HW 6

Uma Nair

12/19/2024

What is the difference between gradient descent and *stochastic* gradient descent as discussed in class? (*You need not give full details of each algorithm. Instead you can describe what each does and provide the update step for each. Make sure that in providing the update step for each algorithm you emphasize what is different and why.*)

In gradient descent, the algorithm computes the gradient of the loss function with respect to the model parameters using the entire training dataset. This means that for each update step, it uses all data points to calculate the average gradient, which can be computationally expensive, especially with large datasets. In stochastic gradient descent, the algorithm updates the model parameters using the gradient of the loss function computed from a single training example (or a small batch of examples). This introduces more variability in the updates, but it can significantly speed up learning, especially for large datasets.

Consider the FedAve algorithm. In its most compact form we said the update step is $\omega_{t+1} = \omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} \nabla F_k(\omega_t)$. However, we also emphasized a more intuitive, yet equivalent, formulation given by $\omega_{t+1}^k = \omega_t - \eta \nabla F_k(\omega_t)$; $w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$.

Prove that these two formulations are equivalent.
(*Hint: show that if you place ω_{t+1}^k from the first equation (of the second formulation) into the second equation (of the second formulation), this second formulation will reduce to exactly the first formulation.*)

To prove that the two formulations of the FedAve algorithm are equivalent, we will follow the steps provided in the hint and show that substituting one formulation into the other results in the first formulation.

1. (first formulation):

$$\omega_{t+1} = \omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} \nabla F_k(\omega_t)$$

2. (second formulation):

- Update rule for each client k:

$$\omega_{t+1}^{k} = \omega_t - \eta \nabla F_k(\omega_t)$$

- Global update rule:

$$\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^{k}$$

From the second formulation, we know that:

$$\omega_{t+1}^{k} = \omega_t - \eta \nabla F_k(\omega_t)$$

Substitute this expression for ω_{t+1}^{k} into the global update rule:

$$\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^{k}$$

Substituting for ω_{t+1}^{k}:

$$\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \left(\omega_t - \eta \nabla F_k(\omega_t)\right)$$

Now, expand the sum:

$$\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} \nabla F_k(\omega_t)$$

Notice that the first term can be simplified as:

$$\sum_{k=1}^{K} \frac{n_k}{n} \omega_t = \omega_t \sum_{k=1}^{K} \frac{n_k}{n} = \omega_t \cdot 1 = \omega_t$$

since $\sum_{k=1}^{K} n_k = n$, the total number of data points across all clients.

Thus, we have:

$$\omega_{t+1} = \omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} \nabla F_k(\omega_t)$$

This expression for ω_{t+1} is exactly the same as the first (compact) formulation of the FedAve algorithm.

Thus, we have shown that the two formulations are equivalent.

Now give a brief explanation as to why the second formulation is more intuitive. That is, you should be able to explain broadly what this update is doing.

In federated learning, multiple clients (or devices) train machine learning models collaboratively without sharing their raw data. Instead, each client computes updates to its local model, and these updates are aggregated centrally to improve a global model. The second formulation is often considered more intuitive because it clearly separates the local and global steps. Each client performs an update based on its local data and the current global model. This step mirrors traditional gradient descent, where the model parameters are adjusted according to the local gradients. After the local updates are computed, the global model is updated by averaging the local models. The averaging is done in a weighted manner, where clients with more data points (larger n_k) have a bigger influence on the global model. In summary, the more intuitive formulation first computes local updates at each client and then averages them to form the global update.These two formulations are mathematically equivalent, but the second one provides more clarity by explicitly showing the local-to-global structure of the algorithm

Prove that randomized-response differential privacy is ϵ-differentially private.

The randomized-response mechanism is used to protect privacy in surveys by allowing individuals to randomize their responses. The mechanism provides a way for individuals to answer sensitive questions without directly revealing their true feelings or answers.

For a binary question (e.g., "Have you ever cheated on a test in university?" where x□{0,1}, where 0 means "No" and 1 means "Yes"), the randomized-response mechanism works as follows:

The randomized-response mechanism works as follows:

$$y = \begin{cases} x & \text{with probability } p, \\ 1 - x & \text{with probability } 1 - p. \end{cases}$$

This mechanism ensures that the individual provides their true answer x with probability p, and a flipped answer $1 - x$ with probability $1 - p$.

Step 2: Calculate the Probability of Outputting $y = 1$

For dataset D, the probability of the mechanism outputting $y = 1$ is:

$$\Pr[M(D) = 1] = \begin{cases} p & \text{if } x = 1, \\ 1 - p & \text{if } x = 0. \end{cases}$$

Similarly, for dataset D', the probability of outputting $y = 1$ is:

$$\Pr[M(D') = 1] = \begin{cases} p & \text{if } x' = 1, \\ 1 - p & \text{if } x' = 0. \end{cases}$$

Step 3: Compute the Ratio of Probabilities

We now compute the ratio of probabilities for obtaining $y = 1$ in datasets D and D':

$$\frac{\Pr[M(D) = 1]}{\Pr[M(D') = 1]} = \begin{cases} 1 & \text{if } x = x', \\ \dfrac{p}{1 - p} & \text{if } x \neq x'. \end{cases}$$

This ratio is 1 when the individual's answers in D and D' are the same, and $\frac{p}{1-p}$ when they differ.

Step 4: Bound the Ratio

To satisfy ϵ-differential privacy, we need to ensure:

$$\frac{p}{1 - p} \leq e^{\epsilon}.$$

Taking the natural logarithm of both sides:

$$\ln\left(\frac{p}{1 - p}\right) \leq \epsilon.$$

This condition holds if p is chosen appropriately, ensuring the ratio is bounded by e^{ϵ}.

Conclusion

Thus, the randomized-response mechanism satisfies ϵ-differential privacy, since the ratio of probabilities for any two neighboring datasets is bounded by e^{ϵ}.

Define the harm principle. Then, discuss whether the harm principle is *currently* applicable to machine learning models. (*Hint: recall our discussions in the moral philosophy primer as to what grounds agency. You should in effect be arguing whether ML models have achieved agency enough to limit the autonomy of the users of said algorithms.*)

The term "harm principle" was first defined by John Stuart Mill, a famous thinker and pioneer in the branch of ethics called Utilitarianism. The harm principle states that people should generally be free to do whatever they wish unless their actions directly harm others. Harm here is typically understood in terms of physical, emotional, social, economic, or psychological

damage to others. This makes sense in terms of consequentalism, as consequentalists always prioritize the consequences an action garners over anything when assessing whether or not said action is morally just. Machine learning models, at least in their current form in today's society, do not have agency in the same way humans do. Agency typically refers to the capacity for intentional action, decision-making, and understanding of consequences. While ML models can make decisions (classifying an image, predicting outcomes, etc.), they do so based on statistical patterns learned from data rather than through any conscious intent or understanding of what "they" are actually doing. Human agency involves the ability to reason about one's environment, understand the potential consequences of actions, and act autonomy. Machine agency involves pattern recognition, optimization based on training data, and performing tasks according to predefined objectives, but without the capacity for conscious awareness or intentional decision-making.hey are tools created and controlled by humans. Therefore, it is not the machine learning models themselves that are "limited" by the harm principle, but rather the ways in which they are used to influence, manipulate, or harm individuals. The harm principle is applicable to ML models in the sense that it is the use of these models by humans or organizations that can lead to harms (such as bias, discrimination, loss of privacy, or manipulation) that justify limiting their deployment or regulation. In this regard, it's the responsibility of developers, policymakers, and society to ensure that ML systems do not infringe on individuals' rights or autonomy in harmful ways.Thus, while ML models do not have agency to limit autonomy directly, their use and deployment can have substantial consequences that justify ethical scrutiny and regulation under the harm principle.