HW 6

Student Name

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 aims at minimizing θ so the algorithm will move in the direction of greatest descent, utilizing a learning rate to determine how large the algorithm should skip down that specified gradient. This algorithm utilizes current parameter values, and *all data* available to compute the minimum value within that step. Then, the learning rate times the steepest gradient (utilizing current parameters and *all data*) is subtracted from the current θ to calculate the new θ value. This continues as an optimization-like algorithm that finds the minimum θ given all data. However, this algorithm might have issues with efficiency, especially if there is a "valley" of sorts present in the full data, where the algorithm might get "stuck" in determining descent direction.

Stochastic gradient descent also aims at minimizing θ so the algorithm will move in the direction of greatest descent, utilizing a learning rate to determine how large the algorithm should skip down that specified gradient. This algorithm utilizes current parameter values, and *random samples* from all data available to compute the minimum value within that step. Then, the learning rate times the steepest gradient (utilizing current parameters and *random samples* from given data) is subtracted from the current θ to calculate the new θ value. This continues as an optimization-like algorithm that finds the minimum θ given random samples from 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 $\omega_{t+1}^k = \omega_t - \eta \nabla F_k(\omega_t); w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$.

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.*)

Utilizing the second formulation above, we can substitute ω_{t+1}^k from the first equation into the second, as follows:

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

Then, we can distribute $\sum_{k=1}^{K} \frac{n_k}{n}$ to the individual components of $\omega_t - \eta \nabla F_k(\omega_t)$, resulting in the following formulation: $\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_t - \sum_{k=1}^{K} \frac{n_k}{n} \eta \nabla F_k(\omega_t)$. Taking out constants from the first sumnation term,

we can re-write the first part of this new formulation as: $\frac{\omega_t}{n}\sum_{k=1}^{K} n_k$ and observe $\frac{1}{n}\sum_{k=1}^{K} n_k$ should be 1, leaving us with: $\omega_{t+1} = \omega_t - \eta\sum_{k=1}^{K}\frac{n_k}{n}\nabla F_k(\omega_t)$, or the first formulation specified above, as desired.

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 implements a formula mirroring the stochastic gradient descent update. The first part of this formulation is as follows: $\omega_t - \eta\nabla F_k(\omega_t)$, which is similar to stochastic gradient descent: $\theta_i - \alpha\nabla F_k(\theta_i, x_{i'}, y_{i'})$. Then, the `FedAve` algorithm can be read as a two-part algorithm; the first following an update step that mirrors stochastic gradient descent, except this finds the *local* minimum ω_{t+1}^k through optimization. The second part weighs these local minimums across the data; multiplying them by the average and adding results. This step is important to hide the data from potential attacks; it serves to ensure one data point does not affect the outcome of the model by a large margin. Overall, the above formulation provides an easier representation of how Federated Learning works, especially in the context of the `FedAve` algorithm; these two steps optimize the objective function in terms of empirical risk and find global weights through local iterations.

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, as defined by John Stuart Mill, says autonomy should extend up until its use results in the objective harm of another agent. This definition includes a restriction on the personal autonomy of an agent, or the capacity to decide for oneself a course of action and pursue it (sometimes independent of a moral character). Personal autonomy as incorporated by the use of *autonomy* in the harm principle can be restricted if there is a clear potential to harm another agent (either indirectly or directly). Then, the harm principle cannot be considered currently applicable to machine learning models, as the algorithms themselves have not achieved "agency". This is given by the fact they do not have the capacity to act in accordance with their own volition. Machine learning algorithms typically rely heavily on training data (supplied by the developer), and the model acts on the information provided by that data rather than their "own volition". Therefore, machine learning models have not independently achieved agency and the harm principle cannot be *directly* applied (either for or against these models). However, one can argue for the indirect application of the harm principle against machine learning models by applying it to check developer autonomy. One example of this type of accountability is the fact that lucrative practices are not sufficient justification in data acquisition; if the developers utilize data in unjust ways, they could potentially pose harm to other agents and are therefore violating the harm principle. Overall, the harm principle cannot be directly applied to current machine learning models, but it can be indirectly applied through their developers to fill the requirement of agency.