DataDiP: An application to make your Data Differentially Private

Ajinkya Chaudhari
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
ajinkya.chaudhari18@vit.edu

Krish Agarwal
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
krish.agarwal18@vit.edu

Akshay Parseja
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
akshay.parseja18@vit.edu

Aayush Agarwal
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
aayush.agarwal18@vit.edu

Abhishek Sharad Ugale
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune, India
abhishekh.ugale18@vit.edu

***Abstract*— Differential privacy provides a way to get useful information about sensitive data without revealing much about any one individual. It enjoys many nice compositionality properties not shared by other approaches to privacy, including, in particular, robustness against side-knowledge. Since its inception in 2006, differential privacy has emerged as the de-facto standard in data privacy, owing to its robust mathematical guarantees, generalised applicability and rich body of literature. Over the years, researchers have studied differential privacy and its applicability to an ever-widening field of topics. Mechanisms have been created to optimise the process of achieving differential privacy, for various data types and scenarios**

With the rise of the data economy, companies are finding enormous value in collecting, sharing and using data. However, working with big data has its own risks and valid concerns should be made about the privacy of the users. For more than a decade, differential privacy has been a focal point for the research and development of new methods for data privacy. Using different mechanisms of IBM's DiffPrivLib library we aim to create 'DataDip', a user friendly platform for uploading a dataset and yielding differentially private results.

Keywords*— *Differential Privacy, machine learning, statistics, big data, API

I. Introduction

As volumes of personal data collected by many organizations increase, the problem of preserving privacy is increasingly important. The potential social benefits of analyzing such datasets drive many organizations to be interested in releasing statistical information about the data. In the field of privacy preserving data analysis, the main goal is to release statistical information about sample databases safely without compromising the privacy of any individuals who's records contribute to the database. These two conflicting objectives pose a challenging trade-off between providing useful information about the population and protecting the privacy of any individuals. As more and more companies amalgamate the insights generated by big data into the backbone of their business, and as a growing number of organizations harness the power of big data to make their enterprise stand out from the competition, most enterprise owners and security experts tend to willfully ignore the 'dark' side of big data. Dealing with big data can be a risk to the privacy of the users and Organizations commonly believe that keeping sensitive data secure from hackers means they're automatically compliant with data privacy regulations. This is not the case.

Data Privacy deals with how data is collected, shared and used. For example Netflix's case of 2007, researchers were able to breach privacy even when data was anonymized. Also people may want to share large datasets to share important conclusions with larger communities, but they may be restricted in doing so since the numeric data left behind after removing personally identifying data (e.g. names, addresses, social security numbers) can still be used to trace back to individual users. Thus, differential privacy is integral for preventing attacks on auxiliary data, which makes it a significant area of research.

Mathematically, the result of an algorithm A is -differentially private if for two datasets D1 and D2 that differ by exactly one element:

$$\Pr[A(D1) \in S] \leq e \ * \Pr[A(D2) \in S]$$

More intuitively, differential privacy describes that given the result of an algorithm on a dataset, it is hard to identify or obtain information pertaining to an individual entry of this dataset from simply looking at the result.

To that motive, we created DataDip– a novel tool that allows users to inject differential privacy into an user's private datasets by adding noise to it. DataDip also allows users to compare performance of machine learning models and the end user will also have an option to save the dataset in the repository provided on the website. The models that our tool supports are linear regression, logistic regression and K means and these have been imported from Diffprivlib The IBM differential Privacy Library.

Our work differs from the previous work in following ways:

1. We designed a tool that can be generalised to any kind of datasets.
2. We designed a tool that would allow users to upload their private datasets to the public.
3. Our tool allows an user to compare private machine learning vs non-private machine learning.

II. RELATED WORK

[1] In this paper Naoise Holohan has presented the IBM Differential Privacy Library, a general purpose, open source Python library for the simulation, experimentation and development of applications and tools for differential privacy. They have presented the functionality of Version 0.1.1 of the library, but also its user-friendly development ethos, smooth integration with the popular NumPy and Scikit-learn packages and wider development potential which will be built upon in the forthcoming months and years. Our tool was made using the models from this library.

[2] They have taken a look at data perturbation utilizing noise addition as a methodology used to provide privacy for published data sets. They also took a look at the statistical considerations when utilizing noise addition. They provided an illustrative example showing that deidentification of data when done in concert with noise addition would add more to the privacy of published data sets while maintaining the statistical properties of the original data set. However, generating perturbed data sets that are statistically close to the original data sets is still a challenge as consideration has to be made for the tradeoff between utility and privacy; the more close the perturbed data is to the original, the less confidential that data set becomes, and the more distant the perturbed data set is from the original, the more secure but then, utility of the data set might be lost when the statistical characteristics of the origin data set are lost.

The notion of differential privacy has received much theoretical attention in the privacy community and has been extensively studied in the literature [3, 4, 5, 6, 7]; a recent survey on differential privacy is provided in [8]. However, most research on differential privacy has focused on exploring theoretical properties of the model. The main focus of study has been how to safely release databases while preserving privacy for a particular function f. For example, [9] studies how to release count queries and [10] touches on more general query functions such 328 J. Lee and C. Clifton as histograms and linear algebraic functions. The concept of global sensitivity was introduced in [11] and it has been shown that releasing a database with noise proportional to the global sensitivity of the query functions achieves differential privacy. Nissim et al. [12] expanded the framework of differential privacy by introducing smooth sensitivity, which reduces the amount of noise added. It is motivated by the observation that, for many types of query functions, the local sensitivity is small while global (worst-case) sensitivity is extremely large. To decide the magnitude of noise, they use a smooth upper bound function S, which is an upper bound on local sensitivity

There have also been several researchers who have conducted previous work in creating tools that aim to provide the ability to query databases and manipulate their contents in a differentially private manner. Although these initiatives have lowered the barrier of entry in implementing data distribution techniques with strong privacy guarantees, at the current state, the average user still encounters hurdles in incorporating differential privacy.

We will now proceed to a presentation of a few tools that aim to provide such services, while briefly explaining their contributions. Lastly, we will demonstrate how Diff can fill in some of the gaps they leave.

[13] At its core, PINQ is a platform that provides a programmatic interface to unmodified data through a SQL-like language. PINQ imposes limitations to what can be learned about the data by performing queries that provide formal guarantees in the form of differential privacy to the users of the platform. It does this by doing the following: First, it allocates a privacy budget that is fixed and correlated to how many clients of the platform are allowed to learn about a particular dataset. This budget is then consumed through queries. After the budget has been exhausted, any further queries are left unanswered. PINQ's greatest contribution is the creation of a framework that permits differentially private interactions on datasets managed by developers that do not require expert knowledge on the topic. It achieves this by abstracting away the complexity that comes by having to enforce privacy away from developers, allowing them to solely focus on the logic of the application. Although PINQ undoubtedly lowers the barrier of entry of using differentially private methods from solely experts to any developer, it is still unrealistic for an average user to take advantage of differential privacy. On the other hand, DataDip, although not as powerful or expressive, allows users to interact with it in the form of a simple web application that does not demand extensive technical knowledge.

[14] Airavat is a MapReduce based platform that provides strong privacy and security guarantees for distributed computations. Its objective enables the usage of untrusted or not thoroughly inspected code to analyze sensitive information and generate aggregate computations that stem from input datasets. It does this without exposing information about particular entries on the set. By using mandatory access control in conjunction with added differential privacy in the form of added Laplacian noise, Airavat provides an end-to-end privacy guarantee.Although the end goals of Airavat and DataDip are similar in the sense that they attempt to simplify the process of applying differential privacy, the scope in which they try to achieve this objective is fundamentally different. While Airavat establishes a foundation on using arbitrary code to simply provide privacy guarantees in a distributed architecture, DataDip aims to simplify the user experience such that the average user would be able to use it and take advantage of its benefits. To this extent, future iterations of DataDip could try to incorporate systems like Airavat. Moreover, it could expand upon its scalability as well as the types of computations it can handle. provides an end-to-end privacy guarantee.

III. METHODOLOGY

The authors designed this tool using local differential privacy as the base methodology. Our primary aim was to help users to understand the benefits of differential privacy and to allow them to share their datasets through a public repository without having to worry about privacy concerns. Second, the authors want to depict the difference between private machine learning and non-private machine learning to allow the users to test their own datasets and compare the two outcomes. In this section we have described how we designed our model.

Our tool has three prominent features:

a. Addition of noise through DP mechanisms

b. Comparison between DP-ML and non private ML
c. Storage of shared datasets

A. Addition of noise through DP mechanisms

Our most important feature involves making data differentially private by the usage of DP mechanisms. This method involves adding statistical noise to single data points to randomise them to the output. We primarily used the mechanisms module from IBM's diffprivlib library for this purpose. As stated in [1], these mechanisms have been built primarily for their inclusion in more complex applications and models, but can be executed and experimented with in isolation. Thus, we could leverage this into our application.

Mechanisms are interacted with using function-specific methods, reducing the need for code and documentation duplication. For example, each mechanism has a set_epsilon() (and/or setepsilondelta()) method to set the ε(and δ) parameters of the mechanism,and many mechanisms have a setsensitivity() method. Similarly, each method has a randomise() method, which takes an input value and returns a differentially private output value, assuming the mechanism has been correctly configured.

In our tool, we made use of five mechanisms out of twelve available. In the table below, you can see the mechanism used for a specific data type.

Table I: Mechanism VS Datatype

Mechanisms	Datatype
Laplace	Numerical Data
Geometric	Numerical Data
Gaussian	Numerical Data
Exponential	Categorical Data
Binary	Binary Data

B. Comparison between DP-ML and non private ML

Our second feature is designed to allow users to test their datasets using machine learning models and compare between private models and non-private models.

The models developed in diffprivlib have been engineered to mirror the behaviour and syntax of the privacy-agnostic versions present in Scikit-learn. This allows for a simple one-step change to switch from a vanilla Scikit-learn machine learning model to one which implements differential privacy with diffprivlib. Thus, we used the models from diffprivlib and compared them with sklearn's. In the following table, we have shown which models are available in the tool and the metrics used to compare them.

Table II: Types of machine learning models

Type	Model	Metric
Classification	Logistic Regression	Classification accuracy
Regression	Linear Regression	R2 score
Clustering	K-Means	Silhouette average

C. Storage of shared datasets

Our third feature is a novel functionality that allows any user to upload their private datasets from thier local device to the repository available in our tool. They can privatise the dataset through our feature 1, and it will automatically upload to our public repository. This feature would allow researchers and industrialists to experiment with large privatised datasets which do not contain any personally identifiable information. The analyst will not be able to back-engineer to the original dataset, as we have ensured the addition of 'local differential privacy'.

D. API Design

Finally, in order to combine all our features into one tool, we designed an API using Flask. We have ensured that through Flask, we have a concrete backend which would process large amounts of data. The API is currently hosted locally, but will be running on servers in the future.

E. WorkFlow

In this section, we have described how an user would use our tool. This is a basic workflow of the entire tool. The user will have the liberty to select any tool and explore the tool other than the features too. The criteria for datasets to be uploaded is that it should have a predefined row of column names and should be of .csv format.

- Open website
- Select the feature of your choice (1,2,3).
- If 1:
    - Input csv file
    - Input details of data (columns to be privatised)
    - Select DP parameters (epsilon, delta, sensitivity)
    - Output: privatised csv file, and move the file to repository.
- If 2:
    - Input csv file
    - Input details of data
    - Select DP parameters
    - Select ML parameters (split, input var, output var, no. of clusters etc.)
    - Output: Numerical output of metrics

- If 3:
    - Download files available in the repository.

IV. RESULTS

We successfully integrated our three features through the usage of Flask API, as mentioned in the previous section. In order to experiment with our tool we used the 'Adults.csv' dataset publicly available over the UCI repository. This dataset has 14 attributes, is multivariate, and has categorical as well as integer values. In this section, we describe the results of experimenting with our tool.

A. Addition of Statistical Noise:

In the following table, we have shown the difference between the first seven original values of the Age column in the Adult dataset and the privatised values. Any adversary cannot deduce the amount of noise added and the parameters used to define it, even though the values alter minimally.

TABLE III- Difference between original and DP values

Original Values	DP values
25	23
38	39
28	28
44	43
18	20
34	34
29	27

B. Comparison of machine learning models:

We tested the second feature using the same dataset. We tested all of the machine learning algorithms available. Therefore, the authors did a random feature selection of input and output variables. The user will have to define specific details about the dataset provided. We used a 70:30 split for all the algorithms. For the differential private parameters, we defined epsilon as 0.1 and sensitivity as 0.1. For classification, our target variable was 'Gender', and l2norm was set to 0.1. For regression, or target variable was 'Salary' as it is a continuous variable. For clustering, we set k=3. Each algorithm outputs numerical metrics which would be interpreted through bar graphs. In order to achieve meaningful results, we suggest that the user does vigorous feature selection. In the table given below, we show the difference between diffprivlib's models and sklearn's models, when tested on our data. The metrics used for comparison are given in section III.A.

TABLE IV- Comparison of ML models

Type of algorithm	Non DP model (sklearn)	DP model (diffprivlib)
Classification	66%	33%
Regression	0.006	-0.323
Clustering	0.557	0.508

V. CONCLUSION

We successfully devised a tool to leverage the usage of differential privacy to help protect personal data and experiment with ML algorithms. We aim that the tool will be useful for users who want to make their data public for research purposes or solely want to make their data private for personal use. We aim to work on following aspects in the near future:

1. Allowing single querying onto datasets in repository.
2. Pre-determine the amount of noise to be added by understanding the data better.
3. Make the tool available for integration to existing systems.
4. Improve generalising ability of the algorithms

We aim to make our tool public in the future.

ACKNOWLEDGMENT

We are thankful to our institute Vishwakarma Institute of Technology for allowing us to take up this work during our academics. We appreciate the guidance and mentorship provided to us by IBM Bangalore and IBM Ireland. Lastly, we would like to thank our academic mentor Dr. Manikrao Dhore, whose valuable inputs enabled us to improve our tool and conduct impactful research.

REFERENCES

[1] Naoise Holohan, Stefano Braghin, Pol Mac Aonghusa, Killian Levacher "Diffprivlib: The IBM Differential Privacy Library"

[2] Kato Mivule, "Utilizing Noise Addition for Data Privacy, an Overview"

[3] Blum, A., Dwork, C., McSherry, F., Nissim, K.: Practical privacy: the SuLQ framework. In: Proceedings of the Twenty-Fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 128–138. ACM, New York (2005)

[4] Blum, A., Ligett, K., Roth, A.: A learning theory approach to non-interactive database privacy. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, pp. 609–618. ACM, New York (2008)

[5] Dwork, C., Nissim, K.: Privacy-preserving data mining on vertically partitioned databases. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 528–544. Springer, Heidelberg (2004)

[6] Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our data, ourselves: Privacy via distributed noise generation. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 486–503. Springer, Heidelberg (2006)

[7] Barak, B., Chaudhuri, K., Dwork, C., Kale, S., McSherry, F., Talwar, K.: Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In: Proceedings of the Twenty-Sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 273–282. ACM, New York (2007)

[8] Dwork, C.: Differential privacy: A survey of results. In: Agrawal, M., Du, D.-Z., Duan, Z., Li, A. (eds.) TAMC 2008. LNCS, vol. 4978, pp. 1–19. Springer, Heidelberg (2008)

[9] Dinur, I., Nissim, K.: Revealing information while preserving privacy. In: Proceedings of the Twenty-Second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 202–210. ACM, New York (2003)

[10] Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)

[11] Dwork, C.: Differential privacy. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006, Part II. LNCS, vol. 4052, pp. 1–12. Springer, Heidelberg (2006)

[12] Nissim, K., Raskhodnikova, S., Smith, A.: Smooth sensitivity and sampling in private data analysis. In: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, pp. 75–84. ACM, New York (2007)

[13] F. McSherry, PINQ: Privacy Integrated Queries. Association for Computing Machinery, Inc. 2009.

[14] Indrajut Roy, Srinath Setty, Ann Kilzer, Vitaly Shmatikov, Emmet Witchel. Airavat: security and privacy for MapReduce. USENIX Association 2010