Made by:
Atoyan George

What is Differential Privacy?

Imagine that all companies, universities, medical centers can share data containing sensitive or confidential information about individuals, Apple and Google can analyze the text that users type on their keyboard every day and make a better product using this without big data breaches. We can do it with differential privacy methods.
Suppose we have a database and want to return output (e.g. Number of users with blue eyes). But we should do it only in a private way. To make the differential privacy process you need to add random values or noise. How exactly to add it and how much depends on a particular task.
Differential privacy provides a mathematically rigorous definition of privacy and is one of the strongest guarantees of privacy available. It is rooted in the idea that carefully calibrated noise can mask a user's data. When many people submit data, the noise that has been added averages out and meaningful information emerges.
Let's take a look at the two main models of differential privacy: ε-differential privacy and (ε, δ) - differential privacy. In a nutshell, the main difference between the models is that the second one assumes the possibility that the attacker can get the whole database into possession where the DP will not work, i.e. it is possible that a very bad case will happen.

ε-differential privacy:

$$\mathbb{P}\left[A(D_1) = O\right] \leq e^{\varepsilon} \cdot \mathbb{P}\left[A(D_2) = O\right]$$

- $\mathbb{P}\left[A(D_1) = O\right]$ is the probability that we get $[O]$ result or some statistic when running the process A on the dataset D_1. $P\left[A(D_2) = O\right]$ is similar but D_2 is the database which differs in only one individual.

- $epsilon$ is a metric of privacy loss at a DP change in data (adding/removing 1 entry data of person). If $epsilon$ = 0, it means process is absolutely private, because $\mathbb{P}\left[A(D_1) = O\right] = \mathbb{P}\left[A(D_2) = O\right]$. Algorithm A doesn't depend on data and thus, protects data perfectly.
- But the smaller $epsilon$ the smaller accuracy you can get. We will get nothing except the noise. In Figures 1 and 2 shown two processes: with $epsilon = 1$ and $epsilon$ is close to zero.

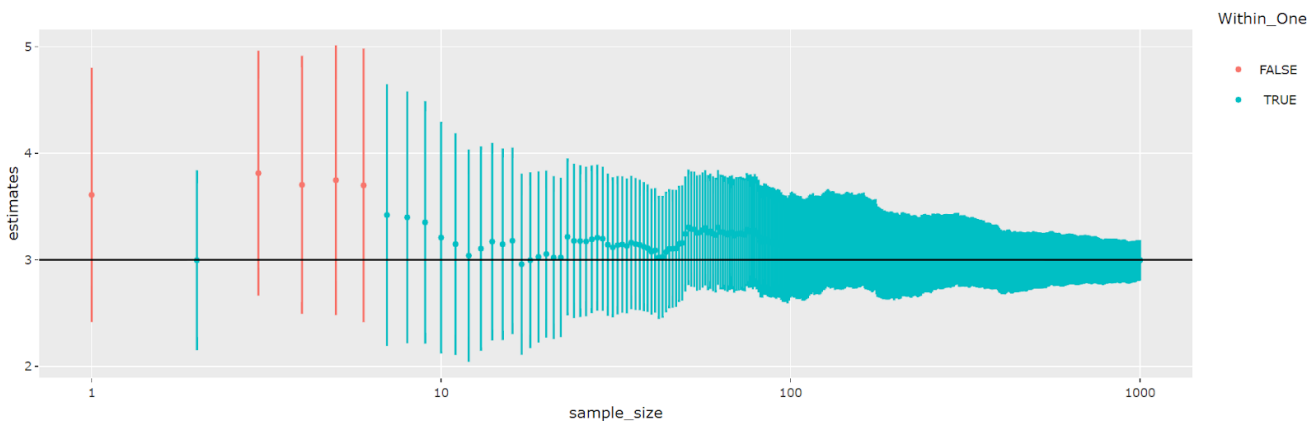
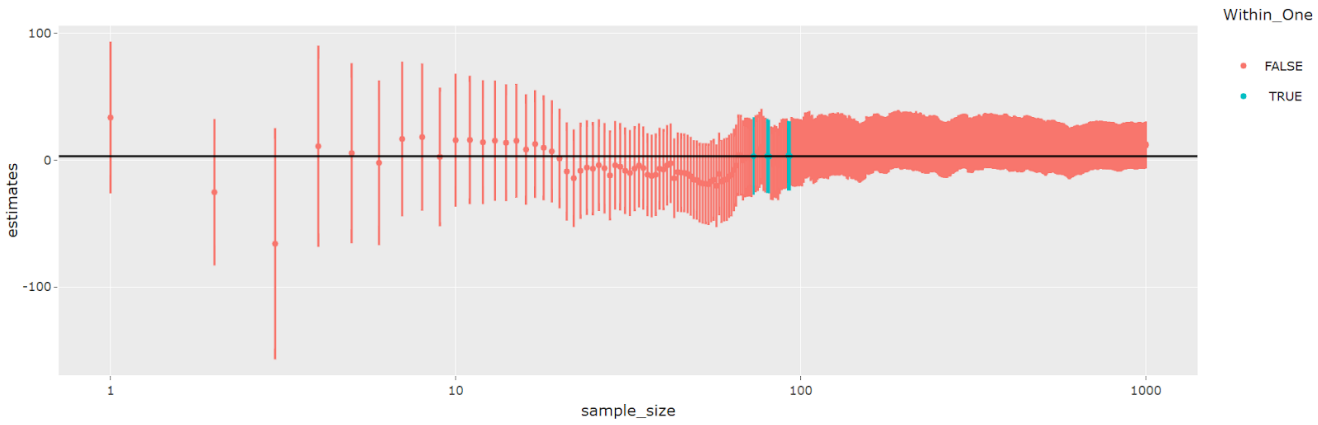


Figure 1. *This simulation demonstrates the Laplacian Noisy Counting mechanism (a differentially private algorithm). In a nutshell, this mechanism preserves the privacy of the ground truth (which is 3). With each new query from an adversary, our data curator returns the ground truth with additional noise. This noise is drawn from a zero-centered Laplace distribution with a parameter equal to (1 / epsilon). The quantity epsilon is a positive, tunable variable, and is the center of entire field of differential privacy. To be brief, the smaller the epsilon, the more private the results. On the x-axis, we have the number of consecutive queries that an adversary sends to the curator. Notably, it is logarithmically scaled. On the y-axis, we've plotted the average of the query replies that the curator sends out, along with the 95% confidence interval of those queries*

Figure 2.



(ε, δ) - differential privacy:

$$\mathbb{P}[A(D_1) \in S] \leq e^{\varepsilon} \cdot \mathbb{P}[A(D_2) \in S] + \delta$$

- δ- it captures the odds that something goes wrong (actually, tt's not really the worst-case scenario)[1]

By using (ε,δ)-differential privacy, we're saying that the algorithm is *almost* ε-differentially private. And here, *almost* means *with probability* 1−δ: the closer δ is to 0, the better.

There are also 2 types of differential privacy: global and local.
Global: in this approach, we aggregate data in one place (e.g. google collects data from users' keyboards on its servers) and only then adds noise. The disadvantage is the high probability of data privacy loss. The advantage is a little noise.
Local: here, we add noise to the data first and then aggregate it (for example, your phone without connecting to external servers can do this). Minus that is high noise in the data, plus that is privacy.

Let's consider two approaches to DP: classical methods and deep learning.

We're running this study on the CIFAR-10 dataset. CIFAR-10 includes 60 thousand pictures with resolution of 32x32x3 with 10 different classes like dog, cat, airplane, and others.

Classical methods

We are using 3 models: gaussian naive bayes and logistic for our classification experiment. All implementations were taken from IBM *diffprivlib* library.
How do they work:

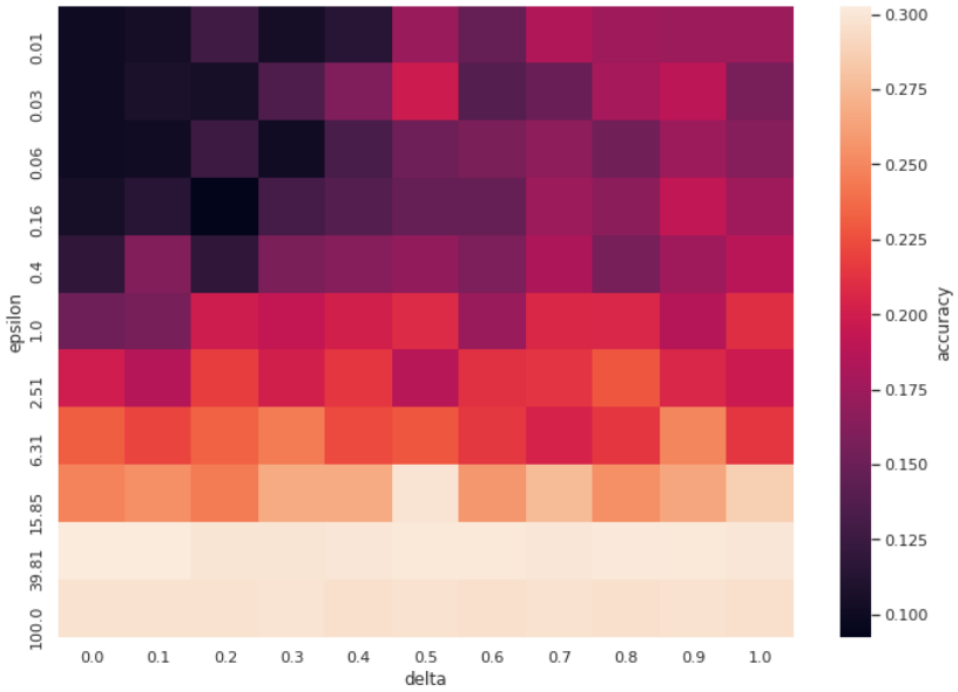
- *Gaussian naive bayes*: adds noise to satisfy differential privacy to the learned means and variances
- *Logistic regression*: adds a Laplace-distributed random vector to the loss function

[1] https://desfontain.es/privacy/privacy-loss-random-variable.html

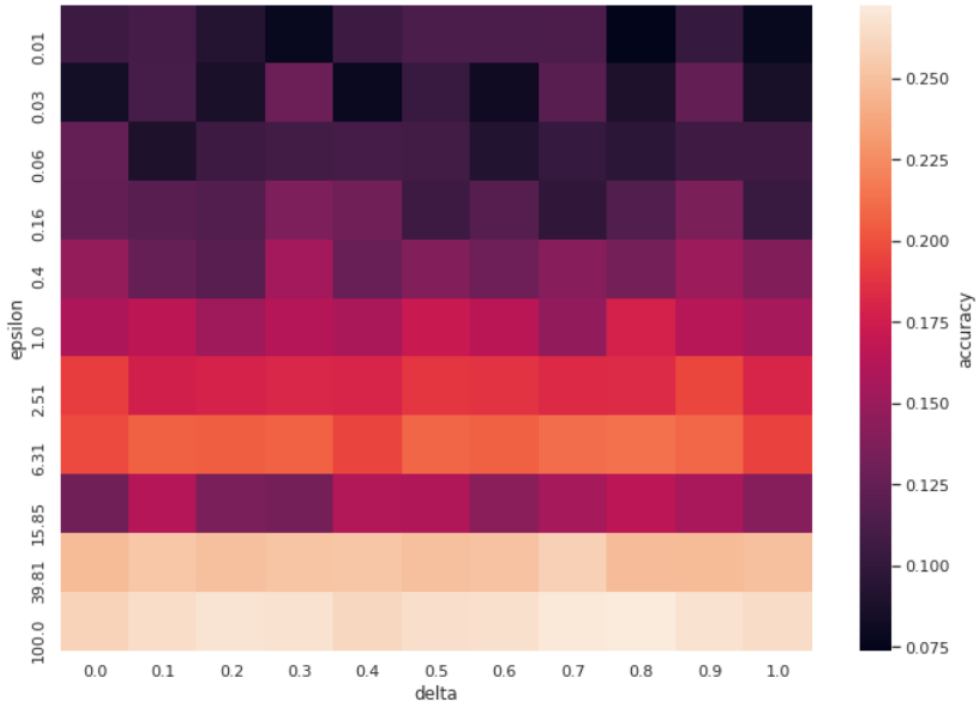
As we can see, random noise can be added to both the data itself and their "attributes". For example, to variances or mathematical expectations.

Also, we already know that epsilon and delta are the most important parameters in (ε, δ) - differential privacy. So, we want to see how the quality of our predictions changes according to different epsilon and delta parameters.

A) Gaussian naive bayes:



B) Logistic regression:



The graphs show 2 patterns:

1. Model quality is dependent of δ:
    - At different epsilon values, the quality of the models may differ at different scales when changing delta. For example, the accuracy of a Logistic regression model changes from ~0.075 to ~0.125 if epsilon = 0.01 and if delta is changed.
    - The scales to which the accuracy values change also depend on the model. For example, the accuracy of Gaussian naïve bayes model with the same delta and epsilon values above changes from ~0.1 to ~0.225, that is, twice as much, when the results of Logistic model are ~1.6 times

2. The lower the epsilon value, the higher privacy guarantee and therefore less model quality, as we add more random noise.

Also, we can notice that the quality of the models does not vary much, no matter what exactly we add to the noise.

Deep learning

Renyi differential privacy is a natural relaxation of the standard notion of differential privacy that preserves many of its essential properties. It can most directly be compared with (epsilon, delta)-differential privacy, with which it shares several important characteristics.

Mechanism	**Differential Privacy**	**Rényi Differential Privacy for α**
Randomized Response	$\left\|\log \frac{p}{1-p}\right\|$	$\alpha>1$: $\frac{1}{\alpha-1}\log\left(p^\alpha(1-p)^{1-\alpha}+(1-p)^\alpha p^{1-\alpha}\right)$ $\alpha=1$: $(2p-1)\log\frac{p}{1-p}$
Laplace Mechanism	$1/\lambda$	$\alpha>1$: $\frac{1}{\alpha-1}\log\left\{\frac{\alpha}{2\alpha-1}\exp(\frac{\alpha-1}{\lambda})+\frac{\alpha-1}{2\alpha-1}\exp(-\frac{\alpha}{\lambda})\right\}$ $\alpha=1$: $1/\lambda+\exp(-1/\lambda)-1=.5/\lambda^2+O(1/\lambda^3)$
Gaussian Mechanism	∞	$\alpha/(2\sigma^2)$

TABLE II
SUMMARY OF RDP PARAMETERS FOR BASIC MECHANISMS.

- **Probabilistic privacy guarantee**

The standard "bad outcomes" guarantee of epsilon-differential privacy is independent of the probability of a bad outcome: it may increase only by a factor of exp(epsilon). Renyi differential privacy even with very weak parameters never allows a total breach of privacy with no residual uncertainty

- **Baseline-dependent guarantees**
  The Renyi differential privacy bound gets weaker for less likely outcomes. Contrasted with the pure epsilon-differential privacy this type of guarantee is conceptually weaker and more onerous in application: in order to decide whether the increased risk is tolerable, one is required to estimate the baseline risk first.

Principles of differential privacy are also used in deep learning algorithms. In laymen terms, differential privacy is all about injecting **noise** (or **"randomness"**) into machine learning system. There's a number of ways you could do it:

- Perturb user's input into a common training pool (eg when a user sends data to a server x% is replaced with random numbers)
- Perturb the underlying data
- Perturb the parameters of the model (eg inject noise into the parameter update process)
- Perturb the loss function (similar to eg L2 regularization)
- Perturb output during prediction (eg to x% of users your API sends random noise)

The challenge with differential privacy is that it's not free. The more protection you want, the more "privacy budget" you allocate, the more your model's performance suffers.

One of the ideas that is popular when talking about applying differential privacy to neural networks is adding noise when calculating gradients. In the paper by Abadi[2], authors introduce modification of classical stochastic gradient descent algorithm to make it DP.

Algorithm 1 Differentially private SGD (Outline)

Input: Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate η_t, noise scale σ, group size L, gradient norm bound C.

Initialize θ_0 randomly

for $t \in [T]$ **do**

Take a random sample L_t with sampling probability L/N

Compute gradient

For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$

Clip gradient

$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$

Add noise

$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$

Descent

$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$

Output θ_T and compute the overall privacy cost (ε, δ) using a privacy accounting method.

Figure 3. Differentially private stochastic gradient descent

As it can be seen, firstly, the gradients are clipped to avoid such a thing called 'gradient explosion'. Basically, clipping makes gradient step smoother which helps to find the global minimum of the loss function. After clipping gradients Gaussian noise is added with mean equal to zero and variation equal to product of noise scale and gradient norm bound. Authors state that such algorithm can achieve differential privacy of the neural network which implies that an

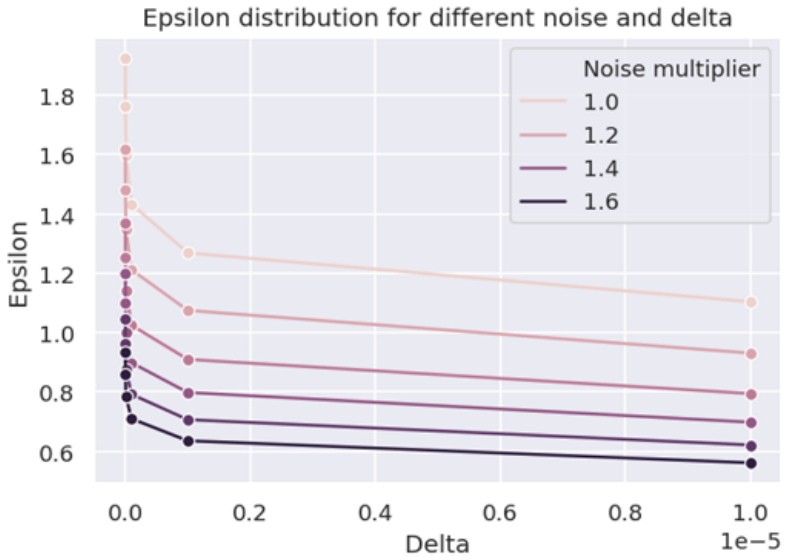
[2] https://arxiv.org/abs/1607.00133

attacker would not be able to recognize whether an exact training sample was included in the training set or not.

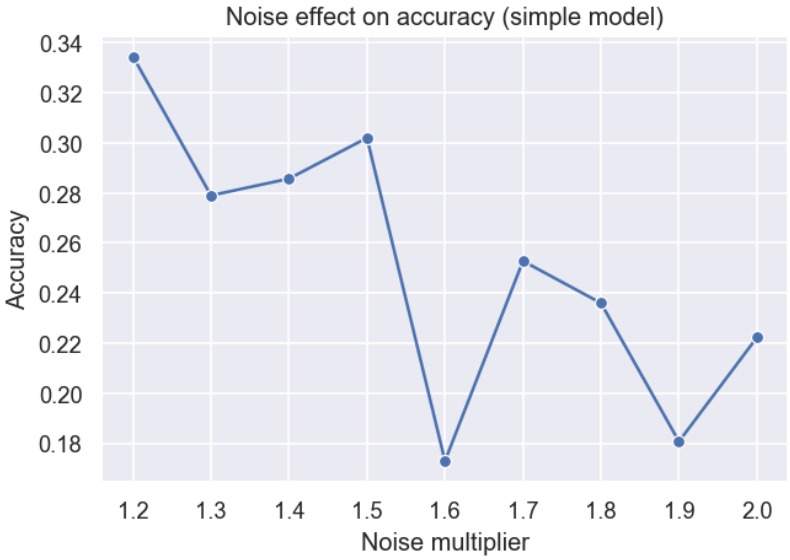
We decided to research how the noise scale is affecting different structures of neural networks. We will run this study on the CIFAR-10 dataset with 3 various deep learning models such as: simple neural network with couple of convolutional layers, ResNet50 and MobileNet.

Since we are bound by time and resources, we decided to run 15 epochs for each architecture with batch size equal to 80. We changed parameters of the noise scale and gradient norm bound to find its effect on accuracy of the models.

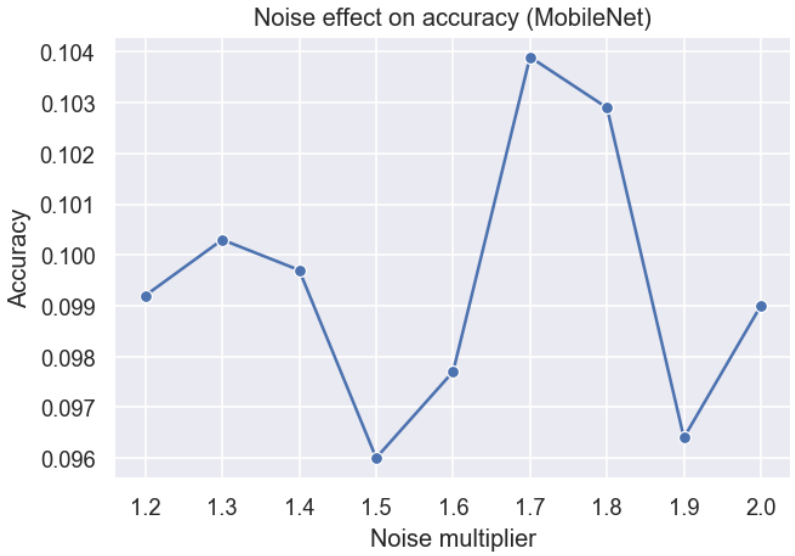
On the graph below you can see epsilon values for different noise scale and delta. The Delta limits the likelihood that the privacy guarantee will not be respected. General idea is to set it to less than the inverse of the size of the training dataset.



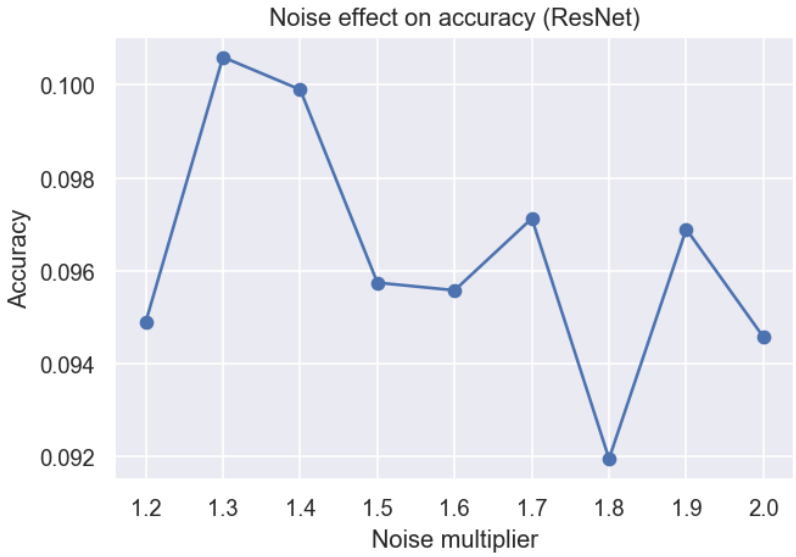
The graph illustrates that there is a significant deterioration in the model's ability to recognize different classes with higher noise scale, accuracy drops by approximately 50% for simple neural network with a couple of convolutional layers.



Regarding the MobileNet, it seems that there is no direct fall in the accuracy. This may be due to the reason that it takes a lot of layers to backpropagate, and the effect of applying noise becomes negligible.



The same situation is acquired with the ResNet. There is no clear evidence that noise scale impacts accuracy in a bad manner. Even though there is a slight drop in the quality of neural network to classify images correctly, we can't make any conclusions.



Summary

As it is noted, the classical methods yield less generalizable models. It is obvious from the experiments that making algorithms differentially private does not guarantee any significant drops in the accuracy.

It can be clearly seen that for deep learning models that quality of the model's prediction is not going to necessarily suffer. Moreover, Deep learning models may get a performance boost from DP, most likely due to buffering the effects of overfit.

What can be done in terms of further research? Firstly, the amount of time spent on training of different DL architectures should be increased to imitate the real word application. Usually, it takes around 200 epochs for ResNet and MobileNet to converge. Secondly, it is interesting to study the accuracy of the various classical ML and DL models, if the noise is added to training dataset, gradients and loss function e.t.c.

To sum everything up, a particular task requires its own individual selection of parameters. Deriving an optimal set of DP parameters remains a challenge to solve via grid search.

Sources:

https://desfontain.es/privacy/
https://towardsdatascience.com/understanding-differential-privacy-85ce191e198a
https://medium.com/secure-and-private-ai-writing-challenge/summary-of-deep-learning-with-differential-privacy-d7ffa2033e8f
https://georgianpartners.shinyapps.io/interactive_counting/
https://arxiv.org/pdf/1702.07476.pdf
https://arxiv.org/abs/1607.00133

Link to the google colab notebooks:

https://colab.research.google.com/drive/13ZEBcQCEbUpxpAm0Fw4gWxSqpJ_egIxS#scrollTo=ef56gCUqrdVn
https://colab.research.google.com/drive/12bdeH6tcVMC448ioGp08KoKb2zAoNp5B?usp=sharing