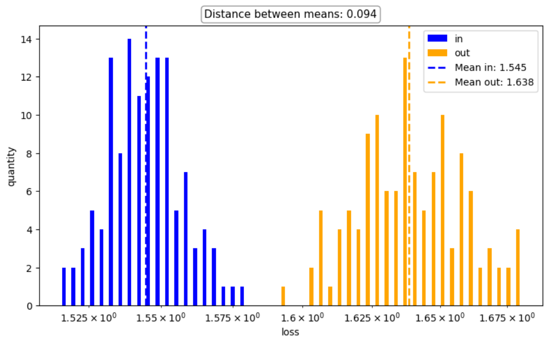
Preen and Smith [PS25] published their findings while we were finishing up this thesis. Because there is a considerable overlap between their work and ours, we will compare the results of the two. In their work, Preen and Smith target multiple tree-based classifiers, in-

cluding random forests trained with the scikit-learn library. They also include numerous membership inference attacks, multiple evaluation metrics, including LiRA, and a high TPR at a low FPR across multiple datasets. Doing so covers most of our experimental setup. However, Preen and Smith aim to draw comparisons across different tree-based models, membership inference attacks, and evaluation metrics. They did not exclusively focus on applying LiRA to random forest models.
By comparing multiple attacks over different models, datasets, and evaluation metrics, they identify risk factors that contribute to the success of membership inference attacks. Preen and Smith [PS25] identified a high maximum depth and a larger number of estimators as high-risk factors, which increase the success of membership inference attacks, regardless of the evaluated model. This aligns with our findings. We also found that increasing these parameters increases the success of LiRA.

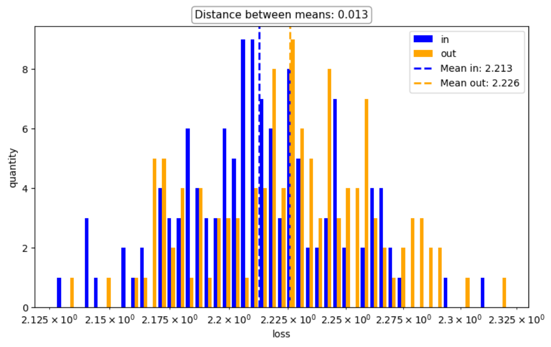
While Preen and Smith [PS25] conducted an extensive evaluation of over 70,000 parameter combinations and datasets, they did not investigate the distributions of members and non-members, as we did. Their focus was on the structure of tree-based classification models and their susceptibility to membership inference attacks. In contrast, we used our results to connect the changes in the selected parameters with our new metric. For this metric, we analysed the member-non-member distributions. By identifying changes in these distributions and the distance between them, these differences correlate with other changes observed at specific parameter values, maximum depth, and the number of estimators. A low distance between means occurs for the same parameters as a low TPR at a low FPR. Our best attack is on the same models as those with the highest distance between means. Thus, we can connect our new metric directly to the attack success.



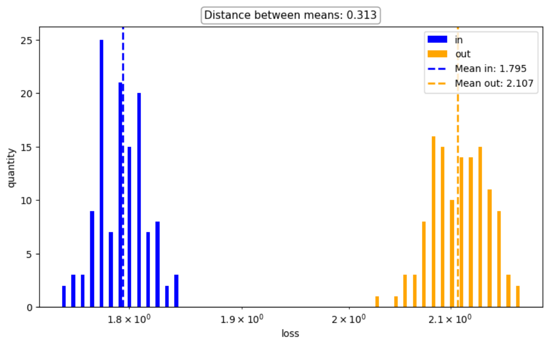
(a) Member-Non-Member Distribution of the loss for the inlier of label five on a random forest with parameters maximum depth 7, estimators 10-50



(b) Member-Non-Member Distribution of the loss for the inlier of label five on a random forest with parameters maximum depth 50, estimators 400-500



(c) Member-Non-Member Distribution of the loss for the outlier of label five on a random forest with parameters maximum depth 7, estimators 10-50



(d) Member-Non-Member Distribution of the loss for the outlier of label five on a random forest with parameters maximum depth 50, estimators 400-500

Figure 7.1: This figure shows the member-non-member distributions for the inlier and outlier of label five. Top row: loss of inlier for label 5, Bottom row: loss of outlier for label 5, left Column: maximum depth 7, estimators 10-50, right Column: maximum depth 50, estimators 400-500

# 8 Conclusion

In this thesis, we successfully adapted LiRA to random forest classifiers. While this was expected, it also demonstrated that the architecture-independent nature of the statistical approach in LiRA can be transferred between different model architectures. Although we decided to change the dataset we were using to train the random forests to the MNIST dataset, we still achieved success. This change does not reflect negatively on our adaptation or LiRA itself. Changing the dataset was necessary due to the poor performance of random forests on the CIFAR-10 dataset.
The successful implementation of our approach enabled us to achieve the results we had theorized. We were able to leverage the member-non-member distributions for our evaluation.

By creating multiple experiments with varying parameter values, we were able to link several metrics together. First, we demonstrated that increasing the model's maximum depth and number of estimators also increases the model's generalization gap. This gap, formalized by Yeom et al. [YGFJ18], can help determine whether a model is overfitted. This observation was highlighted in Table 6.1. Following this observation, we demonstrated that models with higher parameter values also allowed for LiRAs with better results than those with lower parameter values. We displayed this in Figure 6.1. By combining these observations, we can connect the same increase in parameter values to both overfitting and LiRA success.
While we only investigated LiRA on random forest classifiers, Preen and Smith [PS25] investigated multiple attacks on multiple tree-based models. Their results align with ours, as they were also able to demonstrate that higher maximum depth and an increased number of estimators increased both the models' overfitting and the success of membership inference attacks, such as LiRA.

Additionally, we also created a new metric. We used the member and non-member distributions and added the distance between their means. We found that these distances increase with the parameter values, maximum depth, and number of estimators, similar to the generalization gap and the success of LiRA. These distances have been visualized for a single inlier, Figure 6.3, and an outlier, Figure 6.2. To better summarize these distances, we added them to Table 6.2. Here we can see that our proposed metric may be a viable tool for analyzing a model's overfitting or the success of possible membership inference attacks. Because these distances increase with the parameter values, which is the same behavior we and Preen and Smith [PS25] have already demonstrated for overfitting and the success of LiRA.

In summary, we were able to confirm our theories by linking several metrics to an increase in the parameter values of random forest classifiers. While these metrics may already be known to be contributing factors, such as overfitting, we also introduced the distance between the means of the member and non-member distributions for a specific point.

# 9 Outlook

While this thesis has demonstrated the practical application of the LiRA on random forest models, several areas for deeper investigation and broader application remain open.

The distance between the means of the member and non-member distributions has the potential to accurately represent overfitting in random forest models, in addition to the already existing generalization gap. For this, more data points need to be analysed for more parameter values. An extensive study may be able to link overfitting in random forests to a threshold or range in the distance between the distributions of members and non-members.
With such a threshold, further studies using loss-based membership inference attacks may be able to utilize this threshold to determine the attack's success before executing it. By simply computing the model's loss for several data points to obtain an estimate of the average distance between means, this estimate can be used as a prediction for the success of the attacks.

The distance between means could also be used as a training metric. When training new classifiers, evaluating potential overfitting with this metric can indicate when to halt increasing essential parameters, such as the maximum depth. This way, overfitting can be prevented during the training process. The distance between means can be leveraged more precisely during training when compared to an attack scenario. As these distances provide more information, if the whole dataset is known, they can be used to more accurately classify the observed distances as inliers or outliers, relative to the mean of their label.
Finally, applying LiRA to other models would be another area of research. This topic has already been explored in other works, such as that of Preen and Smith [PS25]; however, there are far more model types that haven't been extensively covered than tree-based classifiers.

## References

[AA25] Fatemeh Akbarian and Amir Aminifar. Membership inference attack in random forests. pages 455–460, 01 2025.

[BNL12] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 1467–1474, 2012.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[CCN$^+$22] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles, 2022.

[CW17] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks, 2017.

[Den12] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[GBC16a] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[GBC16b] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[HSS$^+$22] 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), September 2022.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2 edition, 2009.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[Kri12] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[MJ18] Matiur Rahman Minar and Jibon Naher. Recent advances in deep learning: An overview, 2018.

[MP43] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

[NPP33] Jerzy Neyman, Egon Sharpe Pearson, and Karl Pearson. Ix. on the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231(694-706):289–337, 1933.

[PGM$^{+}$19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035. Curran Associates, Inc., 2019.

[PS25] Richard J. Preen and Jim Smith. A hierarchical approach for assessing the vulnerability of tree-based classification models to membership inference attack, 2025.

[PVG$^{+}$11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[Qui86] J. R. Quinlan. Induction of decision trees. In *Machine Learning*, volume 1, pages 81–106. Springer, 1986.

[SRG24] Shlomit Shachor, Natalia Razinkov, and Abigail Goldsteen. Improved membership inference attacks against language classification models, 2024.

[SSSS17] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models, 2017.

[VRD09] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[XKL$^{+}$21] Haoyin Xu, Kaleab A. Kinfu, Will LeVine, Sambit Panda, Jayanta Dey, Michael Ainsworth, Yu-Chung Peng, Madi Kusmanov, Florian Engert, Christopher M. White, Joshua T. Vogelstein, and Carey E. Priebe. When are deep networks really better than decision forests at small sample sizes, and how?, 2021.

[YGFJ18] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting, 2018.

[ZBH$^+$17] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2017.

[ZF18] Zhi-Hua Zhou and Ji Feng. Deep forest. *National Science Review*, 6(1):74–86, 10 2018.