HW 6

Ruby Ashman

11/19/2024

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.*)

The primary difference between gradient descent and stochastic gradient descent is found in the data range used to compute the gradient. Gradient descent utilizes all data in order to determine the gradient, while stochastic gradient descent utilizes a random subset of the data and therefore decreasing the likelihood of the method to be trapped into calculating local minima instead of global minima. The update rule we constructed for gradient descent allowed us to minimize the function by locating the direction of steepest descent, instead of the direction of steepest ascent, which is typically generated. To do this we followed the negative gradient, as demonstrated by the following update rule : $\theta_{i+1} = \theta_i - \alpha \nabla f(\theta_i, X, Y)$, where we subtracted the second term. As previously explained, stochastic gradient descent utilizes a random subset of the data to escape being trapped by local minima, as demonstrated in the update rule we constructed: $\theta_{i+1} = \theta_i - \alpha \nabla f(\theta_i, X_{i'}, Y_{i'})$, where we added an indexing set.

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.*)

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

The second formulation is more intuitive because it provides an update for each of the K clients, which are averaged to give a global update, instead of a local one.

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

$\frac{P[A(D_1) \epsilon S]}{P[A(D_2) \epsilon S]} <= e^{\epsilon}$ $\frac{P[A(Yes)=Yes]}{P[A(No)=Yes]} = \frac{P[Output=Yes|Input=Yes]}{P[Output=Yes|Input=No]} = \frac{3/4}{1/4} = 3 = e^{ln(3)}$ $\frac{P[A(No)=Yes]}{P[A(Yes)=Yes]} = \frac{1}{3}$
$\frac{P[A(D_1) \epsilon S]}{P[A(D_2) \epsilon S]} <= 3 = e^{ln(3)}$

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 harm principle states that one's personal autonomy can be forgone in a scenario where their autonomy will solely cause harm to another agent. I do not think the harm principle is currently applicable in machine learning models, since a user of said models does not have the power to solely cause harm to another if allowed their personal autonomy, while the developer of these machine learning models does however have the power to hurt the user. While it could be argued by excercising their personal autonomy the user could be harming the developer of the model, I think it is important to note the model itself should not be considered a moral agent, and it would be difficult for the excercision of a single person's autonomy to have a large scale effect on the developer. Additionally, it would be difficult for the user of these algorithms to solely be causing harm by excercising their personal autonomy, when they are likely doing so to attempt to protect themselves from harms- such as data leaks.