Differentially Private Naïve Bayes Classification

Jaideep Vaidya
Rutgers, The State University of New Jersey
1, Washington Park, Newark, New Jersey 07102, USA
jsvaidya@business.rutgers.edu

Anirban Basu
KDDI R&D Laboratories, Inc.
2-1-15 Ohara, Fujimino-shi, Saitama 356-8502, Japan
basu@kddilabs.jp

Basit Shafiq
Lahore University of Management Sciences
Street 29, Lahore 54792, Pakistan
basit@lums.edu.pk

Yuan Hong
University at Albany, SUNY
1400 Washington Ave., Albany, NY 12222, USA
hong@albany.edu

***Abstract*—Privacy and security concerns often prevent the sharing of users' data or even of the knowledge gained from it, thus deterring valuable information from being utilized. Privacy-preserving knowledge discovery, if done correctly, can alleviate this problem. One of the most important and widely used data mining techniques is that of classification. We consider the model where a single provider has centralized access to a dataset and would like to release a classifier while protecting privacy to the best extent possible. Recently, the model of differential privacy has been developed which provides a strong privacy guarantee even if adversaries hold arbitrary prior knowledge. In this paper, we apply this rigorous privacy model to develop a Naïve Bayes classifier, which is often used as a baseline and consistently provides reasonable classification performance. We experimentally evaluate the proposed approach, and discuss how it could be potentially deployed in PaaS clouds.**

***Index Terms*—Differential Privacy, Naïve Bayes Classification**

I. Introduction

Data Mining has many applications in the real world, such as in bioinformatics, genetics, medicine, seismology, climatology, economics, and so on. Building and applying any data mining model generally assumes that the underlying data is freely accessible. Often when it involves users' data, this is not realistic. Privacy and security concerns restrict sharing and centralization of users' data. Privacy-preserving data mining [1] has emerged as an effective method to solve this problem. Distributed solutions have been proposed that can preserve privacy while still facilitating data mining. The definition of privacy followed in this line of research is conceptually simple: no site should learn anything new from the *process* of data mining. Specifically, anything learned during the data mining process must be derivable given one's own data and the final result – nothing is learned about any other site's data that isn't inherently obvious from the data mining result. The approach followed in this research has been to select a type of data mining model to be learned and develop a protocol to learn the model while meeting this definition of privacy. However, the typical cryptographic solutions have not been very scalable, and are not effective for large scale datasets, where data mining is primarily needed.

Alternatively, in many situations, a centralized trusted party can be found to serve as a data warehouse. In this case, since the trusted party has full access to the data, it can learn the appropriate data mining model. However, while the raw data can be safely stored in such a repository, giving access to even summarized knowledge can enable inference-based attacks, hence creating privacy breaches.

Recently, the model of Differential Privacy [2] has been developed to enable data release while providing a strong privacy guarantee even in situations where adversaries hold arbitrary prior knowledge. Differential Privacy is a very strong notion of privacy wherein it is possible to guarantee that a randomization algorithm working on two datasets close to each other will produce very similar outputs. Differential Privacy is the de facto model of choice for private data release and has been recently used for developing several privacy-preserving classification models.

The Naïve Bayes classifier is simple but highly effective. This combination of its simplicity and effectiveness has led to its use as a baseline standard by which other classifiers are measured. With various enhancements, it is highly accurate and receives practical use in many applications (e.g., text classification [3]). In this paper, we consider the case where a single provider has centralized access to a dataset and would like to release a classifier while ensuring privacy of users' data. We apply the rigorous privacy model of differential privacy to construct a privacy preserving Naïve Bayes classifier. We experimentally validate the proposed approach and discuss how it can be deployed on a Platform-as-a-Service cloud, such as the Google App Engine[1].

II. Preliminaries

We first give a brief overview of Naïve Bayes classification, followed by an overview of Differential Privacy.

A. Naïve Bayes Classification

Naïve Bayes is a highly practical Bayesian learning method and is particularly suited to high dimensional tasks. It is

[1]See: http://appengine.google.com.

often used as a baseline classifier and despite its simplicity, often outperforms more sophisticated methods. The following description is based on the discussion in Mitchell [3]. The Naïve Bayes classifier takes an arbitrary number of continuous or categorical variables and classifies an instance to belong to one of several classes. Thus, it applies to learning tasks where each instance x is described by a conjunction of attribute values and the target function $f(x)$. The target function can take on any value from some finite set C. A set of training examples of the target function is provided, and a new instance is presented, described by the tuple of attribute values $< a_1, a_2, \ldots, a_n >$. The learner is asked to predict the target value, or classification, for this new instance.

The Bayesian approach to classifying the new instance is to assign the most probable target value, c_{MAP}, given the attribute values $< a_1, a_2, \ldots, a_n >$ that describe the instance.

$$c_{MAP} = \underset{c_j \in C}{argmax} \left(P(c_j|a_1, a_2, \ldots, a_n)\right) \tag{1}$$

Using Bayes theorem,

$$\begin{aligned} c_{MAP} &= \underset{c_j \in C}{argmax} \left(\frac{P(a_1, a_2, \ldots, a_n|c_j)P(c_j)}{P(a_1, a_2, \ldots, a_n)}\right) \\ &= \underset{c_j \in C}{argmax} \left(P(a_1, a_2, \ldots, a_n|c_j)P(c_j)\right) \end{aligned} \tag{2}$$

The Naïve Bayes classifier makes the further simplifying assumption that the attribute values are conditionally independent given the target value. Therefore,

$$c_{NB} = \underset{c_j \in C}{argmax} \left(P(c_j) \prod_i P(a_i|c_j)\right) \tag{3}$$

where c_{NB} denotes the output by the Naïve Bayes classifier.

The conditional probabilities $P(a_i|c_j)$ need to be estimated from the training set. The prior probabilities $P(c_j)$ also need to be fixed in some fashion (typically by simply counting the frequencies from the training set). The probabilities for differing hypotheses (classes) can also be computed by normalizing the values received for each hypothesis (class).

Probabilities are computed differently for nominal and numeric attributes.

1) Nominal Attributes: For a nominal attribute X with r possible attributes values $x_1, \ldots, x_r$, the probability $P(X = x_k|c_j) = \frac{n_{kj}}{n}$ where n is the total number of training examples for which $C = c_j$, and n_{kj} is the number of those training examples that also have $X = x_k$.

To make the classifier more robust to cases where some counts may be zero (resulting in the overall probability being zero even if the other attributes are extremely likely), a Laplace estimator is normally applied, which simply consists of adding 1 to all of the counts before normalization to ensure that none of the counts are zero.

2) Numeric Attributes: In the simplest case, numeric attributes are assumed to have a "normal" or "Gaussian" probability distribution. It is also possible to use instead the lognormal, gamma, and Poisson density function or to use binning to discretize the values. Another possibility is to use a kernel estimation method to approximate more complex distributions. However, we restrict our attention to the normal distribution in this paper and leave the rest for future work.

The probability density function for a normal distribution with mean μ and variance σ^2 is given by

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{4}$$

The mean μ and variance σ^2 are calculated for each class and each numeric attribute from the training set. Now the required probability that the instance is of the class c_j, $P(X = x'|c_j)$, can be estimated by substituting $x = x'$ in equation 4.

B. Differential Privacy

Differential Privacy [2], [4] provides a formal and quantifiable privacy guarantee irrespective of an adversary's background knowledge and available computational power. Differential Privacy is actually a condition on the data release mechanism and not on the dataset. A randomized algorithm is considered to be differentially private if for any pair of neighboring inputs, the probability of generating the same output, is within a small multiple of each other, for the entire output space. This means that for any two datasets which are close to one another, a differentially private algorithm will behave approximately the same on both data sets. This notion provides sufficient privacy protection for users regardless of the prior knowledge possessed by the adversaries.

Definition 1 (ϵ-Differential Privacy): A randomization algorithm $\mathcal{R}$ satisfies ϵ-differential privacy if for any two neighboring datasets D_1 and D_2 (differing in one record), and any output $D \in Range(\mathcal{R})$, we have $e^{-\epsilon} \leq \frac{Pr[\mathcal{R}(D_1)=D]}{Pr[\mathcal{R}(D_2)=D]} \leq e^{\epsilon}$.

Definition 2 (Sensitivity): For any query q over the input datasets, the sensitivity of q is $\Delta q = \max ||q(D_1) - q(D_2)||$ for any two neighboring datasets D_1 and D_2.

Queries with lower sensitivity can better tolerate the data modifications from added noise. The work in [4] shows that to release a (perturbed) value $f(x)$ while satisfying privacy, it suffices to add Laplace noise with standard deviation $S(f)/\epsilon$. We make use of this in our work below.

III. Making Naïve Bayes Differentially Private

We now present how to make the standard Naïve Bayes Classifier differentially private. The basic idea is to derive the sensitivity of the classifier parameters and to use those to add Laplacian noise, so that the derived classifier is then guaranteed to be differentially private.

A. Deriving Sensitivity

We first discuss how the sensitivity is derived for categorical attributes, and then discuss the case for numeric attributes.

1) Categorical Attributes: We do make a simplifying assumption that all possible values for the given categorical attributes are already known (i.e., we do not consider the case where the extra new record encodes a "brand" new category.) As long as every category has at least 2 records in the dataset from which the Naïve Bayes classifier is learned, this is true.

Recall that for a nominal attribute X with r possible attributes values $x_1, \ldots, x_r$, the probability $P(X = x_k|c_j) = \frac{n_{kj}}{n}$ where n is the total number of training examples for which $C = c_j$, and n_{kj} is the number of those training examples that also have $X = x_k$.

The sensitivity computation can be done on the counts or on the likelihoods. Since it is easier to carry this out on the counts, we use this approach. Note the difference in the counts due to a new record is simply 1, since the presence of the record can increment the count for a particular `<class-value, attribute-value>` combination by 1. Therefore, the sensitivity of each n_{kj} is 1 for all attribute values x_k and class values c_j.

2) Numeric Attributes: In the case of numeric attributes, the probability $P(X = x'|c_j)$ depends on the mean μ_j and standard deviation σ_j, where the mean μ_j and variance σ_j^2 are calculated for class j based on the values of attribute X from the training set. Therefore, we need to derive the sensitivity for both the mean and standard deviation. Unlike the case of categorical attributes, to derive the sensitivity for numeric attribute we need to fix the size of the dataset since the extent to which the mean and standard deviation are affected by a single record, depends on the number of records from which that mean and standard deviation have been computed. Therefore, in the following derivation we denote the size of the dataset by n. We also need to define a bound on the values that the attribute can have (i.e, bound the domain). For example, assume the values for attribute X_j must lie in the range $[l_j, u_j]$.

Now, we show how to compute the sensitivity of the mean. Given, mean = μ_j, the sum of all of the record values for attribute X_j is $\mu_j * n$. Now, when we consider a neighboring dataset with one extra record, the sum of the attribute values for the neighboring dataset is bounded by $[\mu_j * n + l_j, \mu_j * n + u_j]$. Therefore, the mean μ'_j of the neighboring dataset is bounded by $[(\mu_j * n + l_j)/(n+1), (\mu_j * n + u_j)/(n+1)]$. Hence, sensitivity (s) is bounded by $[(\mu_j * n + l_j)/(n+1) - \mu_j, (\mu_j * n + u_j)/(n+1) - \mu_j]$. Simplifying, the sensitivity is bounded by $[(l_j - \mu_j)/(n+1), (u_j - \mu_j)/(n+1)]$.

Note that the values of attribute X_j are bounded by $[l_j, u_j]$, which implies that μ_j is also bounded by $[l_j, u_j]$. Therefore in the worst case, the difference in μ_j and μ'_j is bounded by $(u_j - l_j)/(n+1)$. Therefore, the sensitivity for the mean is:

$$sensitivity = (u_j - l_j)/(n+1) \tag{5}$$

Now, we compute the sensitivity of the standard deviation. Given, mean = μ_j and standard deviation σ_j, Now, when we consider a neighboring dataset with one extra record, as derived earlier the mean μ'_j of the neighboring dataset is bounded by $[(\mu_j * n + l_j)/(n+1), (\mu_j * n + u_j)/(n+1)]$. Note that $\sigma^2 = 1/n * (\sum_j (x_j - \mu_j)^2)$. In the case of the standard deviation, the maximum change can occur if all of the records in the original dataset are at one extreme and the value of the additional record is at the other extreme (i.e., all records in the original dataset have value l_j, and the additional record has value u_j]. In such a case the standard deviation σ_j of the original dataset is 0. We now derive the standard deviation σ'_j of the neighboring dataset.

Note that in the worst case described above, $\mu_j = l_j$ and $\mu'_j = (n * l_j + u_j/n + 1)$. Now, the variance ${\sigma'_j}^2$ is given in equation 6.

Therefore, $\sigma'_j = \sqrt(n) * (u_i - l_i)/(n+1)$. Since, $\sigma_j = 0$, the sensitivity is:

$$sensitivity = \sqrt(n) * (u_j - l_j)/(n+1) \tag{7}$$

Note that in both cases, the sensitivity for the prior probabilities $P(c_j)$ can be computed in a similar fashion. Specifically, the sensitivity of the prior count is 1 (since an additional record can simply be of that class).

Algorithm 1 Computing differentially private parameters for Naïve Bayes

Require: ϵ, the privacy parameter for differential privacy
Require: Laplace(a, b) samples the Laplace distribution with mean a and scale b
for each attribute X_j **do**
  if X_j is categorical **then**
    sensitivity, $s \leftarrow 1$
    scale factor, $sf \leftarrow s/\epsilon$
    $\forall$ counts n_{kj}, $n'_{kj} = n_{kj}$+ Laplace$(0, sf)$
    Use n'_{kj} to compute $P(x_i|c_j)$
  else if X_j is numeric **then**
    compute sensitivity, s for mean μ_j as per equation 5
    scale factor, $sf \leftarrow s/\epsilon$
    $\mu'_j \leftarrow \mu_j$+ Laplace$(0, sf)$
    compute sensitivity, s for standard deviation σ_j as per equation 7
    scale factor, $sf \leftarrow s/\epsilon$
    $\sigma'_j \leftarrow \sigma_j$ + Laplace$(0, sf)$
    Use μ'_j and σ'_j to compute $P(x_i|c_j)$
  end if
end for
for each class c_j **do**
  count $nc'_j \leftarrow nc_j$+ Laplace$(0, 1)$
  Use nc'_j to compute the prior $P(c_j)$
end for

B. Algorithm

Now that we have derived how to compute the sensitivity, the actual differentially private Naïve Bayes procedure is quite simple. As described earlier, the key idea is to derive the sensitivity for each attribute appropriately based on whether it is categorical or numeric. Following this, Laplacian noise of the appropriate scale (and mean 0) is added to the parameters (the counts for categorical attributes, the means and standard deviations for numeric attributes). This process is described in Algorithm 1. The computed parameters are then used to classify a new instance in the standard Naïve Bayes fashion as described in Section II-A.

$$
\begin{aligned}
\sigma_j'^2 &= 1/(n+1) * (\sum_{i=1}^{n}(l_j - (n * l_j + u_j/n+1))^2) + (u_j - (n * l_j + u_j/n+1))^2 \\
&= 1/(n+1) * ((n * (l_j - u_j/n+1)^2) + (n * (u_j - l_j)/n+1)^2) \\
&= 1/(n+1) * ((n * (l_j - u_j)^2/(n+1)^2) + (n^2 * (u_j - l_j)^2/(n+1)^2)) \\
&= 1/(n+1) * (u_j - l_j)^2/(n+1)^2 * n(1+n) \\
&= n * (u_j - l_j)^2/(n+1)^2 \qquad (6)
\end{aligned}
$$

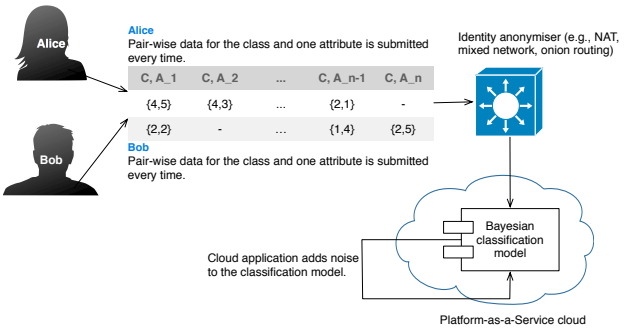


Fig. 1: A typical deployment on a PaaS cloud

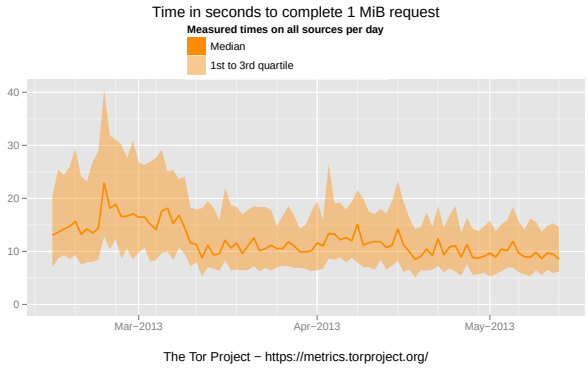


Fig. 2: Tor latency for a 1MiB request.

While the above description is theoretically complete, there are some implementation considerations. Specifically, when Laplacian noise is added, since the noise can be negative as well, it is possible for the mean, standard deviation, counts, and class priors to become negative. While this is not a problem for the mean, it is a problem for all of the rest. Instead of choosing positive values from the Laplace distribution (which would make it bounded), we instead resample the Laplace distribution as many times as necessary to ensure that the modified values are non-negative.

IV. Deployment scenario on a cloud platform

Given the volume of the data involved, it is realistic to assume that organisations will deploy such a privacy preserving classifier on the cloud. In figure 1, we present a typical deployment scenario on a Platform-as-a-Service cloud, such as the Google App Engine. Typically, during the training phase, the training data is submitted (e.g., by Alice and Bob, each owning a subset of the attributes) in attribute and target class pairs, through some form of network anonymizer that is sufficient to delink the identities of individual submissions from their submitters. For example, if Alice submitted one pair of such training data using the same wireless network as Bob then to the cloud, the training data will appear to come from one single WAN side IP even though there were two different submissions from two different entities. This identity anonymization step is necessary only if the submitted training data is considered private, which may not always be the case. Anonymizers, beyond the simple network address translation mechanism may also be used, such as the Tor network. However, some anonymizers may result in high latencies during submitting large volumes of the training data to the cloud thus slowing down the process of building the classifier. For example, figure 2 shows the time it takes for Tor to complete a 1MiB request as recorded in the period between March and May, 2013.

While the training data is added, the application running on the cloud adds Laplace noise and generates a compact model according to algorithm 1. This model – the classifier, which is used to make classifications is differentially private, and therefore, resilient against privacy threats from the cloud itself.

We are yet to experiment with the scenario of cloud deployment that we have presented here. It constitutes one of our future work exercises.

V. Experimental Evaluation

We implemented our differentially private Naïve Bayes algorithm on Weka [5], which is an open source collection of machine learning algorithms for data mining tasks implemented in Java. Apart from providing algorithms, it is a general implementation framework, along with support classes and documentation. It is extensible and convenient for prototyping purposes.

We have implemented a DiffPrivNaiveBayesSimple class based on the NaiveBayesSimple class and integrated it into Weka version 3.6. It is possible to set the value of ϵ and n for differential privacy. We experimented with several real datasets from the UCI repository [6]. The results are reported for the Nursery, Mushroom, Adult, Skin, Optical Digit Recognition datasets. These datasets have varying number of instances, attributes and class values giving a realistic picture of the performance of our algorithm.

Since the Differentially Privacy Naïve Bayes adds random (Laplacian) noise, in all the experiments, 5 iterations have been run with 10-fold cross validation and averaged. The baseline is the standard Naïve Bayes classifier against which our differentially private Naïve Bayes classifier is compared for varying values of $n = \{1000, 5000, 10000, 100000, 1000000\}$ and $\epsilon = \{1.0, 0.1, 0.05, 0.01, 0.005, 0.001\}$.

Figures 3a – 3e depict the results of the experiments. It is clear that with the decreasing value of ϵ, more noise is added, resulting in degradation in the accuracy. Generally the accuracy loss is quite steep beyond a specific value of ϵ which is dependent on the dataset. For Mushroom and Nursery, the accuracy loss is rather small for $\epsilon >= 0.005$. To a much smaller extent, the accuracy loss is also dependent on the value of n. As n increases, the sensitivity decreases thus resulting in less noise. This is more obvious for the Skin and Adult datasets, while the results for the Mushroom, Nursery, and Optical Digit Recognition do not vary much. We do not report the time taken to construct the Differentially Private classifier, since the additional time required beyond that for the standard Naïve Bayes is negligible.

VI. Related Work

There has been significant work on developing privacy models for centralized data release. Dwork et al. [2], [4], [7] have proposed the rigorous privacy definition of differential privacy, which provides sufficient privacy protection for users regardless of the prior knowledge possessed by the adversaries. This has been extended to various contexts of privacy-preserving data release and data analysis, e.g., user behavior recommender [8], graph based applications [9], frequent pattern mining [10], [11], query log statistics [12] and publishing [13]. Jagannathan et. al [14] propose ways to construct a differentially private Random Decision Tree (RDT) classifier from a centralized dataset. However, this does not work for Naïve Bayes Classification, as in our case. Cormode [15] shows how a differentially private Naïve Bayes classifier can still be used to infer "private" attributes. Thus, they show how to build the classifier. However, it only works for categorical attributes. Our work also extends this to numeric attributes, and has a detailed experimental evaluation and integration into Weka.

There has also been significant work in the area of privacy-preserving data mining. Several solution approaches have been suggested. One approach is to add "noise" to the data before the data mining process, and then use reconstruction techniques that mitigate the impact of the noise from the data mining results [16]. Indeed, the recent differentially private algorithms follow the same line of research to build queries or data analysis application by adding noise [7] or running particular randomization mechanism [13], [17] to achieve quantifiable notion of privacy. The work on protecting privacy in the data mining process (especially for classification) can be classified into two kinds. First, besides differentially private approaches mentioned above, Mohammed et al. [18] proposed an anonymization approach to generalize the raw data and then add noise to guarantee differential privacy, where the output data can be used to build a decision tree induction classifier. An alternative approach is to use cryptographic techniques to protect privacy, which was first used for the construction of decision trees by [19]. This work closely followed the secure multiparty computation approach, achieving "perfect" privacy, i.e., nothing is learned that could not be deduced from one's own data and the resulting tree. The key insight was to trade off computation and communication cost for accuracy, improving efficiency over the generic secure multiparty computation method. There has since been work to address other classification tasks [20], [21], but the work in this paper is orthogonal to all of these. The most closely related is the work in [22] which develops privacy-preserving techniques to compute the Naïve Bayes classifier from both horizontally and vertically partitioned data. However, since the current work assumes centralized data as opposed to a distributed data model, and develops a differentially private classifier, it is also orthogonal to that work.

VII. Conclusion

In this paper we have developed a differentially private Naïve Bayes Classifier. We have implemented our algorithm and integrated it into the Weka toolkit. Our experimental results show that the Differentially Private Naïve Bayes performs very well and is able to keep up with the baseline Naïve Bayes classifier even while providing very strong privacy. We have also discussed how such a classifier could be deployed on a PaaS cloud. In the future, we plan to look at how other classification techniques can also be made differentially private and integrated into Weka, and explore how differentially private classification can be outsourced into the cloud.

References

[1] J. Vaidya, C. Clifton, and M. Zhu, *Privacy-Preserving Data Mining*, 1st ed., ser. Advances in Information Security. Springer-Verlag, 2005, vol. 19.

[2] C. Dwork, "Differential privacy," in *33rd International Colloquium on Automata, Languages and Programming (ICALP 2006)*, Venice, Italy, Jul. 9-16 2006, pp. 1–12.

[3] T. Mitchell, *Machine Learning*, 1st ed. McGraw-Hill Science/Engineering/Math, 1997.

[4] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *TCC*, 2006, pp. 265–284.

[5] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. San Francisco: Morgan Kaufmann, Oct. 1999.

[6] C. Blake and C. Merz, "UCI repository of machine learning databases," 1998.

[7] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *EUROCRYPT*, 2006, pp. 486–503.

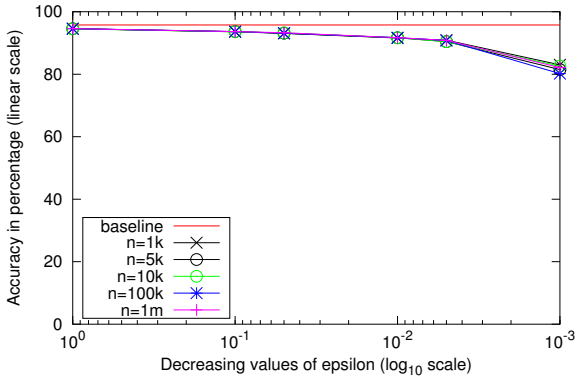
[8] F. McSherry and I. Mironov, "Differentially private recommender systems: building privacy into the net," in *KDD*, 2009, pp. 627–636.

[9] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *ICDM*, 2009, pp. 169–178.

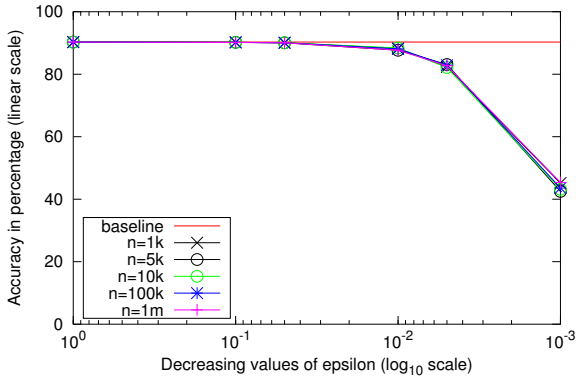
[10] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta, "Discovering frequent patterns in sensitive data," in *KDD*, 2010, pp. 503–512.

[11] N. Li, W. H. Qardaji, D. Su, and J. Cao, "Privbasis: Frequent itemset mining with differential privacy," *PVLDB*, vol. 5, no. 11, pp. 1340–1351, 2012.

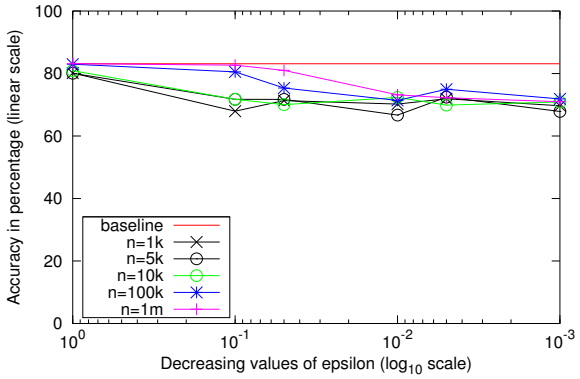
[12] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas, "Releasing search queries and clicks privately," in *WWW*, 2009, pp. 171–180.



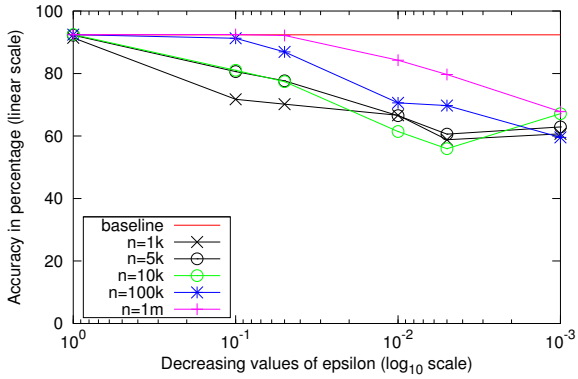
(a) Mushroom dataset: 8K records, 22 attributes and 2 classes.



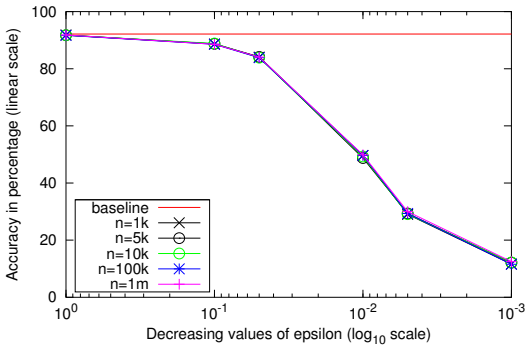
(b) Nursery dataset: 13K records, 8 attributes and 5 classes.



(c) Adult dataset: 48K records, 14 attributes and 2 classes.



(d) Skin dataset: 245K records, 3 attributes and 2 classes.



(e) Opt. Digit: 5.6K records, 64 attributes and 10 classes.

Fig. 3: Accuracy graphs for various datasets.

[13] Y. Hong, J. Vaidya, H. Lu, and M. Wu, "Differentially private search log sanitization with optimal output utility," in *EDBT*, 2012.

[14] G. Jagannathan, K. Pillaipakkamnatt, and R. N. Wright, "A practical differentially private random decision tree classifier," in *Proceedings of the 2009 IEEE ICDM Workshops*, ser. ICDMW '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 114–121.

[15] G. Cormode, "Personal privacy vs population privacy: learning to attack anonymization," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '11. New York, NY, USA: ACM, 2011, pp. 1253–1261.

[16] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of the 2000 ACM SIGMOD Conference on Management of Data*, ACM. Dallas, TX: ACM, May 14-19 2000, pp. 439–450.

[17] Y. Duan, "Privacy without noise," in *CIKM*, 2009, pp. 1517–1520.

[18] N. Mohammed, R. Chen, B. C. M. Fung, and P. S. Yu, "Differentially private data release for data mining," in *KDD*, 2011, pp. 493–501.

[19] Y. Lindell and B. Pinkas, "Privacy preserving data mining," *Journal of Cryptology*, vol. 15, no. 3, pp. 177–206, 2002.

[20] R. Wright and Z. Yang, "Privacy-preserving bayesian network structure computation on distributed heterogeneous data," in *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Seattle, WA: ACM, Aug.22-25 2004.

[21] J. Vaidya, C. Clifton, M. Kantarcioglu, and A. S. Patterson, "Privacy-preserving decision trees over vertically partitioned data," *ACM Trans. Knowl. Discov. Data*, vol. 2, no. 3, pp. 1–27, 2008.

[22] J. Vaidya, M. Kantarcioglu, and C. Clifton, "Privacy preserving naive bayes classification," *International Journal on Very Large Data Bases*, vol. 17, no. 4, pp. 879–898, Jul. 2008.