Effects of Noise on Machine Learning Algorithms Using Local Differential Privacy Techniques

Krishna Chaitanya Gadepally
Dept. of Electrical and Electronics Engineering
Birla Institute of Techology and Science, Pilani
Hyderabad, India
krishna.gadepally@gmail.com

Sameer Mangalampalli
Data and Machine Learning Engineering
Comatrix Labs
Hyderabad, India
sameer.m1@comatrixlabs.io

2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS) | 978-1-6654-4067-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/IEMTRONICS52119.2021.9422609

***Abstract*— Noise has been used as a way of protecting privacy of users in public datasets for many decades now. Differential privacy is a new standard to add noise, so that user privacy is protected. When this technique is applied for a single end user data, it's called local differential privacy. In this study, we evaluate the effects of adding noise to generate randomized responses on machine learning models. We generate randomized responses using Gaussian, Laplacian noise on singular end user data as well as correlated end user data. Finally, we provide results that we have observed on a few data sets for various machine learning use cases.**

Keywords—Differential Privacy, Randomization, Privacy

I. INTRODUCTION

Data is being collected in large amounts from a web browser to purchasing of data for applying machine learning algorithms to engage with customers. Machine learning algorithms work by studying a lot of data and updating their parameters to encode the relationships between features and the output in that data. If Machine learning is used to solve important tasks, like making a cancer diagnosis model, credit card eligibility, customer engagement, customer lifetime value, then features need to be developed using personally private information leading to invasion of privacy. In a study by PWC more than 83% of participants did not want to share their due to fear of privacy loss. [1]

In the 1990s, the American state of Massachusetts made medical data publicly available which included minimal demographic information: birth date, zip code and gender, in addition to the diagnosis. A computer scientist Latanya Sweeney purchased a voter list with 54,000 names. Sweeney was able to link the demographic information in the medical records released by the government to the demographic information in the voter database and in many cases, she was able to match an unlabelled medical record to the patient's name using the voter list she had bought earlier. She tried this data-matching technique for the then-governor William Weld. Just six people in the city of Cambridge shared his birthday. Out of the six, only three of them were men. Weld was the only one who lived in the correct zip code. Cynthia Dwork came up with a new way to define privacy as a quantifiable measure, also giving a formal guarantee that information is not leaked, called "differential privacy". [2]

"Differential privacy" describes a promise, made by a data holder, or curator, to a data subject: "You will not be affected, adversely or otherwise, by allowing your data to be used in any study or analysis, no matter what other studies, data sets, or information sources, are available." [3]
Differentially private algorithms are able to answer a large number of aggregates, statistical queries approximately, so that the approximate answers can draw roughly the same conclusions as if the original data. Local differential privacy is a model of differential privacy with the added restriction that approximates personal responses of an individual such that we do not reveal too much about the user's personal data.
Some of the methods in Differential privacy methods are:

Randomization
Laplace Method
Exponential Mechanism

In this paper we explore methods for local differential privacy by adding noise to generate randomized responses for features in Machine learning models.

II. DEFINITIONS

Our idea is to explore methods adding noise and generate randomized responses to be used in features. Here are few ways noise can be expressed mathematically:

1. Noise limitations expressed mathematically:

Gaussian noise

x: Gaussian random variable
The probability density function, p of a Gaussian random function, x is:

$$p(x) = \frac{1}{\sigma\left(\sqrt{2\pi}\right)} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where μ is the mean and σ is the standard deviation.

Laplacian mechanism

The Laplace Distribution (centred at 0) with scale b is the distribution with probability density function:

978-1-6654-4067-7/21/$31.00 ©2021 IEEE

$$L(x|b) = \frac{1}{2b} e^{-\left(\frac{|x|}{b}\right)}$$

where b is a scale parameter.

2. Pearson Correlation

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 (y_i - \bar{y})^2}}$$

where:
r = correlation coefficient
x_i = values of the x-variable in a sample
$\bar{x}$ = mean of the values of the x-variable
y_i = values of the y-variable in a sample
$\bar{y}$ = mean of the values of the y-sample

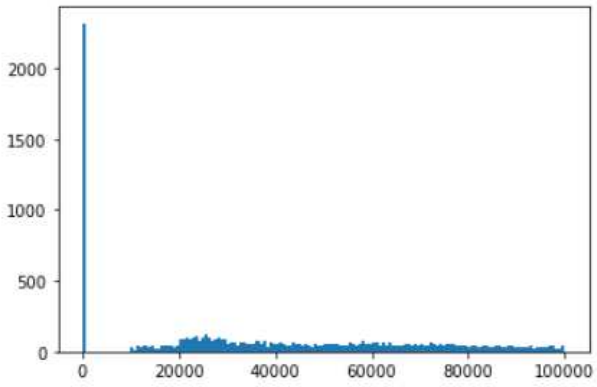
III. METHODOLOGY

A sample dataset with a selected subset of features is shown below. 'Engaged' is the output of the dataset.

Engaged	Income	Number of Open Complaints	Number of Policies	Total Claim Amount
0	71941	0	2	198.23
1	21604	0	1	379.20

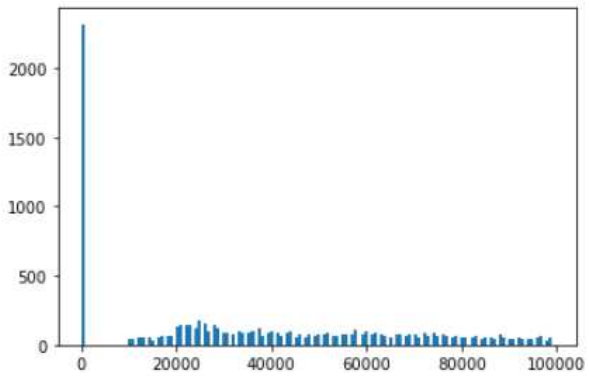
Before noise was added to Income:

Shown below is a histogram plot of 'Income' before noise addition to 'Income'.

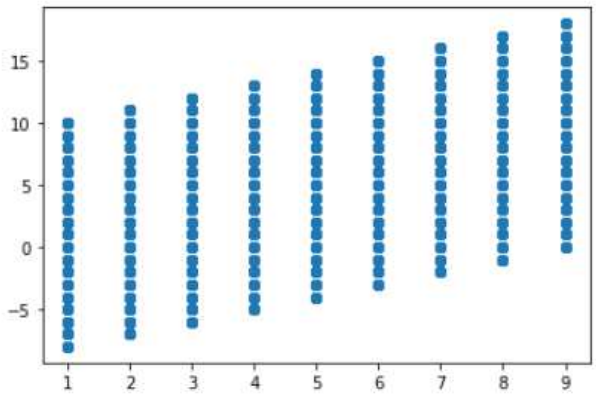


After noise is added to Income:

Shown below is a histogram plot of 'Income' after noise addition to 'Income'.



Shown below is a scatter plot which shows how 'Number of Policies' values' change as noise is added.
The horizontal axis corresponds to 'Number of Policies' variable before noise addition.
The vertical axis corresponds to 'Number of Policies' variable after noise addition.



Pseudo Code	
Input	f(x) = {f1, f2, ..., fn}: features of x G(x): Gaussian noise function

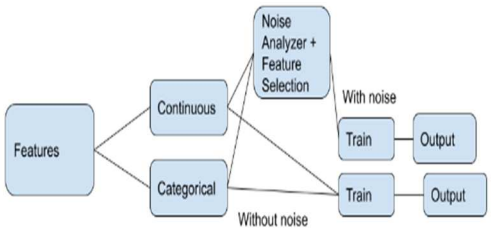
Function	Repeat for all features f in {f1, f2, ..., fn}:
	Take a random feature column, fx ∈ f
	For each M in {Logistic Regression, Random Forest Classifier}:
	if fx is categorical:
	call pseudoCodefunction1
	else
	call pseudoCodefunction2

Pseudo Code -- Function 1	
Input	Example data x1, x2, …, xn in X Random sample percentage for selecting values to add noise
Function	For x ∈ X:

	Take random sample x with sampling probability L/N Compute For each value compute the noise g(x): Clip noise g(x) to g(x)' Add Noise x' = x + g(x)' within a bound Fx' append x' where Fx' is the new noise feature

Pseudo Code -- Function 2	
Input	Input: Example data x1, x2, ..., xn Random sample percentage for selecting values to add noise.
Pre-process	If Categorical introduce a new random value based on data: Method 1: Increase a max bound Method 2: Introduce a categorical value in range Bounds changed and gaussian function updated
Function	For x ∈ X: Take random sample x with sampling probability L/N Compute For each value compute the noise g(x): Add Noise x' = x + g(x) within a bound Fx' append x' where Fx' is the new noise feature

IV. RESULTS



The results were determined using the above workflow for ML training. The Noise Analyzer and Feature Selection module were used to add noise to the features. This was done using Python and scikit-learn in Jupyter notebooks. Finally, the ML models were trained with noise and without noise as expressed in the flow.

The IBM-Watson Customer Marketing Value data set was selected. 'Engaged' is the output column. 'Yes' and 'No' in 'Response' feature were converted into 1 and 0 respectively and made the output column. Categorical variables were factorized and added to the dataset.

Eleven features were chosen for randomization, namely, 'Customer Lifetime Value', 'Education', 'Gender', 'Income', 'Location Code', 'Marital Status', 'Months Since Last Claim', 'Months Since Policy Inception', 'Number of Open Complaints', 'Number of Policies' and 'Sales Channel' from the data set.

Model: Logistic Regression

1. Before Gaussian noise and randomized responses were added to the features, a logistic regression model was trained on the data set. The training accuracy was 86.41%.

Feature chosen	Trained on noisy dataset; Tested on original dataset	Trained on original dataset; tested on noisy dataset
Customer Lifetime Value	86.41%	86.39%
Education	86.40%	86.32%
Gender	86.40%	86.40%
Income	86.39%	86.45%
Location Code	86.40%	86.30%
Marital Status	86.40%	86.41%
Months Since Last Claim	86.36%	86.19%
Months Since Policy Inception	86.42%	86.26%
Number of Open Complaints	86.40%	86.22%
Number of Policies	86.35%	86.11%
Sales Channel	86.45%	86.38%

Model: Random Forest Classifier

2. Before Gaussian noise and randomized responses were added to the features, a random forest classifier was trained on the data set. The training accuracy was 100%.

Feature chosen	Trained on noisy dataset; tested on original dataset	Trained on original dataset; tested on noisy dataset
Customer Lifetime Value	100%	100%
Education	100%	99.26%
Gender	100%	100%
Income	100%	100%
Location Code	100%	99.37%
Marital Status	100%	99.75%

Months Since Last Claim	100%	98.9%
Months Since Policy Inception	99.99%	97.89%
Number of Open Complaints	100%	99.98%
Number of Policies	99.99%	100%
Sales Channel	100%	99.77%

3. In contrast to the previous results, here three features are considered instead of one for randomization.

'Customer Lifetime Value', 'Income' and 'Marital Status' are chosen for randomization.

Model	Trained on noisy dataset; tested on original dataset	Trained on original dataset; tested on noisy dataset
KNN	100%	100%
SVM	85.68%	85.68%

'Months Since Last Claim', 'Months Since Policy Inception', 'Number of Open Complaints' are chosen for randomization.

Model	Trained on noisy dataset; tested on original dataset	Trained on original dataset; tested on noisy dataset
KNN	96.66%	94.70%
SVM	85.68%	85.68%

4. 'LocationCode' was found to have the highest Pearson Correlation value, which meant that 'LocationCode' was the feature which had the closest to a linear relationship with the output 'Engaged'.

Model	Trained and tested on original dataset	Trained on noisy dataset; tested on original dataset	Trained on original dataset; tested on noisy dataset
KNN	100%	99.9%	99.68%
SVM	85.68%	85.68%	85.68%
Logistic Regression	86.41%	86.38%	86.30%
RF	100%	100%	99.42%

V. CONCLUSION

From our experiments we can safely conclude machine learning applications will not be affected by differential privacy preserving mechanisms and in fact can become allies. If proper care is taken to add noise to the data, we have seen the models are almost alike with noise and without noise. We could easily extend the methodology to build features using SQL join, distributed joins as well as advanced data aggregation methods. There are several real world machine learning applications like targeting customers, marketing funnels, segmentation, churn prediction in ecommerce, communication, retail as well as health care applications where differential privacy and local differential privacy techniques could be applied in order to protect the privacy of the end user data. Personalization while preserving end user privacy is something that we all want and we hope we were able to present a strong case for it in our study.

VI. REFERENCES

[1] PwC. 2021. *Consumers Trust Your Tech Even Less Than You Think*.

[2] Dwork, C., 2021. *Differential Privacy*.

[3] Dwork, C. and Roth, A., 2014. *The algorithmic foundations of differential privacy*.

[4] Privacytools.seas.harvard.edu. 2021. *Differential Privacy*.

[5] Geng, Q. and Viswanath, P., 2021. *Optimal Noise Adding Mechanisms for Approximate Differential Privacy*.

[6] Mironov, I., Talwar, K. and Zhang, L., 2021. *Renyi Differential Privacy of the Sampled Gaussian Mechanism*.

[7] Erlingsson, U., Pihur, V. and Korolova, A., 2021. *RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response*.

[8] Agrawal, D., Aggarwal, C. 2001. *On the Design and Quantification of Privacy Preserving Data Mining Algorithms.*

[9] Chawla, S., Dwork, C., McSherry, F., Smith, A., Wee, H., 2005. *Toward Privacy in Public Databases*.