HW 6

Grace Sun

4/10/2024

1

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 computes the gradient of the loss function using the entire dataset at each iteration to update the parameters. This approach provides a smooth descent towards the minimum, but it can be slow for large datasets. Gradient descent can end up being stuck in local minimums instead of continuing towards an absolute minimum.

The update step for gradient descent is: $\theta_{i+1} = \theta_i - \alpha \nabla f(\theta_i, x, y)$

Stochastic gradient descent updates the parameters using the gradient computed from a subset of randomly selected data at each iteration. SGD is faster for large datasets and is able to find the absolute minimum without stopping in the local minima.

The update step for stochastic gradient descent is: $\theta_{i+1} = \theta_i - \alpha \nabla f(\theta_i, x_i, y_i)$

The difference between the two update steps is the last two values in the parentheses of the gradient. For gradient descent, (x, y) references all of the data, while for stochastic gradient descent, x_i, y_i represents a subset of randomly selected data.

2

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.)

First, substitute ω_{t+1}^k from the first equation into the second equation (both of the second formulation).

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

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

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

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

This is the first formulation and it is thus proven that these two forms are equivalent.

3

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 splits the update step into two easier to understand parts: a local and a global step. The local step is when individual clients updates their local model parameters independently, and then the global step is when all of the updated local models are aggregated together to create a global model that is weighted by the proportion of data contributed by each client. These two steps reflect the name of the algorithm (Federated Average), and the averaging of local steps is reflected in the summation and fraction in the global step. These steps help preserve data privacy and optimizes a global model at the same time by keeping the raw data with each individual client.

4

Explain how the harm principle places a constraint on personal autonomy. 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 personal autonomy should extend up until its use results in the objective harm of another agent. This means that people are autonomous and free to do as they please until their actions will harm others. Since people should not have the autonomy to hurt others, this places a constraint on personal autonomy. Machine learning models have achieved enough agency to limit the autonomy of users of these algorithms, since using these algorithms is putting groups of people in harm's way. While this harm is not as direct or obvious as punching someone, the use of ML models can put people in harms way by perpetuating biases and causing privacy violations. A good example of harm as injustice resulting from the use of a ML algorithm is the COMPAS algorithm, which predicts levels of recidivism in inmates as a tool for a judge to decide whether or not to grant parole. Since this algorithm is more likely to incorrectly assign black individuals as high risk for reoffending than white individuals, black individuals who are not granted parole in a decision heavily influenced by COMPAS are put in harms way by having to remain in prison longer. Privacy violations can put individuals in harms way when their private information is leaked, which can cause negative impacts such as increased insurance rates or even the loss of a job. An example of private information leaking from a ML model is ChatGPT, which reveals full email signatures sometimes while answering another prompt. The aforementioned harms emphasize that the harm principle can be used to constrain users' autonomy when using ML algorithms to prevent harm to others.