SYNTHETIC DATA GENERATION WITH DIFFERENTIAL PRIVACY VIA BAYESIAN NETWORKS

ERGUTE BAO, XIAOKUI XIAO, JUN ZHAO, DONGPING ZHANG, AND BOLIN DING

***Abstract*—This paper introduces PrivBayes, a method that ensures differential privacy in the generation of synthetic datasets. It was employed in the 2018 Differential Privacy Synthetic Data Challenge hosted by NIST. PrivBayes focuses on creating synthetic datasets that retain the privacy of individual contributors while providing utility for analysiss.**

***Index Terms*—Differential privacy, synthetic data generation, Bayesian network.**

I. INTRODUCTION

Maintaining data privacy when published is an important issue, as differential privacy remains the best solution to this problem due to established privacy regulations and growing concerns about data breaches for the sake of itself,… but when we have data with very many attributes . Existing methods face challenges in such cases, as they add significant amounts of noise to the data compared to the actual data, making the published data less informative and potentially useless in research

The main idea of PrivBayes involves the generation of synthetic data from sensitive input data sets while maintaining privacy. It does this through several steps. Essenially PrivBayes is a Bayesian network, which serves two main purposes: (i) it is a private and concise representation of relationships between features in a dataset, and (ii) it is a low - dimensional cumulative distrbution of surrounding datasets the whole of the encounter

PrivBayes then injects noise into each of these low-level distributions, resulting in a noise that is very close to the original distribution. Importantly, the Bayesian network is designed and constructed in such a way as to conform to privacy principles and conform to miimum classification

Finally, using Bayesian networks and noise classification, PrivBayes generates artificial tuples, which are then released to act as a private repository to replace the original data This procedure ensures privacy but allows synthsis data a useful for research or other purpose

The main advantage of Privbayes is it avoids the curse of dimensionality by injecting noise into low dimension marginal distributions instead of the original high dimension data.

PROBLEM STATEMANT:

Let D be the input dataset with n participants,each associated with d attributes. We denote the attribute set as **A'** where |**A'**|= d. This data set D is the joint probability distributions of its attributes .Let this distribution be P(**A').** The goal of our task is to find P***(A)** such that it resembles the original dirstribution but satisfying with their privacy.

As discussed above it is done using Bayesian networks which narrows the dimensions of our distributions and also we take care that it satisfies differential privacy.

The detailed method how to perform it will be discussed below.

II. YOUR PROCEDURE OR YOUR METHOD

Now first we look into the condition for the generated data to be said differentially private.

Differential Privacy: We say to datasets are neighboring if one dataset is obtained by adding a record to other dataset. Differential privacy requires that any release of information about an input dataset should be done via a randomized algorithm **M.** such that the output of M does not reveal much information about any particular participant (tuple) in the input dataset. The output distributions of M should be similar on any two neighboring input datasets.

Now we say that the two neighbouring datasets are

differentially private if it follows the following condition

$$\Pr\left[\mathcal{M}(D_1) \in \mathcal{O}\right] \leqslant e^{\varepsilon} \cdot \Pr\left[\mathcal{M}(D_2) \in \mathcal{O}\right] + \delta.$$

Where D_1 and D_2 are the neighbouring datasets.
δ =0 for differentially private and value of ε tells how private is our new data generated.
In summary, PrivBayes operates through three distinct phases.
i)Network Learning: We construct a Bayesian network N using the attributes of **A'** it is a fully connected network of attribute and parent pair (AP pairs).
ii)Distribution Learning: Here we generate the joint distribution and conditional distributions for the Bayesian network.
iii)Data synthesis :From the result of the above methods we try to derive and approximate equation for D and then we generate tuples to form synthetic data D*.
We adopt DP and rely heavily on the analytic gaussian mechanism .
For example let the data be dataset D1 containing three attributes 'age', 'higrade', and 'income', where 'income' and 'age' are numerical attributes representing the income and age of a participant, respectively. For simplicity, we assume that each numeric attribute has a domain that is discretized into a number of ranges, e.g., attribute 'age' with range [0, 99] and it has subranges as [0,4],[5,9],……. The 'higrade' attribute is a categorical attribute that represents the highest grade of school attended by the participant.
Let the Bayesian network for the example be

1)Network Learning:
In network learning of PrivBayes, two distinct steps are involved. Initially, we calculate a differential privacy-protected score for each attribute pair. This score reflects the degree of correlation between attributes, with higher scores indicating stronger correlations and vice versa. Subsequently, a Bayesian network is constructed based solely on these differential privacy-protected scores, without reliance on additional information from the original dataset. It's important to note that the second step doesn't utilize any portion of the privacy budget allocated for the algorithm.

We calculate the score of two attributes and the attributes with high score are declared as parent attribute pair(AP pair).The score is calculated using the following formula.

$$\mathcal{S}(A_i, A_j) = \sup_{X_i \times X_j \subset \mathcal{A}_i \times \mathcal{A}_j} \left| C[X_i, X_j] - \frac{C[X_i]C[X_j]}{n} \right|$$

If A_i is age in our case then $C[X_i]$ is the number of individuals in a particular range of age (as assumed above).
sup denotes to calculate the total variation distance(tvd) between the two distributions in the $C[X_i, X_j]$ and $C[X_i]*C[X_j]$.
We observe **S** is minimum when the two attributes are independent and it is large when they are corelated.
Therefore, S could be a good score function for our purpose.
This step doesn't affect the privacy of the data since it only takes the number not the exact info of the tuple.
Based on this observation, we generate a differentially private score **S'** for each attributes pair by adding Gaussian noise with variance σ^2 the value of sigma is choosen such that it satisfies (ε, δ)DP. It is discussed later.

Algorithm 1: Generating noisy scores for attribute pairs

Data: The original dataset D
Result: The differentially private score of each attribute pair
foreach *attribute pair* (A_i, A_j) **do**
  compute $\tilde{\mathcal{S}}(A_i, A_j) \leftarrow \mathcal{S}(A_i, A_j) + \mathcal{N}(0, {\sigma_{NL}}^2)$;
end
return $\tilde{\mathcal{S}}(A_i, A_j)$ for all i, j;

Now comes the next part we construct the Bayesian network based on the scores calculated.
Given the score **S'**(A_i,A_j) for each attribute pair and we construct bayesian network for D by first running algo2and then algo 3.

Algorithm 2: Constructing the first AP pair in the Bayesian network $\mathcal{N}$

Data: The noisy scores of all the attribute pairs
Result: An AP pair
Identify the attribute pair (A_i, A_j) that maximizes $\frac{\tilde{\mathcal{S}}(A_i, A_j)}{|\mathrm{dom}(A_i)| \cdot |\mathrm{dom}(A_j)|}$;
Initialize $\Pi = \{A_j\}$, and $\mathcal{A}' = \{A_i, A_j\}$;
Initialize $x = \frac{\tau}{|\mathrm{dom}(A_i)| \cdot |\mathrm{dom}(A_j)|}$;
while $\mathcal{A}' \subset \mathcal{A}$ **do**
  Identify the attribute A_l in $\mathcal{A} \backslash \mathcal{A}'$ that maximizes $\sum_{A \in \Pi \cup \{A_l\}} \frac{\tilde{\mathcal{S}}(A, A_l)}{|\mathrm{dom}(A)| \cdot |\mathrm{dom}(A_l)|}$;
  if $\frac{x}{|\mathrm{dom}(A_l)|} > 1$ **then**
    $x \leftarrow \frac{x}{|\mathrm{dom}(A_l)|}$;
    Insert A_l into Π;
  end
  Insert A_l into $\mathcal{A}'$;
end
return (A_i, Π);

Algo 2 is used to find the root node and its child and then we use algo 3 to find the remaining AP pairs .

Algorithm 3: Constructing a subsequent AP pair in the Bayesian network $\mathcal{N}$

Data: The network $\mathcal{N}$ and the noisy scores of all the attribute pairs
Result: A subsequent AP pair in the network $\mathcal{N}$
Let $\mathcal{A}'$ be the set of attributes that appear in the AP pairs that have been constructed;
Initialize $AP \leftarrow \varnothing$;
foreach $A_j \in \mathcal{A} \backslash \mathcal{A}'$ **do**
  Identify the attribute set $\Pi' \subseteq \mathcal{A}'$ that gives the largest value sum of $\sum_{A \in \Pi'} \tilde{\mathcal{S}}(A_j, A)$ through a greedy approach, while satisfying: $|\mathrm{dom}\,(\Pi' \cup \{A_j\})| \leqslant \tau$.
  Update $AP \leftarrow AP \cup \{(A_j, \Pi')\}$;
end
return the best AP pair among all candidates in AP.

The choice of τ in the algo is important since it decides the size of the parent set.
An AP pair that contains a large number of attributes tends to capture more information from D, but its corresponding marginal distribution is more vulnerable to noisy injection since the cells in the marginal distribution table tend to have smaller values. Motivated by this, we impose an upper bound τ on the number of cells (domain size) allowed in a marginal distribution.
Let the number of attributes in the first AP pair be k then we construct d-k-1 AP pairs by invoking algo 3 by d-k-1 times.Briefly we find a set a attributes which are already part of the Bayesian network and then we find an attribute which is not part of it such that the tvd score is maximum as discussed above.
Algo 2 and 3 do not consume any privacy budget, since they utilize only the noisy scores of attribute pairs.
ii)Distribution Learning:

Algorithm 4: Generating differentially private joint and conditional distributions corresponding to the network $\mathcal{N}$

Data: The noiseless joint and conditional distributions of the original dataset
Result: The noisy version of the joint and conditional distributions
Initialize $\mathcal{P}^* = \varnothing$;
Materialize the joint distribution $P(A_1, ..., A_k)$;
Generate differentially private $P^*(A_1, ..., A_k)$ by adding noise Gaussian$(0, \sigma_{NC}{}^2)$;
for $i = k + 1$ **to** d **do**
  Materialize the joint distribution $P(A_i, \Pi_i)$;
  Generate differentially private $P^*(A_i, \Pi_i)$ by adding noise Gaussian$(0, \sigma_{NC}{}^2)$;
  Set negative values in $P^*(A_i, \Pi_i)$ to 0;
  Normalize all values in $P^*(A_i, \Pi_i)$ so that they sum up to 1;
  Derive $P^*(A_i \mid \Pi_i)$ from $P^*(A_i, \Pi_i)$, and add it to $\mathcal{P}^*$;
end
return $\mathcal{P}^*$;

Let k be the number of attributes in the first AP pair constructed by Algo 2. and $A_1,\ldots, A_k$ denote these k attributes. Let (A_{i+k+1}, π_i) denote the i-th AP pair constructed by the i-th invocation of Algo 3.Here we need to approximate the joint distributuion of $P(A_1,\ldots, A_k)$and also the condtional distributiions of $P(A_i|\pi_i)$. Algo 4 is the pseudocode for approximating these distributions while preserving privacy.It first generalizes the joint distribution and the adds gaussian noise to it to obtain a noisy distribution $P^*(A_1,\ldots, A_k)$.Thne the algo generalizes the $P(A_i,\pi_i)$. And injects gaussian noise to it and forms $P^*(A_i,\pi_i)$ and thenit finds $P^*(A_i|\pi_i)$ using conditional probabilities.

Having made this clear, we shall graphically describe how Gaussian noise is added to probability distributions. For instance, look at the marginal distribution of 'age'. This approach also works for joint distributions that have several attributes and conditional distributions as well. We then count the number of tuples falling in each subset within a discrete range of age from the input dataset. Tallying these figures throughout all subsets yields a histogram with a sensitivity of 1. For example, let's take the first histogram that counts as (1, 6, ..., 1) with accumulated count n =100. Hence in an initial input data set there would be found in this case 1, 6, ... , 1 persons of different age groups ranging from zero to four years old, five to nine years old up to ninety-five to ninety-nine years olds respectively. On the contrary, a distorted histogram can be represented as (1,3,...,0) with a cumulative count equal to 93. Moreover noisy histogram cannot be divided by n =100 straightforwardly in order to get the private distribution for age because n is sensitive information. The noisy marginal distribution for attribute 'age' is computed by (1/93,3/93,……0/93) where the denominator is the sum of all entries in the histogram,instead of n.

iii)Data Synthesis:

From these distributions we can derive the following equation.

$$P(\mathcal{A}) \approx P^*_{\mathcal{N}}(\mathcal{A}) = P^*(A_1, ..., A_k) \cdot \prod_{i=k+1}^{d} P^*(A_i \mid \Pi_i)$$

A general approach to generate tuples from P*(**A**) is to first sample $A_1,\ldots, A_k$ from $P^*(A_1,\ldots, A_k)$ and then we sample A_i from $P^*(A_i|\pi_i)$ for i=k+1 to d.Though this method does not give differentially private. This is due to the negative values in the noisy marginals,and inconsistences among marginal distributions. The probabilities with negative values we round up them to zero and inconsistency refers to when the sum of the probabilities of the attributes of AP pair doesn't add up to parents probabilities then we enforce consistency and sample.

Table 2. Example noisy marginal distributions for network $\mathcal{N}_1$.

(a) Marginal distribution for 'age'

'age'	probability
[0, 4]	0.04
[5, 9]	-0.1
...	...
[95, 99]	0.08

(b) Marginal distribution for 'age' and 'higrade'

'age' \ 'higrade'	01	02
[0, 4]	0.02	0.02
[5, 9]	-0.1	0
...	...	...
[95, 99]	0	0

Table 3. Processed noisy marginal distributions for network $\mathcal{N}_1$.

(a) Marginal distribution for 'age'

'age'	probability
[0, 4]	0.05
[5, 9]	0
...	...
[95, 99]	0.1

(b) Marginal distribution for 'age' and 'higrade'

'age' \ 'higrade'	01	02
[0, 4]	0.025	0.025
[5, 9]	0	0
...	...	...
[95, 99]	0	0

Table 4. Noisy marginal distributions for network $\mathcal{N}_1$ that are consistent on attribute 'age'.

(a) Marginal distribution for 'age'

'age'	probability
[0, 4]	0.05
[5, 9]	0
...	...
[95, 99]	0.067

(b) Marginal distribution for 'age' and 'higrade'

'age' \ 'higrade'	01	02
[0, 4]	0.025	0.025
[5, 9]	0	0
...	...	...
[95, 99]	0.0335	0.0335

from table 2 to 3 we round off the negatives to 0 and from 3 to 4 we ensure that the distributions are consistent.

Table 5. Histograms for tuple generation

(a) Histogram for 'age'

'age'	count
[0, 4]	500
[5, 9]	0
...	...
[95, 99]	670

(b) Histogram for 'age' and 'higrade'

'age' \ 'higrade'	01	02
[0, 4]	250	250
[5, 9]	0	0
...	...	...
[95, 99]	335	335

Table 6. Histograms for tuple generation after generating one tuple.

(a) Histogram for 'age'

'age'	count
[0, 4]	500
[5, 9]	0
...	...
[95, 99]	669↓

(b) Histogram for 'age' and 'higrade'

'age' \ 'higrade'	01	02
[0, 4]	250	250
[5, 9]	0	0
...	...	...
[95, 99]	334↓	335

One limitation of the "sampling with replacement" method is that it may fail to produce tuples having low but non-zero frequencies in the original input dataset. As a result, we suggest an alternative method: sampling without replacement. For instance, consider Figure 1 which shows a Bayesian network structure and Corresponding noisy marginal tables for tables 4a and 4b respectively. Initially, we convert the distributions shown in Tables 3a and 3b into histograms by scaling the probabilities by multiplying them with N(number of samples=10,000). In this way, we obtain Tables 5a and 5b.

Next, we go ahead to generate one tuple at a time. The first tuple comes from table Table5a , after table Table5b . After generating a tuple, we update Tables entries by decrementing these entries by one unit each.. Therefore if the number range of 'age' attribute for this tuple is 95 to 99 and that for higrade is equal to '01', then Tables are updated thus: Table6a &

Table6b. This process repeats until all tuples have been generated.

While experimenting with our approach of not replacing samples during sample piking resulted in a significant improvement,.

RESULT

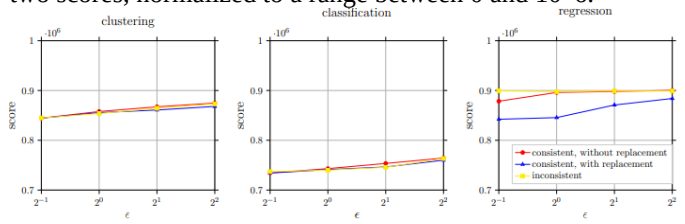
A dataset used for analysis in this section is the Colorado Public Use Microdata Sample (PUMS) data which contains 662,000 records with 98 categorical and continuous variables.

Performance metrics which are evaluated in this case include:

Clustering: Involves comparing the three-way marginal distributions between synthetic datasets and original ones on randomly selected marginals. This measures absolute differences between low-dimensional distributions obtained from both datasets by scaling it to range between 0(indicating perfect match) and 2 (no overlap). The resulting score is calculated as 106 times (1 decreased by half of the absolute difference). It is again averaged over 100 repetitions.

Classification: This metric selects one-third of all attributes randomly. For a numerical attribute, a random interval is picked while for a categorical attribute, some potential values are chosen randomly. The evaluation of how often tuples having those selected values appear in both original and synthetic datasets are done applying natural logarithms to these frequencies. Classification score is defined as the root mean square difference in log-frequencies between the original and synthetic datasets over 300 repetitions, divided by ln(p1/1000) and then multiplied by 10^6.

Regression: This metric computes an average score derived from two components. The first aspect involves measuring the mean squared error between the Gini coefficients of the original and synthetic datasets for all cities, calculated from the joint distribution of gender and income. The second aspect entails ranking the cities based on gender pay gap for both datasets and computing the mean-square deviation between the resulting city ranks. The final score is an average of these two scores, normalized to a range between 0 and 10^6.



Various levels of privacy budget (e) from 0.5 to 4 with δ ∼ 10−9 were used to measure the performance of PrivBayes.

The performance of PrivBayes is measured across different privacy budgets with ε varying between 0.5 and 4 and δ ∼ 10^-9.

For example, the evaluation metrics shown in Figure 2 above demonstrate how they have assessed the performance of PrivBayes. However, as ε increases, its data utility also improves. This metric approximates low-dimensional marginal distributions best among all other metrics which are the ultimate goal for optimization in PrivBayes. Nonetheless, despite not being devised specifically for classification or regression tasks, regression scores in PrivBayes are far better than their classification counterparts possibly due to intricate aspects linked to categorization exercises; furthermore, slight differences exist between clustering and classification measures when sampling from inconsistent or consistent distributions respectively. Interestingly enough, under tight privacy restrictions (small ε), a naïve sampling strategy works better than an optimal one – this implies that it might be possible to use effective heuristics to sample from noisy disjunctive marginal distributions

III. Conclusion

This paper highlights Differential Privacy Synthetic Data Challenge where we deployed PrivBayes. It also presents key findings from the challenge's result data and performance measures. We have subsequently explored alternative approaches to overcome two specific limitations of PrivBayes post-challenge. First, low-dimensional noisy distributions have disparities as stated above , thus limiting direct tuple formation. Second, PrivBayes constructs a Bayesian network with at most d low-dimensional distributions where d is the number of attributes in the input dataset. This restricts its modeling ability and consequently affects the usefulness of the synthetic data it generates.

.

Acknowledgment

I wish to thank ERGUTE BAO, XIAOKUI XIAO, JUN ZHAO, DONGPING ZHANG, AND BOLIN DING for their's work in the research paper of the SYNTHETIC DATA GENERATION WITH DIFFERENTIAL PRIVACY VIA BAYESIAN NETWORKS(. And all the one who helped me to complete the paper.Special to thanks to my Professor and TA for helping me.