Differentially Private Selection using Smooth Sensitivity

Iago C. Chaves
Universidade Federal do Ceará
Fortaleza, Ceará, Brazil
Email: iago.chaves@lsbd.ufc.br

Victor A. E. Farias
Universidade Federal do Ceará
Fortaleza, Ceará, Brazil
Email: victor.farias@lsbd.ufc.br

Amanda Perez
Fundação Getulio Vargas
Rio de Janeiro, Rio de Janeiro, Brazil
Email: perez.amanda@fgv.edu.br

Diego Mesquita
Fundação Getulio Vargas
Rio de Janeiro, Rio de Janeiro, Brazil
Email: diego.mesquita@fgv.br

Javam C. Machado
Universidade Federal do Ceará
Fortaleza, Ceará, Brazil
Email: javam.machado@lsbd.ufc.br

***Abstract*—Differentially private selection mechanisms offer strong privacy guarantees for queries aiming to identify the top-scoring element r from a finite set $\mathcal{R}$, based on a dataset-dependent utility function. While selection queries are fundamental in data science, few mechanisms effectively ensure their privacy. Furthermore, most approaches rely on *global sensitivity* to achieve differential privacy (DP), which can introduce excessive noise and impair downstream inferences. To address this limitation, we propose the *Smooth Noisy Max* (SNM) mechanism, which leverages *smooth sensitivity* to yield provably tighter (upper bounds on) expected errors compared to global sensitivity-based methods. Empirical results demonstrate that SNM is more accurate than state-of-the-art differentially private selection methods in three applications: percentile selection, greedy decision trees, and random forests.**

1. Introduction

Differential privacy (DP) establishes a mathematically rigorous framework to avoid information leakage upon releasing the outcome of a query. More formally, achieving DP entails ensuring that the outcome of a query is statistically near-indistinguishable for similar databases. This is typically done by endowing the original query with a *mechanism*, which randomizes the query's output. Importantly, DP mechanisms are tailored to the space of outcomes of the query they aim to protect.

In particular, private selection mechanisms address non-numerical queries and play a crucial role in private machine learning and data analysis, with applications in classification [26], synthetic data generation [10, 40], dimensionality reduction [9], and top-k queries [24]. However, despite their vast applicability, there exist only a few mechanisms for private selection, including the exponential mechanism [29], the report-noisy-max algorithm [15], permute-and-flip [28], and the local dampening mechanism [17].

Most of these algorithms are based on adding noise depending on the notion of global sensitivity, which measures the most significant impact over the utility function of adding or removing an entry from all possible databases and for all possible outcomes. This approach is guided by the worst-case scenario, which usually adds high noise [7, 22, 38, 39], potentially harming the accuracy of results. To mitigate that, Nissim, Raskhodnikova, and Smith [30] propose the concept of smooth sensitivity, an instance-based sensitivity that depends locally on the input database $\mathbf{x}$; nevertheless, their paper focus only on numerical queries [30]. The local dampening algorithm already applies a similar concept (local sensitivity) to the context of selection queries [17], but it still shows some limitations, mainly related to stability and time complexity.

We propose a novel differentially private selection algorithm, termed Smooth Noisy Max (SNM), which employs smooth sensitivity for noise addition. Specifically, SNM corrupts the utility score of each potential outcome $r \in \mathcal{R}$ with random noise (e.g., from a Laplace, Laplace Log-Normal, or Student's T distribution) scaled according to a factor proportional to the instance-based sensitivity. Notably, we show that SNM is provably more accurate than alternatives based on global sensitivity and produces better empirical results than the prior art.

Problem Statement. We address the challenge of *private data selection*, aiming to ensure that the selection process remains both privacy-preserving and capable of producing meaningful outcomes. Let $\mathbf{x} \in \mathcal{X}$ be a sensitive database, represented as a *multiset* of records from $\mathcal{X}$, where each entry x_i corresponds to a record in $\mathcal{X}$. Consider a data selection function $f : \mathcal{X} \to \mathcal{R}$ that takes $\mathbf{x}$ as input and produces an outcome $r \in \mathcal{R}$. The central challenge is to release $f(\mathbf{x})$ in a differentially private manner—ensuring that the output reveals minimal information about any individual record in $\mathbf{x}$—while maintaining the utility and relevance of the results.

Contributions. The main contribution of this paper is the first primitive for private selection relying on smooth sensitivity. Moreover, applying this concept to non-numerical selection requires addressing significant technical obstacles.

For instance, the most intuitive way to adapt existing algorithms (e.g., exponential mechanism) is replacing the global sensitivity by the smooth one. However, our Theorem 7.2 shows this does not result in a differentially private algorithm. We also provide utility guarantees showing SNM is never worse than existing methods under mild conditions. In summary, the contributions of this work are:

i) We prove that the concept of smooth sensitivity cannot be utilized along with the exponential mechanism. Therefore, we extend the smooth sensitivity, originally defined for numerical data, to the data selection setting;
ii) We propose the Smooth Noisy Max (SNM), a differentially private data selection algorithm that applies our extended notion of smooth sensitivity;
iii) We provide differential privacy guarantees for Smooth Noisy Max, along with theoretically rigorous utility guarantees showing that Smooth Noisy Max is never worse than its competitors under mild conditions;
iv) We conducted an empirical comparison[1] of Smooth Noisy Max with competing methods across three applications: percentile selection, greedy decision trees, and random forests. Our findings indicate that SNM consistently outperforms state-of-the-art methods in terms of accuracy and expected error.

This paper is structured as follows: Section 2 provides basic definitions regarding DP. Section 3 reviews the prior art on private selection. Section 4 presents the Smooth Noisy Max algorithm. Section 5 applies Smooth Noisy Max to percentile selection. Section 6 explores a private decision tree approach. Section 7 discusses a novel random forest algorithm using Smooth Noisy Max. Finally, Section 8 concludes the paper with future directions.

2. Preliminaries

Let database $\mathbf{x}$ be a set of records drawn from a universe $\mathcal{X}$, and f a query over $\mathbf{x}$. In differential privacy, the goal is to ensure that the outcome of a computation/algorithm, denoted by $\mathcal{A}$, does not reveal much sensitive information about any individual in a database. At the same time, the algorithm $\mathcal{A}$ ensures data processing without disclosing individual information, even if an adversary has almost complete knowledge of all other individuals in the database. Differential privacy uses a randomized algorithm, i.e., a mechanism that adds controlled noise to the data, and it is based on a privacy budget parameter, typically denoted as ε, representing the desired level of privacy protection. We formalize the database as a *multiset* of records of $\mathcal{X}$. Therefore, the distance between two databases can be determined by counting the records that differ between them. More specifically, this distance is quantified using the symmetric difference of two sets, denoted as $d(\mathbf{x}, \mathbf{y}) = |\mathbf{x} \oplus \mathbf{y}|$.

Definition 2.1 ((ε, δ)-Differential privacy [15])**.** A randomized algorithm $\mathcal{A}$ satisfies (ε, δ)-differential privacy if, for any two databases $\mathbf{x}$ and $\mathbf{y}$ that differ in at most one record, and for any possible output $S \subseteq \mathcal{Y}$ over the outcome space $\mathcal{Y}$ of the algorithm

$$Pr[\mathcal{A}(\mathbf{x}) \in S] \leq e^{\varepsilon} Pr[\mathcal{A}(\mathbf{y}) \in S] + \delta$$

where $Pr[\cdot]$ stands for probability of an event. When $\delta = 0$, the algorithm is ε-differentially private. We refer to ε-differential privacy as pure differential privacy. Conversely, (ε, δ)-differential privacy, where $\delta > 0$, is referred to as approximate differential privacy.

An alternative interpretation of differential privacy is presented in Remark 3.1 of Dwork and Roth [15], utilizing the concept of δ-approximate max divergence. This perspective reformulates differential privacy in terms of distributional distance measures.

Definition 2.2 (δ-Approximate Max Divergence [15])**.** The δ-Approximate Max Divergence between two random variables X and Y taking values from the same domain is defined to be:

$$D_{\infty}^{\delta}(X||Y) = \max_{S \subseteq \mathcal{Y} \,:\, Pr[X \in S] \geq \delta} \left[\log \left(\frac{Pr[X \in S] - \delta}{Pr[Y \in S]} \right) \right]$$

Lemma 2.3 (Approx. Differential Privacy [15])**.** Note that a mechanism $\mathcal{A}$ is (ε, δ)-differentially private if and only if on every two neighboring databases $\mathbf{x}, \mathbf{y} : D_{\infty}^{\delta}(\mathcal{A}(\mathbf{x})||\mathcal{A}(\mathbf{y})) \leq \varepsilon$ and $D_{\infty}^{\delta}(\mathcal{A}(\mathbf{y})||\mathcal{A}(\mathbf{x})) \leq \varepsilon$.

The quantity of noise introduced is proportional to the global sensitivity of the query. Global sensitivity, denoted by Δf, quantifies the maximum change in a function's f output when a single individual's data is modified, reflecting the largest difference between outputs for databases differing by one record.

Definition 2.4 (Global sensitivity [15])**.** The global sensitivity of a function $f : \mathcal{X} \to \mathbb{R}$ is defined:

$$\Delta f = \max_{\substack{\mathbf{x}, \mathbf{y} \in \mathcal{X} \\ d(\mathbf{x}, \mathbf{y}) \leq 1}} |f(\mathbf{x}) - f(\mathbf{y})|$$

However, its practical utility is often limited due to excessive noise generation, as the Laplace mechanism's scale parameter is $\frac{\Delta f}{\varepsilon}$ [15], leading to high noise levels for functions like k-clique counting [39] and median queries [30]. Local sensitivity, denoted by $LS_f(\cdot)$, is a database-specific measure of the maximum change in output resulting from individual data modifications. Consequently, employing instance-specific sensitivity can help reduce the amount of noise introduced.

It is crucial to highlight that the global sensitivity is the maximum local sensitivity over all databases, $\Delta f = \max_{\mathbf{x} \in \mathcal{X}} LS_f(\mathbf{x})$. Nonetheless, using the local sensitivity, instead of global, would reduce the amount of noise produced by the random algorithm so much that it would not satisfy the differential privacy definition [30].

To address the problem of achieving differential privacy for numerical queries with instance-based sensitivity, the work of Nissim, Raskhodnikova, and Smith [30] proposed

1. The source code and other artifacts have been made available at https://github.com/iagocc/smooth-noisy-max

the smooth sensitivity framework, which smooths the local sensitivity at a distance t.

The local sensitivity at a distance t measures the maximum local sensitivity LS_f over all databases up to the distance t from $\mathbf{x}$, i.e., up to t modifications on the database $\mathbf{x}$. It is important to note that it is a generalization of the local sensitivity $LS_f(\mathbf{x}, 0) = LS_f(\mathbf{x})$, a particular case when the distance is set to 0.

Definition 2.5 (Local sensitivity at distance t [30])**.** For a query $f : \mathcal{X} \to \mathbb{R}^k$ and a database $\mathbf{x} \in \mathcal{X}$, the local sensitivity of f at $\mathbf{x}$ at distance t is defined as:

$$LS_f(\mathbf{x}, t) = \max_{\mathbf{y} \in \mathcal{X} \mid d(\mathbf{x},\mathbf{y}) \leq t} LS_f(\mathbf{y})$$

The sensitivity measure itself may inadvertently disclose individual information. Moreover, adjusting noise based on local sensitivity may risk potential data leakage. To determine the appropriate noise magnitude, the work by Nissim, Raskhodnikova, and Smith [30] utilizes a smooth upper bound on local sensitivity. Specifically, they define a function S that not only provides an upper limit on LS_f across all points but also ensures that $\ln(S(\cdot))$ maintains low sensitivity.

Definition 2.6 (Smooth bound [30])**.** For $\beta > 0$, a function $S : \mathcal{X} \to \mathbb{R}^+$ is a β-smooth upper bound on the local sensitivity of a function f if it satisfies the following requirements:

$$\forall \mathbf{x} \in \mathcal{X} : S(\mathbf{x}) \geq LS_f(\mathbf{x})$$
$$\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}, d(\mathbf{x}, \mathbf{y}) \leq 1 : S(\mathbf{x}) \leq e^{\beta} S(\mathbf{y})$$

Definition 2.7 (Smooth sensitivity [30])**.** For $\beta > 0$, the β-smooth sensitivity of f is:

$$\mathcal{S}_{f,\beta}(\mathbf{x}) = \max_{t=0,1,\ldots,|\mathbf{x}|} \left(e^{-t\beta} \cdot LS_f(\mathbf{x}, t)\right)$$

The smooth sensitivity $\mathcal{S}_{f,\beta}$ is the smallest function to satisfy the smooth bound requirements (Definition 2.6) [30]. The smooth sensitivity adjusts the contribution of the local sensitivity based on the distance between a database and $\mathbf{x}$. The β parameter, which serves as a smoothing factor, is strategically chosen to mitigate inadvertent data disclosure risks that may arise when employing local sensitivity directly. The global sensitivity Δf is also a smooth upper bound on the local sensitivity, i.e., the global sensitivity satisfies the Definition 2.6.

Corollary 2.8 (Smooth sensitivity upper bound)**.** For a query f, a database $\mathbf{x}$, the global sensitivity Δf is an upper bound of smooth sensitivity $\mathcal{S}_{f,\beta}$ i.e., $\mathcal{S}_{f,\beta}(\mathbf{x}) \leq \Delta f$.

Mechanisms that the addition of noise is proportional to the smooth sensitivity are contingent upon whether the noise distribution meets the criteria necessary for achieving differential privacy, i.e., (α, β)-admissibility.

Definition 2.9 (Admissible Noise Distribution [30])**.** A probability distribution on $\mathbb{R}^k$, given by a density function h, is (α, β)-admissible if, for $\alpha = \alpha(\varepsilon, \delta)$, $\beta = \beta(\varepsilon, \delta)$, the following two conditions hold for all $\Delta \in \mathbb{R}^k$ and $\lambda \in \mathbb{R}$ satisfying $\|\Delta\|_1 \leq \alpha$ and $|\lambda| \leq \beta$, and for all measurable subsets $S \subseteq \mathbb{R}^k$.

(i) *(Sliding)* $\Pr_{Z \sim h}[Z \in S] \leq e^{\frac{\varepsilon}{2}} \Pr_{Z \sim h}[Z \in S + \Delta] + \frac{\delta}{2}$

(ii) *(Dilation)* $\Pr_{Z \sim h}[Z \in S] \leq e^{\frac{\varepsilon}{2}} \Pr_{Z \sim h}[Z \in S \cdot e^{\lambda}] + \frac{\delta}{2}$

3. Private Selection

This section covers works on private selection from the literature, as this paper specifically addresses the private selection problem. Private selection refers to selecting the best item, or outcome, option from a set of possible outputs while ensuring the individual's data privacy. Formally, we want to build a private algorithm for a query $f : \mathcal{X} \to \mathcal{R}$ where all possible outcomes for f are discrete, e.g., categorical values. In the private selection setting is necessary a utility function $u : \mathcal{X} \times \mathcal{R} \to \mathbb{R}$ that maps a database $\mathbf{x}$ and an output $r \in \mathcal{R}$ to a utility score $u(\mathbf{x}, r)$. This utility function is application-based, and the higher the utility values are, the better the outcome is for the database.

We now review the prior art on differentially private data selection algorithms, which comprises the well-established exponential [29] and the report-noisy-max [15] mechanisms, as well as the recently proposed permute-and-flip [28] and the local dampening [17] mechanisms.

Exponential Mechanism. The exponential mechanism in the private selection setting is the *de facto* standard. It samples possible outputs from $\mathcal{R}$ with a probability that grows exponentially with their utility function u.

Definition 3.1 (Exponential Mechanism [29])**.** The exponential mechanism $\mathcal{M}^{\exp}_{u,\varepsilon}(\mathbf{x}, r)$ selects an outcome $r \in \mathcal{R}$ as follows: $\mathcal{M}^{\exp}_{u,\varepsilon}(\mathbf{x}, r) \propto \exp\left(\frac{\varepsilon u(\mathbf{x},r)}{2\Delta u}\right)$, where Δu is the global sensitivity of the utility function u.

McSherry and Talwar [29] showed that the exponential mechanism satisfies ε-DP through global sensitivity, i.e., using noise with scale modulated by the global sensitivity.

Definition 3.2 (Global Sensitivity [29])**.** Let $u : \mathcal{X} \times \mathcal{R} \to \mathbb{R}$ be a utility function that maps a pair of a database and an outcome to a score. The global sensitivity of u is:

$$\Delta u = \max_{r \in \mathcal{R}} \max_{\mathbf{x},\mathbf{y} \in \mathcal{X} \mid d(\mathbf{x},\mathbf{y}) \leq 1} |u(\mathbf{x}, r) - u(\mathbf{y}, r)|$$

Permute-and-flip. Another private selection algorithm, called Permute-and-Flip, was proposed by McKenna and Sheldon [28]. The algorithm works by iterating over the set of outcomes $\mathcal{R}$ in a random order, and for each element r, it flips a biased coin with a certain probability. If the flipped coin lands tails, then r is removed from all possible outcomes. Otherwise (if it lands heads), r is the returned outcome for the mechanism. The likelihood of obtaining heads follows an exponential pattern concerning the quality score, thereby boosting the mechanism to produce results with superior quality scores. While the permute-and-flip algorithm achieves ε-differential privacy, this guarantee only applies under the global sensitivity Δu. For problems with

high global sensitivity, the algorithm might suffer from reduced accuracy, as well as the exponential mechanism.

The work proves that the expected error of permute-and-flip is never worse than that of the exponential mechanism. Moreover, it shows that the exponential mechanism can be viewed as a rejection sampling algorithm that samples uniformly from the outcome set $\mathcal{R}$ with replacement. On the other hand, the permute-and-flip works like an exponential mechanism but sampling without replacement from $\mathcal{R}$.

Report-noisy-max. Nevertheless, another private selection algorithm is proposed by Dwork and Roth [15] called the report-noisy-max. The algorithm adds independent noise to each outcome utility score and returns the outcome with the highest noisy score. Dwork and Roth [15] proposes the algorithm with noise sampled by the Laplace distribution. However, the algorithm can be generalized to other noise distributions, such as the Gumbel and Exponential distributions.

This algorithm is a broad private selection method. Specifically, the report-noisy-max with the Exponential distribution, denoted by $\mathcal{N}^{\texttt{exp}}$, samples noise from $\texttt{Expo}\left(\varepsilon/2\Delta u\right)$, and this version also has strong utility guarantees shown by the Theorem 3.3. It is identical to permute-and-flip [13]. Moreover, the report-noisy-max with the Gumbel distribution $\texttt{Gumbel}\left(2\Delta u/\varepsilon\right)$ is identical to the exponential mechanism [14]. Nevertheless, the report-noisy-max only holds the differential privacy requirements under the global sensitivity of the utility function, which might lead to poor accuracy under certain scenarios [7, 22, 38, 39].

The report-noisy-max inadvertently discards information [12]. More precisely, without incurring any supplementary privacy costs, it can disclose an estimate of the difference between the two largest noisy utility values.

Theorem 3.3. Consider the report-noisy-max with exponential distribution $\mathcal{N}^{\texttt{exp}}$ algorithm. Let $\mathbf{x} \in \mathcal{X}$ be a fixed database, ξ be the error and $\mathcal{R}$ be the set of all possible outcomes. Then, for a given $t > 0$, the following inequalities hold:

(i) $Pr\left[\xi(\mathcal{N}^{\texttt{exp}}, \mathbf{x}) \geq \frac{2\Delta u\left(\ln(|\mathcal{R}|)+t\right)}{\varepsilon}\right] \leq e^{-t}$;

(ii) $\mathbb{E}\left(\xi(\mathcal{N}^{\texttt{exp}}, \mathbf{x})\right) \leq \frac{2\Delta u\left(\ln(|\mathcal{R}|)+1\right)}{\varepsilon}$.

Local Dampening Mechanism. In specific scenarios, the global sensitivity may not be suitable because the global sensitivity is large and the signal-to-sensitivity ratio (i.e. $^{\text{utility}}/_{\text{sensitivity}}$) is too low, implying inaccurate results. To address this issue, the local dampening mechanism [17] designs an instance-based sensitivity to work along with a novel mechanism based on the exponential one. It also proposes new adapted versions of the local sensitivity at a distance t to the private selection setup.

Definition 3.4 (Local Sensitivity for private selection [17])**.** Let $u : \mathcal{X} \times \mathcal{R} \to \mathbb{R}$ be a utility function that maps a pair of a database and an outcome to a score. The local sensitivity is defined as:

$$LS_u(\mathbf{x}) = \max_{r \in \mathcal{R}} \max_{\substack{\mathbf{y} \in \mathcal{X} \\ d(\mathbf{x},\mathbf{y}) \leq 1}} |u(\mathbf{x}, r) - u(\mathbf{y}, r)|$$

Definition 3.5 (Local Sensitivity at distance t for private selection [17])**.** Let $u : \mathcal{X} \times \mathcal{R} \to \mathbb{R}$ be a utility function that maps a pair of a database and an outcome to a score. The local sensitivity of a function u for the database $\mathbf{x}$ at distance t is defined as:

$$LS_u(\mathbf{x}, t) = \max_{\substack{\mathbf{y} \in \mathcal{X} \\ d(\mathbf{x},\mathbf{y}) \leq t}} LS_u(\mathbf{y})$$

Whereas the local sensitivity at distance t provides an overview of the utility u variation in its neighborhood, it lacks in granting more information about the utility function u with a specific outcome r in its neighborhood. Therefore, the paper [17] proposes a novel generalization of local sensitivity called the element local sensitivity. It measures the sensitivity of a utility function u for a specific outcome r at a distance t.

The computation of the element's local sensitivity is only sometimes feasible because it could be NP-hard. Therefore, the paper proposes a definition that represents an heuristic to compute an upper bound to the element's local sensitivity, named admissible function δ_u.

The local dampening attenuates the utility function in a specific way to make the signal-to-sensitivity ratio larger. This function is called D_{u,δ^u} and uses an admissible function δ^u that provides a dampened and scaled version of the original utility function. And finally, the local dampening mechanism $\mathcal{M}^{\texttt{dam}}_{u,\varepsilon,\delta_u}$ selects an element $r \in \mathcal{R}$ with probability proportional to $\exp\left(\frac{\varepsilon \cdot D_{u,\delta_u}(\mathbf{x},r)}{2}\right)$.

The local dampening mechanism satisfies ε-differential privacy if δ is admissible. It also performs at least equal to the exponential mechanism when the sensitivity function meets specific scenarios, such as stability. However, there are a few caveats to the local dampening mechanism, particularly related to the inversion problem, the necessity for stability, and the time complexity.

4. Smooth Noisy Max

This paper introduces Smooth Noisy Max (SNM), an algorithm that tackles the differentially private selection problem. The proposed algorithm is inspired by the report-noisy-max. SNM offers significant advantages over the existing methods, such as simplicity, ease of implementation, and accuracy performance. In particular, our novel approach adopts an instance-based sensitivity instead of global sensitivity, as the latter can often be excessively large, leading to a low signal-to-sensitivity ratio (i.e., $^{\text{utility}}/_{\text{sensitivity}}$) and, consequently, inaccurate results.

More precisely, SNM applies the smooth sensitivity.

Definition 4.1 (Smooth sensitivity, adapted from Nissim, Raskhodnikova, and Smith [30])**.** For $\beta > 0$, the β-smooth sensitivity of the utility function u is:

$$\mathcal{S}_{u,\beta}(\mathbf{x}) = \max_{t=0,1,\ldots,|\mathbf{x}|} \left(e^{-t\beta} \cdot LS_u(\mathbf{x}, t)\right)$$

The smooth sensitivity attenuates the local sensitivity (Definition 3.5) based on the distance from $\mathbf{x}$. Applying an instance-based sensitivity, such as smooth sensitivity, within a private selection algorithm is not always feasible for differential privacy. For instance, the exponential mechanism can not be used directly with the smooth sensitivity (see Theorem 7.2). On the other hand, the proposed Smooth Noisy Max algorithm can take advantage of the smooth framework and consequently decrease the signal-to-sensitivity ratio of the method. Additionally, it can keep the same differentially private guarantees of the standard report-noisy-max and ensures better accuracy.

SNM adds noise proportional to a smooth upper bound on the local sensitivity (e.g. smooth sensitivity $\mathcal{S}_{u,\beta}$) to its utility value for each possible outcome r for the query f at database $\mathbf{x}$, i.e., $u(\mathbf{x}, r)$. The noise, expressed by a random variable Z, is drawn from an (α, β)-admissible probability density function (Definition 2.9). For the sake of simplicity, we refer to $\mathcal{S}_{u,\beta}$ as $\mathcal{S}$. This procedure is explained in Algorithm 1.

Algorithm 1: Smooth Noisy Max Algorithm

for $r \in \mathcal{R}$ **do**
  $\tilde{u}(\mathbf{x}, r) \leftarrow u(\mathbf{x}, r) + \frac{2\mathcal{S}(\mathbf{x})}{\alpha} \cdot Z$;
return $\arg\max_{r \in \mathcal{R}} \tilde{u}(\mathbf{x}, r)$

4.1. Privacy Guarantees

In Theorem 4.2, we prove that the Smooth Noisy Max algorithm ensures (ε, δ)-differential privacy.

Theorem 4.2. The Smooth Noisy Max $\mathcal{A}_{u,\varepsilon}$ algorithm is (ε, δ)-differentially private if h is an (α, β)-admissible noise probability density function, and Z a random variable sampled according to h.

Proof. See proof on appendix A.1

Corollary 4.3. The Smooth Noisy Max $\mathcal{A}_{u,\varepsilon}$ algorithm with sampled noise from the Student's T distribution is ε-differentially private. By scaling the Student's T distribution in accordance with the smooth sensitivity, pure differential privacy is assured [8].

Corollary 4.4. The Smooth Noisy Max $\mathcal{A}_{u,\varepsilon}$ algorithm with sampled noise from the Laplace distribution is (ε, δ)-differentially private, when β parameter is defined by ${}^{\varepsilon}/_{2\log(2/\delta)}$ [30].

Corollary 4.5. The Smooth Noisy Max $\mathcal{A}_{u,\varepsilon}$ algorithm with sampled noise from the Laplace Log-Normal ($\mathtt{LLN}(\sigma)$) distribution is (ε, δ)-differentially private [8], when α parameter is defined as $e^{-{}^{3}/_{2}\sigma^{2}}\,(\varepsilon - {}^{\beta}/_{\sigma})$.

We can also improve the noise addition under the monotonicity property. The utility function u is monotonic in the database if adding an element to the database cannot cause the value of the function to decrease, e.g., counting queries.

Corollary 4.6. When the utility function u is monotonic in the database, then the Smooth Noisy Max $\mathcal{A}_{u,\varepsilon}$ scales the noise only by a factor of $\frac{\mathcal{S}(\mathbf{x})}{\alpha}$.

4.2. Utility Analysis

A significant characteristic of the Smooth Noisy Max algorithm is that it provides strong utility guarantees. Given a database $\mathbf{x}$, we can find the error bound of the private algorithm by a specific parameter t. The algorithm's accuracy is assessed based on the largest utility score $u^* = \max_{r \in \mathcal{R}} u(\mathbf{x}, r)$. It will be highly unlikely that the returned element r has a utility score significantly less than $O\left(u^* - ({}^{\mathcal{S}_{u,\beta}(\mathbf{x})}/_{\varepsilon}) \ln |\mathcal{R}|\right)$ when the noise distribution is Laplace.

Lemma 4.7. Given a fixed database $\mathbf{x} \in \mathcal{X}$, for the Smooth Noisy Max $\mathcal{A}$ algorithm with a standard Laplace distribution as noise function and any $t > 0$, the error $\xi(\mathcal{A}, \mathbf{x})$ satisfies

$$Pr[\xi(\mathcal{A}, \mathbf{x}) \geq t] \leq |\mathcal{R}| \exp\left(-\frac{\varepsilon t}{4\mathcal{S}_{u,\beta}(\mathbf{x})}\right).$$

Proof. See proof on appendix A.2

Theorem 4.8. Let $\mathbf{x} \in \mathcal{X}$ be a fixed database. Then, for a given $t > 0$, the Smooth Noisy Max $\mathcal{A}$ algorithm with standard Laplace noise distribution will have the following properties:

(i) $Pr\left[\xi(\mathcal{A}, \mathbf{x}) \geq \frac{4\mathcal{S}_{u,\beta}(\mathbf{x})\,(\ln(|\mathcal{R}|)+t)}{\varepsilon}\right] \leq e^{-t}$;
(ii) $\mathbb{E}\left(\xi(\mathcal{A}, \mathbf{x})\right) \leq \frac{4\mathcal{S}_{u,\beta}(\mathbf{x})\,(\ln(|\mathcal{R}|)+1)}{\varepsilon}$.

The utility bounds presented by Theorem 4.8 provide tools to compare and show that the Smooth Noisy Max outperforms our related work, i.e., report-noisy-max, exponential mechanism, and permute-and-flip. Firstly, we analyze the utility of the Smooth Noisy Max in contrast to the report-noisy-max with exponential noise, shown by Theorem 4.10.

Definition 4.9. An algorithm $\mathcal{A}$ is said to be *never worse* than some other algorithm $\mathcal{B}$ when, given a dataset $\mathbf{x}$:

(i) $Pr\left[\xi\left(\mathcal{A}, \mathbf{x}\right) \geq t\right] \leq Pr\left[\xi\left(\mathcal{B}, \mathbf{x}\right) \geq t\right]$ for all $t \geq 0$;
(ii) $\mathbb{E}[\xi\left(\mathcal{A}, \mathbf{x}\right)] \leq \mathbb{E}[\xi\left(\mathcal{B}, \mathbf{x}\right)]$.

Theorem 4.10. The Smooth Noisy Max $\mathcal{A}$ with Laplace noise distribution is *never worse* than $\mathcal{N}^{\exp}$ report-noisy-max algorithm with exponential noise when $\mathcal{S}_{u,\beta}(\mathbf{x}) \leq \frac{\Delta u}{2}$.

Proof. Using the lemma 4.7, we can obtain

$$Pr\left[\xi\left(\mathcal{A}, x\right) \geq t\right] \leq \frac{|\mathcal{R}|}{|\mathcal{R}_*|} \exp\left(-\frac{\varepsilon t}{4\mathcal{S}_{u,\beta}(\mathbf{x})}\right),$$

where $|\mathcal{R}_*|$ is the set of outcomes with the highest utility value. We observe that when $\mathcal{S}_{u,\beta}(\mathbf{x}) \leq \frac{\Delta u}{2}$, then

$$\frac{|\mathcal{R}|}{|\mathcal{R}_*|} \exp\left(-\frac{\varepsilon t}{4\mathcal{S}_{u,\beta}(\mathbf{x})}\right) \leq \frac{|\mathcal{R}|}{|\mathcal{R}_*|} \exp\left(-\frac{\varepsilon t}{2\Delta u}\right),$$
$$Pr\left[\xi\left(\mathcal{A}, \mathbf{x}\right) \geq t\right] \leq Pr\left[\xi\left(\mathcal{N}^{\exp}, \mathbf{x}\right) \geq t\right].$$

Furthermore, by the Theorem 3.3, the first statement (*i*) holds. We want to prove the second statement (*ii*). The

expected error can be expressed in terms of complementary cumulative distribution function:

$$\mathbb{E}(\xi(\mathcal{A}, x)) = \int_0^\infty Pr[\xi(\mathcal{A}, x) \geq t]dt.$$

We shown that $Pr\left[\xi\left(\mathcal{A}, x\right) \geq t\right] \leq Pr\left[\xi\left(\mathcal{N}^{\texttt{exp}}, x\right) \geq t\right]$, thus:

$$\mathbb{E}(\xi(\mathcal{A}, x)) - \mathbb{E}(\xi(\mathcal{N}^{\texttt{exp}}, x)) =$$
$$\int_0^\infty Pr\left[\xi\left(\mathcal{A}, x\right) \geq t\right] - Pr\left[\xi\left(\mathcal{N}^{\texttt{exp}}, x\right) \geq t\right] dt \leq 0.$$

Thus, the Smooth Noisy Max with Laplace noise distribution is never worse than the report-noisy-max algorithm with exponential noise when $\mathcal{S}_{u,\beta}(\mathbf{x}) \leq \frac{\Delta u}{2}$. □

Ding et al. [13] shows that the report-noisy-max with exponential noise is identical to the permute-and-flip, so as we know, by Theorem 4.10 the Smooth Noisy Max is never worse than report-noisy-max algorithm with exponential noise when $\mathcal{S}_{u,\beta}(\mathbf{x}) \leq \frac{\Delta u}{2}$, and consequently never worse than permute-and-flip mechanism under the same constraint.

The utility of our proposed method also outperforms the exponential mechanism and report-noisy-max with Gumbel noise when $\mathcal{S}_{u,\beta}(\mathbf{x}) \leq \frac{\Delta u}{2}$. Since, by transitivity, SNM surpass the permute-and-flip that exceeds the exponential mechanism [28]. Additionally, the exponential mechanism is identical to report-noisy-max with Gumbel noise [14], therefore the Smooth Noisy Max is *never worse* than report-noisy-max with Gumbel noise. All these results are expressed by Corollary 4.11.

Corollary 4.11. When $\mathcal{S}_{u,\beta}(\mathbf{x}) \leq \frac{\Delta u}{2}$, SNM $\mathcal{A}$ algorithm is *never worse* than $\mathcal{M}^{\texttt{pf}}$ permute-and-flip, $\mathcal{M}^{\texttt{exp}}$ exponential mechanism, and $\mathcal{N}^{\texttt{gum}}$ report-noisy-max with Gumbel noise.

The lack of utility bounds for the Local Dampening mechanism [17] hampers a comparative assessment with our Smooth Noisy Max. Nevertheless, the paper conducts an exhaustive empirical analysis in the subsequent sections.

Other commonly used admissible distributions include the Student's T and Laplace Log-Normal distributions [8]. The upper bounds of these utility functions may not readily suggest the better admissible noise distribution for a given problem. To aid in identifying a suitable distribution, one can use Chebyshev's inequality to compare the distributions by focusing on their variances. For example, consider a comparison between the Laplace distribution and the Student's T distribution. The variance of the Laplace distribution is $2b^2$, where b is the scale parameter. For the Student's T distribution with degrees of freedom d, the variance is defined as $\frac{d}{d-2}$ for $d > 2$. According to Chebyshev's inequality, the Student's T distribution has a lower upper bound than the Laplace distribution when $\frac{d}{d-2} < 2b^2$. Similarly, we can compare the Laplace distribution with the Laplace Log-Normal distribution, which has a variance of $2e^{2\sigma^2}$. When the scale parameter b satisfies $b > e^{\sigma^2}$, the Laplace Log-Normal distribution presents a lower upper bound than the Laplace distribution according to Chebyshev's inequality.

5. Application — Percentile Selection

In this section, we address the percentile selection problem. The task is to return the *p-th* percentile value from a set of real numbers. Nissim, Raskhodnikova, and Smith [30] and McKenna and Sheldon [28] have dealt with similar tasks. Nissim, Raskhodnikova, and Smith's work [30] addressed the challenge of privately releasing the numerical median of a dataset. McKenna and Sheldon [28] work attacks a similar problem, also for the median of the data, returning the bin value where it belongs.

5.1. Problem Statement

Given a dataset $\mathbf{x}$ represented as a vector $[x_1, \ldots, x_n]$. For simplicity's sake, assume that every database $\mathbf{x}$ is ordered such that $x_1 \leq \ldots \leq x_n$. Suppose that all the values lies in $[0, \Lambda]$, $0 \leq x_1 \leq \ldots \leq x_n \leq \Lambda$. The task is to return the percentile value where its element x_i is as close as possible to the p-th percentile element.

5.2. Private Mechanism and Sensitivity Analysis

Following the task statement, various private selection algorithms are applicable, including the exponential mechanism, permute-and-flip, local dampening, and our proposed Smooth Noisy Max variants. The algorithms select any value from a discrete subset of $[0, \Lambda]$, i.e., $\mathcal{R} \subseteq [0, \Lambda]$. We designed a utility function u_p that assigns a maximum score of 1 when element i matches the p-th element's value and a minimum score of 0 in all other cases; see Definition 5.1.

Definition 5.1 (Utility function for percentile selection problem)**.** Consider a database $\mathbf{x} \in \mathcal{X}$, $n = |\mathbf{x}|$, and $i \in \mathbb{Z}^{0+}$ a non-negative integer. The utility is defined as follows:

$$u_p(\mathbf{x}, i) = \begin{cases} 1, & \text{if } x_i = x_k, \text{where } k = \left\lfloor \frac{p \cdot n}{100} \right\rfloor; \\ 0, & \text{otherwise.} \end{cases}$$

Recall that the exponential mechanism and the permute-and-flip require the global sensitivity Δu_p, the local dampening requires the element local sensitivity, and the Smooth Noisy Max expects the smooth sensitivity $\mathcal{S}_{u_p}$.

Global Sensitivity. The following example can show a worst-case scenario. For instance, let $p = 50$ implying that $k = \left\lfloor \frac{n}{2} \right\rfloor$. Let $\mathbf{x}$ be a dataset with $n > 2$ and even, where $x_{<k} = 0$ and $x_{\geq k} = \Lambda$. Let $\mathbf{y}$ be a neighboring dataset of $\mathbf{x}$, where one element $x_{\geq k}$ has been removed. Thus we have $u(\mathbf{x}, k) = 1$, and $u(\mathbf{y}, k) = 0$ which implies that $u(\mathbf{x}, k) - u(\mathbf{y}, k) = 1$. Thus, $|u(\mathbf{x}, r) - u(\mathbf{y}, r)| \leq 1$ for all $r \in \mathcal{R}$, and any two neighboring datasets $\mathbf{x}, \mathbf{y}$.

Proposition 5.2. Let u_p be the utility function given by Definition 5.1. Then, the global sensitivity for percentile selection problem is $\Delta u_p = 1$.

$$\mathbf{x} = [2\ 3\ 5\ \ldots\ \overbrace{5}^{k}\ \ldots\ 5\ 6\ 7]$$
$$\boldsymbol{u}_p^{\mathbf{x}} = [\underbrace{0\ 0\ 1\ \ldots}_{\boldsymbol{j}_l}\ \ 1\ \ \underbrace{\ldots\ 1\ 0\ 0}_{\boldsymbol{j}_g}]$$
$$j = \max\left(\sum \boldsymbol{j}_l, \sum \boldsymbol{j}_g\right)$$

Figure 1. The utility function u_p maps elements of $\mathbf{x}$ to a utility value. In this example, the $x_k = 5$, and $\boldsymbol{u}_p^{\mathbf{x}}$ is the utility vector for the dataset $\mathbf{x}$. The subsets $\boldsymbol{j}_l$ and $\boldsymbol{j}_g$ partition the dataset into elements less than and greater than index k, respectively. The final equation computes j as the maximum of the summed utility values, i.e., the number of elements that have the same value of x_k in each partition.

Local Sensitivity. One must first compute the local sensitivity at a distance t to compute the smooth sensitivity. Let $\mathbf{x} \in \mathcal{X}$ be a dataset, and $j = \min\left(\sum_{i=0}^{k-1} u(\mathbf{x}, x_i), \sum_{i=k+1}^{n} u(\mathbf{x}, x_i)\right)$ the smallest sequence of p-th value repetition length at left or right of position k. Thus, the p-th percentile value will remain the same until $2j+1$ insertions and deletions from $\mathbf{x}$ because of the floor function in the k definition (Definition 5.1). Figure 1 provides an illustrative example of how j is computed.

Proposition 5.3. Let $\mathbf{x} \in \mathcal{X}$ be a dataset, $j = \min\left(\sum_{i=0}^{k-1} u(\mathbf{x}, x_i), \sum_{i=k+1}^{n} u(\mathbf{x}, x_i)\right)$ as described above, and u_p as in Definition 5.1. Then, the local sensitivity at distance t for the problem of percentile selection is given by:

$$LS_{u_p}(\mathbf{x}, t) = \begin{cases} 1, & \text{if } t \geq 2j+1; \\ 0, & \text{otherwise.} \end{cases}$$

Now, it is possible to calculate the smooth sensitivity of the percentile selection problem using the smooth sensitivity defined by Definition 4.1. The local sensitivity remains zero until $t < 2j+1$ and changes to one when $t \geq 2j+1$. Since the LS_{u_p} is constant when $t \geq 2j+1$, the smooth sensitivity will be max when $t = 2j+1$.

Proposition 5.4. The smooth sensitivity (as defined in Definition 4.1) of a dataset $\mathbf{x} \in \mathcal{X}$ considering the utility function u_p from Definition 5.1 is given by

$$\mathcal{S}_{u_p,\beta}(\mathbf{x}) = \exp(-(2j+1)\cdot\beta),$$

where $j = \min\left(\sum_{i=0}^{k-1} u(\mathbf{x}, x_i), \sum_{i=k+1}^{n} u(\mathbf{x}, x_i)\right)$.

5.3. Experimental Evaluation

Datasets. We tested PATENT, HEPTH, and INCOME datasets from Hay et al. [23]. The PATENT dataset contains 32,558 tuples with a high percentage of zero-valued entries at 97.80%. In contrast, the HEPTH dataset comprises 347,414 tuples but only 21.17% zero-valued entries, indicating more varied data. Lastly, the INCOME dataset is the largest, with 20,787,122 tuples and 44.97% zero-valued entries, reflecting a moderate level of homogeneity in its data. An essential attribute for those datasets is the amount of p-th value repetitions because of the utility function.

Methods. We consider six approaches to private percentile selection problem: i) exponential mechanism (EM) using global sensitivity; ii) permute-and-flip (PF) using global sensitivity; iii) local dampening (LD) using the element local sensitivity $\hat{\delta}(\mathbf{x}, t, r) = LS_{u_p}(\mathbf{x}, t) = \max_{r' \in \mathcal{R}} LS_{u_p}(\mathbf{x}, t, r)$ from utility function (Definition 5.1); iv) local dampening (LD2) utilizing both the utility and setup presented in Farias et al.'s work, adjusted to suit our specific problem statement; v) Smooth Noisy Max via Laplace distribution (SNM-LAP) with the smooth sensitivity $\mathcal{S}_{u_p}$; and vi) Smooth Noisy Max via Student's T distribution (SNM-T) with the smooth sensitivity $\mathcal{S}_{u_p}$.

Evaluation. We measured the absolute expected error (AEE) of each method for every specific scenario: $|\xi(\mathcal{A}, \mathbf{x})| = |x_k - \mathbb{E}(\mathcal{A}, \mathbf{x})|$. Understanding each outcome's associated probability is needed to find the expected value. Meanwhile, for the exponential mechanism, permute-and-flip, and local dampening, the probabilities of each outcome are straightforward to identify through the probability mass function of each mechanism. However, finding the probability of each outcome of the SNM algorithm is not straightforward. Reasoning about the output probability of other candidates is a condition for finding the probability of the output of a particular candidate. The specific candidate utility random variable should be greater than all others. This intricate probability function leads us to solve the following integral to find those probabilities numerically.

$$Pr[\mathcal{A}(\mathbf{x}) = r] = \int_{-\infty}^{\infty} f(i) \cdot \prod_{r \neq s} F\left(\frac{u_p(\mathbf{x}, r) - u_p(\mathbf{x}, s)}{N(\mathbf{x})} + i\right) di,$$

where $N(\mathbf{x}) = {}^{2\mathcal{S}_{u,\beta}(\mathbf{x})}/_{\alpha}$, f and F represent the probability density function and the cumulative density function of the distribution used, respectively. Figure 2 shows the result varying the privacy budget $\varepsilon \in [10^{-1}, 10^{2}]$ and $p = 50, 90, 99$. For the Student's T distribution the degree of freedom was set to 3. Each dataset has a ground-truth percentile value (GT) for each percentile. The desired behavior is that with a small privacy budget, the method outputs a value near the ground-truth value.

All versions of Smooth Noisy Max (SNM-T, SNM-LAP) have better accuracy when the dataset has several repetitions of the p-th value, i.e., a significant j value. For instance, the median perspective ($p = 50$) in the PATENT dataset has $j = 0$, HEPTH has $j = 2$, and INCOME has $j = 10$. The datasets show different scenarios to assess the SNM algorithms compared to the competitors. The EM and PF methods have similar expected values in all scenarios. For the HEPTH dataset with $p = 50$, SNM-T method achieves a similar expected value, the difference in absolute values of a maximum of 5, with 69% and 85% less privacy budget than LD and LD2 methods, respectively. For $p = 90$, the

SNM-T needs less than 51% and 76% privacy budget than the LD and LD2 methods, respectively. For $p = 99$, the behavior is similar when SNM-T requires less than 51% and 76% budget compared with LD and LD2. For the PATENT dataset, we observe up to 51%, 70% and 70%, for $p \in \{50, 90, 99\}$ respectively, in privacy budget saving when compared to LD. In the PATENT dataset with $p = 50$, LD2 method quickly reaches the desired value. And for $p \in \{90, 99\}$ we observe up to 70% of privacy budget saving when compared to LD2. For the INCOME dataset, when $p = 50$, the difference is more evident due to the high j value, but for $p \in \{90, 99\}$, the performance is quite the same in the other datasets.

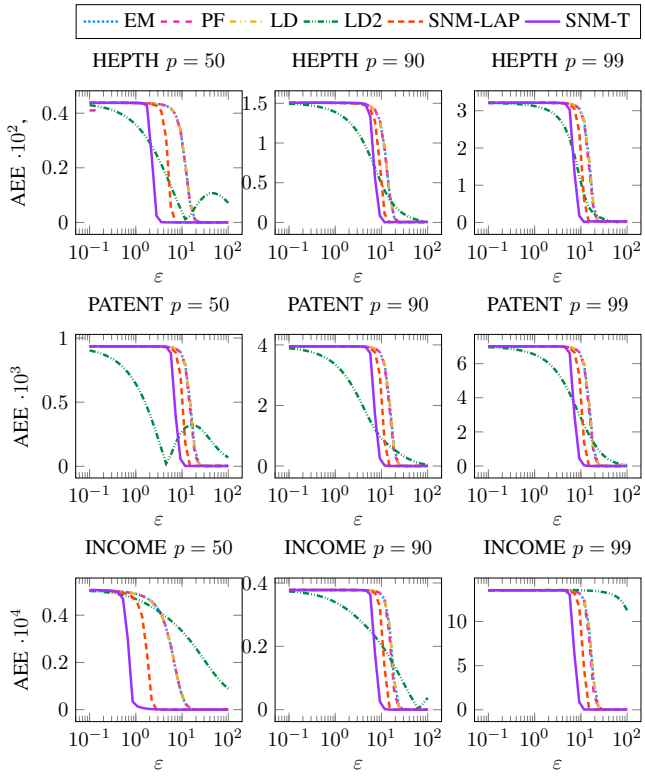

Figure 2. Comparison of private selection methods for percentile selection. Plots show the absolute expected error (AEE) as a function of the privacy budget $\varepsilon \in [10^{-1}, 10^2]$. The x-axis uses log scale. Overall, SNM-LAP and SNM-T achieve lower expected errors than other methods for all ε.

In the LD2 experiments on the HEPTH and PATENT datasets with $p = 50$, we observed a peculiar trend: the absolute expected error initially drops to low levels swiftly. However, as the privacy budget increases, the error, counterintuitively, increases. This rapid convergence appears to be coincidental, with the algorithm still in the process of converging. Figure 3 visually captures this behavior.

6. Application — Greedy Decision Tree

Decision trees are compelling methods for classification and regression tasks [26]. A decision tree is a graphical

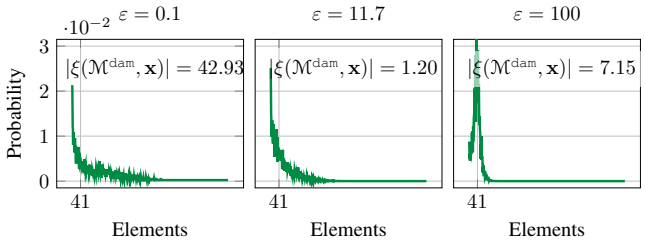

Figure 3. Local Dampening (LD2) probabilities on the HEPTH dataset with $p = 50$. The first plot demonstrates that the probability of selecting element 41 (median) is low when the privacy budget is minimal. The second graph depicts a scenario with very low expected error, suggesting that the observed low expected error occurs by chance. The last plot illustrates that with an increased privacy budget, LD2 converges effectively.

representation of a set of rules, where each node represents a decision based on attributes from the training dataset.

The tree topology is settled by the training algorithm that receives, as input, a dataset and outputs a decision tree. The ID3 algorithm [34] is one of the most known decision tree algorithms. It recursively selects the best attribute, according to some measure, to split the data until a stopping criterion is met. In this work, the split criterion is based on the Max Operator [21], which is the summation of each attribute value of the class with the highest frequency.

6.1. Problem Statement

A decision tree induction algorithm takes as input a dataset $\mathcal{T}$ with attributes $A = \{A_1, \ldots, A_d\}$ and a class attribute C and produces a decision tree. The task is to build a decision tree in a differentially private manner. Specifically, we base our approach on one of the most known differentially private tree induction algorithms, the Differentially Private ID3 algorithm [4].

6.2. Private Mechanism and Sensitivity Analysis

Blum et al. [4] introduced the SuLQ framework, where they design a differentially private version of ID3 as an application. The adapted application of the ID3 algorithm takes advantage of two SuLQ operators: i) `NoisyCount`: a Laplace mechanism operator to provide a private estimate for a count query and ii) `Partition`: an operator that splits the dataset into disjoint subsets.

The primary disadvantage of the ID3 algorithm proposed by Blum et al. is its inefficient use of the privacy budget when evaluating the information gain for each attribute separately. The work presented by Friedman and Schuster [21], described by Algorithm 2, offers a more effective alternative using the exponential mechanism to evaluate each attribute independently, assessing all attributes simultaneously in a single query, resulting in the selection of an appropriate attribute for splitting. Line 13 is the exponential mechanism call that selects an attribute based on its information gain, which is the utility function. The function `BuildDiffID3` in algorithm 2 starts by checking properties like the number

Algorithm 2: Differentially Private ID3 (from [21])

1 **Function** `GlobalDiffPID3(` *dataset* $\mathcal{T}$*, attribute set* A*, class attribute* C*, depth* d*, privacy budget* ε `)` **do**
2 $\varepsilon' \leftarrow \varepsilon/2\cdot(d+1)$;
3 **return** `BuildDiffPID3(`$\mathcal{T}, A, C, d, \varepsilon'$`)`
4 **Function** `BuildDiffPID3(` *dataset* $\mathcal{T}$*, attribute set* A*, class attribute* C*, depth* d*, privacy budget* ε `)` **do**
5 $t \leftarrow \max_{a \in A} |a|$;
6 $N_{\mathcal{T}} \leftarrow \texttt{NoisyCount}_{\varepsilon}(\mathcal{T})$;
7 **if** $A = \emptyset$ *or* $d = 0$ *or* $^{N_{\mathcal{T}}}/_{t|C|} < \sqrt{2}/2$ **then**
8 $\mathcal{T}_c \leftarrow \texttt{Partition}(\mathcal{T}, \forall c \in C : r_c = c)$;
9 $\forall c \in C : N_c \leftarrow \texttt{NoisyCount}_{\varepsilon}(\mathcal{T}_c)$;
10 **return** *a leaf labeled with* $\arg\max_c N_c$
11 $\bar{A} \leftarrow \mathcal{M}_{ig}^{\text{exp}}(\mathcal{T}, \varepsilon, A)$; `/* Exp. mechanism call */`
12 $\mathcal{T}_i \leftarrow \texttt{Partition}(\mathcal{T}, \forall i \in \bar{A} : r_{\bar{A}} = i)$;
13 $\forall i \in \bar{A} : \text{Subtree}_i \leftarrow$ `BuildDiffPID3(`$\mathcal{T}_i, A \backslash \bar{A}, C, d-1, \varepsilon$`)`;
14 **return** *a tree with a root node labeled* $\bar{A}$ *and edges labeled 1 to* $\bar{A}$ *each going to Subtree*$_i$

of attributes and the number of instances that are used as termination criteria to construct the leaves (lines 5-8). In lines 9-10, the algorithm partitions the dataset based on class labels and counts the instances for each class label. It also employs the Laplace mechanism for each class label count to select the class label for the leaf. Lines 13-16 build new decision rules recursively by privately choosing the attribute with the largest information gain value using the exponential mechanism. Moreover, it splits the dataset according to the selected attribute value and produces recursively new subtrees for each dataset partition.

Several works address the private construction of decision trees and random forest [18–20, 25, 31, 35]. However, only Farias et al. [17] addresses the greedy decision tree construction algorithm applying local sensitivity. The approach proposed by Fletcher and Islam [19] uses smooth sensitivity in the random forest algorithm through random decision trees. In this section, we focus on the greedy decision tree process. The following section will address the random forest application with random decision trees.

Our differentially private greedy decision tree application is similar to Algorithm 2. We simply replace the exponential mechanism on line 13 with our Smooth Noisy Max, applying a utility function based on the max operator [21] that represents the summation of each attribute value of the class with the highest frequency.

Definition 6.1 (Max Operator)**.** Consider a dataset $\mathcal{T}$, and an attribute A_i, the Max operator is defined as follows: $MaxOp(\mathcal{T}, A_i) = \sum_{j \in A_i} \max_c \tau_{j,c}^{A_i}$, where $\tau_{j,c}^{A_i}$ counts the records in $\mathcal{T}$ with attribute $A_i = j$ and class $C = c$.

In our experiments, we observed that we should design a utility function representing a good split criterion and take advantage of smooth sensitivity definition to benefit from local sensitivity. Therefore, we define a utility function based on the max operator u_{mo}. That function outputs 1 only for the attribute $A_i \in A$, which is the highest value of $MaxOp$ among all others $A_k \in A$, and 0 otherwise.

Definition 6.2 (Greedy decision tree utility)**.** Consider a dataset $\mathcal{T}$, and an attribute A_j, the utility is defined as:

$$u_{mo}(\mathcal{T}, A_j) = \begin{cases} 1, & \text{if } A_j = \arg\max_{A_i \in A} MaxOp(\mathcal{T}, A_i); \\ 0, & \text{otherwise.} \end{cases}$$

Global Sensitivity. The global sensitivity for u_{mo} is 1 [21].

Local Sensitivity. To compute the smooth sensitivity, it is crucial to have a clear understanding of the local sensitivity at a distance of t. Additionally, it is worth noting that the utility value will remain unchanged until k additions or deletions occur in the training dataset $\mathcal{T}$. Here, k refers to the difference between the highest $MaxOp$ attribute and the second-highest attribute in the dataset.

Proposition 6.3. Let $\mathcal{T}$ be a dataset, A_j an attribute, and the utility be as described in Definition 6.2. Then, the local sensitivity at distance t for a greedy decision tree is:

$$LS_{u_{mo}}(\mathcal{T}, t) = \begin{cases} 1, & \text{if } t \geq k; \\ 0, & \text{otherwise.} \end{cases}$$

The local sensitivity remains zero until $t < k$ and changes to one when $t \geq k$. Since the $LS_{u_{mo}}$ is constant when $t < k$, the smooth sensitivity will be max when $t = k$.

Proposition 6.4. The smooth sensitivity for a greed decision tree is given by $\mathcal{S}_{u_{mo}}(\mathcal{T}) = e^{-k \cdot \varepsilon}$, where $\mathcal{T}$ is a dataset, and u_{mo} is the utility function described in Definition 6.2.

6.3. Experimental Evaluation

Datasets. We consider three tabular datasets: i) The *National Long Term Care Survey (NLTCS)* [27], comprising 16 binary attributes of $21,574$ surveyed individuals; ii) the *American Community Surveys (ACS)* dataset [37], which includes information from $47,461$ rows with 23 binary attributes, sourced from the 2013 and 2014 ACS sample sets in IPUM-S-USA; and iii) the *Adult* dataset [3], containing $45,222$ records (excluding those with missing values), featuring 12 attributes, where 8 are discrete and 4 are continuous.

Methods. We experimented with several mechanisms, changing the default selection algorithm described in line 13 of the Algorithm 2. i) Exponential mechanism (EM) with information gain using global sensitivity; ii) Permute-and-flip (PF) with information gain using global sensitivity; iii) Shifted Local dampening (SLD) with information gain using the element local sensitivity [17]; iv) Smooth Noisy Max using the Laplace Log-Normal distribution (SNM-LLN); v) Smooth Noisy Max using the Student's T distribution (SNM-T); vi) Smooth Noisy Max using the Laplace distribution (SNM-LAP).

All variants of the Smooth Noisy Max algorithm use a utility function based on the Max Operator as the split criterion, leveraging smooth sensitivity $\mathcal{S}_{u_{mo}}$. Notably, SNM-LLN and SNM-LAP ensure approximate differential privacy ($\delta > 0$) rather than ε-differential privacy.

Evaluation. We measured the accuracy of each mechanism varying the max tree depth $d \in \{2, 5\}$ and the privacy budget $\varepsilon \in \{0.01, 0.05, 0.1, 0.5, 1.0, 2.0\}$. Each trial was measured using 10-fold validation, and each scenario ran 5 times. Figure 4 shows the average accuracy of those scenarios.

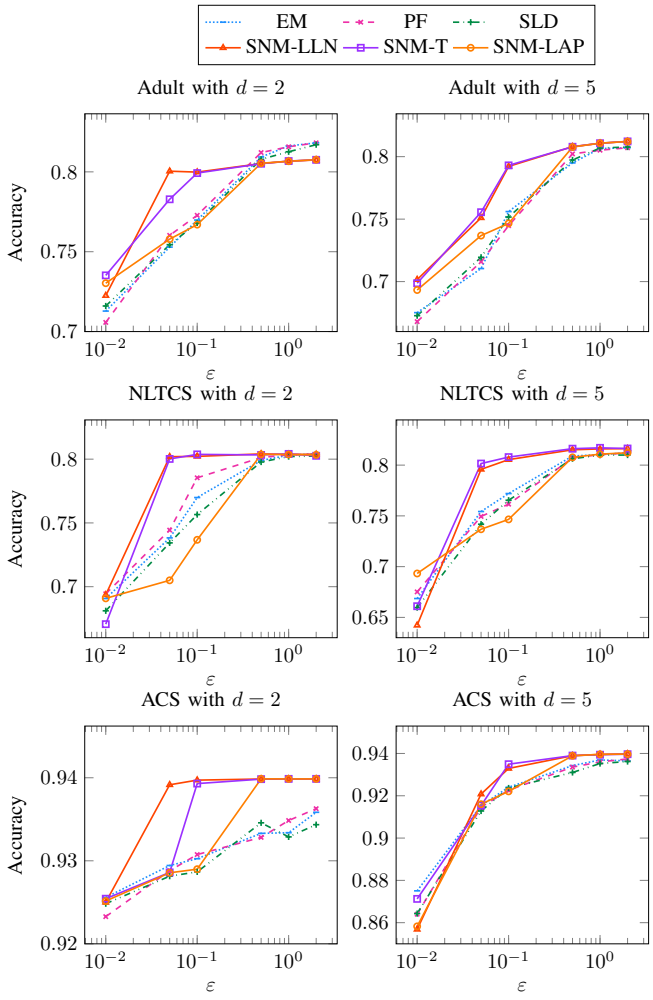

Figure 4. Comparison of private selection methods for the greedy decision tree application. The plots show the mean accuracy of greedy decision tree experiments - 5 runs of 10-fold cross-validation, where $d \in \{2, 5\}$ and $\varepsilon \in \{0.01, 0.05, 0.1, 0.5, 1, 2\}$. X axis is in log scale. All SNM variants consistently achieve superior accuracy compared to competing methods. Notably, the performance of SNM-T is especially significant, as it ensures ε-dp.

We observed that SNM with Student's T and Laplace Log-Normal are the best-performing method in most scenarios. In the Adult dataset with $d = 2$, SNM-LLN and SNM-T are the best-performing methods for small ε. However, when the budget is higher, all the competitors have better accuracy, indicating that with the Adult dataset with shallow trees ($d = 2$), the information gain split criteria works better than the max operator even with a higher signal-to-sensitivity ratio when compared with the max operator. Nevertheless, with $d = 5$, SNM-LLN and SNM-T perform better for all ε values. When the dataset is NLTCS, SNM-T has up to 8.58% of improvement in accuracy when compared with the competitors. SNM-T improves up to 1.15% with the ACS dataset compared to the other methods.

7. Application — Random Forest

Classification based on decision tree algorithms are remarkable tools for data mining [21]. They also serve as core building block for random forests [5]. Random forest is a supervised learning algorithm that combines the predictions of several decision trees, an ensemble of predictors. The algorithm starts by building a set of decision trees and then applies a majority voting to the outcomes of those trees.

The decision tree is a supervised learning algorithm based on a tree structure, where each intermediate node represents a decision based on a feature, and each leaf node represents a label. The algorithm starts from the root node and, based on comparing the feature value with a threshold on numerical features, it splits the tree. If the feature value exceeds the threshold, the algorithm goes to the right child node. Otherwise, it goes to the left child node. When the feature selected is categorical, the node has one child for each possible categorical value, and the comparison is made by checking the equality of attribute value. The algorithm continues until it reaches a leaf node when the node's majority label is the tree's outcome.

This section presents an application of a differentially private random forest algorithm using the Smooth Noisy Max as a selection mechanism. The method is a random decision tree designed to save privacy budget in the splitting process. We describe and test the random forest algorithm with several selection mechanisms, including our Smooth Noisy Max, under different scenarios and datasets to compare its results against our competitors.

7.1. Problem Statement

A random forest algorithm takes as input a dataset $\mathbf{x}$ with attributes $F = \{F_1, \ldots, F_d\}$, a max depth parameter h, and a parameter c that represents the forest size. The task is to build a forest with c trees $\mathcal{T} = \{\tau_1, \ldots, \tau_c\}$ in a differential private manner.

7.2. Random Decision Trees

The most common approaches to building a decision tree are ID3 [34], CART [6], and C4.5 [33]. They are based on some purity measures as splitting criteria. However, they have a lower generalization performance [5]. To overcome this problem, the random decision tree algorithm applies random splitting criteria. The generalization helps ensemble methods like the random forest to add diversity to the ensemble and, therefore, improve the performance [20].

In a greedy decision tree algorithm, the splitting process of a node depends on the input data in the same way that the leaf node class counts dictated by the data, which may leak some information. Considering information leakage,

we should prevent these privacy breaches using differential privacy. We must spend some privacy budget whenever data needs to be queried. So, seeking to save privacy budget, the random decision trees apply random split criteria to avoid the usage of privacy budget and save it for the leaf node class counts queries [20].

Fletcher and Islam [19] propose a random forest algorithm based on random decision trees that satisfies differential privacy. The algorithm applies the exponential mechanism in the leaves to select the majority label.

The work of Fletcher and Islam is summarized in Algorithm 3, which starts by splitting the dataset into c chunks. Then, each chunk x_i builds a random tree τ_i. Finally, the algorithm applies the exponential mechanism to select the majority label of the forest and adds the tree to the forest $\mathcal{T}$.

Algorithm 3: Random Forest Algorithm [19]

```
1  Function buildForest(Dataset x, Forest Size c, Features F, Depth h) do
2      for i ∈ split(x, c) do
3          τ ← setMajority(buildTree(x_i, F, h, 0));
4          T ← T ∪ τ ;
5  Function buildTree(Dataset x, Features F, Max Depth h, Depth d) do
6      T ← {};
7      if d < h then
8          Uniformly select attribute f from F to split current node;
9          if f is continuous then
10             Uniformly select split point p from the f's domain;
11             x_l, x_r ← split(x, f, p) ;
12             T ∪ buildTree(x_l, F, h, d + 1) ∪ buildTree(x_r, F, h, d + 1);
13     else
14         F ← F\f ;
15         forall a ∈ f do
16             x_a ← getData(x, f, a) ;
17             T ∪ buildTree(x_a, F, h, d + 1);
18     return T
```

In line 2, the dataset is partitioned into c chunks and iterated over it. The build tree function is called in line 3. The build tree function is a conventional recursive approach in that the features are randomly chosen for each node and the split point using only the data's domains, regardless of the data itself. The novel part of the proposed algorithm is the set majority function also in line 3. The set majority function applies the exponential mechanism to select the majority label of the leaf node through a specifically designed utility function. The proposed utility function, shown by Definition 7.1, outputs 1 for the label with the highest count in the leaf node and 0 otherwise.

Definition 7.1 (Utility Function [19])**.** The utility function u is defined as:

$$u(\mathbf{x}, r) = \begin{cases} 1, & \text{if } r = \arg\max_{i \in \mathcal{R}} n_i; \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

where n_i is the number of samples of class i in the leaf node.

The global sensitivity of the utility function (Definition 7.1) is 1. The work of Fletcher and Islam [19] applies the smooth sensitivity instead of global sensitivity to reach a better signal-to-noise ratio. However, as proven in the Theorem 7.2 below, it does not satisfy differential privacy.

Theorem 7.2. The exponential mechanism setting $\mathcal{M}^{\exp}_{u,\varepsilon}(\mathbf{x}, r) \propto \exp\left(\frac{\varepsilon u(\mathbf{x},r)}{2\mathcal{S}_{u,\beta}(\mathbf{x})}\right)$ does not satisfy ε-differential privacy with smooth sensitivity instead of global sensitivity.

Proof. Assuming that the exponential mechanism with smooth sensitivity satisfies ε-differential privacy, consider an approval voting example. Here, voters can endorse multiple candidates instead of choosing just one. In this scenario, the utility function assigns a value of 1 to the candidate with the highest votes and 0 to all others.

The utility function exhibits a smooth sensitivity of $\mathcal{S}_{u,\beta}(\mathbf{x}) = \exp(-j\varepsilon)$, where j is the vote disparity between the top candidate and the runner-up in dataset $\mathbf{x}$ (Theorem 7.3). The local sensitivity of the utility function u remains zero until the vote gap j is large enough to affect the comparison, at which point it jumps to 1. The smooth sensitivity peaks when $t = j$, yielding $\mathcal{S}_{u,\beta}(\mathbf{x}) = \exp(-j\varepsilon)$.

For example, consider the output set $\mathcal{R} = [\text{C1}, \text{C2}, \text{C3}, \text{C4}, \text{C5}]$ with the vote count vector $\mathbf{v} = [22, 8, 17, 4, 0]$ from dataset $\mathbf{x}$. Here, candidate C1 leads with 22 votes, followed by others, with C2 receiving 8 votes, and so forth. The utility function (Definition 7.1) assigns a score of 1 solely to candidate C1. The vote difference between the leading candidate and the second-most voted, denoted as j, is 5.

To ensure the differential privacy definition is necessary to address all possible neighboring datasets from $\mathbf{x}$, for instance, the dataset $\mathbf{y}$ by adding one more vote for the second-most voted candidate (C3). Therefore, the j parameter reduces to the value for 4, implying a smooth sensitivity value of 0.135. Using the privacy budget as 0.5, we have $Pr[\mathcal{M}^{\exp}_{u,0.5}(\mathbf{x}, \text{C3})] = 0.04$ and $Pr[\mathcal{M}^{\exp}_{u,0.5}(\mathbf{y}, \text{C3})] = 0.10$, following the Definition 2.1:

$$Pr[\mathcal{M}^{\exp}_{u,0.5}(\mathbf{x}, \text{C3})] \leq e^{0.5} Pr[\mathcal{M}^{\exp}_{u,0.5}(\mathbf{y}, \text{C3})] \Rightarrow$$
$$0.04 \leq 0.16 \Rightarrow \top$$
$$Pr[\mathcal{M}^{\exp}_{u,0.5}(\mathbf{y}, \text{C3})] \leq e^{0.5} Pr[\mathcal{M}^{\exp}_{u,0.5}(\mathbf{x}, \text{C3})] \Rightarrow$$
$$0.10 \leq 0.07 \Rightarrow \bot$$

Therefore, by contradiction, the exponential mechanism setting does not hold ε-differential privacy with smooth sensitivity instead of global sensitivity. □

To address that issue, we replace the exponential mechanism with our SNM in Fletcher and Islam's random forest algorithm as the differentially private selection procedure.

Algorithm 4 details our set majority function, which implements the Smooth Noisy Max algorithm. It begins by traversing all leaves of the tree τ (line 2). For each leaf l, the algorithm retrieves the label counts of the leaf node in line 3. Subsequently, the Smooth Noisy Max algorithm is

Algorithm 4: Set Majority Labels with Smooth Noisy Max

```
1 Function setMajority(Tree τ) do
2     for l ∈ ℓ do
3         labelCounts ← l.counts ;
4         l.maj ← SNM(labelCounts);
```

applied to select the majority label of the leaf node in line 4. It is important to note that to execute the Smooth Noisy Max algorithm, the smooth sensitivity of the utility function is required, as demonstrated in Theorem 7.3.

Theorem 7.3 (Smooth sensitivity of Def. 7.1 [20])**.** The smooth sensitivity of the utility function u (definition 7.1) is: $\mathcal{S}_{u,\beta}(\mathbf{x}) = \exp(-j\varepsilon)$, where j is the difference between the most frequent and the second-most frequent labels in $\mathbf{x}$.

7.3. Experimental Evaluation

This section presents the datasets, methods, and experimental evaluation results. We selected six datasets to evaluate the performance of our proposed method compared with other baselines.

Methods. Our evaluation employs the standard random forest algorithm (Algorithm 3). We term the non-private implementation of Algorithm 3 as WDP. The experiment employs various selection mechanisms, including the exponential mechanism (EM), permute-and-flip (PF), local dampening mechanism (LD), Smooth Noisy Max with Laplace Log-Normal distribution (SNM-LLN), Smooth Noisy Max with Student's T distribution (SNM-T), and Smooth Noisy Max with Laplace distribution (SNM-LAP). We configure all privacy-preserving mechanisms, excluding the non-private method, with the utility function defined in Definition 7.1. EM and PF utilize a global sensitivity of 1.0. We empirically determine the element local sensitivity across each dataset for local dampening. We follow Theorem 7.3 to find the smooth sensitivity within our Smooth Noisy Max. Our goal is to measure the accuracy impact of choosing the Smooth Noisy Max as a private selection method.

Evaluation. We measured the accuracy of those methods over ten executions using the accuracy metric. The process split the dataset in 80% for the training step and 20% for evaluation purposes. The privacy budget varies by $\{0.01, 0.05, 0.1, 1, 2\}$. We also set each random forest with 32 trees. The max tree depth was set for each dataset using the Theorem 2 from Fletcher and Islam's work [20].

Results. Our experimental procedure compares our method using the accuracy metric in 6 datasets. The datasets were selected based on their size, number of features, and number of classes. The Adult dataset comprises 48,842 instances with 6 continuous and 8 discrete features, a maximum tree depth of 9, and 2 classes. Compas contains 4,732 entries, 9 continuous and 4 discrete features, a depth of 5, and 11

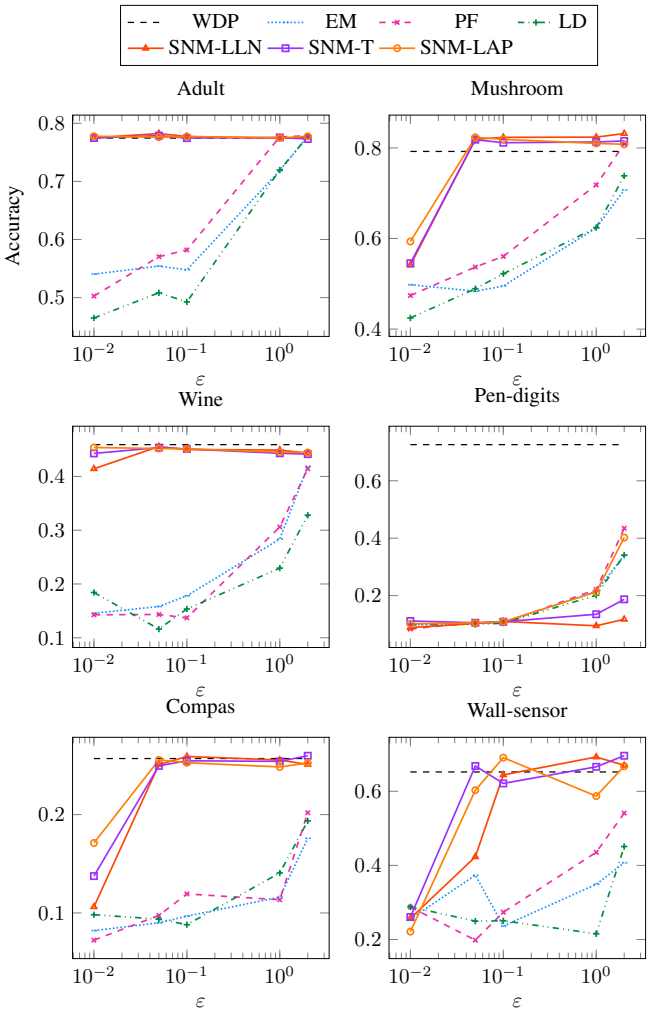

Figure 5. Comparison of private selection methods for the random forest problem. The plots show mean accuracy for WDP, EM, PF, LD, and SNM variants of random forest with 32 random trees varying $\varepsilon \in \{0.01, 0.05, 0.1, 1, 2\}$. X is in log scale. The SNM flavors constantly reach the standard non-private random forest accuracy level. When compared with other private selection methods, the variants of SNM surpass in almost all ε values.

classes. The Wine dataset involves 4,898 samples, all 11 features being continuous, a depth of 10, and 7 classes. Mushroom includes 8,124 entries, 22 discrete features, a maximum depth of 11, and 2 classes. The Pen-digits dataset, one of the largest with 109,092 instances, features 17 continuous attributes, a depth of 12, and 10 classes. Finally, Wall-sensor offers 5,456 samples, 4 continuous features, a depth of 4, and 4 classes. Figure 5 shows the result of our proposed random forest algorithm using the Smooth Noisy Max with the other selection algorithms varying the budget parameter.

Firstly, we focus on the experiments using the Mushroom [36] and Adult [2] datasets. Both datasets have mostly discrete attributes and few classes but differ in size. The Adult dataset has more than 48 thousand tuples compared to almost 8 thousand in the Mushroom dataset. The max

depth was set to 9 and 11 for Adult and Mushroom datasets. Looking at the results of our experiment in the Adult dataset, we can observe that even with a small privacy budget, we can deliver excellent accuracy results, achieving the version without private guarantees. Using the Mushroom dataset, when the budget is 0.1, all the versions of Smooth Noisy Max surpasses the standard random forest, i.e., our private method is better than the privateless version. By our method, the randomness input can improve the power of the tree generalization [5], leading to better accuracy. Fixing with a privacy budget of 1, our method is similar to permute-and-flip's accuracy performance, but using the privacy budget of 0.01, we can deliver the same accuracy performance as the permute-and-flip with 100 times more budget (see Figure 5).

The Wine Quality dataset [11] has almost 5 thousand records with 11 continuous features and zero discrete features. The Pen-Based Recognition of Handwritten Digits (Pen-digits) dataset [16] has more than 109 thousand records with 17 continuous features and zero discrete features. The maximum depth was set to 10 and 12 for the wine and pen-digit datasets. The random forest results with all the versions of SNM reach almost the non-private version in the wine dataset, outperforming all the adversaries even with very low privacy budget values. In the experiments using the pen-digits dataset, all the private methods underperform, mainly because of the dataset's j (Theorem 7.3) value. The difference between the highest class count and the second highest is narrow in the pen-digits dataset, implying a smooth sensitivity almost equal to global sensitivity.

The Compas and the Wall-sensor datasets have similar sizes but differ in the number of classes. The Compas (Correctional Offender Management Profiling for Alternative Sanctions) dataset [32] has 11 classes, and the Wall-Following Robot Navigation Dataset (Wall-sensor) [1] has only four classes. Figure 5 shows that our proposed SNM versions outperform the private selection adversaries using the Compas and wall-sensor datasets. Even with many classes, the proposed random forest algorithm employing SNM with a small privacy budget reaches the standard random forest without any privacy concerns.

8. Conclusion

This paper introduces the Smooth Noisy Max, a novel differentially private selection algorithm. We formally describe our approach, its privacy attributes, and its utility. We demonstrate that under mild conditions, our algorithm's utility, leveraging the Laplace distribution, consistently matches or exceeds that of competing methods while satisfying differential privacy criteria. Smooth Noisy Max utilizes local sensitivity across various private selection scenarios. Additionally, we comprehensively compare our mechanism against established methods such as local dampening, report-noisy-max, permute-and-flip, and exponential mechanisms. We empirically evaluated our approach on three different applications: i) Percentile selection; ii) Greedy decision trees; and, iii) Random forest.

In the experiments, we faced a limitation of our proposed algorithm. The notion of local sensitivity at a distance t quickly converges to global sensitivity due to the max operator (from Definition 3.5) iterating over all the possible outcomes. It was necessary to design specific utility functions to overcome that situation, e.g., only the best answer has a non-zero utility. Another limitation of smooth sensitivity is its computational complexity, which leads to algorithms with high time demands. To address this challenge, we have employed simple utility functions that the smooth sensitivity is analytically solvable.

We presume that applying the notion of element local sensitivity [17] should solve this limitation. As future work, we aim to prove the use of the element local sensitivity with the Smooth Noisy Max. Additionally, formalizing problems as single functions can be challenging, as many real-world situations are complex and involve multiple objectives. While theoretically possible, combining these through a single function, such as weighting or ranking them, might not always perfectly capture the nuances of some problems. Therefore, differentially private multi-objective selection is a research topic in our pipeline.

Acknowledgments

This work was partially supported by CAPES/Brazil under grant number 88882.454584/2019-01. This work was also supported by the Laboratório de Sistemas e Banco de dados (LSBD), the Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro FAPERJ (SEI-260003/000709/2023), the São Paulo Research Foundation FAPESP (2023/00815-6), and the Conselho Nacional de Desenvolvimento Científico e Tecnológico CNPq (404336/2023-0).

References

[1] Marcus Veloso Ananda Freire and Guilherme Barreto. *Wall-Following Robot Navigation Data*. 2009. DOI: 10.24432/C57C8W. URL: https://archive.ics.uci.edu/dataset/194.

[2] Barry G. Becker and Ronny Kohavi. *Adult*. https://doi.org/10.24432/C5XW20. Accessed on YYYY-MM-DD. Apr. 1996. DOI: 10.24432/C5XW20. URL: https://doi.org/10.24432/C5XW20.

[3] Catherine L Blake and Christopher J Merz. *UCI repository of machine learning databases*. 1998.

[4] Avrim Blum et al. "Practical privacy: the SuLQ framework". In: *Proceedings of the Twenty-fourth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 13-15, 2005, Baltimore, Maryland, USA*. Ed. by Chen Li. ACM, 2005, pp. 128–138. DOI: 10.1145/1065167.1065184. URL: https://doi.org/10.1145/1065167.1065184.

[5] Leo Breiman. "Random Forests". In: *Mach. Learn.* 45.1 (2001), pp. 5–32. DOI: 10.1023/A:1010933404324. URL: https://doi.org/10.1023/A:1010933404324.

[6] Leo Breiman et al. *Classification and Regression Trees*. Wadsworth, 1984. ISBN: 0-534-98053-8.

[7] Mark Bun and Thomas Steinke. "Average-Case Averages: Private Algorithms for Smooth Sensitivity and Mean Estimation". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al. 2019, pp. 181–191. URL: https://proceedings.neurips.cc/paper/2019/hash/3ef815416f775098fe977004015c6193-Abstract.html.

[8] Mark Bun and Thomas Steinke. "Average-Case Averages: Private Algorithms for Smooth Sensitivity and Mean Estimation". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al. 2019, pp. 181–191. URL: https://proceedings.neurips.cc/paper/2019/hash/3ef815416f775098fe977004015c6193-Abstract.html.

[9] Kamalika Chaudhuri, Anand D. Sarwate, and Kaushik Sinha. "A near-optimal algorithm for differentially-private principal components". In: *J. Mach. Learn. Res.* 14.1 (2013), pp. 2905–2943. DOI: 10.5555/2567709.2567754. URL: https://dl.acm.org/doi/10.5555/2567709.2567754.

[10] Rui Chen et al. "Differentially Private High-Dimensional Data Publication via Sampling-Based Inference". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*. Ed. by Longbing Cao et al. ACM, 2015, pp. 129–138. DOI: 10.1145/2783258.2783379. URL: https://doi.org/10.1145/2783258.2783379.

[11] Paulo Cortez et al. "Modeling wine preferences by data mining from physicochemical properties". In: *Decis. Support Syst.* 47.4 (2009), pp. 547–553. DOI: 10.1016/J.DSS.2009.05.016. URL: https://doi.org/10.1016/j.dss.2009.05.016.

[12] Zeyu Ding et al. "Free gap estimates from the exponential mechanism, sparse vector, noisy max and related algorithms". In: *VLDB J.* 32.1 (2023), pp. 23–48. DOI: 10.1007/S00778-022-00728-2. URL: https://doi.org/10.1007/s00778-022-00728-2.

[13] Zeyu Ding et al. "The Permute-and-Flip Mechanism is Identical to Report-Noisy-Max with Exponential Noise". In: *CoRR* abs/2105.07260 (2021). arXiv: 2105.07260. URL: https://arxiv.org/abs/2105.07260.

[14] David Durfee and Ryan M. Rogers. "Practical Differentially Private Top-k Selection with Pay-what-you-get Composition". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al. 2019, pp. 3527–3537. URL: https://proceedings.neurips.cc/paper/2019/hash/b139e104214a08ae3f2ebcce149cdf6e-Abstract.html.

[15] Cynthia Dwork and Aaron Roth. "The Algorithmic Foundations of Differential Privacy". In: *Found. Trends Theor. Comput. Sci.* 9.3-4 (2014), pp. 211–407. DOI: 10.1561/0400000042. URL: https://doi.org/10.1561/0400000042.

[16] Fevzi. Alimoglu E. Alpaydin. *Pen-Based Recognition of Handwritten Digits*. 1996. DOI: 10.24432/C5MG6K. URL: https://archive.ics.uci.edu/dataset/81.

[17] Victor A. E. de Farias et al. "Local dampening: differential privacy for non-numeric queries via local sensitivity". In: *VLDB J.* 32.6 (2023), pp. 1191–1214. DOI: 10.1007/S00778-022-00774-W. URL: https://doi.org/10.1007/s00778-022-00774-w.

[18] Sam Fletcher and Md Zahidul Islam. "A Differentially Private Decision Forest". In: *Thirteenth Australasian Data Mining Conference, AusDM 2015, Sydney, Australia, August 2015*. Ed. by Kok-Leong Ong et al. Vol. 168. CRPIT. Australian Computer Society, 2015, pp. 99–108. URL: http://crpit.scem.westernsydney.edu.au/abstracts/CRPITV168Fletcher.html.

[19] Sam Fletcher and Md Zahidul Islam. "Differentially private random decision forests using smooth sensitivity". In: *Expert Syst. Appl.* 78 (2017), pp. 16–31. DOI: 10.1016/J.ESWA.2017.01.034. URL: https://doi.org/10.1016/j.eswa.2017.01.034.

[20] Sam Fletcher and Md. Zahidul Islam. "Decision Tree Classification with Differential Privacy". In: *ACM Computing Surveys* 52.4 (Aug. 2019), pp. 1–33. DOI: 10.1145/3337064.

[21] Arik Friedman and Assaf Schuster. "Data mining with differential privacy". In: *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*. Ed. by Bharat Rao et al. ACM, 2010, pp. 493–502. DOI: 10.1145/1835804.1835868. URL: https://doi.org/10.1145/1835804.1835868.

[22] Alon Gonem and Ran Gilad-Bachrach. "Smooth Sensitivity Based Approach for Differentially Private PCA". In: *Algorithmic Learning Theory, ALT 2018, 7-9 April 2018, Lanzarote, Canary Islands, Spain*. Ed. by Firdaus Janoos, Mehryar Mohri, and Karthik Sridharan. Vol. 83. Proceedings of Machine Learning Research. PMLR, 2018, pp. 438–450. URL: http://proceedings.mlr.press/v83/gonem18a.html.

[23] Michael Hay et al. "Principled Evaluation of Differentially Private Algorithms using DPBench". In: *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*. Ed. by Fatma Özcan, Georgia Koutrika, and Sam Madden. ACM, 2016, pp. 139–154. DOI: 10.1145/2882903.2882931. URL: https://doi.org/10.1145/2882903.2882931.

[24] Ihab F Ilyas, George Beskales, and Mohamed A Soliman. "A survey of top-k query processing techniques in relational database systems". In: *ACM Computing Surveys (CSUR)* 40.4 (2008), pp. 1–58.

[25] Geetha Jagannathan, Krishnan Pillaipakkamnatt, and Rebecca N. Wright. "A Practical Differentially Private Random Decision Tree Classifier". In: *ICDM Workshops 2009, IEEE International Conference on Data Mining Workshops, Miami, Florida, USA, 6 December 2009*. Ed. by Yücel Saygin et al. IEEE Computer Society, 2009, pp. 114–121. DOI: 10.1109/ICDMW.2009.93. URL: https://doi.org/10.1109/ICDMW.2009.93.

[26] Sotiris B Kotsiantis, Ioannis Zaharakis, P Pintelas, et al. "Supervised machine learning: A review of classification techniques". In: *Emerging artificial intelligence applications in computer engineering* 160.1 (2007), pp. 3–24.

[27] Kenneth G Manton. "National Long Term Care Survey". In: *Encyclopedia of Aging, Second Edition. Springer, New York* (1999).

[28] Ryan McKenna and Daniel Sheldon. "Permute-and-Flip: A new mechanism for differentially private selection". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle et al. 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/01e00f2f4bfcbb7505cb641066f2859b-Abstract.html.

[29] Frank McSherry and Kunal Talwar. "Mechanism Design via Differential Privacy". In: *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*. IEEE Computer Society, 2007, pp. 94–103. DOI: 10.1109/FOCS.2007.41. URL: https://doi.org/10.1109/FOCS.2007.41.

[30] Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. "Smooth sensitivity and sampling in private data analysis". In: *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*. Ed. by David S. Johnson and Uriel Feige. ACM, 2007, pp. 75–84. DOI: 10.1145/1250790.1250803. URL: https://doi.org/10.1145/1250790.1250803.

[31] Abhijit Patil and Sanjay Singh. "Differential private random forest". In: *2014 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2014, Delhi, India, September 24-27, 2014*. IEEE, 2014, pp. 2623–2630. DOI: 10.1109/ICACCI.2014.6968348. URL: https://doi.org/10.1109/ICACCI.2014.6968348.

[32] ProPublica. *GitHub - propublica/compas-analysis: Data and analysis for "Machine Bias" — github.com*. https://github.com/propublica/compas-analysis. [Accessed 22-Feb-2023]. 2016.

[33] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993. ISBN: 1-55860-238-0.

[34] J. Ross Quinlan. "Induction of Decision Trees". In: *Mach. Learn.* 1.1 (1986), pp. 81–106. DOI: 10.1023/A:1022643204877. URL: https://doi.org/10.1023/A:1022643204877.

[35] Santu Rana, Sunil Kumar Gupta, and Svetha Venkatesh. "Differentially Private Random Forest with High Utility". In: *2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14-17, 2015*. Ed. by Charu C. Aggarwal et al. IEEE Computer Society, 2015, pp. 955–960. DOI: 10.1109/ICDM.2015.76. URL: https://doi.org/10.1109/ICDM.2015.76.

[36] UCI Machine Learning Repository. *Mushroom*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5959T. 1987.

[37] Integrated Public Use Microdata Series. "Version 6.0". In: *Minneapolis: University of* (2015).

[38] Lichao Sun et al. "Differentially Private Deep Learning with Smooth Sensitivity". In: *CoRR* abs/2003.00505 (2020). arXiv: 2003.00505. URL: https://arxiv.org/abs/2003.00505.

[39] Jun Zhang et al. "Private Release of Graph Statistics using Ladder Functions". In: *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*. Ed. by Timos K. Sellis, Susan B. Davidson, and Zachary G. Ives. ACM, 2015, pp. 731–745. DOI: 10.1145/2723372.2737785. URL: https://doi.org/10.1145/2723372.2737785.

[40] Jun Zhang et al. "PrivBayes: private data release via bayesian networks". In: *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*. Ed. by Curtis E. Dyreson, Feifei Li, and M. Tamer Özsu. ACM, 2014, pp. 1423–1434. DOI: 10.1145/2588555.2588573. URL: https://doi.org/10.1145/2588555.2588573.

Appendix A.
Smooth Noisy Max proofs

A.1. Privacy proof

Theorem 4.2. The Smooth Noisy Max $\mathcal{A}_{u,\varepsilon}$ algorithm is (ε, δ)-differentially private if h is an (α, β)-admissible noise probability density function, and Z a random variable sampled according to h.

Proof of Theorem 4.2. Consider two neighbor databases $\mathbf{x}$ and $\mathbf{y}$. Fix any $i \in \mathcal{R}$ and let $\vec{z_i} = \{z_1, \ldots, z_{|\mathcal{R}|}\} \backslash \{z_i\}$ be the fixed noises for all outputs except the ith output. We will argue for each $\vec{z_i}$ independently, similarly to what was done by Dwork and Roth [15] (Claim 3.9). For simplicity of notation, denote $N(\mathbf{x}) = {}^{2\mathcal{S}_{u,\beta}(\mathbf{x})}/_{\alpha}$, and the Smooth Noisy Max as $\mathcal{A}$. Then, the probability of $i \in \mathcal{R}$ being the output of the algorithm, given the noises $\vec{z_i}$, is

$$Pr[\mathcal{A}(\mathbf{x}) = i | \vec{z_i}] = Pr\left[u(\mathbf{x}, i) + N(\mathbf{x}) \cdot Z \geq \max_{j \in \mathcal{R}; j \neq i} \{u(\mathbf{x}, j) + z_j\}\right].$$

Let $\tilde{u}_* = \max_{j \in \mathcal{R}; j \neq i} \{u(\mathbf{x}, j) + z_j\}$ and $\tilde{u}'_* = \max_{j \in \mathcal{R}; j \neq i} \{u(\mathbf{y}, j) + z_j\}$. Then:

$$Pr[\mathcal{A}(\mathbf{x}) = i | \vec{z_i}] = Pr\left[Z \geq \frac{\tilde{u}_* - u(\mathbf{x}, i)}{N(\mathbf{x})}\right].$$

For the sake of simplicity, define $g(i) = \frac{\tilde{u}_* - u(\mathbf{x}, i)}{N(\mathbf{x})}$ and $g'(i) = \frac{\tilde{u}'_* - u(\mathbf{y}, i)}{N(\mathbf{y})}$, so that:

$$Pr[\mathcal{A}(\mathbf{x}) = i | \vec{z_i}] = Pr\left[Z \geq g(i)\right].$$

Using the definition 2.2 for neighboring databases $\mathbf{x}, \mathbf{y}$, and letting $Z_X \sim \mathcal{A}(\mathbf{x})$, $Z_Y \sim \mathcal{A}(\mathbf{y})$:

$$D_\infty^\delta(Z_X || Z_Y) = \max_{\substack{S \subseteq \mathcal{R}: \\ Pr[Z_X \in S] \geq \delta}} \left[\log\left(\frac{Pr[Z_X \in S] - \delta}{Pr[Z_Y \in S]}\right)\right].$$

As our algorithms draws results from the discrete set of outputs, we can write:

$$\begin{aligned}
D_\infty^\delta(Z_X || Z_Y) &= \max_{\substack{S \subseteq \mathcal{R}: \\ Pr[Z_X \in S] \geq \delta}} \left[\log\left(\frac{\sum_{r \in S} Pr[Z_X = r] - \delta}{\sum_{r \in S} Pr[Z_Y = r]}\right)\right], \\
&= \max_{\substack{S \subseteq \mathcal{R}: \\ Pr[Z_X \in S] \geq \delta}} \left[\log\left(\frac{\sum_{r \in S} \int Pr[Z_X = r | \vec{z_r}] Pr[\vec{z_r}] d\vec{z_r} - \delta}{\sum_{r \in S} \int Pr[Z_Y = r | \vec{z_r}] Pr[\vec{z_r}] d\vec{z_r}}\right)\right], \\
&= \max_{\substack{S \subseteq \mathcal{R}: \\ Pr[Z_X \in S] \geq \delta}} \left[\log\left(\frac{\sum_{r \in S} \int Pr\left[Z_X \geq g(r)\right] Pr[\vec{z_r}] d\vec{z_r} - \delta}{\sum_{r \in S} \int Pr\left[Z_Y \geq g'(r)\right] Pr[\vec{z_r}] d\vec{z_r}}\right)\right].
\end{aligned}$$

Since $Z \sim h$ and h is admissible, we can use the sliding property:

$$\begin{aligned}
D_\infty^\delta(Z_X || Z_Y) &\leq \max_{\substack{S \subseteq \mathcal{R}: \\ Pr[Z_X \in S] \geq \delta}} \left[\log\left(\frac{\sum_{r \in S} \int Pr\left[Z_X \geq g(r) - g(r) + \frac{\tilde{u}'_* - u(\mathbf{y}, r)}{N(\mathbf{x})}\right] \cdot e^{\frac{\varepsilon}{2}} Pr[\vec{z_r}] d\vec{z_r} + \frac{\delta}{2} - \delta}{\sum_{r \in S} \int Pr\left[Z_Y \geq g'(r)\right] Pr[\vec{z_r}] d\vec{z_r}}\right)\right], \\
&\leq \max_{\substack{S \subseteq \mathcal{R}: \\ Pr[Z_X \in S] \geq \delta}} \left[\log\left(\frac{\sum_{r \in S} \int Pr\left[Z_X \geq \frac{\tilde{u}'_* - u(\mathbf{y}, r)}{N(\mathbf{x})}\right] \cdot e^{\frac{\varepsilon}{2}} Pr[\vec{z_r}] d\vec{z_r} - \frac{\delta}{2}}{\sum_{r \in S} \int Pr\left[Z_Y \geq g'(r)\right] Pr[\vec{z_r}] d\vec{z_r}}\right)\right].
\end{aligned}$$

The first inequality results from the sliding property since h is admissible. Notice that this property can be applied above because, by the properties of smooth and local sensitivities, and since $\mathbf{x}$ and $\mathbf{y}$ are neighbors:

$$\begin{aligned}
-g(r) + \frac{\tilde{u}'_* - u(\mathbf{y}, r)}{N(\mathbf{x})} = \frac{-\tilde{u}_* + u(\mathbf{x}, r) + \tilde{u}'_* - u(\mathbf{y}, r)}{N(\mathbf{x})} &= \alpha \frac{-u(\mathbf{y}, r) + u(\mathbf{x}, r) - \tilde{u}_* + \tilde{u}'_*}{2\mathcal{S}_{u,\beta}(\mathbf{x})}, \\
&\leq \frac{\alpha}{2LS(\mathbf{x})} \left(\underbrace{u(\mathbf{x}, r) - u(\mathbf{y}, r)}_{\leq LS(\mathbf{x})} + \underbrace{\tilde{u}'_* - \tilde{u}_*}_{\leq LS(\mathbf{x})} \right), \\
&\leq \alpha \frac{2LS(\mathbf{x})}{2LS(\mathbf{x})} = \alpha.
\end{aligned}$$

Further we can apply the dilation property since h is admissible and $\ln \frac{N(\mathbf{x})}{N(\mathbf{y})} = \ln \frac{\mathcal{S}_{u,\beta}(\mathbf{x})}{\mathcal{S}_{u,\beta}(\mathbf{y})} \leq \beta$ (see Definition 2.6):

$$\begin{aligned}
D_\infty^\delta(Z_X || Z_Y) &\leq \max_{\substack{S \subseteq \mathcal{R}: \\ Pr[Z_X \in S] \geq \delta}} \left[\log \left(\frac{\sum_{r \in S} \int Pr\left[Z_X \geq \frac{\tilde{u}'_* - u(\mathbf{y}, r)}{N(\mathbf{x})} \cdot \frac{N(\mathbf{x})}{N(\mathbf{y})} \right] \cdot e^{\frac{\varepsilon}{2}} \cdot e^{\frac{\varepsilon}{2}} Pr[\vec{z_r}] d\vec{z_r} + \frac{\delta}{2} - \frac{\delta}{2}}{\sum_{r \in S} \int Pr\left[Z_Y \geq g'(r) \right] Pr[\vec{z_r}] d\vec{z_r}} \right) \right], \\
&= \max_{\substack{S \subseteq \mathcal{R}: \\ Pr[Z_X \in S] \geq \delta}} \left[\log \left(\frac{\sum_{r \in S} \int Pr\left[Z_X \geq \frac{\tilde{u}'_* - u(\mathbf{y}, r)}{N(\mathbf{y})} \right] \cdot e^{\varepsilon} Pr[\vec{z_r}] d\vec{z_r}}{\sum_{r \in S} \int Pr\left[Z_Y \geq g'(r) \right] Pr[\vec{z_r}] d\vec{z_r}} \right) \right], \\
&= \max_{\substack{S \subseteq \mathcal{R}: \\ Pr[Z_X \in S] \geq \delta}} \left[\log \left(\frac{e^{\varepsilon} \cdot \sum_{r \in S} \int Pr\left[Z_X \geq g'(r) \right] Pr[\vec{z_r}] d\vec{z_r}}{\sum_{r \in S} \int Pr\left[Z_Y \geq g'(r) \right] Pr[\vec{z_r}] d\vec{z_r}} \right) \right], \\
&= \varepsilon.
\end{aligned}$$

By symmetry, we can also prove that $D_\infty^\delta(Z_Y \| Z_X) \leq \varepsilon$. Then, by Definition 2.3, we conclude that SNM is (ε, δ)-differentially private. □

A.2. Utility proof

Lemma 4.7. Given a fixed database $\mathbf{x} \in \mathcal{X}$, for the Smooth Noisy Max $\mathcal{A}$ algorithm with a standard Laplace distribution as noise function and any $t > 0$, the error $\xi(\mathcal{A}, \mathbf{x})$ satisfies

$$Pr[\xi(\mathcal{A}, \mathbf{x}) \geq t] \leq |\mathcal{R}| \exp\left(-\frac{\varepsilon t}{4\mathcal{S}_{u,\beta}(\mathbf{x})} \right).$$

Proof of Lemma 4.7. Define $u^*(\mathbf{x}) = \max_{r \in \mathcal{R}} u(\mathbf{x}, r)$, so for each possible outcome $r \in \mathcal{R}$, the error can be written as $\xi(\mathcal{A}, \mathbf{x}) = u^*(\mathbf{x}) - u(\mathbf{x}, r)$. Thus, for $t > 0$:

$$Pr[\xi(\mathcal{A}, \mathbf{x}) \geq t] = Pr[u(\mathbf{x}, \mathcal{A}(\mathbf{x})) \leq u^*(\mathbf{x}) - t]. \tag{2}$$

For simplicity of notation, define the following subsets of $\mathcal{R}$:

(i) $\mathcal{R}_t = \{r \in \mathcal{R} : u(\mathbf{x}, r) \leq u^*(\mathbf{x}) - t\}$;
(ii) $\mathcal{R}_* = \{r \in \mathcal{R} : u(\mathbf{x}, r) = u^*(\mathbf{x})\}$.

Also, consider the noisy utility $\tilde{u}(\mathbf{x}, r) = u(\mathbf{x}, r) + (2\mathcal{S}_{u,\beta}(\mathbf{x})/\alpha) \cdot z_r$, where $z_r \sim \texttt{Lap}(0, 1)$, and its maximal value $\tilde{u}^*(\mathbf{x}) = \max_{r \in \mathcal{R}} \tilde{u}(\mathbf{x}, r)$. Notice that the probability of the output being in $\mathcal{R}_t$ is the same probability of existing some element in $\mathcal{R}_t$ with the greatest noisy utility. This way, the probability expressed in Equation 2 is equivalent to the probability of existing some $r \in \mathcal{R}_t$ such that $\tilde{u}(\mathbf{x}, r) = \tilde{u}^*(\mathbf{x})$. In other words, $\exists r \in \mathcal{R}_t : \tilde{u}(\mathbf{x}, r) = \tilde{u}^*(\mathbf{x})$. Then:

$$Pr[\xi(\mathcal{A}, \mathbf{x}) \geq t] = Pr[\exists r \in \mathcal{R}_t : \tilde{u}(\mathbf{x}, r) = \tilde{u}^*(\mathbf{x})] = Pr[\cup_{r \in \mathcal{R}_t} [\tilde{u}(\mathbf{x}, r) = \tilde{u}^*(\mathbf{x})]] \leq \sum_{r \in \mathcal{R}_t} Pr[\tilde{u}(\mathbf{x}, r) = \tilde{u}^*(\mathbf{x})].$$

Let r' be the most probable output in $\mathcal{R}_t$. In this case, we can write:

$$\begin{aligned}
Pr[\xi(\mathcal{A}, \mathbf{x}) \geq t] &\leq |\mathcal{R}_t| \ Pr[\tilde{u}(\mathbf{x}, r') = \tilde{u}^*(\mathbf{x})] = |\mathcal{R}_t| \ Pr[\tilde{u}(\mathbf{x}, r') \geq \tilde{u}^*(\mathbf{x})] = |\mathcal{R}_t| \ Pr[(2\mathcal{S}_{u,\beta}(\mathbf{x})/\alpha) \cdot z_{r'} \geq \tilde{u}^*(\mathbf{x}) - u(\mathbf{x}, r')], \\
&\leq \frac{|\mathcal{R}_t| \ Pr[z_{r'} \geq (\tilde{u}^*(\mathbf{x}) - u(\mathbf{x}, r')) \cdot (\alpha/2\mathcal{S}_{u,\beta}(\mathbf{x}))]}{Pr[\mathcal{A}(\mathbf{x}) \in \mathcal{R}_*]}.
\end{aligned}$$

Notice that $Pr[\tilde{u}(\mathbf{x}, r') = \tilde{u}^*(\mathbf{x})] = Pr[\tilde{u}(\mathbf{x}, r') \geq \tilde{u}^*(\mathbf{x})]$, since $\tilde{u}^*(\mathbf{x})$ is the maximal noisy utility. Now, consider $r^* = \arg\max_{r \in \mathcal{R}} u(\mathbf{x}, r)$ and z_* as the noise associated with r^*. Thus, we can write:

$$\begin{aligned} Pr[\mathcal{A}(\mathbf{x}) \in \mathcal{R}_*] &= Pr[\cup_{r \in \mathcal{R}_*}[\mathcal{A}(\mathbf{x}) = r]], \\ &= \sum_{r \in \mathcal{R}_*} Pr[\mathcal{A}(\mathbf{x}) = r], \\ &= |\mathcal{R}_*| \; Pr[\mathcal{A}(\mathbf{x}) = r^*], \\ &= |\mathcal{R}_*| \; Pr[z_* \geq (\tilde{u}^*(\mathbf{x}) - u(\mathbf{x}, r^*)) \cdot (\alpha / 2\mathcal{S}_{u,\beta}(\mathbf{x}))]. \end{aligned}$$

The equality above is valid because $u(\mathbf{x}, r) = u^*(\mathbf{x}) \; \forall r \in \mathcal{R}_*$, so the chance of any of them being the output depends only on the noise, resulting in independent events with equal probability. As a consequence:

$$Pr[\xi(\mathcal{A}, \mathbf{x}) \geq t] \leq \frac{|\mathcal{R}_t| \; Pr[z_{r'} \geq (\tilde{u}^*(\mathbf{x}) - u(\mathbf{x}, r')) \cdot (\alpha / 2\mathcal{S}_{u,\beta}(\mathbf{x}))]}{|\mathcal{R}_*| \; Pr[z_* \geq (\tilde{u}^*(\mathbf{x}) - u(\mathbf{x}, r^*)) \cdot (\alpha / 2\mathcal{S}_{u,\beta}(\mathbf{x}))]}.$$

However, as $z_{r'}, z_* \sim \texttt{Lap}(0, 1)$, $u(\mathbf{x}, r^*) = u^*(\mathbf{x})$ and $u(\mathbf{x}, r') \leq u^*(\mathbf{x}) - t$:

$$\begin{aligned} \frac{|\mathcal{R}_t| \, Pr[z_{r'} \geq (\tilde{u}^*(\mathbf{x}) - u(\mathbf{x}, r')) \cdot (\alpha / 2\mathcal{S}_{u,\beta}(\mathbf{x}))]}{|\mathcal{R}_*| \, Pr[z_* \geq (\tilde{u}^*(\mathbf{x}) - u(\mathbf{x}, r_*)) \cdot (\alpha / 2\mathcal{S}_{u,\beta}(\mathbf{x}))]} &= \frac{\frac{|\mathcal{R}_t|}{2} \exp\left(-\frac{\alpha(\tilde{u}^*(\mathbf{x}) - u(\mathbf{x}, r'))}{2\mathcal{S}_{u,\beta}(\mathbf{x})}\right)}{\frac{|\mathcal{R}_*|}{2} \exp\left(-\frac{\alpha(\tilde{u}^*(\mathbf{x}) - u(\mathbf{x}, r_*))}{2\mathcal{S}_{u,\beta}(\mathbf{x})}\right)}, \\ &\leq \frac{|\mathcal{R}_t|}{|\mathcal{R}_*|} \frac{\exp\left(-\frac{\alpha(\tilde{u}^*(\mathbf{x}) - u^*(\mathbf{x}) + t)}{2\mathcal{S}_{u,\beta}(\mathbf{x})}\right)}{\exp\left(-\frac{\alpha(\tilde{u}^*(\mathbf{x}) - u^*(\mathbf{x}))}{2\mathcal{S}_{u,\beta}(\mathbf{x})}\right)}, \\ &= \frac{|\mathcal{R}|}{|\mathcal{R}_*|} \exp\left(-\frac{\alpha t}{2\mathcal{S}_{u,\beta}(\mathbf{x})}\right). \end{aligned}$$

We know that $\alpha = \frac{\varepsilon}{2}$ (see Nissim, Raskhodnikova, and Smith, Lemma 2.9 [30]). Then, from the result above, we can finally conclude that:

$$Pr[\xi(\mathcal{A}, \mathbf{x}) \geq t] \leq |\mathcal{R}| \exp\left(-\frac{\varepsilon t}{4\mathcal{S}_{u,\beta}(\mathbf{x})}\right).$$

□

B. Meta-Review

The following meta-review was prepared by the program committee for the 2025 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

B.1. Summary

This paper considers the differentially private selection problem, in which we must select an item from a set based on a dataset-dependent utility function, with differential privacy. The authors propose an algorithm called Smooth Noisy Max (SNM), which uses the notion of smooth sensitivity to reduce the error of classical algorithms (both theoretically and in practice). The authors demonstrate the utility of their approach on several downstream problems.

B.2. Scientific Contributions

- Provides a Valuable Step Forward in an Established Field

B.3. Reasons for Acceptance

1) The selection problem is relatively old. This paper proposes a new algorithm and theoretical analysis that outperforms widely-used methods depending on global sensitivity. The results in this work are of theoretical interest, and can be of practical interest for some problem settings.