TRANSACTIONS ON DATA PRIVACY 7 (2014) 51–70

Combining Binary Classifiers for a Multiclass Problem with Differential Privacy

Vera Sazonova*, **Stan Matwin***,**

*Faculty of Computer Science, Dalhousie University, 6050 University, Halifax Canada

** Institute for Computer Science of the Polish Academy of Science Warsaw, Poland

E-mail: vsazonov@uottawa.ca, stan@sc.dal.ca

Abstract. Multiclass classification problem is often solved by combing binary classifiers into ensembles. While this is required for inherently binary classifiers, such as SVM, it also provides performance advantages for other classifiers. In this paper, we address the problem of combining binary classifiers into ensembles in the differentially private data publishing framework, where the data privacy is achieved by anonymization. The main idea of this paper is to counter the inevitable loss of data quality due to anonymization of the data by building an ensemble of binary classifiers, and then to use an error-correcting approach to obtain a class decision from this ensemble. We describe the proposed algorithm and present the results of extensive experimentation on synthetic and UC Irvine data. We find that while building ensembles after anonymization leads to no change in classifier accuracy, preparing the data for ensembles prior to anonymization improves accuracy in most of the cases.

1 Introduction

There has been a vast amount of digital information collected about individuals in the recent years. Such personal data presents tremendous opportunities for data miners, but it also poses a threat to individual's privacy, as the raw information often contains sensitive data [1]. It is thus important to develop techniques to allow for successful retrieval of information without the breach of privacy. Differential privacy [8] is a a recent theoretical privacy model that is currently used as the privacy standard. In a nutshell differential privacy requires an output of a query of a database to be insensitive to the values of any particular record. Differential privacy provides formal guarantees that do not depend on an adversary's computing power or background knowledge.

One of the most common tasks in data mining is training classifiers that can be used to categorize new data. Training classifier on private data poses additional challenges as no sensitive individual information should be transmitted to the classifier. Typically the following interaction protocol is assumed. The private data is stored in a database, which is maintained by a curator. A client who wants to mine data does so by interacting with the database through the curator.

Several models for differentially private data mining have recently been proposed [16, 12, 21, 22]. We can separate them in two approaches. In one, a classifier is build by a client

by successive queries to a private database. The database curator adjusts the level of noise added to the response to maintain a certain privacy of the data [16, 12, 22, 18]. A different approach is taken in the field of differentially private data publishing [21, 15, 17, 4, 9, 23, 13]. Here the database curator anonymizes the data, or creates a synthetic data to satisfy the privacy requirements, and then publishes it. The client may then build a classifier using the published data.

The majority of the published work concentrates on binary classification. Multiclass problems, however, are abundant in the real world and thus multi-class classifiers are necessary for the end users. For cases when the use of an inherently binary classifier (for example support vector machine) is necessary, the problem may be resolved by employing various ensemble techniques, particularly error-correcting output codes [7, 3]. In this paper we study the problem of building a multiclass classifier from an ensemble of binary classifiers in the differentially private data publishing setting, following the work of [21].

Anonymizing the data increases its privacy, but at the same time it misleads the learning of a classifier. Error correcting techniques have been known to help recover from errors made by classifiers in a multiclass classification setting. We investigate if such techniques are also beneficial in differentially private framework. The main idea of this paper is therefore to counter the inevitable loss of data quality due to anonymization of the data by building an ensemble of binary classifiers, and then to use an error-correcting approach to obtain a class decision from this ensemble.

We propose a variant of the approach where the ensembles are built by the database curator prior to publishing. We compare it to the baseline of the classical approach in which the ensembles are build by the client on the published data. We investigate experimentally the performance of these approaches, comparing them between each other. Our main finding is that publishing datasets prepared for learning an ensemble influences the performance and leads to an increase in classifier accuracy for about a half of the data sets considered, however, using these ensemble methods after the data has been anonymized on average does not affect the accuracy of the classifier. This effect is particularly notable for weaker privacy guarantees.

The contribution of this paper is as follows: (1) we introduced a novel algorithm for an ECOC-based classifier in a differentially private data publishing scenario; (2) we conducted a series of experiments to compare the classifier's performance to a baseline of a standard ECOC-based classifier on a differentially private published dataset. We investigated the dependence of the relative accuracy achieved for different ECOC parameters, such as, type of ECOC and ECOC decoding method.

In section 2 we describe the differential privacy[8] framework and the previously proposed [21] PPDP algorithm. In section 4 we present our model. In section 5 we describe the datasets used in this study, and in section 6 we describe our experiments, giving the details of implementation needed for replicability of our results. Finally, we conclude with section 7.

2 Privacy Preserving Methods

2.1 Differential Privacy

Differential privacy[8] is a recent privacy definition that guarantees the outcome of a calculation to be insensitive to any particular record. The results of any computation are the same with or without any particular record.

DEFINITION. *(ϵ-differential privacy).* A randomized computation M provides differential privacy if for any datasets A and B with symmetric difference $|A\Delta B| = 1$ and for all possible outcomes of a computation $C \in Range(M)$

$$Pr[M(A) \in C] \leq Pr[M(B) \in C] \times e^{\epsilon} \tag{1}$$

The parameter ϵ allows us to control the desired level of privacy. Lower values of ϵ indicate stronger privacy guarantees.

The definition of differential privacy maintains a sequential composition property [19]. When successive differentially private computations are executed on a data set, the compound computation is differentially private with ϵ set by the sum of individual privacy values.

Differential privacy also maintains a parallel composition [19]. A set of computations performed on disjoint data sets is differentially-private with ϵ set by the largest privacy guarantee of its parts.

2.1.1 Exponential Mechanism

The *Exponential Mechanism* [20] is an ϵ-differentially private mechanism to select the best among a discrete set of options, where "best" is defined by a *score* function s: $dataset \times R \rightarrow \Re$, where $s(B, r)$ is interpreted as the quality of the result r, for the dataset B. The Exponential Selection Mechanism, E selects a result r from a the distribution satisfying

$$Pr[E(B) = r] \propto \exp(\epsilon \times s(B, r)/2) \tag{2}$$

2.1.2 Laplace Mechanism

The *Laplace Mechanism* [8] is an ϵ-differentially private mechanism to report a approximate sums of bound functions over a dataset. If f is a function of records in the dataset adding a Laplace noise with a parameter $(1/\epsilon)$ to it before publishing ensures its ϵ-differential privacy. In other words if for a function f: $B \rightarrow \Re^d$, an algorithm that adds noise drawn from $Lap(\Delta f/\epsilon)$ to each of the d outputs, where Δf is f's sensitivity, is differentially private.

2.2 Differential Privacy Data Publishing

Most of differentially private data publishing methods concentrate on creating a noisy or a synthetic set of data, that can be optimized to give a correct response under a certain chosen set of queries [15],[23],[13],[4]. Some other methods group data according to some entropy measure and publish their noisy counts [21]. Other techniques include creating synthetic datasets similar to the dataset of interest, and adjusting the weight of each data point according to its similarity to the original data [17].

2.2.1 DiffGen

Mohammed et al. [21] introduce a differentially private data publishing technique, in which a data set can be published in a non-interactive manner. The details of this algorithms can be seen in [21], we reproduce it here as a reference in algorithm 1. In this technique the data is initially generalized to the top of each attribute's taxonomy (taxonomies are provided for nominal attributes, and adaptively calculated for numerical attributes)

Algorithm 1 DiffGen

Input: $data$, privacy ϵ,
number of specializations H
for each attribute **do**
generalize $data$ to the top of respective taxonomy
calculate specialization $score$
end for
set $partitionCounts$ to include all $data$
for $i = 1$ **to** H **do**
select attribute A according to $score$ using *Exponential Mechanism* with $\epsilon/2H$
specialize $data$ on A
update $score$
update $partitionCounts$
end for
return $partitionCounts$ + Lap($\epsilon/2$)

and one large partition containing all data is created. For each attribute a score is calculated. The score for specialization data on a certain attribute is calculated with a utility function. We used a sum of highest class frequencies over all child values for the current specialization of the attribute in the taxonomy tree. Other possibility could for example be *information gain*.

At each round of specialization the best candidate for specialization is randomly selected using the *Exponential Mechanism* described above in section 2.1.1. The probability of an attribute being selected is exponential, with the exponent proportional to the product of the attribute's score computed above and the privacy budget. The data is specialized on this attribute, partitioned with respect to the selected attribute's child values, and the score for the attribute is recalculated according to possibility of further specialization.

After the desired number of specialization rounds the data is separated among several partitions, each with a specific attribute values. Before publishing the attributes values and the partition counts, the partition counts are scrambled by the addition of a random number drawn from a Laplace distribution. The centre of the Laplace distribution is proportional to the remaining privacy budget.

2.2.2 Differentially Private Synthetic Dataset for Data Publishing

Hardt et al.[15] introduced a simple and practical algorithm for differentially private data release, that based on *Exponential Mechanism* and Multiweights Scaling. The algorithm creates an synthetic data set which gives the same response to a set of supplied queries as the real dataset.

For a desired number of approximation rounds, the algorithm selects a query out of the pool using *Exponential Selection* and the difference between the real and the synthetic datastets' response to a query as the *score* function. Once chosen, the algorithm employs *Multiweights Scaling* to adjust the synthetic counts to better approximate the response.

3 Converting Multiclass to Binary

For many natural data sets the task of classification is complicated by the fact that data consists of several different classes. Some algorithms, e.g. decision trees fit this case naturally, assigning a different class label to different leaves. Some others, e.g. support vector machines (SVM) are inherently binary.

Several binary classifiers can be combined together in various ways to produce a classifier capable of assigning multiple labels. These methods, in essence, train multiple classifiers, each on a subset of the data with certain class labels combined and certain left out to produce a binary problem.

Algorithm 2 extract sub data

Input: $data$, $code$ for one classifier
$subData \leftarrow null$
for $i = 1$ **to** number of classes **do**
  if $code_i \neq 0$ **then**
    $extractedData \leftarrow data$ with $class = i$
    if $code_i = +1$ **then**
      $extractedData$ assigned class $+1$
    else
      $extractedData$ assigned class -1
    end if
    add $extractedData$ to $subData$
  end if
end for
return $subData$

We define the subset of the data and how the class labels are combined in terms of a table. The rows of the table represent each available class label, and the columns - binary classifiers. The entries in the table can be $+1, -1$, or 0. The classes are, thus, represented by unique sequences of $+1, -1$ and 0, known as the *Error Correcting Output Codes*.

To train this combined classifier, for each binary classifier, the appropriate subset of the dataset is extracted and converted to binary: the data with class label marked 0 are not used for training, and the data with class labels marked $+1$ and -1 are combined into a positive and negative class label, respectively. The classifiers are then trained. To label an example, each of the binary classifiers assigns a label. The resulting class label vector is then compared to each of the classes encodings to find the closest match. The exact definition of the 'closest match' depends on the decoding scheme used.

Having a good code is essential for obtaining good results. Ideally, the code would entries for the classes that are most dissimilar under the decoding method used as a metric.

Many different codes and decoding techniques exist. For a review see: [10, 3, 7]. Error correcting codes can be divided into three categories: general binary (1-ALL, Dense Random, Exhaustive), general ternary (1-1, Sparse Random) and problem-specific (ECOC-1, DECOC). *1-ALL* code discriminates each class against all others combined. *1-1* code discriminates each class against each other class. *Exhaustive* code is the largest possible binary code for a given size problem.

Dense Random code usually is a subset of the *Exhaustive* code; it is used as the *Exhaustive* code quickly becomes too large for computation. *Sparse Random* code is a ternary random code, with a certain probability to have a zero entry. *Dense Random* code would have this

probability equal to zero. Random codes of different lengths can be created. It has been empirically determined [3] that the optimal code length for a problem of n classes is $15 \log_2(n)$ for a *Sparse Random* code and $10 \log_2(n)$ for a *Dense Random* code.

The details of problem-specific codes can be looked up in [10], typically the code starts from a random code and then a validation set is used to optimize the performance of this code on a particular dataset.

Decoding techniques are methods that allow us to calculate the distance between the result vector produced by the ensemble of classifiers and the codes for individual classes. These techniques vary in how they can take into account the certainty of the classifier, how they deal with the zero entries in the codes. For a good review of decoding techniques see [10].

Algorithm 2 describes the algorithm for extracting a data set based on the code.

4 Differential Privacy with ECOC

Algorithm 3 ClientEns

Input: $data$, privacy ϵ, ECOC $code$
publish ϵ-private version of $data$
for $i = 1$ **to** number classifiers **do**
  $subData \leftarrow$ data defined by column i of $code$
  $ensCls_i \leftarrow$ train base classifier on $subData$
end for
return $ensCls$

Algorithm 4 CuratorEns

Input: $data$, privacy ϵ, ECOC $code$
$n \leftarrow$ number of non-disjoint subsets defined by $code$
for $i = 1$ **to** number of classifiers, m **do**
  $subData \leftarrow$ data defined by column i of $code$
  publish ϵ/n-private version of $subData$
  $ensCls_i \leftarrow$ train base classifier on $subData$
end for
return $ensCls$

In the data-publishing scenario, such as, for example, [21], [15], [4], [13], [23] or [17] an ensemble-based classifier can be used by the client as well as any other classifier, since the sensitive data has already been anonymized by the curator of the database prior to publishing. We refer to this scheme as the *ClientEns*. The data flow for this method is:

1. Curator:
   (a) publish the differentially private version of the dataset
2. Client:
   (a) create an ensemble of datasets according to the ECOC table

(b) train individual binary classifiers

However, the curator could, potentially, create an ensemble of anonymized datasets on which the client would then run hers individual binary classifiers. We will refer to this scheme as the *CuratorEns*. The data flow would be the following:

1. Curator:
   (a) create an ensemble of datasets according to the ECOC table
   (b) publish the differentially private version of each dataset
2. Client:
   (a) train individual binary classifiers

While the first method does not require any additional considerations as the data is already differentially private prior to the use of the ensemble classifiers, in the second method, care must be taken to assure that when anonymizing each dataset subset in the ensemble results in the differentially private data with the same privacy level.

4.1 Privacy Considerations for *CuratorEns*

	1	2	3	4	5	6
A	1	0	1	0	1	0
B	1	0	0	1	0	1
C	0	1	1	0	0	1
D	0	1	0	1	1	0

Table 1: 1-1 code for 4 classes. Classifiers 1 and 2, 3 and 4, as well as 5 and 6 use mutually disjoint subsets of data

In the most general case, for the code of length m, there will be m different datasets published. Which data forms each of these subsets is defined by the ECOC table. The publication of each of these datasets needs to be ϵ-differentially private. For m datasets, then, using the composability property [19] of differential privacy we see that all m datasets are $(m \times \epsilon)$-differentially private. The privacy guarantee for the publication of each dataset must then be tightened by a factor of $1/m$ in order to arrive to the original requirement of ϵ-differentially private publication.

We should note, however, that for ternary codes certain classifiers might use non-overlapping datasets. For example, as shown in table 4.1, classifiers 1 and 2, are using disjoint sets. Same is true for classifiers 3 and 4, and 5 and 6. If publishing each dataset is ϵ-differentially private, publishing all, according to the sequential composition property of differential privacy, is still ϵ-differentially private. The factor for tightening the privacy guarantee for each dataset will depend on the number of non-disjoint subsets that remain. The exact scaling factor will be code dependent and vary with its "sparseness".

In the case of the *1-1* code we can calculate the number of non-disjoint subsets analytically. The length of the code is $m = \binom{n}{2} = n(n-1)/2$, there are exactly two datasets in each columns, and there are $n/2$ columns that define disjoint datasets. The privacy guarantee has to be tightened by $n-1$ for the publication of each dataset. In the example of table 4.1 there are three pairs of mutually disjoint data subsets: 1 and 2, 3 and 4, 5 and 6.

4.2 Privacy Guarantees

Let us know look at the algorithms 3 and 4 to ensure that they remain ϵ-differentially private.

ClientEns algorithm is trivially ϵ-differentially private, as it builds upon already anonymized and published data.

If there are n non-disjoint subsets defined by $code$, then the algorithm *CuratorEns* basically reduces to the following step repeated m times, where m is total number of data subsets, i.e. the number of classifiers.

$$\text{publish } \epsilon/n\text{-differentially private data subset defined by column } i \text{ of } code \tag{3}$$

This implies that $m - n$ times the (ϵ/n)-differentially private publishing procedure is run on disjoint datasets, and using the sequential composition property theses $m - n$ steps are together as well (ϵ/n)-differentially private. The rest of the data subsets are non-disjoint, and using the composability of differential privacy, the overall publishing of the datasets will be $(n \times \epsilon/n)$-differentially private.

4.3 Running Times

In both algorithms, *CuratorEns* and *ClientEns* there are three main contributions to the running time: creation of the code, data anonymization, and training of classifiers. Let T_{ECOC} be the time it takes to build a code of length m, $T_{anonymization}$ be the time it takes to publish the differentially private version of the full dataset, and T_{cls} be the time it takes to train a classifier.

The *ClientEns*, then, will execute in

$$T_{ClientEns} = \underbrace{T_{anonymization}}_{\text{curator}} + \underbrace{T_{ECOC} + m \times T_{cls}}_{\text{client}} \tag{4}$$

The *CuratorEns* will execute in

$$T_{CuratorEns} = \underbrace{T_{ECOC} + m \times T_{anonymization}}_{\text{curator}} + \underbrace{m \times T_{cls}}_{\text{client}} \tag{5}$$

Here, for simplicity, we include the factor by which the anonymization of a datasets' subset defined by the code and training of a classifier thereon will be reduced, in the codes' width parameter m.

As we can see, the running time difference between the two algorithm lies in the number of data anonymization turn. While this is a significant increase in the running time, it only be seen by the *Curator* and had to be done once per the publishing of the dataset.

5 Data

Use of ECOC as the core of our approach mandated the use of data with multiple classes. The generalization procedure required a relatively large dataset. We thus chose four datasets from the UC Irvine repository [11] that satisfied or partially satisfied these requirements: ECOLI, GLASS, STATLOG and SHUTTLE.

ADULT dataset from UCI Machine Learning Repository [11] is a staple dataset used for privacy preserving data mining. It contains census information for adults in the US, and is

NAME	ATTRIBUTES	CLASSES	SIZE
ECOLI	8	8	336
GLASS	9	6	214
STATLOG	19	7	2,310
SHUTTLE	9	7	43,500
ADULT-MARITAL-STATUS	15	7	48,842
ADULT-RELATIONSHIP	15	5	48,842
SYNTHETIC	3-10	3-10	300 - 100K

Table 2: Data sets employed in the study

used to predict income value to be more or less than 50k. The dataset contains 8 nominal attributes and 7 numerical attributes. We could not use this data set in its traditional forms as it did not comply with requirement of multiple classes set by ECOC. To get around this problem we decided to chose a different nominal attribute as a class label. Out of these eight only two had more than two values, and also yielded classifiers with some predictive power (better than a classifier with all labels removed except the class label). The two remaining possible class labels were: MARITAL STATUS, RELATIONSHIP, with 7 and 5 possible class values, respectively.

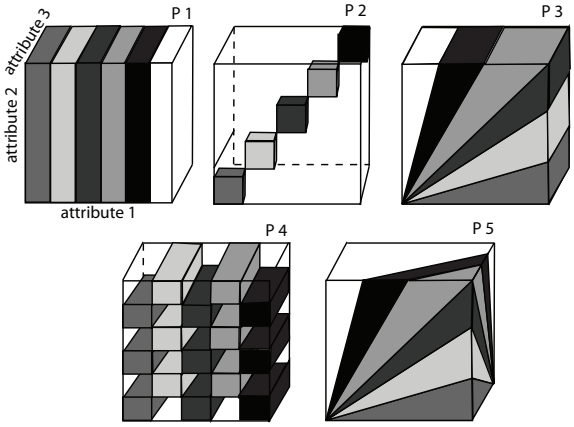

Figure 1: Three different types of three-attribute classification problems with 6 classes. Each shade represents a different class value.

We have also created five types of synthetic datasets, in the spirit of [2], described in more detail below. Table 5 describes the properties of each dataset. Synthetic datasets were created using a data generator. The datasets had between three and ten continuous numeric attributes, and a class attribute, calculated according to one of the five geometrical problems referred to as P1, P2, P3, P4 and P5, shown in figure 1. The number of classes could be set from tree to ten, and the number of data samples per class was varied from 30 per class to 10,000 per class.

Assuming n is the number of classes, for P1, the span of attribute one was divided into n

equal intervals, each corresponding to one class value. The values of other attributes did dot influence the class value.

For P2, the span of each attribute was divided into $n-1$ equal intervals, the intersection of which defined a set of $n-1$ hyper-cubes on the main diagonal of the data attributes hyper-space. These $n-1$ squares defined the first $n-1$ classes, the remaining space in the attribute hyper-space defined the last class value.

For P3, the attribute1-attribute2 plane was divided into n triangles, each corresponding to a certain class value. The values of other attributes did not influence the class assignment. Problem P3, is hard for a decision tree as the tree can not reproduce a diagonal line in the attribute1-attribute2 plane, it approximates it by a "staircase-like" division, resulting in a very large, complicated tree.

For P4, similarly to P2, the spans of attribute1 was each divided into $n-1$ equal intervals, and the span of attribute2 in six equal intervals. For each interval on the attribute1 axis, three non-adjacent squares defined one class. For adjacent intervals on the attribute1 axis, the squares were offset by one, defining, overall, a check-board pattern. The rest of the attribute space was attributed to remaining class. The values of other attributes did not influence the class assignment. Note that we expect problem P4 to be hard as in essence it reduces to an XOR, which is known to be hard for decision tree algorithms.

Finally, P5 was a generalization of P3 for an arbitrary number of attributes. In P5, for each two attributes i and j, the attribute$i$-attribute$j$ plane was divided into n triangles by $n-1$ boundary line. n hyperplanes connecting these lines defined n class values. Like P3, P5 is hard for a decision tree.

6 Experiments

We will now examine performance of *ClientEns* and *CuratorEns* algorithms in different conditions with different datasets. Unless otherwise mentioned we used SVM classifier, *1-1* ECOC, and *ELB* decoding technique and $\epsilon = 3.0$. We studied the dependence of the relative performance of the two aforementioned algorithms on the type of the classifier, ECOC type, decoding technique and privacy constraint. While in our experiments we used the data publishing method proposed by [21], we would like to underline, that any differentially-private data publishing method can be used in these two scenarios.

We implemented data publishing algorithm introduced in [21] to be compatible with WEKA [14]. We also implemented the necessary error-correcting codes and decoding methods from the ECOC library provided in [10] to be compatible with the use of WEKA classifiers. For SVM we used WEKA's *SMO* algorithm and for decision trees the C4.5 algorithm.

6.1 Experimental Procedure

In our experiments we used 10-fold cross-validation (3-fold for ADULT) unless the testing set was provided in the UC Irvine repository. To eliminate the effect of the non-deterministic *Exponential Mechanism* of algorithm 1, we ran five different experimental trials (ten for SHUTTLE and ADULT) on the same training set with different random seeds. For each set of conditions, we thus calculated fifty values of accuracy (ten for SHUTTLE, thirty for ADULT). The average of these values is shown as one experimental data point on figures below, and their standard deviation is shown as error bars.

6.2 Anonymization Algorithm

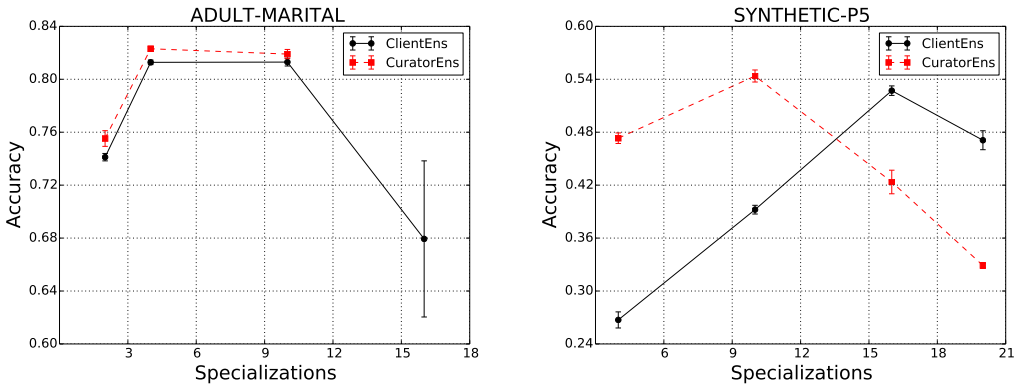

Figure 2: Accuracy vs. the number of specializations for a) ADULT dataset with MARITAL STATUS class label and for b) synthetic dataset for problem P5 with 8 classes 8 attributes and 7000 pts per class. Using *1-1* code, *ELB* decoding technique, and ϵ = 3.0. Black circles - *ClientEns*, red squares - *CuratorEns*

The left panel of Figure 2 presents a set of results for experiments for the ADULT dataset with MARITAL STATUS used as a class label. The graph shows accuracy vs. the number of specialization used in the anonimization procedure presented in algorithm 1 with the privacy constraint of $\epsilon = 3.0$. The *ClientEns* is shown in black circles and *CuratorEns* - in red squares. Similarly, in the right panel of Figure 2 we can see the accuracy vs. the number of specialization in the anonimization procedure for a synthetic dataset for problem P5 with six classes, six attributes and 3000 datapoints per class.

For both datasets we observe that there is a maximum in the accuracy vs. number of specializations curve for each algorithm. For small number of specializations the data values are initialized to the top of their respective attribute taxonomy trees, for most of attributes. This is equivalent to deleting data for these attributes, and thus the resulting accuracy is low. For larger number of specializations, the average size of a partition decreases. When the average size of a partition is of the same order as the mean of the Laplace distribution in the count scrambling stage of the anonimization algorithm, the resulting data becomes useless and the accuracy decreases. This implies that there is a sweet spot where accuracy is maximum. For *ClientEns* this maximum is achieved at a higher number of specializations then for *CuratorEns*.

We would like to distance ourselves from the details of any particular anonymization procedure, and compare only the potential of algorithms *ClientEns* and *CuratorEns*. In order to do that, for each experimental scenario, we compare the maximum achieved accuracies by both algorithms as the anonymization procedure is tuned (i.e. the number of specialization varied).

The middle column of table 6.3 presents the results obtained in this manner with SVM, *1-1*, *ELB* and $\epsilon = 3.0$.

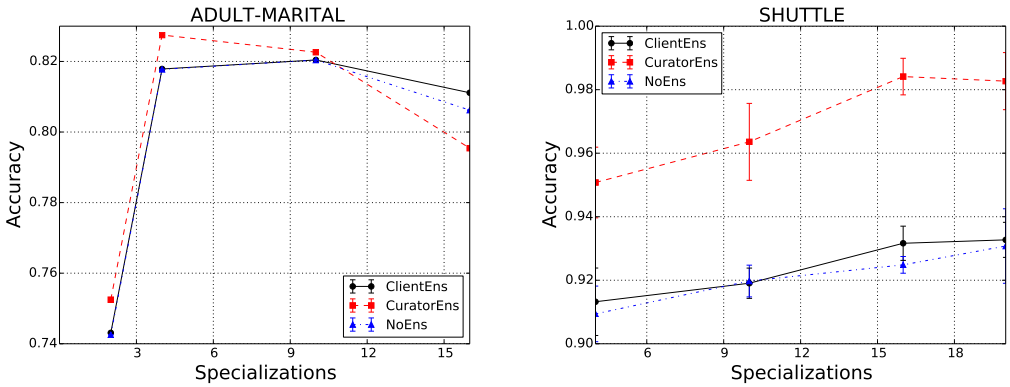

Figure 3: Accuracies of *ClientEns* (black circles), *CuratorEns* (red squares), and *NoEns* (blue triangles) as a function of the number of specializations for ADULT-MARITAL and SHUTTLE datasets using decision trees (C4.5) as a base classifier, *1-1* code, *ELB* decoding technique and $\epsilon = 3.0$

6.3 Other Classifiers

While ECOC techniques are typically used with inherently binary classifiers they can be used with any classifier. In this section we compare the accuracy of a differentially-private decision tree classifier with the accuracy of a decision tree classifier (C4.5) used in the *ClientEns* and *CuratorEns* techniques.

FIgure 3 shows accuracy vs. the number of specialization rounds of *DifGen* algorithm for a differentially private decision tree (DT), which we will reffer to *NonEns* in blue triangles, *ClientEns* with DT in black squares, and *CuratorEns* in red circles for two different datasets. As we can see for ADULT-MAR and SHUTTLE datasets, *CuratorEns* achieves better accuracies than both *ClientEns* and NonEns, while there is no statistically significant difference between *ClientEns* and *NoEns*. In the *CuratorEns* ECOC helps recover from certain errors introduced by the anonymization procedure. However, in *ClientEns* ECOC is run on already anonymized set, and in essence is useless. While we only show results from two datasets, *ClientEns* and *NoEns* achieve similar, statistically indistinguishable accuracies for all datasets in the study.

The maximum accuracy achieved by *ClientEns* and *CuratorEns* achieved for each dataset in the study can be seen in the rightmost column of table 6.3, next to the SVM results. Note that, while SVM and C4.5 achieve different accuracies on the same datasets, the respective advantage or disadvantage of *CuratorEns* over *ClientEns* is the same: i.e. for datasets where *CuratorEns* performs better than *ClientEns* with the SVM, it also performs better with C4.5 and vice versa.

We conclude that first the advantage of *CuratorEns* is problem-dependent, second, for classifiers that do not require ECOC, *ClientEns* is not advatageous at all.

Two questions remain: for which kind of datasets and for which ECOC conditions is *CuratorEns* advantageous?

To address the first question let us note that in the table 6.3 *CuratorEns* produced lower accuracies for ECOLI, GLASS, STATLOG and ADULT-RELATIONSHIP. From table 5 we learn that with the exception of ADULT-RELATIONSHIP dataset, all these datasets have few datapoints (200 - 2,300). This is suggestive of *CuratorEns* being advantageous for larger datasets.

NAME	CLIENTENS / CURATORENS			
	SVM		C4.5	
ECOLI	**0.64**	/ 0.49	**0.58**	/ 0.47
GLASS	**0.50**	/ 0.41	**0.50**	/ 0.35
STATLOG	**0.72**	/ 0.67	**0.82**	/ 0.77
SHUTTLE	0.93	/ **0.95**	0.93	/ **0.98**
ADULT-MARITAL-STATUS	0.813	/ **0.823**	0.82	/ **0.827**
ADULT-RELATIONSHIP	**0.764**	/ 0.750	**0.773**	/ 0.747
SYNTHETIC 6 CLASSES, 6 ATTRIBUTES, 7000 SAMPLES PER CLASS				
P1	0.994	/ 0.992	0.994	/ 0.993
P2	0.969	/ **0.989**	0.990	/ **0.994**
P3	0.628	/ **0.669**	0.628	/ **0.694**
P4	0.900	/ 0.895	0.900	/ 0.898
P5	0.634	/ **0.662**	0.642	/ **0.673**

Table 3: Accuracies of ClientEns and CuratorEns using *1-1* code, and *ELB* decoding technique for two different base classifiers: SVM and C4.5. Boldface indicate significant difference between techniques.

6.4 Accuracy Difference versus Synthetic Dataset Parameters

For synthetic data we could do a systematic study of how different dataset parameters, such as number of attributes, number of classes and dataset size (i.e. number of datapoints per class) influence the respective performances of *CuratorEns* and *ClientEns*. The result can be see in Figure 4 a) and b) for problems P2 and P5, respectively. We omitted this study for problems P1, P3 and P4 as they are similar to P2 and P5, respectively. The experiments were done with *1-1* code, *ELB* decoding technique and $\epsilon = 3.0$.

Figure 4 present the difference in the maximum accuracy (as described in the previous section) between *CuratorEns* and *ClientEns* for different values of number of attributes, number of classes and number of datapoints per class. Red (blue) squares represent statistically significant positive (negative) difference; and pink (light blue) squares statistically insignificant positive (negative) difference. Black squares represent no data being available for those experimental parameters.

First thing that we can observe is that for all synthetic problems, for larger datasets, *CuratorEns* performs better than *ClientEns*. This is consistent with the UC Irvine datasets, where smaller datasets gave better accuracies with *ClientEns* and larger with *CuratorEns* .

Secondly, there seems to be a much less strong dependence on the number of classes and attributes, however, the sign of the dependence (which algorithm performs better for small number of classes) is problem dependent.

6.5 Privacy Guarantees

As explained in section 2.1, in ϵ-differentially private framework the strength of privacy guarantees is expressed by the parameter ϵ. The lower the value of ϵ the stronger the guarantees. Figure 5 shows accuracy of *ClientEns* and *CuratorEns* as we loosen the privacy guarantees by increasing ϵ for ADULT-MARITAL dataset.

As we can see, for high values of privacy, *ClientEns* shows higher accuracies, but as ϵ is increased and the privacy guarantees are loosened the accuracy of both algorithms in-

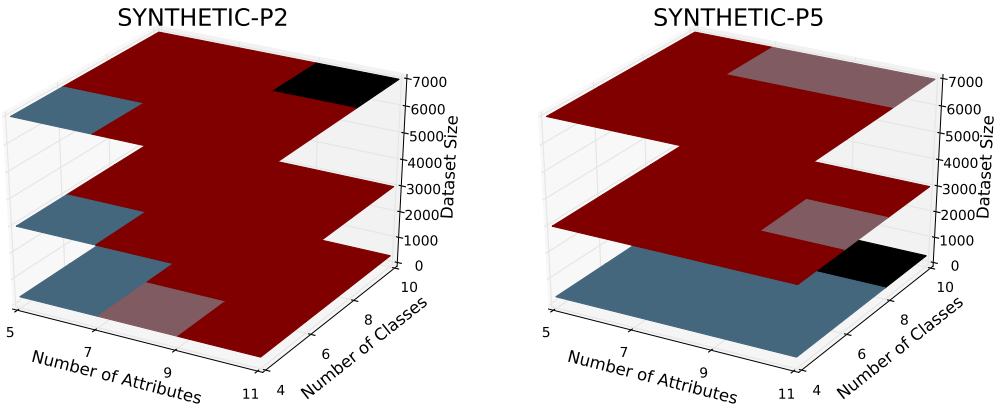

Figure 4: Difference between accuracies achieved by CuratorEns and ClientEns as a function of number of attributes, number of classes, dataset size (300, 3000, and 7000 datapoints per class) for two synthetic problems P2 and P5. Red - statistically significant positive difference (CuratorEns achieves better accuracy). Pink / Light Blue - statistically insignificant difference (positive / negative). Black - no data available. Using *1-1* code, *ELB* decoding technique and ϵ = 3.0

crease. *ClientEns*'s accuracy plateaus quickly, while that of *CuratorEns* continues to increase steadily and overpasses the accuracy of *ClientEns*. The cross-over in accuracies happens around $\epsilon = 1.0$. The difference in accuracies at higher values of ϵ are statistically significant.

The process of differentially-private anonymization can be though of as error injection into the real dataset. ECOC can recover from a certain of these errors. As the number of errors injected goes down (ϵ increases), the percentage of errors recovered from by ECOC is higher and higher. At some value of ϵ a point will be reached where ECOC the usage of ECOC is, so to speak, maximized, that is that ECOC recovers from all errors it can recover from. At this point a plateauing in accuracy occurs.

From section 4.1 we know that for *CuratorEns*, each copy of ECOC is executed on a ϵ/n-differentially private dataset, where n is the code width. This effectively reduces the value of ϵ in Figure 5 and thus *CuratorEns* reaches its terminal accuracy at higher values of ϵ.

From the user point of view there are two point that we need to underline. First, for *ClientEns* there is no reason to increase privacy guarantees a lot, as plateauing happens at relatively low values of ϵ. Second, to take advantage of the higher accuracy of *CuratorEns* we need to be able to work with lower privacy guarantees.

6.6 ECOC Parameters

In the rest of the paper we will investigate the dependence of the algorithms performance on various ECOC parameters: code type and decoding technique.

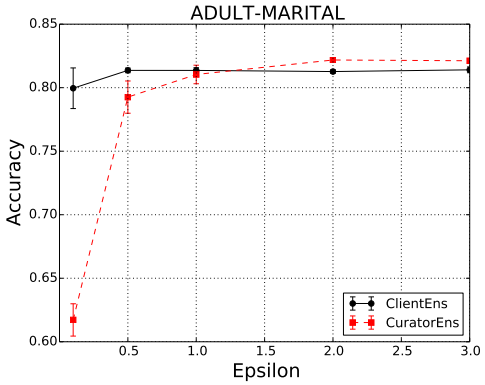

Figure 5: Accuracies of *ClientEns* (black circles), and *CuratorEns* (red squares) as a function of privacy guarantees (ϵ) for ADULT-MARITAL dataset using *1-1* code and *ELB* decoding technique.

6.6.1 Code Type

As described in section 3, many different codes can be used to encode a particular problem. These codes might lead to different performances in both *ClientEns* and *CuratorEns* methods. We tested five different codes: *1-All*, *Dense Random*, *Sparse Random* with probability of finding zero of 0.4 and 0.2, and *1-1* on three different datasets: ADULT with MARITAL STATUS class label, SHUTTLE and a synthetic dataset of problem P5 with 8 classes, 8 attributes and 7000 datapoints per class.

1-All and *1-1* codes are unique for value of the number of classes and are of set length. *Dense Random* and *Sparse Random* codes can be of variable length, plus several possibilities per a given length exist. For consistency, for each number of classes value, one code of each type: *Dense Random*, *Sparse Random, 0.1* and *Sparse Random 0.4* were created with the optimal length suggested by [10]. The same code was used for all datasets with equal number of classes.

Figure 6 presents a a comparison in accuracies between CuratorEns and ClientEns for these five different codes. All random codes were 56 columns in length. There are two features common to all four datasets. First there is a trend of increasing accuracy for "sparser" codes for both *ClientEns* and *CuratorEns*, even though the latter curve has a dip for two out of four datasets. Secondly, *ClientEns* does not exhibit as strong of a dependence on the code type as *CuratorEns* does. And finally, *CuratorEns* outperforms *ClientEns* only for the *1-1* code.

These three observations are not surprising. As noted in section 3, the "sparseness" of the code is very important for the *CuratorEns* method, as it increases the number of disjoint data subsets and allows to generalize on the same privacy budget. There is inevitably a variability between the number of disjoint data subsets between different codes so we may not expect a simple monotonic dependency. However, it is clear that the largest number of disjoint sets is provided by the *1-1* codes. The resulting performance of the code consists then of two components - the applicability of the code to the dataset, and the number of disjoint sets it allows for. The performance of the former is reflected by the performance of

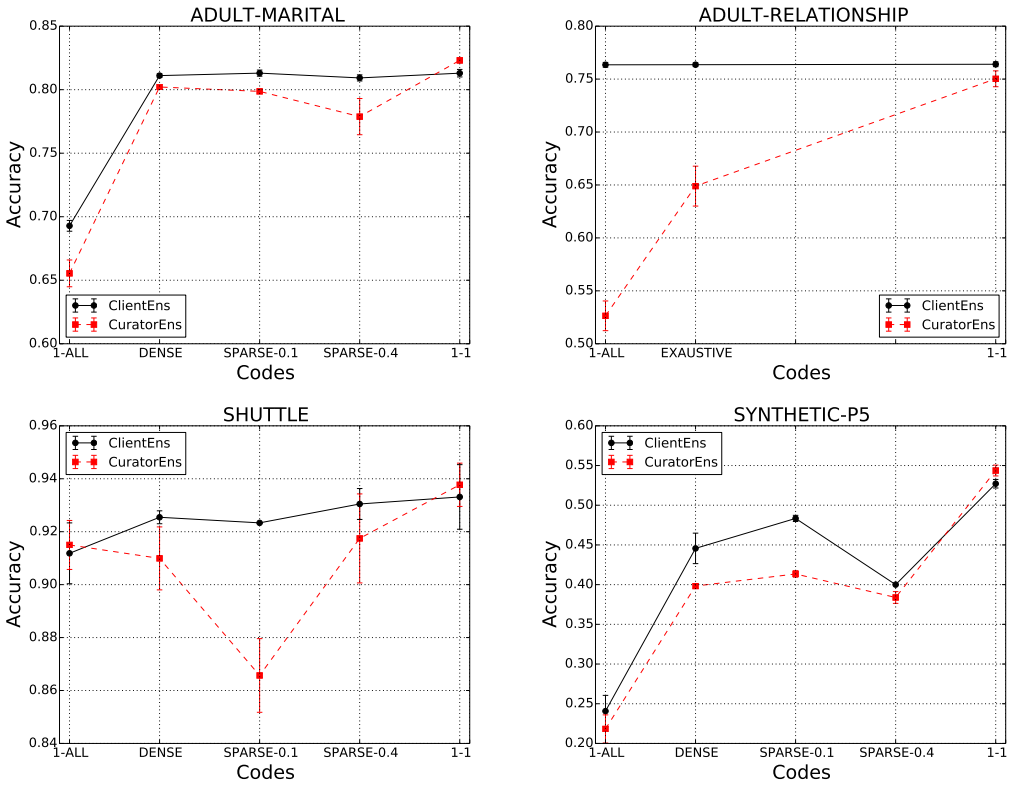

Figure 6: Accuracy of *ClientEns* (black circles) and *CuratorEns* (red squares) using different codes with *ELB* decoding technique and $\epsilon = 3.0$. From left to right: *1-All*, *Dense Random* width=56, *Sparse Random* with p(0)=0.1 and width=56, *SparseRandom* with p(0)=0.4 and width=56, *1-1*. For datasets ADULT-MARITAL, ADULT-RELATIONSHIP, SHUTTLE, SYNTHETIC-P5 with 6 classes, 6 attributes, 7000 pts/class

the *ClientEns*, leaving the performance of *CuratorEns* be a convolution of *ClientEns* and the number of disjoint sets allowed by the code.

Finally, for random codes, the code width can also be factor. We tested the effect of the code length for *Sparse Random* code with the probability of zero of 0.1 with ADULT-MARITAL dataset and found no statistically significant dependence of the accuracy on the code width for either algorithm.

6.6.2 Decoding ECOC Techniques

For datasets ADULT-MARITAL, ADULT-RELATIONSHIP, and SYNTHETIC for problem P5 with 6 attributes, 6 classes and 7000 datapoints per class, we also compared the accuracy achieved by *ClientEns* and *CuratorEns* with two different decoding techniques: *Hamming Distance (HD)* and *Exponential Loss-Based (ELB)* decoding. As table 6.6.2 shows, while the actual accuracy result might change with the decoding technique the relative accuracy achieved by the two algorithms remains the same for both decoding methods.

Overall, our finding from this section is three-fold. Firstly, *CuratorEns* is only advan-

NAME	CLIENTENS / CURATORENS	
	ELB	HD
ADULT-MARITAL-STATUS	0.813/ **0.823**	0.815 / **0.823**
ADULT-RELATIONSHIP	**0.764** / 0.750	**0.764** /0.750
SYNTHETIC 6 CLASSES, 6 ATTRIBUTES, 7000 SAMPLES PER CLASS		
P5	0.634 / **0.662**	0.522 / **0.527**

Table 4: Accuracies of CuratorEns and ClientEns using *1-1* code and *ELB* and *HD* decoding techniques. Boldface indicate significant difference between accuracies.

tageous using *1-1* code. Secondly, the advantage of *CuratorEns* is privacy constraint dependent. Lower privacy guarantees allows *CuratorEns* achieve higher accuracies, with the cross-over around $\epsilon = 1.0$. Lastly, ECOC decoding technique does not affect the relative advantage of *CuratorEns* over *ClientEns*.

7 Conclusions

In this paper we present novel algorithm for combining differential privacy data publishing task with an ensemble-based data-mining. Perturbations to the data lay at the core of differential privacy yet perturbations decrease the accuracy of data make data-mining tasks, for example, classifications. Ensemble methods are known to address this issues. Particularly, error-correcting output codes are known to be capable to compensate errors introduced by individual classifiers. Our algorithm allows to take advantage of this capability of ECOC and in certain cases recover from perturbations introduced by differentially-private data publishing.

We compared our algorithm to the use of ECOC on already differentially-private published data for several different datasets and investigated their relative accuracies as a function of ECOC parameters. Our findings are two-fold. First, we find that setting up ECOC before anonymization can yield better or similar accuracies for most large datasets with lower privacy guarantees. This accuracy advantage is independent of the classifier, used. Moreover, for classifier that do not require ECOC to operate, using ECOC is only advantageous in our novel approach.

Secondly, this accuracy increase, however, is present only for the *1-1* error-correcting code. The decoding method used affects the traditional and novel algorithms in the same manner and does not affect their relative advantage.

We recommend this technique be used on most large datasets, where the privacy guarantees can be loose.

This algorithm does require an increase in the running time, but only on the curator-side. The client execution time is not changed.

One of the unfortunate limitation of our algorithm is the fact that if ever more advantageious error-correcting codes are discovered, the dataset would have to be republished.

In future work we want to investigate the use of other ensemble methods in the non-interactive data publishing scenario. Random forest [6] and bagging [5] and the natural next candidates.

Acknowledgments

This work was partially supported by grants of Natural Sciences and Engineering Research Council of Canada. We thank Nicola Fanizzi from University of Bari for helpful discussions that suggested the original idea.

Appendix

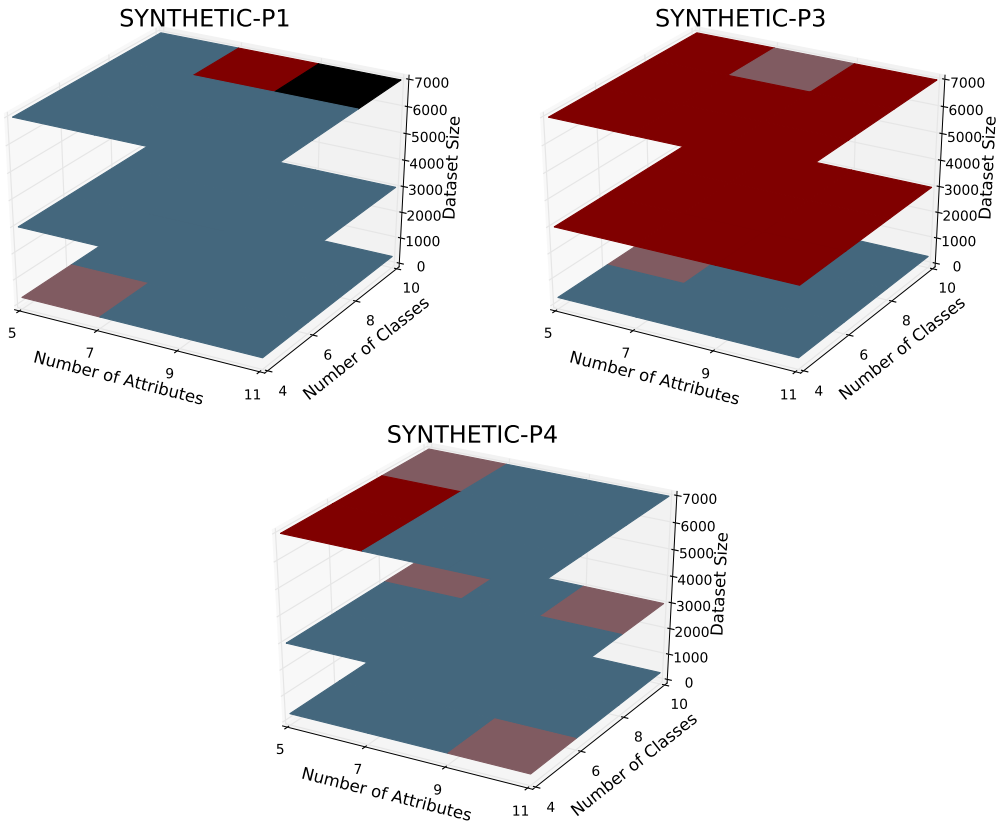

Figure 7: Difference between accuracies achieved by CuratorEns and ClientEns as a function of number of attributes, number of classes, dataset size (300, 3000, and 7000 datapoints per class) for three synthetic problems P1, P3 and P4. Red - statistically significant positive difference (CuratorEns achieves better accuracy). Pink / Light Blue - statistically insignificant difference (positive / negative). Black - no data available. Using *1-1* code, *ELB* decoding technique and $\epsilon = 3.0$

References

[1] Charu C. Aggarwal and Philip S. Yu. *Privacy-Preserving Data Mining: Models and Algorithms*. Springer Publishing Company, Incorporated, 1 edition, 2008.

[2] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. volume 29, pages 439–450, New York, NY, USA, May 2000. ACM.

[3] Erin L. Allwein, Robert E. Schapire, and Yoram Singer. Reducing multiclass to binary: a unifying approach for margin classifiers. *J. Mach. Learn. Res.*, 1:113–141, September 2001.

[4] Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to noninteractive database privacy. *J. ACM*, 60(2):12:1–12:25, May 2013.

[5] Leo Breiman. Bagging predictors. *Mach. Learn.*, 24(2):123–140, August 1996.

[6] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[7] Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *CoRR*, cs.AI/9501101, 1995.

[8] Cynthia Dwork. Differential privacy. In *ICALP*, pages 1–12. Springer, 2006.

[9] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil Vadhan. On the complexity of differentially private data release: Efficient algorithms and hardness results. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 381–390, New York, NY, USA, 2009. ACM.

[10] Sergio Escalera, Oriol Pujol, and Petia Radeva. On the decoding process in ternary error-correcting output codes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(1):120–134, 2010.

[11] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[12] 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*, KDD '10, pages 493–502, New York, NY, USA, 2010. ACM.

[13] Anupam Gupta, Aaron Roth, and Jonathan Ullman. Iterative constructions and private data release. In *Proceedings of the 9th International Conference on Theory of Cryptography*, TCC'12, pages 339–356, Berlin, Heidelberg, 2012. Springer-Verlag.

[14] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

[15] Moritz Hardt, Katrina Ligett, and Frank McSherry. A simple and practical algorithm for differentially private data release. In *NIPS*, pages 2348–2356, 2012.

[16] Geetha Jagannathan, Krishnan Pillaipakkamnatt, and Rebecca N. Wright. A practical differentially private random decision tree classifier. In *Proceedings of the 2009 IEEE International Conference on Data Mining Workshops*, ICDMW '09, pages 114–121, Washington, DC, USA, 2009. IEEE Computer Society.

[17] Zhanglong Ji and Charles Elkan. Differential privacy based on importance weighting. *Machine Learning*, 93(1):163–183, 2013.

[18] Chao Li and Gerome Miklau. An adaptive mechanism for accurate query answering under differential privacy. *Proc. VLDB Endow.*, 5(6):514–525, February 2012.

[19] Frank McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. *Commun. ACM*, 53(9):89–97, 2010.

[20] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 94–103, Washington, DC, USA, 2007. IEEE Computer Society.

[21] Noman Mohammed, Rui Chen, Benjamin C. M. Fung, and Philip S. Yu. Differentially private data release for data mining. In *KDD*, pages 493–501, 2011.

[22] Manas A. Pathak and Bhiksha Raj. Large margin gaussian mixture models with differential privacy. *IEEE Transactions on Dependable and Secure Computing*, 9(4):463–469, 2012.

[23] Grigory Yaroslavtsev, Cecilia M. Procopiuc, Graham Cormode, and Divesh Srivastava. Accurate and efficient private release of datacubes and contingency tables. In *Proceedings of the 2013 IEEE International Conference on Data Engineering (ICDE 2013)*, ICDE '13, pages 745–756, Washington, DC, USA, 2013. IEEE Computer Society.