


Investigating the Behavior of Predicate Singling Out Attacks

Thomas Sedlmeyr 

TUM School of Computation, Information and Technology - Informatics, Technical University of Munich

 thomas.sedlmeyr@tum.de

September 21, 2024

Abstract — Singling out is one of the concrete privacy violations acknowledged by the General Data Protection Regulation (GDPR) and plays a significant role in discussions about data protection risks. Recent research has introduced the concept of predicate singling out (PSO), where an adversary identifies a unique record in a dataset by exploiting the output of a data-release mechanism. This occurs when the adversary finds a specific condition, or predicate, that matches exactly one data sample with a probability significantly higher than a statistical baseline. A widely used approach to minimize and control the exposure of private information when publishing datasets is to release synthetic data using differential privacy, which is a formal technique to provide provable privacy protection.

In the first part of our study, we conduct an empirical analysis to assess the success rate of singling-out attacks on synthetic datasets, with a focus on how the chosen privacy budget impacts the effectiveness of these attacks. In the second part, we demonstrate that trained decision trees are especially susceptible to singling-out attacks. We introduce an attack method that identifies vulnerable nodes and leverages the corresponding decision paths to construct predicates capable of isolating a specific sample in the dataset. To mitigate this vulnerability, we also introduce an algorithm that prunes an already trained decision tree, thereby enhancing the robustness against singling-out attacks. The source code needed to reproduce our experiments can be found here: https://github.com/TUM-AIMED/PSO_attacks.

1 Introduction

In recent years, deep learning models have revolutionized numerous fields, achieving remarkable performance in tasks such as image recognition and natural language processing. A key driver of these advancements has been the availability of large datasets. However, in privacy-sensitive domains such as healthcare, there are only a small number of datasets available due to privacy concerns. One potential solution to

this issue is using generative models, which learn the underlying distribution of sensitive data and sample synthetic datasets from this distribution, such as diffusion models. Ideally, these synthetic datasets would maintain the overall utility of the original data while containing only a fraction of private information. Nevertheless, it has been demonstrated that these generative models can leak sensitive personal data [10, 3].

While generating synthetic data alone offers no guaranteed protection against information leakage from the original dataset, incorporating differential privacy into the process is essential to ensure provable privacy guarantees [6]. Differential privacy has been mathematically proven to provide Predicate Singling Out security [4]. However, even mathematically proven guarantees are not always sufficient, as a significant gap between theoretical privacy guarantees and actual privacy leakage due to implementation mistakes has been demonstrated [1]. This highlights the need for empirical evaluations to assess the robustness of these algorithms. Therefore, we aim to analyze the success rate of singling-out attacks on synthetic datasets, focusing on how the chosen privacy budget influences the effectiveness of these attacks. Predicate singling-out attacks are concerning as recent work [5] demonstrates that predicate singling-out attacks can lead to the identification of individuals within a dataset. This highlights the significant privacy risks, as isolating a person in a dataset can compromise their anonymity and expose sensitive information.

Furthermore, we demonstrate the susceptibility of decision trees to PSO attacks. Decision trees are widely used for tabular data due to their ease of training and interpretability. However, this interpretability also makes them particularly vulnerable, as they reveal a significant amount of information about the underlying data, which can be exploited in attacks. We train decision trees with different hyperparameters on multiple dataset subsets derived from three distinct datasets to explore their vulnerability to predicate singling-out attacks. By analyzing the trained models, we identify specific vulnerable nodes that can be leveraged to

construct valid predicates for executing a successful singling-out attack. Furthermore, we present a pruning algorithm that removes all vulnerable nodes from the tree, to prevent our proposed singling out attacks.

2 Related Work

One popular approach for synthesizing privacy-preserving datasets is PrivBayes [18]. The data generation process with PrivBayes is based on taking the input dataset D and constructing a Bayesian network from it. This network has two main functions: (i) it accurately models the relationships among the attributes in D , and (ii) it facilitates the approximation of the data distribution using a set of low-dimensional marginals P derived from D . Once the Bayesian network is trained, noise is injected into each marginal in P to ensure that differential privacy is upheld. The noisy marginals, along with the Bayesian network, are then used to approximate the original data distribution. In the final step, PrivBayes produces a synthetic dataset by sampling from this approximated distribution, which is then released. A key advantage of PrivBayes is its ability to address the challenges associated with high-dimensional data by introducing noise at the level of low-dimensional marginals P , rather than across the entire high-dimensional dataset D [18].

Various types of attacks have been conducted against synthetic datasets. One type of attack are the so-called membership inference attacks (MIA), where an attacker attempts to determine whether specific data records were included in the training process of the generative model [9]. In a previous study [1], MIA attacks were used to demonstrate that the implementation of synthetic data generation with differential privacy was flawed, as it leaked significantly more information than the allocated privacy budget should have allowed.

When it comes to attacks against decision trees, previous work has been focused on performing reconstruction attacks, where the goal is to infer or reconstruct the underlying data used to train the decision tree [7, 8]. These attacks typically aim to create a probabilistic representation of the dataset that was used to train the decision tree. To the best of our knowledge, no research has been conducted to perform predicate singling-out attacks on trained decision trees.

3 Attack Scenarios

3.1 Attack Against Synthetic Datasets

We analyze scenarios where a malicious entity, i.e. an attacker, attempts to isolate instances in a dataset. We assume that the dataset X , which the attacker targets, is sampled from a larger population dataset D , which is public and free of any duplicate samples. We also assume that X is sampled without replacement, and we can write $X \subset D$. The person who publishes the synthetic data, to whom we will refer as the data publisher, trains a generative model M_θ in a privacy-secure manner. The data publisher then samples synthetic data from M_θ and publishes it.

We investigate two main types of attacks: black-box and white-box attacks. In the white-box attack scenario, the attacker has access to both D and M_θ and tries to find a predicate that singles out a specific row in X . For the black-box attack, the attacker utilizes D and the synthetic dataset Y to find this predicate.

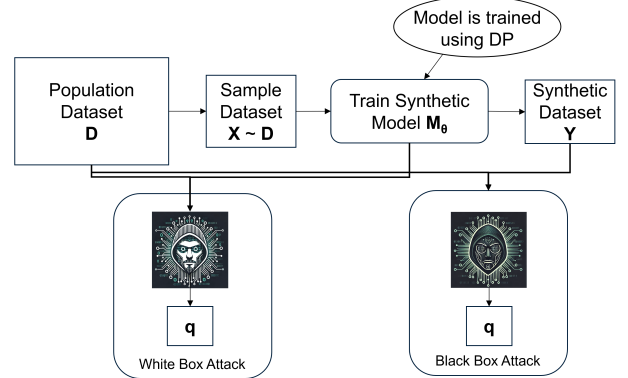


Figure 1 For the white box attack, the attacker utilizes the population dataset D and the parameters of the trained model M_θ . For the black box attack, the attacker uses D and the synthetic dataset Y

We use the following as a baseline to interpret the results of our attacks against synthetic datasets. In this baseline scenario, the attacker selects a random sample $r \in D$ and uses it to construct a predicate by checking that each attribute matches the corresponding attribute of r . If the attacker is fortunate and selects an r such that $r \in X$, the resulting predicate would isolate exactly one sample, assuming that X contains no duplicates. Therefore, the probability of singling out a record in X with this simple baseline attack depends on $|X|$ and $|D|$ and can be computed with:

$$P_{SO} = \frac{|X|}{|D|} \quad (1)$$

Our proposed attack works on increasing the probability that we choose an r that is contained in X . We achieve this by executing multiple membership inference attacks (MIAs) against M_θ in a white-box scenario or against Y in a black-box scenario. The goal of these MIAs is to identify a sample x_T that was very likely used when training M_θ . If the MIA indicates that $x_T \in X$, we construct a predicate that isolates x_T within X by asserting that each attribute matches the corresponding attribute of x_T .

For selecting the record for our attack, we leverage the fact that $X \subset D$ and focus on finding records $r \in D \wedge r \in X$. To obtain this information, we employ shadow modeling techniques [9, 12]. The core concept of this method involves generating pairs of datasets that differ by only a single sample, training a generative model on each dataset, and then sampling a synthetic dataset from each model. Finally, a meta classifier is trained to detect the membership of x_T , using features generated from the synthetic dataset Y for the black-box attack or features computed from the synthetic model M_θ when performing a white-box attack.

To compute the features used for training the meta-classifier, we adopted the approach outlined by [1]. For the black-box attack, the features consist of responses to queries targeting x_T , derived from the shadow synthetic datasets. For the white-box attack, we leverage the joint conditional probabilities and the adjacency matrix of the trained PrivBayes model as features.

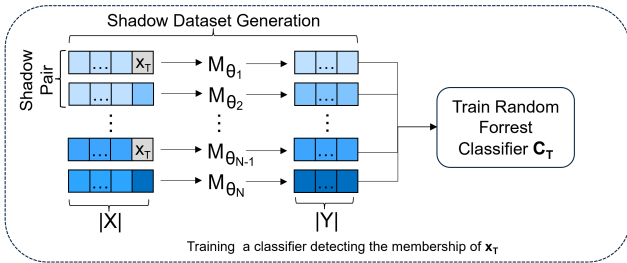


Figure 2 The attacker generates multiple shadow pairs, which are datasets sampled from D , which only differ in a single sample. For each of these datasets, the attacker then trains a synthetic model and generates a synthetic dataset. For the white box attack, features are generated from the trained synthetic models M_{θ_i} for the black box attack the features are generated from the synthetic datasets Y_i . Finally, a classifier is trained to detect the membership of the record x_T given these features.

Relying on a single classifier to predict the membership of x_T is insufficient, as there is no guarantee that $x_T \in X$. To address this, we extend the approach by constructing shadow datasets for multiple targets, generating a set of n classifiers, each tasked with detecting the membership of a different target. For selecting the target records for the attack, we first sort the records in D by their vulnerability [15] and choose the top n most vulnerable ones.

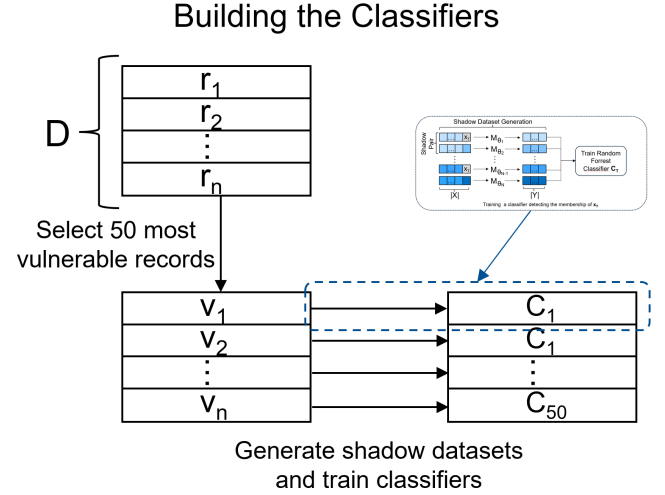


Figure 3 The attacker constructs shadow datasets for each of the 50 most vulnerable records and trains 50 classifiers, each capable of identifying the membership of a single record within the synthetic dataset.

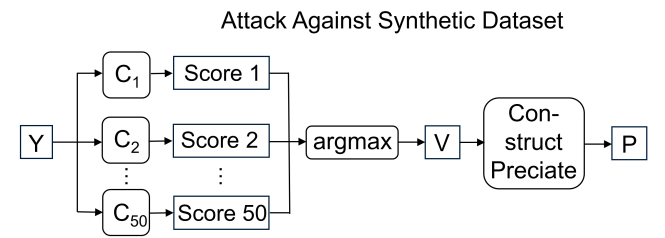


Figure 4 When performing the singling-out attack, the attacker applies each classifier to the synthetic dataset and uses the record from the classifier that yields the highest score to construct the predicate.

To estimate how many classifiers are needed to ensure at least one vulnerable record is contained in X , we compute the probability $P_{\text{no_classifier_in_X}}$ that, for a dataset X sampled from D , no record for any of the k classifiers is contained in X . As mentioned previously, we assume that X was sampled without replacement from D . The probability $P_{\text{no_classifier_in_X}}$ can therefore be computed with:

$$P_{no_classifier_in_X} = \prod_{i=0}^{k-1} \frac{|D| - |X| - i}{|D| - i} \quad (2)$$

By computing $P_{\text{no_classifier_in_X}}$ for various numbers of classifiers, we obtain the plot shown in Figure 5. As anticipated, $P_{\text{no_classifier_in_X}}$ decreases as the number of classifiers increases. This is because the likelihood grows that at least one record is contained in X for that detection a classifier has been trained on. Specifically, when there are 30 classifiers, $P_{\text{no_classifier_in_X}}$ becomes negligibly small.

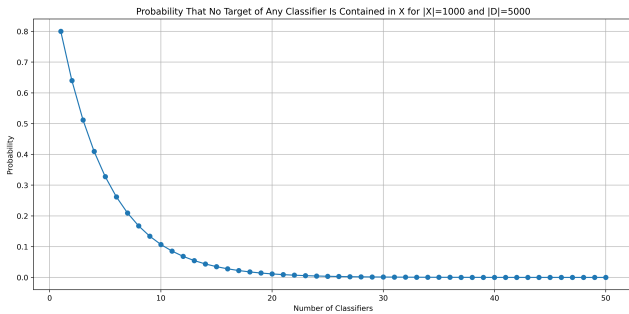


Figure 5 visualizing $P_{\text{no_classifier_in_X}}$ for an increasing number of classifiers where $|X| = 1000$ and $|D| = 5000$

Figure 5 shows that having 30 perfect classifiers yields good performance. We observed that using even more than 30 classifiers further improves the singling out accuracy. Therefore, we opted to use 50 classifiers, the maximum number we could build with our computational resources.

3.2 Attack Against Trained Decision Trees

Furthermore, we propose a method for performing a singling-out attack on a trained decision tree by leveraging the tree and the dataset’s column names as input. This is feasible because column names are not typically stored within the decision tree, such as those generated by frameworks like scikit-learn [17].

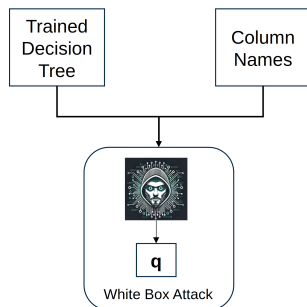


Figure 6 The attacker gets the trained decision tree and the names of the columns of the dataset as input.

The attack involves identifying a predicate that isolates a specific sample from the dataset used to train the decision tree. The attack begins by locating at least one vulnerable node in the tree. A node is considered vulnerable if it satisfies the following condition:

- The class distribution within the node includes at least one class that is represented by only a single sample.

Next, the attacker traces the decision path from the vulnerable node back to the tree’s root by concatenating the conditions along the path and the class label using the logical ‘AND’ operator. Since this metadata is not necessary for inference, the publisher of the decision tree could remove or obfuscate it. To address this, we also analyze the accuracy of singling-out attacks that do not rely on this metadata. In such cases, we construct predicates from the decision path for each leaf node, combined with every possible class label. We prioritize leaves at larger depth, as longer decision paths are more likely to yield valid predicates for the singling-out attack. The singling-out accuracy for this attack is calculated as the inverse of tries needed to construct a valid predicate averaged over all attacked trees.

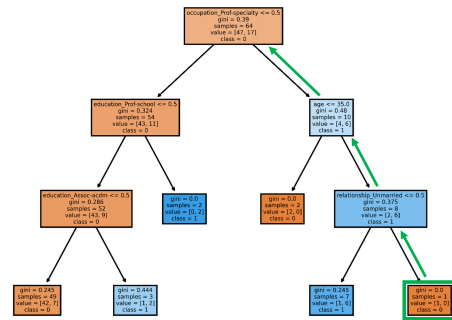


Figure 7 shows how the attacker traces back the decision path for a vulnerable node for generating the corresponding predicate $y = 0 \wedge relationship_Unmarried > 0.5 \wedge age > 35 \wedge occupation_Prof_specialty > 0.5$

For regression trees, vulnerable nodes can only be identified when a leaf node contains less than two samples due to the lack of distinct classes that would provide more granular information. We can convert the decision path of such a vulnerable node into a valid predicate by adding the constraint $\& \text{ } > \textit{leaf_node_value}$ assuming no duplicates in the dataset. When metadata is unavailable, we use the same strategy as for decision trees: starting with the node at the largest depth, we generate the predicate for the singling-out attack by

using the decision path and the leaf node value, by adding the constraint $\> leaf\ node\ value$.

3.3 Pruning Algorithm

To mitigate the risk of predicate singling-out attacks on an already trained decision tree, we developed a pruning algorithm that removes all vulnerable nodes. Our implementation is compatible with decision trees from the scikit-learn package [17]. Due to constraints in scikit-learn’s implementation, the structure of a trained decision tree cannot be directly modified. Therefore, our pruning algorithm eliminates vulnerable nodes by converting their parent nodes into leaf nodes and obfuscating the values of the remaining nodes, which are not part of the pruned decision tree. This approach ensures that no decision path exists that can be used to reconstruct any query that would single out a sample of the dataset.

4 Datasets and Synthetic Algorithms

To analyze our proposed attacks, we experimented with three tabular datasets and utilized two synthetic data generation algorithms.

1. Cardio [2]: The dataset comprises 70,000 patient records, evenly split between those with and without cardiovascular disease. It includes 11 features: 4 demographic, 4 examination, and 3 social history. The features are a mix of numerical, categorical, and binary values. We trim and bin the dataset to 11 categorical attributes.
2. Adult [13]: The data is used to predict whether an income of a person’s surpasses \$50K using Census data. We refine it by reducing and grouping it into 11 categorical attributes.
3. House 16H [11]: It offers 16 features for predicting the median house price in a region based on the demographic composition and the state of the housing market in that area.

For generating the synthetic datasets, we used two packages for synthetic data generation: the PrivBayes implementation of the DPART package [14] and the PrivBayes implementation of the package Synthetic Data Generation [16].

To assess how the quality of the data deteriorates for smaller values of ϵ , we computed the Jensen-Shannon

distance between the original dataset and the synthetic one. The Jensen-Shannon distance measures the similarity between two probability distributions. It is symmetric and always finite, ranging from 0 (identical distributions) to 1 (maximally different). The Jensen-Shannon distance is derived from the Kullback-Leibler divergence and provides a more balanced comparison between distributions.

Additionally, we compared the performance of a random forest model trained on the original data versus a classifier trained solely on the synthetic data by evaluating it on a test set from the original distribution. The experiments were conducted on 10 randomly sampled subsets of the corresponding dataset, each consisting of 1,000 samples, and the results were then averaged.

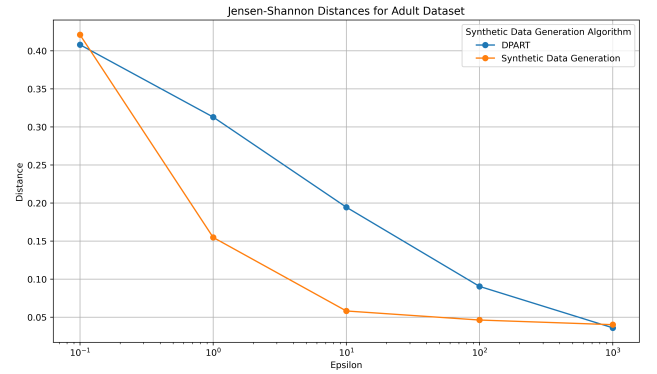


Figure 8 Jensen-Shannon distance between the synthetic and the original data computed on the adult dataset

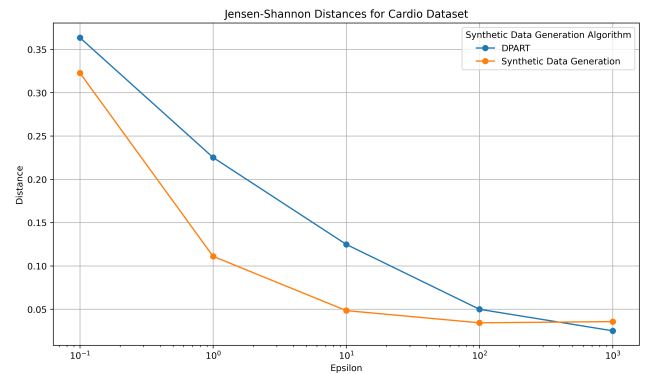


Figure 9 Jensen-Shannon distance between the synthetic and the original data computed on the cardio dataset

It can be observed that, as expected, the Jensen-Shannon distance decreases over time as epsilon increases. However, a notable difference exists between DPART and the Synthetic Data Generation package, as they exhibit different distance values for the same epsilon levels. This discrepancy could be attributed to

the distinct implementations, with DPART employing more efficient dependency-inference approaches.

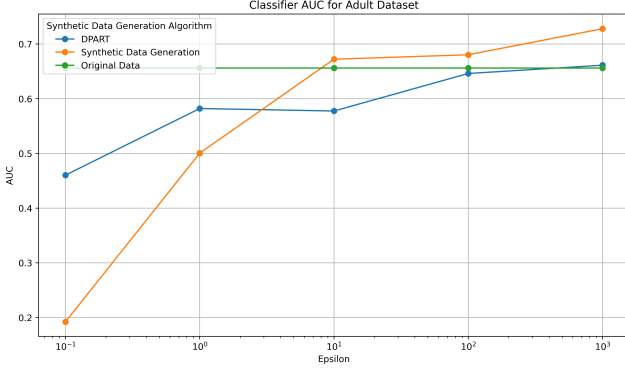


Figure 10 Performance of a classifier trained on the synthetic and the and original datasets for the adult dataset

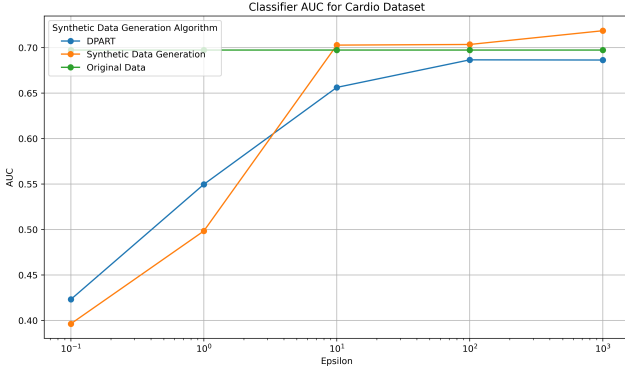


Figure 11 Performance of a classifier trained on the synthetic and the and original datasets for the cardio dataset

When comparing the performance of the random forest classifier trained on both synthetic datasets, we observe an improvement in prediction accuracy as the privacy budget increases. Notably, when the epsilon value reaches 10, the classifier trained on the synthetic data produced by the Synthetic Data Generation package achieves a higher AUC than the model trained on the original data. In the case of the PrivBayes implementation in DPART the classifier’s AUC does not surpass that of a model trained on the original data. But it still demonstrates satisfactory performance for epsilon values greater than 10.

5 Results

5.1 Attack Against Synthetic Data

To evaluate the proposed attacks, we conducted both black-box and white-box attacks on the adult and cardio datasets. We generated 1,000 synthetic datasets for

each scenario, each sampled from a PrivBayes model trained on a distinct sampled dataset X . We maintained a consistent fraction of $\frac{|X|}{|D|} = 0.2$ across all experiments. The final singling-out accuracy was calculated as the ratio of successful singling-out attacks to the total number of synthetic datasets 1,000.

Interestingly, the white-box attack did not achieve a higher accuracy than the baseline accuracy of 0.2. For training each classifier, we generated 1,000 shadow pairs. For the black-box attack, we sampled 10 synthetic datasets from each of the resulting 2,000 PrivBayes models, resulting in 20,000 training samples for each classifier. For the white-box attack, we utilized 2,000 training computed from the trained PrivBayes models.

Figure 12 - Figure 15 depict the results for the black box attacks against the synthetic datasets.

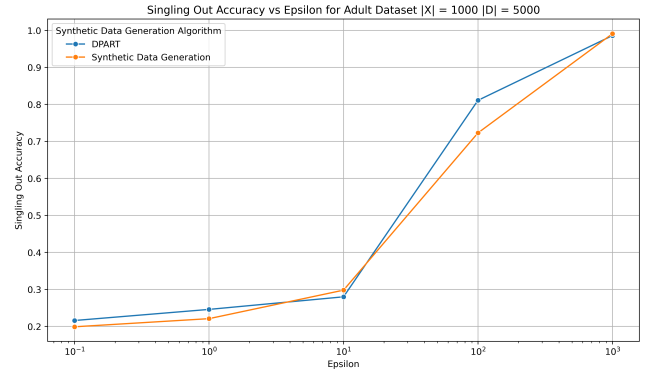


Figure 12 Singling out accuracies for different privacy budgets ϵ for the adult dataset where $|X| = 1,000$ and $|D| = 5,000$.

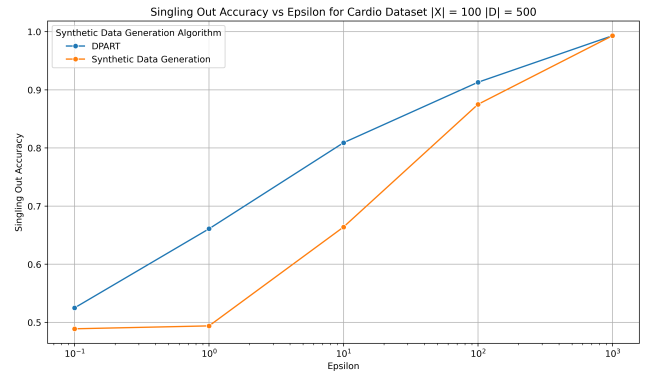


Figure 13 Singling out accuracies for different privacy budgets ϵ for the cardio dataset where $|X| = 100$ and $|D| = 500$.

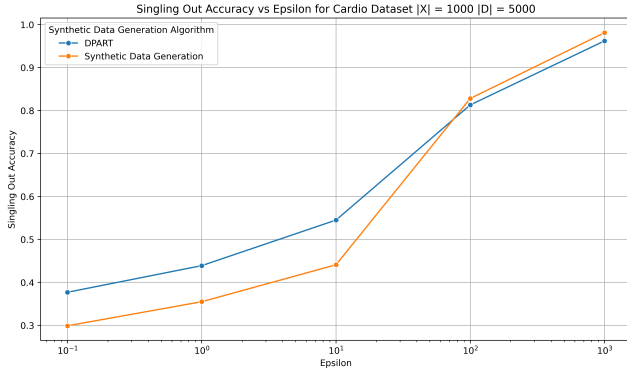


Figure 14 Singling out accuracies for different privacy budgets ϵ for the cardio dataset where $|X| = 1,000$ and $|D| = 5,000$.

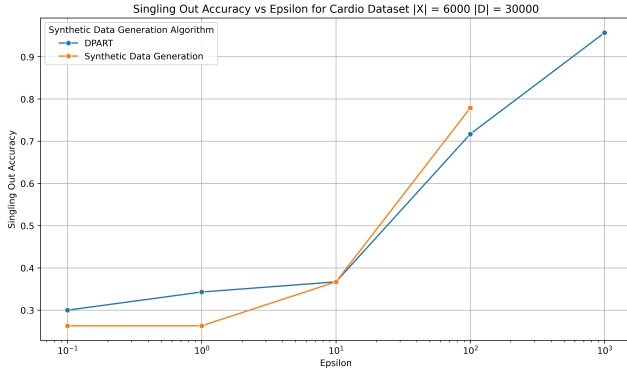


Figure 15 Singling out accuracies for different privacy budgets ϵ for the cardio dataset where $|X| = 6,000$ and $|D| = 30,000$. The value for $\epsilon = 1000$ for the Synthetic Data Generation package is not provided because its computation was infeasible with our available computational resources.

In all experiments, the singling-out accuracy increases with larger values of ϵ . Additionally, datasets with fewer samples exhibit higher singling-out accuracy than those with more samples. In Figure 12, which presents the results for the synthetic adult dataset, we see that for $\epsilon = 0.1$, both synthetic data generation frameworks only achieve the baseline accuracy. Interestingly, on the Cardio dataset, our attack achieves higher singling-out accuracies even for $\epsilon = 0.1$, which could indicate that detecting vulnerable records in this type of data is easier.

We also experimented with weighting the scores of each classifier based on their performance metrics computed on a separate test set, which was not used during training. The goal was to give more weight to more reliable classifiers. However, this approach resulted in worse overall performance. We also tried excluding the worst-performing classifiers according

to their test metrics, but this did not yield better results either. Therefore, we opted to apply the argmax function directly to the raw classifier scores.

5.2 Attack Against Trained Decision Trees

To evaluate the results of our proposed attack against trained decision trees, we constructed 100 datasets, each comprising 1,000 to 10,000 records randomly sampled from the adult, cardio, and house datasets. We then trained a decision tree on each sampled dataset and executed our two proposed attacks. A small random search for hyperparameter optimization was conducted to fit a tree on each sampled dataset, with the following bounds: $max_depth \in (5, 64)$, $min_samples_split \in (2, 32)$, and $min_samples_leaf \in (1, 16)$. The singling-out accuracy for the attack based on the tree’s metadata was computed as the fraction of trees containing at least one vulnerable node. For the attack based solely on the tree structure, without using any metadata, we counted the attempts needed to produce a valid predicate for each tree, averaged over all 100 trained trees.

	Meta Data	No Meta Data
adult dataset	97.00	10.56
cardio dataset	94.00	8.28
house dataset	0.0	6.37

Table 1 Singling-out accuracies in % of our proposed attack for different datasets

For the adult and cardio datasets, it is very likely that at least one vulnerable node exists. However, in the house dataset, none of the trees contained any vulnerable nodes. This is because the hyperparameter search resulted for the parameter $min_samples_leaf$ in values greater than 2. For the approach that does not use any metadata, an average of only 16 attempts was needed to construct a valid predicate.

We also applied our pruning algorithm to the trained trees and measured how much the model performance changes after pruning the trees. The result was that

6 Discussion

We demonstrated that predicate singling-out attacks can be successfully executed on both synthetic datasets and trained decision trees. In this section, we discuss strategies that machine learning practitioners can employ to mitigate the risk of such attacks when

publishing synthetic datasets or releasing decision trees.

As discussed in ??, smaller values of ϵ can lower the risk of singling-out attacks, although this often comes at the expense of data utility. Our proposed attack builds on the shadow modeling attack, which assumes that the adversary can generate thousands of synthetic shadow datasets. To mitigate this risk, one potential countermeasure is to design synthetic data generation algorithms that are deliberately time-consuming, ensuring they take a specific minimum amount of time, thereby making it more difficult for attackers to generate multiple shadow datasets. However, this strategy introduces the trade-off of increased computational costs during synthetic dataset creation.

To prevent singling-out attacks against trained decision trees, the most straightforward approach is to remove or obfuscate metadata that is not required for inference. Another technique is to directly construct more robust trees during training. However, to the best of our knowledge, there is no hyperparameter in decision tree training that guarantees more than one sample per class at each leaf node. While the *min_samples_leaf* hyperparameter ensures that each leaf contains at least a specified minimum number of samples, it can still lead to vulnerable nodes in the tree. To mitigate this, we have developed and published a pruning algorithm that can be applied to an already trained decision tree, reducing the risk of singling-out attacks. Applying our proposed pruning algorithm did not lead to worse prediction performance of the tree.

For regression trees, increasing the value of hyperparameter *min_samples_leaf* increases an attacker’s difficulty in identifying a valid predicate on the first try. However, since regression trees deal with continuous data, an attacker could exploit the label information at leaf node level in combination with the corresponding decision path to estimate appropriate values, potentially leading to predicates that successfully single out a sample.

References

- [1] M. S. M. S. Annamalai, G. Ganey, and E. De Cristofaro. “What do you want from theory alone?” Experimenting with Tight Auditing of Differentially Private Synthetic Data Generation”. In: *arXiv preprint arXiv:2405.10994* (2024).
- [2] *Cardiovascular Disease Dataset*. Accessed: 2024-08-13.
- [3] N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramèr, B. Balle, D. Ippolito, and E. Wallace. “Extracting training data from diffusion models”. In: *32nd USENIX Security Symposium (USENIX Security 23)*. 2023, pp. 5253–5270.
- [4] A. Cohen and K. Nissim. “Towards formalizing the GDPR’s notion of singling out”. In: *Proceedings of the National Academy of Sciences* 117.15 (2020), pp. 8344–8352.
- [5] G. D’Acquisto, A. Cohen, M. Naldi, and K. Nissim. “From Isolation to Identification”. In: *Privacy in Statistical Databases*. Ed. by J. Domingo-Ferrer and M. Önen. Cham: Springer Nature Switzerland, 2024, pp. 3–17. ISBN: 978-3-031-69651-0.
- [6] C. Dwork, F. McSherry, K. Nissim, and A. Smith. “Calibrating noise to sensitivity in private data analysis”. In: *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*. Springer. 2006, pp. 265–284.
- [7] J. Ferry, U. Arvudji, S. Gambs, M.-J. Huguet, and M. Siala. “Probabilistic dataset reconstruction from interpretable models”. In: *2024 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. IEEE. 2024, pp. 1–17.
- [8] S. Gambs, A. Gmati, and M. Hurfin. “Reconstruction attack through classifier analysis”. In: *Data and Applications Security and Privacy XXVI: 26th Annual IFIP WG 11.3 Conference, DBSec 2012, Paris, France, July 11-13, 2012. Proceedings 26*. Springer. 2012, pp. 274–281.
- [9] F. Guépin, M. Meeus, A.-M. Crețu, and Y.-A. de Montjoye. “Synthetic is all you need: removing the auxiliary data assumption for membership inference attacks against synthetic data”. In: *European Symposium on Research in Computer Security*. Springer. 2023, pp. 182–198.
- [1] M. S. M. S. Annamalai, G. Ganey, and E. De Cristofaro. “What do you want from theory alone?” Experimenting with Tight Auditing

- [10] B. Hilprecht, M. Härterich, and D. Bernau. “Monte carlo and reconstruction membership inference attacks against generative models”. In: *Proceedings on Privacy Enhancing Technologies* (2019).
- [11] *House 16H*. Accessed: 2024-08-13.
- [12] F. Houssiau, J. Jordon, S. N. Cohen, O. Daniel, A. Elliott, J. Geddes, C. Mole, C. Rangel-Smith, and L. Szpruch. “Tapas: a toolbox for adversarial privacy auditing of synthetic data”. In: *arXiv preprint arXiv:2211.06550* (2022).
- [13] R. Kohavi and B. Becker. “Adult income dataset”. In: *Dataset available at <https://archive.ics.uci.edu/ml/datasets/adult>* (1996).
- [14] S. Mahiou, K. Xu, and G. Ganev. “dpart: Differentially private autoregressive tabular, a general framework for synthetic data generation”. In: *arXiv preprint arXiv:2207.05810* (2022).
- [15] M. Meeus, F. Guepin, A.-M. Crețu, and Y.-A. de Montjoye. “Achilles’ heels: vulnerable record identification in synthetic data publishing”. In: *European Symposium on Research in Computer Security*. Springer. 2023, pp. 380–399.
- [16] D. Noors. *synthetic_data_generation*. GitHub repository. 2023.
- [17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.
- [18] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. “Privbayes: Private data release via bayesian networks”. In: *ACM Transactions on Database Systems (TODS)* 42.4 (2017), pp. 1–41.