HW 6

Tara Hinton

1/21/2024

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 the d-dimensional vector of partial derivatives representing the direction of steepest ascent, and follows in the direction opposition of the gradient until stopping at a critical parameter. All data is used in finding the gradient of descent; however, this means that we sometimes "get stuck" in local minima rather than finding global minima. The update step for GD is this: $\theta_{i+1} = \theta_i - \alpha \nabla f(\theta_i, X, Y)$, where α is the step size.

Stochastic gradient descent increases variability in calculating gradients, helping us find a global rather than local extrema. We use a random subset of the data as opposed to the entire dataset, so SDG converges to a global optimal solution much more reliably. Our SDG update rule is: $\theta_{i+1} = \theta_i - \alpha \nabla f(\theta_i, X_i', Y_i')$.

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.*)

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

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.

This formulation breaks Federated Average into two intuitive parts. The first part shoes us how we use local parameters. Each local client takes one step of SGD using the current local data. The second part shows us how we use these local data to make a global update weighted by the proportion of data corresponding to the Kth client (ie, the server is taking a weighted average of local models to make a global update, thus protecting individual clients' data).

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

If $\epsilon > 0$ and datasets D1 and D2 differ by exactly 1 element, algorithm A is said to be differentially private if the following is true:

P[A(D1)ϵS]/ P[A(D2)ϵS] <= {e}^ϵ; where S is a subset of the output space and D1 and D2 are datasets that differ by exactly one element.

As {e}^ϵ approaches 1, the similarity of the input and the output is closer.

#Testing out

We'll use our randomized response class example, where students flip a coin to determine what their response to queries about whether or not they cheated on an exam. If the student flips a heads, they must flip the coin again, but tell the truth regardless of the outcome of the second flip (HH or HT in the outcome space); if the student flips a tails, they must flip the coin again. If the next flip is a heads, they must say that they cheated regardless of the truth, while flipping a tails means that they must say that they did not cheat regardless of the truth (in the outcome space, these outcomes are represented as TH or TT).

Let's apply this to our epsilon differential privacy, assuming that a given student answered "yes" to our query:

P [output = "yes" given that our input = "yes"]/P[output = "yes" given that out input = "no"] OR (Probability that a student stated they cheated given that they did)/(Probability that a student stated they cheated given that they did not)

So, considering the output space of HH, HT, TH, TT, our probabilites become: (3/4)/*(1/4), which is 3.

So, we get: 3 = {e}^ϵ, which means ϵ is ln(3).

Therefore, Randomized Response Differential Privacy is epsilon differentiable where $\epsilon = \ln(3)$.

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.*)

In this course, we have defined the harm principle to mean that personal autonomy is bounded, and it is so bounded by the point at which exercising autonomy causes objective harm to another moral agent.

Kant grounded his conception of moral agents in the presence of human dignity. In class, we explored the contributions of rationality, self-awareness, and sentience to these characteristics of "human dignity." If operating purely upon the idea that rationality defines a moral agent, we would be obligated to include AI in this definition (and thereby the harm principle) – machine learning can certainly make logical conclusions and statements. Self-awareness is a trickier one, because it excludes many living things such as animals. Sentience as a criteria narrows the scope of rationality, but is similarly difficult to prove in regards to machine learning. We might look to a Non-Turing test to validate claims of sentience, but failure of a Non-Turing test cannot disprove sentience alone – it must be paired with a Turing test to confirm intelligence. If machine learning passes a paired Turing and Non-Turing test, I would argue that it fits the criteria for a moral agent – therefore making the harm principle applicable on one count.

Still, there is a second question of whether or not machine learning exercising personal autonomy means harm to other moral agents. We've defined personal autonomy as the capacity to decide for oneself and pursue a course of action in one's life, sometimes independent of any particular moral content. I argue that machine learning does not currently have full personal autonomy. It must be prompted by a user,

having limited ability to do beyond what it has been asked to do. Machine learning cannot decide its own fate or make creative choices without being prompted. While machine learning does make strategic choices using information, the scope of its reaction is dependent upon human input. However, might this not be comparable to the state of an elderly person, child, or another person without access to autonomy? Is this lack of personal autonomy in machine learning imposed by humans, or a natural result of an absence of sentience and lack of capability? These are good questions to consider, but this failure to demonstrate personal autonomy means that machine learning cannot be made subject to the harm principle just yet. Users of machine learning are still the primary moral agents that should be held accountable for any harm caused.