# HW 6

Yuewei Wang

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.*)

*In gradient descent, the algorithm computes the gradient of the loss function with respect to the model's parameters across the entire training dataset. This gradient represents the average direction in which the loss function increases most steeply. The model's parameters are then adjusted in the opposite direction of this gradient, aiming to reduce the loss function.* $\theta_{\text{new}} = \theta_{\text{old}} - \eta \cdot \nabla_\theta J(\theta)$ *Stochastic gradient descent updates the model parameters using the gradient of the loss function with respect to the parameters for only one training example at a time. This method can lead to faster iterations, as it does not require a complete pass through the entire dataset to update the parameters.* $\theta_{\text{new}} = \theta_{\text{old}} - \eta \cdot \nabla_\theta J_i(\theta)$ *GD uses all data points to compute the gradient, whereas SGD uses a single data point at each iteration.SGD can be more efficient for large datasets, as it updates parameters continuously and does not require the entire dataset to be in memory.*

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* $\omega_{t+1}^k$ *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.*)

- substitute $\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^k$ to $\omega_{t+1}^k = \omega_t - \eta \nabla F_k(\omega_t)$ $\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \left(\omega_t - \eta \nabla F_k(\omega_t)\right)$ expand the equation we get: $\omega_{t+1} = \sum_{k=1}^{K} \left( \frac{n_k}{n} \omega_t - \frac{n_k}{n} \eta \nabla F_k(\omega_t) \right)$ so $\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_t - \sum_{k=1}^{K} \frac{n_k}{n} \eta \nabla F_k(\omega_t)$ since $\sum_{k=1}^{K} n_k = n$ so

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

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.

*it directly reflects the two-phase approach of federated learning:1. Each client updates its model locally using its data, as shown by* $\omega_{t+1}^{k} = \omega_t - \eta \nabla F_k(\omega_t)$*. This step represents individual learning where each client optimizes based on its dataset. 2. Updated local models are then combined into a global model, indicated by* $\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^{k}$*. This weighted averaging reflects the collective contribution of all clients, with more data-heavy clients having greater influence.*

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 autonomy of ML developers and users is vast, allowing for the creation and implementation of models in many sectors, including healthcare, finance, and law enforcement. However, this autonomy comes with the responsibility to ensure that these models do not harm individuals or groups. For instance, an ML model used in hiring processes must be scrutinized to ensure it doesn't unfairly discriminate against certain demographics. Although ML models do not have self-awareness or intentions like humans, their actions—driven by algorithms and data inputs—can have significant real-world consequences. For example, a biased dataset can lead an ML system to make unfair or harmful decisions, affecting people's lives and opportunities. Thus, even without traditional agency, ML models can be agents of harm in a practical sense. Developers and users, as the human agents behind these models, bear the responsibility to mitigate any potential harm, adhering to the harm principle. This involves careful design, testing, and monitoring of ML systems to identify and address biases, errors, or other issues that could lead to harmful outcomes.*