HW 6

Ava Klissouras

11/12/2024

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.*)

Gradient descent finds a minimum by proceeding along a curve in the opposite of the direction of steepest ascent (gradient). It accomplishes this through a series of update steps (described below) until the minimum is reached. Stochastic gradient descent is similar, but it performs gradient descent upon a random subset of the data for each update step. This adds increased variability in calculating gradients, to prevent getting "stuck" in local minima.

The update step for regular gradient descent is $\omega_{i+1} = \omega_i - \alpha \nabla f(\omega_i, x, y)$ *where* α *is the learning rate (which can be thought of intuitively as the step size),* x *is the features, and* y *is the target variable.* ω_{i+1} *is the new position on the curve, and* ω_i *the previous. "f" is the function of which we are trying to locate the minimum. By contrast, the update for stochastic gradient descent is* $\omega_{i+1} = \omega_i - \alpha \nabla f(\omega_i, x_I, y_I)$*, where* x_I, y_I *is a random subset of the data.*

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

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.*)

Placing

$$\omega_{t+1}^k$$

from the first equation of the second formulation, into the second equation of the first formulation, gives:

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

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

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

Recall that

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

.

Then our above expression becomes

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

which is equivalent to the "compact form" of the update step.

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 ω_{t+1}^k represents the kth local update, and calculating $\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$ updates the global parameters by taking a weighted average of the local updates, where the weight is given by $\frac{n_k}{n}$.

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

For epsilon > 0, and randomized algorithm A, A is ϵ - differentially private if and only if for all datasets D_1, D_2 differing in exactly one element, and subsets $S \in im(A)$, $\frac{P[A(D_1) \in S]}{P[A(D_2) \in S]} \leq e^{\epsilon}$. Want to show this is the case for randomized response differential privacy.

Consider a scenario in which participants are asked to divulge which of 2 classes they belong to. Call them 1 and 0. The participant is asked to flip a fair coin (but any arbitrary decision-making method with a 50% probability of each outcome could be chosen). If the coin lands on heads, the participant must tell the truth. If it lands on tails and then heads, they must say "1" regardless of the truth, and if it lands on tails and then tails, they must say "0".

Consider the case where S is "1" (meaning the participant's response is "1".)

Namely, we want to show that there exists an ϵ such that $\frac{P[A(1)=1]}{P[A(0)=1]} \leq e^{\epsilon}$, where A(x) = y means the participant's true class is x, and the algorithm has them respond "y".

P[A(1) = 1] is 3/4 because if the participant's true answer is "1", they will respond with "1" 3/4 of the time (Flipping heads with a probability of 1/2 means they tell the truth, and flipping tails then heads with a probability of 1/4 means they reply "1". 1/2 + 1/4 = 3/4).

P[A(0) = 1] is 1/4 because if the participant's true answer is "0," the only way they will reply with 1 is if they flip tails and then heads, which occurs with a probability of 1/4.

So, $\frac{P[A(1)=1]}{P[A(0)=1]} = \frac{3/4}{1/4} = 3$.

Conversely, if we considered $\frac{P[A(0)=1]}{P[A(1)=1]}$, this value would be 1/3. So, both cases are bounded above by 3.

Consider that $3 = e^{ln(3)}$. So, $\frac{P[A(1)=1]}{P[A(0)=1]} = \frac{P[A(D_1) \in S]}{P[A(D_2) \in S]} \leq e^{\ln(3)}$, and so ϵ is ln(3), and our proof is concluded.

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 restricts personal autonomy when exercising said autonomy causes objective harm to another moral agent. For instance, it limits the autonomy of a developer (or, in this case, a machine learning model) dubiously soliciting and exchanging data without informed consent. The Harm Principle has its roots in the writings of J.S. Mill, who was a utilitarian. By principles of consequentialism (which envelops Utilitarianism), it does not matter if the machine learning model intended *to cause harm; the only thing that matters is whether it* did *cause harm. Moreover, Mill's definition of harm is nonconsensual. If one knowingly consents to something that could be construed as a harm, it can no longer considered be harm. Therefore, the harm principle is applicable to machine learning models only when the users of these models (or those whose data is used to train and test the models) are unaware of the negative consequences brought about by the model. For instance, if the model is susceptible to a data extraction or membership inference attack, but those whose personal information constitutes training data were unaware of the potential for these attacks, the model would violate the Harm Principle.*

However, we must also consider an important question: do Machine Learning models possess enough agency to be held accountable for the consequences described above? According to Kant, agency is linked to rationality, which many Machine Learning models arguably possess (as they make decisions rooted in mathematical foundations). However, perhaps we could argue that a Machine Learning model which discriminates arbitrarily does not possess rationality. Moreover, if agency is interpreted as consciousness/possessing humanlike qualities, rather than rationality, not all Machine Learning models can be said to have agency. If they fail the Turing test (which distinguishes computer responses from human responses) and/or the non-Turing test for consciousness, perhaps we can claim they lack rationality and thus agency.

Therefore, whether the harm principle applies to ML models depends on the definition of agency which we subscribe to. If we find that the model is rational and/or conscious, and those who are harmed by the model's existence were not informed of the possibility of harm, then perhaps we can consider the harm principle to be applicable.