A Differentially Private Random Decision Forest Using Reliable Signal-to-Noise Ratios

Conference Paper · November 2015

DOI: 10.1007/978-3-319-26350-2_17

2 authors:

Sam Fletcher
Charles Sturt University
8 PUBLICATIONS **26** CITATIONS
SEE PROFILE

Md Zahidul Islam
Charles Sturt University
101 PUBLICATIONS **704** CITATIONS
SEE PROFILE

A Differentially Private Random Decision Forest using Reliable Signal-to-Noise Ratios

Sam Fletcher & Md Zahidul Islam
safletcher@csu.edu.au & zislam@csu.edu.au

School of Computing and Mathematics, Charles Sturt University
Bathurst, Australia

Abstract. When dealing with personal data, it is important for data miners to have algorithms available for discovering trends and patterns in the data without exposing people's private information. Differential privacy offers an enforceable definition of privacy that can provide each individual in a dataset a guarantee that their personal information is no more at risk than it would be if their data was not in the dataset at all. By using mechanisms that achieve differential privacy, we propose a decision forest algorithm that uses the theory of Signal-to-Noise Ratios to automatically tune the algorithm's parameters, and to make sure that any differentially private noise added to the results does not outweigh the true results. Our experiments demonstrate that our differentially private algorithm can achieve high prediction accuracy.

Keywords: Differential Privacy, Noise, Decision Tree, Data Mining.

1 Introduction

Collecting data about people – whether it be for economic, medical, political, militaristic or academic purposes – is becoming increasingly commonplace. With it, comes the question of privacy: what safeguards are data collectors required to enforce to protect the privacy of those whose data they are collecting? The world at large has agreed that privacy is a human right [18]; fortunately, this right can still be upheld without forgoing the benefits of data collection wholesale. Instead, privacy preservation techniques can be employed to remove or distort identifying markers about individuals in the data without ruining the underlying trends and patterns in the data.

Balancing the loss of information with the gain in privacy has been an active area of research for nearly two decades [11]. In the early years, a privacy-preservation technique known as generalization was at the forefront of research, with k-anonymity [17] gaining a lot of support and leading to the development of machine learning techniques that minimized the loss of information in the data while still preserving privacy [11]. One thing these approaches lacked however was a strong definition of "privacy". This was rectified by differential privacy, proposed in 2006 [4–6, 16, 15]. It made the following promise to each individual

in the data: "Any information that could be discovered about you with your data in the dataset could also, with high probability, be discovered *without* your data in the dataset". In other words, the output of any query Q performed on dataset D will be indistinguishable from the output of the same query Q performed on dataset D', where D' differs from D by one record (the record of an individual).

Using differential privacy, we propose a data mining algorithm that builds an ensemble of randomized decision trees (i.e a forest), queries the data in a differentially private manner, and outputs a classifier capable of high accuracy even with very high privacy requirements. We approach the problem by phrasing it in terms of the Signal-to-Noise Ratio of our queries, and using signal averaging to reduce the noise in the queries.

1.1 Problem Statement

Dataset D is a two-dimensional matrix of rows and columns, where each row (i.e. record) $r \in D$ describes a single individual, and each column is an attribute a in the set of attributes A. Each r possesses one discrete value $v \in a; \forall a \in A$. We symbolize that record r has value v for attribute a by writing $r_a = v$. Each r also has a class value c, from the class attribute C. The aim of a decision forest is to correctly predict r_C (the class value c of record r) for records $r \in B : B \cap D = \emptyset$, where B and D are drawn from the same population.

A user is given limited access to D, in which they are allowed to query D in an ϵ-differentially private way. For any given query Q, the value of ϵ can be equal to or less than the total privacy budget β provided to the user by the data owner. We will be dividing β into smaller parts for each query Q. Our aim is to build τ decision trees by only submitting ϵ-differentially private queries Q to D, and without exceeding our total budget β. The decision trees need to have acceptably high prediction accuracy in order to be valuable.

Decision Trees [13] work by iteratively selecting attributes in a dataset that can most accurately classify a class attribute.[1] When an attribute is selected, the records in the dataset are split up according to what value they have for the chosen attribute. For each of these partitions, the process is then repeated until a user-defined termination condition is met.

1.2 Our Contributions

Our novel contributions can be summarized as the following:

- We re-phrase the problem of making a differentially private data mining algorithm in a novel way, by using Signal-to-Noise Ratio theory to assess the noise added to differentially private queries (Sect. 3).
- We present a differentially-private randomized decision forest algorithm (referred to as DP-RF) in which the structure of the decision trees is decided before querying the dataset D at all (Sect. 4).

[1] The class attribute is the attribute that the user wishes to accurately predict the value of for future records, where the value is not known.

– Our algorithm automatically tunes all parameters, with the only inputs being: access to the secure dataset D, the domains of the attributes A, and the privacy budget β (Sections 4.3, 4.4 and 4.5).
– We take full advantage of the benefits of randomly built decision trees, while identifying the assumptions usually made about decision trees that no longer hold, and providing solutions (Sect. 4.6).

We also provide code for our algorithm online.[2]

2 Previous Work

2.1 Differential Privacy

We provide a brief summary of the main components of differential privacy that we use in this paper. We refer the reader to [6] for a more thorough introduction. Differential privacy can be formally defined as follows:

Definition 1 (Differential Privacy [4]). *A query $Q : Q(D) \rightarrow Y$ satisfies ϵ-differential privacy if for all datasets D and D' differing by at most one record,*

$$Pr(Q(D) = y \in Y) \leq e^{\epsilon} \times Pr(Q(D') = y \in Y) \ . \tag{1}$$

Common values for ϵ range from 0.005 to 0.1. This definition allows a data collector to make a strong promise to each individual in D: that for any query Q, the output observed is $1/e^{\epsilon}$ as likely to occur even if they had not been in D. It does not promise that a malicious user cannot find out any information about them, but it does promise that any information they can find, they could have found without the individual even being in D.

In order for Definition 1 to be possible for query Q to achieve, there must be a randomized component in Q, preventing any output y from being 100% likely. One mechanism commonly used to inject randomness into queries is the Laplace Mechanism. Before we define this mechanism, we first need to define the "sensitivity" of Q:

Definition 2 (Sensitivity [5]). *A query Q has sensitivity $\Delta(Q)$, where:*

$$\Delta(Q) = \max_{K,K'} |Q(K) - Q(K')| \tag{2}$$

and K and K' are any datasets that differ by at most one record.

Using Definition 2, we now define:

Definition 3 (The Laplace Mechanism [5, 6]). *A query Q satisfies ϵ- differential privacy if it outputs $y + Lap(\frac{\Delta(Q)}{\epsilon})$, where $y \in Y : Q(D) \rightarrow Y$ and $Lap(x)$ is an i.i.d. random variable drawn from the Laplace distribution with mean 0 and scale x (i.e. variance $2x^2$).*

[2] Our code can be found at `http://csusap.csu.edu.au/ zislam/`, or you can email us.

We will later take advantage of two more theorems that have been proven about differential privacy:

Definition 4 (The Composition Theorem [16]). *The application of queries Q_i, each satisfying ϵ_i-differential privacy, satisfies $\sum_i \epsilon_i$-differential privacy.*

Definition 5 (The Parallel Composition Theorem [15]). *Let D_i be a disjoint subset of dataset D, and let $Q_i(D_i)$ satisfy ϵ-differential privacy; then $\sum_i Q_i(D_i)$ also satisfies ϵ-differential privacy.*

2.2 Previous Differentially Private Decision Trees

The work most closely related to ours is the differentially private random decision forest proposed by [14] in 2012. The main differences between our proposed algorithm and theirs stem from our re-framing of the scenario in terms of Signal-to-Noise Ratios. Our algorithm automatically tunes the required parameters, while the work in [14] requires the user to manually set parameters, or otherwise uses heuristics. Their heuristics are based on the combinatorial reasoning used in an earlier randomized decision tree algorithm [7], which demonstrated that randomly built trees can actually produce high quality classifiers, even in scenarios without privacy restrictions. Other work since then has supported their findings [12]. Outside of randomly built decision trees, another differentially private data mining algorithm was proposed in 2010 [10], in which one decision tree was built using the Gini Index in the same way that CART does [3]. This work has since been improved upon [8].

3 Signal-to-Noise Ratio

Signal-to-Noise Ratio is a comparison between some sort of measurement (signal) and the background noise accompanying that measurement [19]. The ratio of the "proper" or "real" signal to the noise is the Signal-to-Noise Ratio, or SNR. The mathematics underpinning the SNR concept is applicable in any scenario where the following assumptions are met: the signal and the noise are uncorrelated; the signal would be the same if it was measured again (ignoring the noise); and the noise is random, with a mean of 0 and a constant variance for repeated measurements of the signal. Differential privacy meets these assumptions: noise added with the Laplace Mechanism is only dependent on ϵ and Δ, neither of which are affected by the signal; the number of records meeting a query's criteria will not change if the same dataset is queried again; and the Laplace distribution has a mean of 0 and a known, constant variance of $2x^2$.

Expressing the SNR mathematically is quite simple:

Definition 6 (Signal-to-Noise Ratio). *The Signal-to-Noise Ratio of a measurement can be expressed as $\frac{signal}{noise}$, where the signal and noise are expressed in the same units. An alternate way of writing this is:*

$$SNR = \frac{\mu}{\sigma} \tag{3}$$

where μ is the signal mean or expected value and σ is the standard deviation of the noise.

We take advantage of Definition 6 in Sect. 4.

3.1 Signal Averaging

Once a problem has been phrased in terms of the Signal-to-Noise ratio, there is additional property we are able to take advantage of: signal averaging. Signal averaging provides an explicit definition of the intuition that if noise can increase or decrease a signal with equal probability, then summing multiple signals together will result in a total that is less noisy:

$$\text{SNR} = \frac{\mu}{\sigma} = \frac{\sum_x^X \mu_x}{\sqrt{|X|\sigma^2}} \tag{4}$$

where X is the set of signals, and $|X|$ is the size of that set.

4 Our Differentially Private Random Decision Forest

It has been demonstrated in the past that randomized decision trees can have surprisingly high performance, and often have much lower computational complexity compared to their less random counterparts [7, 12]. They gain their computational efficiency from the fact that they do not need to use the training data D in order to build a tree. This is a valuable advantage when trying to achieve differential privacy, as the less times we need to query the training data the better. The overall design philosophy of a differentially private machine learning algorithm is to be as efficient at spending the privacy budget β as possible. Below, we present our novel algorithm for building a differentially private random forest (DP-RF), and achieve better accuracy than any differentially private decision tree/forest algorithm proposed before it, even with very low privacy budgets.

4.1 Overview of Our Algorithm, DP-RF

Our algorithm, described in detail in the sections below, can be summarized as the following steps:

1. Based off the size of the privacy budget β, the size of the dataset D and the domain sizes of the attributes A, automatically tune the following parameters:
   - τ, the number of trees in the decision forest, which in turn dictates the ϵ spent per tree (Sect. 4.4).
   - θ, the minimum support threshold for nodes in the trees (Sect. 4.3).
2. Build a randomized decision forest using τ and θ, and no user-inputted parameters (Sect. 4.3).

3. Query the dataset using the Laplace Mechanism to learn the class counts in each leaf (Sect. 4.2).
4. Prune away nodes with SNR < 1, using signal averaging for non-leaf nodes (Sect. 4.5).
5. Find the node with the highest confidence in each path from the root node to a leaf node, in each tree (Sect. 4.6)
6. Predict the class value of future records by voting on the most confident predictions made by each tree (Sect. 4.7).

4.2 Querying the Leafs of a Tree

By following any chain of directed edges from a decision tree's root to a leaf, we have what is called a "decision rule". Each decision rule is a collection of attribute values, defined by the attributes from A that each node splits on, with each directed edge having one value a_v from the attribute a in the node it came from. Every record r in the training data D possesses a value $a_v; \forall a \in A$ that make it match one and only one decision rule in the decision tree. We therefore say that each record "fits into" or "belongs to" the leaf at the end of the chain of nodes its values match.

The Parallel Composition Theorem (Definition 5) means that for a decision tree – where every record appears in one and only one leaf – we can perform an ϵ-differentially private query Q on every leaf and use a total of ϵ out of the privacy budget β [14].

This is precisely what we will be doing: performing a single query Q on each leaf of a decision tree. That single query will be, "How many records have each class value?". In other words, we will get a histogram of the class counts in each leaf. The sensitivity Δ of this query Q is $\Delta(Q) = 1$, because the bins in a histogram are disjoint – the removal or addition of a single record can affect at most one bin of the histogram. Using the Laplace Mechanism (Definition 3) we can achieve ϵ-differential privacy by adding $\text{Lap}(1/\epsilon)$ to each class count, in each leaf of a decision tree.

4.3 Building a Random Tree

The building of a random decision tree is straight-forward, and does not require the training data D at all. An attribute a is chosen as the root node of the tree, with a directed edge (i.e. a branch of the tree) being created for each value $a_v \in a$. This process is recursively performed for the nodes at the end of each directed edge. The chosen attribute a at each node is selected from the set of attributes not chosen previously in the recursion chain, so that for any chain of nodes (i.e. path) from the root node to the bottom of the tree, no attribute appears twice.

The recursion ends for a given chain in the tree when either of the following termination criteria are met: (1.) there are no attributes left that have not been used previously in the recursion chain; or (2.) The estimated support of the

current node is so low that the estimated SNR is below 1 (i.e the noise outweighs the signal).

Criteria 2 uses the Signal-to-Noise Ratio, described in Sect. 3, and refers to the "estimated support" of the node. We define "estimated support" as the number of records that are estimated to match the attribute values $a_v; \forall a$ on the directed edges that led to the current node. The number of records is predicted by using the assumption that the subset of records D_i in a given node will be divided equally amongst all the directed edges leaving that node. In other words, starting from the root node and working our way down the tree, we make a rough estimate of the support of each node by dividing it by the domain size of each attribute used earlier in the chain.

This has the effect of not restricting the entire tree to the same maximum depth, unlike the differentially-private random decision tree algorithm proposed by [14]. Assuming that each value of an attribute has an equal portion of the records in D is clearly not an accurate assumption, but it does not need to be for our purposes. While rough, we mostly need the order of magnitude of a node's support, and we apply pruning later (see Sect. 4.5) to clean up the rough estimates. By estimating the support of nodes without using the training data, we avoid having to spend any of the privacy budget β on queries.

In the case of differential privacy, there is an additional reason to define a minimum support threshold: the larger the support, the less likely it is that noise will disrupt the signal. If we apply a query Q to a leaf with very small support, the noise could easily completely overwhelm the signal.

In order for the SNR to be above 1, the estimated support of a node must be larger than the minimum support threshold. We define the minimum support threshold as:

$$\theta = |C| \times \sqrt{2} \times \frac{1}{\epsilon} \ . \tag{5}$$

This is derived from the Laplace Distribution's variance, $\sigma^2 = 2x^2$ (see Definition 3), and the Signal-to-Noise Ratio, μ/σ. Because the scale of the noise we are adding is $1/\epsilon$ we can substitute that in for x, and by taking the square root of the variance we are left with the standard deviation, as seen in the formula for the SNR (Definition 6). Combining these observations leaves us with:

$$\frac{\mu}{\sigma} = \frac{\mu}{\sqrt{2 \times (1/\epsilon)^2}} = \frac{\epsilon\mu}{\sqrt{2}} \ . \tag{6}$$

A simple rearrangement lets us see that the signal μ (i.e. the support of a node) must be larger than $\mu > \sqrt{2}/\epsilon$ in order for $\mu/\sigma > 1$. Note that this assumes that $Lap(1/\epsilon)$ is only being added to the support of a node once. However, it as actually being added to each node a number of times equal to the size of the class attribute: $|C|$. We therefore need to increase the noise component of the SNR accordingly:

$$\mu > \frac{|C|\sqrt{2}}{\epsilon} \ , \tag{7}$$

and thus we arrive at (5): the estimated support μ must be larger than θ.

After recursively building upon each node until the estimated support of each node is less than the minimum support threshold θ, we can then learn how many records in the training data D fit into each leaf. Learning about the records in each leaf requires querying the dataset, and spending ϵ amount of the privacy budget β. This process is described in full in Sect. 4.2.

Thus we have built a random decision tree. We then repeat this tree-building process a number of times equal to τ to end up with a forest of random decision trees. The parameter τ is automatically defined by our algorithm, and is described below in Sect. 4.4. τ defines the fraction of the privacy budget that each decision tree can spend. When differentially-privately querying the leafs of one decision tree, ϵ is defined as:

$$\epsilon = \frac{\beta}{\tau} \ . \tag{8}$$

4.4 Defining the Number of Trees

To promote as diverse a collection of random trees as possible, we can make the root node of each tree unique by making sure a different attribute is chosen each time. Diversity is well known to be an advantageous property of decision forests [2] – hence why randomly built trees are valuable even when there are no privacy restrictions involved [7, 12]. Extending this idea, we can define the number of trees as being equal to the number of attributes – thus the dataset will be partitioned based off the values of every attribute, giving us a chance to learn how predictive each attribute can be on their own. Any more trees and we would start having redundant root nodes, and thus be getting less value for the privacy budget spent on it than we did for the earlier trees.

Limiting the number of trees is only necessary because we have a privacy budget we must adhere to. As explained in Sect. 4.2, we can query the dataset about the (disjoint) leafs in a tree using the same portion of the privacy budget. However we cannot use the same portion of the privacy budget for multiple trees – each tree is a re-partitioning of the same data, and so they are not disjoint. This means that the more trees we have, the more we have to divide the privacy budget among them due to Definition 4 (see (8)).

It is therefore possible that if there is a large number of attributes, the budget may be spread too thin to output useful query results. In our case, "too thin" can be interpreted as "the minimum support threshold is too high to build a tree". We address this by reducing the number of trees to a point where the minimum support threshold becomes acceptable. We do so by imagining a scenario in which the dataset D is split into a number of partitions D_i equal to the average domain size of the attributes A – we now have the "stump" of a decision tree. We then repeat this, dividing each node on the next level of the tree by the average domain size of the attributes. We now have a tree with a depth of 3: the dataset is divided up twice by attributes (with average domain sizes), with the last level of the tree being leafs. We consider this to be the minimally acceptable

tree size. For this minimally acceptable tree size to be possible (at least in the average case), the minimum support threshold must be as follows:

$$\theta < \frac{|D|}{\delta^2} \ , \tag{9}$$

where δ is the average domain size of the attributes in A, and θ is defined by (5). Recall that there is no maximum depth defined in our algorithm: some parts of the decision tree might have a depth shorter than 3, while other parts of the same tree might have a much larger depth. Thus we define τ as the largest number, up to the number of attributes $|A|$, that satisfies (9) and (5).

4.5 Pruning to a Reliable Depth

Once the forest of random decision trees has been built, the next step in our algorithm is to prune each decision tree. This is somewhat similar to the pruning done in other decision tree algorithms [13], including the pruning done in Private-RDT [14]. The only pruning done in Private-RDT is to remove leafs with no records in them. We propose a more sophisticated approach, but first a remark on their approach. It is important to note that even if there are 0 records in a leaf, Laplace noise still needs to be added to each class count. While of course a count cannot go below 0, a count that was originally 0 can be raised above 0 by the noise. This drastically reduces the amount of pruning that is done in Private-RDT, and keeps leafs that most likely have a Signal-to-Noise Ratio of 0 (i.e. they are 100% noise). Our proposed approach solves this problem.

Earlier, in Sect. 4.3, we estimated the support of each node in order to estimate its Signal-to-Noise Ratio and decide whether to branch from the node further. Now that we have queried the server and have the (noisy) class counts of each leaf, we no longer have to use such rough estimates for each node's support – we can re-check which nodes have acceptable SNRs, and which do not. We can also take advantage of signal averaging, described in Sect. 3.1. The idea is simple: by summing together the class counts of all the leafs that have the same parent node, we know the class counts of the parent node. This can be repeated up the tree until we know the class counts of every node in the tree. Not only that, but the class counts of the parent node are actually *less noisy* than those of the child nodes. We can see this in (4), as well as the intuition behind it in Sect. 3.1. The noise is reduced by a factor equal to the square root of the number of leaf nodes being summed together.

Using this knowledge, we now not only have more accurate SNRs for the leafs than we did in Sect. 4.3, but we also have increasingly more accurate SNRs for the nodes above the leafs, the further we traverse up the tree. Our pruning is simple: we remove any nodes with SNR < 1. Because of the reduction in noise in nodes using the sum of class counts in its children, it becomes easier and easier for nodes higher up the tree to have SNR > 1. When re-written for this scenario, (4) becomes:

$$\text{SNR} = \frac{\epsilon \sum_{L}^{\text{Leafs}} \left(\sum_{c}^{C} L_c \right)}{|C| \sqrt{2 \times |\text{Leafs}|}} \ , \tag{10}$$

where *Leafs* is the set of all leaf nodes that can be reached by traversing down from the current node (not all leafs in the entire tree), and L_c is the class count for $c \in C$ in leaf L. Observe that (10) reduces down to (6) when considering one leaf and one class value – they both originate from the SNR theory.

4.6 Finding the Most Confident Prediction in Each Rule

Once unreliable leafs have been pruned away, we are left with only nodes where the true counts of the class values outweigh the noise. Our aim now is to use these nodes to predict the class values of future records. For a given future record r and a given decision tree, we can find the leaf that r belongs to, and know the decision rule that it obeys.

Usually in non-randomly built decision trees, a node is only split into more nodes if an attribute can be found that increases the confidence of the decision rule's prediction of r's class value [3]. The predicted class value of r is therefore often defined as the most common class value in the leaf of the decision rule (i.e. the majority class value). The "confidence" of the prediction is then the fraction of the records in the leaf that have the majority class value. However in the case of a randomly built decision tree, there is nothing preventing a node above the leaf from having a higher confidence than the leaf.

We therefore find the most confident prediction in each decision rule, and then use that prediction for any future records that obey that rule.

4.7 Voting

When predicting the class value of a future record r, the process of finding the most confident prediction in the decision rule that r obeys is repeated for each decision tree in the forest. r obeys one decision rule per tree, and our algorithm selects one node out of each decision rule with the highest confidence, as described in Sect. 4.6. It is unlikely that all the predictions are predicting the same class value, so a voting mechanism is required to decide which class value is the decision forest's final prediction.

Our algorithm votes using the following process: (1.) The final prediction is the class value with the highest confidence; if two different class values are predicted with equal confidence, then (2.) sum the confidences of each class value from the nodes selected in Sect.4.6 (including the non-majority class values) and (3.) the final prediction is the class value with the highest summed confidence.

5 Experiments

We test our proposed algorithm (DP-RF) using 9 datasets from the UCI Machine Learning Repository [1]. We use 10-fold stratified cross-validation repeated 30 times to calculate the average prediction accuracy [9] of our algorithm, using various privacy budgets. We test the following privacy budgets: $\beta = 0.01$, 0.05, 0.1, 0.25, 0.5, 1.0, 2.0. We compare our technique to Private-RDT, proposed by

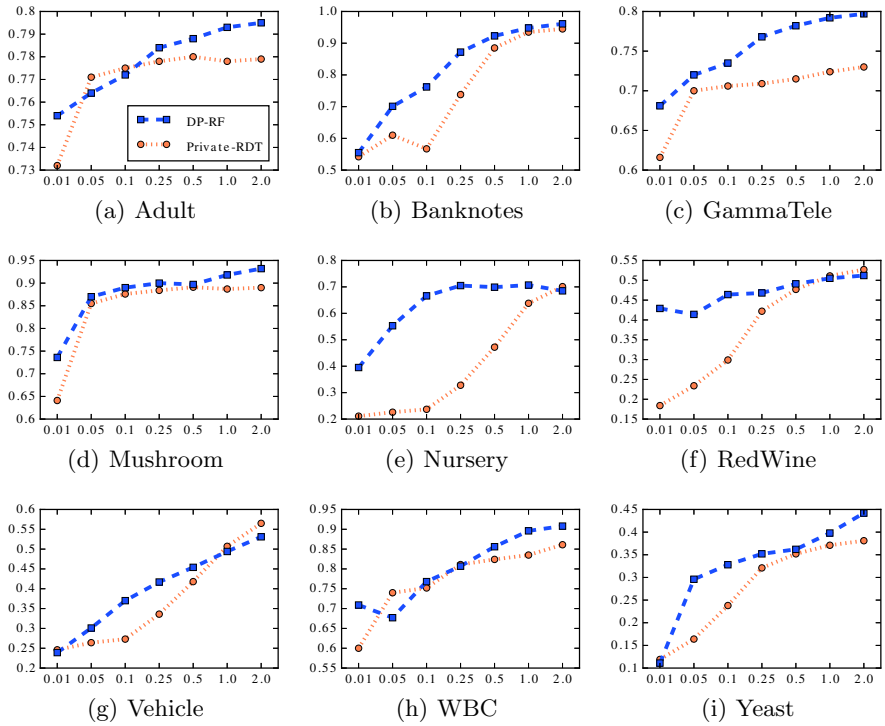

Fig. 1. Comparing the prediction accuracy of our technique (DP-RF) to Private-RDT [14] at different ϵ values, using 9 different datasets from [1]. The y axis represents the prediction accuracy of the classifier and the x axis represents β.

[14] (Sect. 2.2). For continuous attributes, we discretize them by splitting their domain into 5 bins of equal range. We use the default parameters recommended in [14] for Private-RDT. Note that high values of β (especially values ≥ 1) are unlikely in real-world situations, and are presented to demonstrate the trends.

Our results are presented in Fig. 1. For all datasets, our algorithm has higher prediction accuracy than Private-RDT on average. Out of the 63 measurements (7 values for β per dataset) our algorithm out-performs Private-RDT in 55 cases, sometimes only under-performing by $< 1\%$ (i.e. 0.01 on the y axis). At its best our algorithm can beat Private-RDT by 45%; at its worst it loses by 7%.

6 Conclusion

By re-framing the problem of building a differentially private decision forest in terms of the Signal-to-Noise Ratios, we are able to propose intuitive methods for automating the tuning of necessary parameters. We are also able to guarantee that any predictions made about future records are made using class counts that not only have high confidence, but also outweigh any noise that might have

been added to them. Our results prove the success of this approach, and pave the way for extending the application of Signal-to-Noise Ratio theory to other implementations of differential privacy.

References

1. Bache, K., Lichman, M.: UCI Machine Learning Repository (2013), http://archive.ics.uci.edu/ml/
2. Breiman, L.: Random forests. Machine learning pp. 1–35 (2001)
3. Breiman, L., Friedman, J., Stone, C., Olshen, R.: Classification and regression trees. Chapman & Hall/CRC (1984)
4. Dwork, C.: Differential Privacy. In: Automata, languages and programming. vol. 4052, pp. 1–12. Springer Berlin Heidelberg, Venice, Italy (2006)
5. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. Theory of Cryptography pp. 265–284 (2006)
6. Dwork, C., Roth, A.: The Algorithmic Foundations of Differential Privacy, vol. 9. Now Publishers (2013)
7. Fan, W., Wang, H., Yu, P., Ma, S.: Is random model better? On its accuracy and efficiency. Third IEEE International Conference on Data Mining (2003)
8. Fletcher, S., Islam, M.Z.: A Differentially Private Decision Forest. In: Proceedings of the 13th Australasian Data Mining Conference. Sydney, Australia (2015)
9. Fletcher, S., Islam, M.Z.: Quality evaluation of an anonymized dataset. In: 22nd International Conference on Pattern Recognition. IEEE, Stockholm (2014)
10. Friedman, A., Schuster, A.: Data Mining with Differential Privacy. In: 16th SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 493–502. ACM, Washington, DC, USA (2010)
11. Fung, B., Wang, K., Chen, R., Yu, P.: Privacy-preserving data publishing: A survey of recent developments. ACM Computing Surveys (CSUR) 42(4), 14 (2010)
12. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine Learning 63(1), 3–42 (Mar 2006)
13. Han, J., Kamber, M., Pei, J.: Data mining: concepts and techniques. Morgan Kaufmann Publishers (2006)
14. Jagannathan, G., Pillaipakkamnatt, K., Wright, R.: A practical differentially private random decision tree classifier. Transactions on Data Privacy 5 (2012)
15. McSherry, F.: Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In: Proceedings of the 35th SIGMOD international conference on Management of data. pp. 19–30. ACM, Providence, USA (2009)
16. McSherry, F., Talwar, K.: Mechanism Design via Differential Privacy. 48th Annual IEEE Symposium on Foundations of Computer Science pp. 94–103 (2007)
17. Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(5), 571–588 (2002)
18. UN General Assembly: Universal Declaration of Human Rights (1948)
19. Van Drongelen, W.: Signal processing for neuroscientists: an introduction to the analysis of physiological signals. Academic Press (2006)