# HW 6

### Sofia Zhang

### 11/19/2024

# 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 cost function using the entire dataset, resulting in precise but computationally expensive updates $\theta_{i+1} = \theta_i - \alpha\nabla f(\theta_i, X, Y)$. To avoid getting stuck in local extremes rather than the global extreme, stochastic grardient descent can be used. It computes a vector of a random subset of the data in each subset, thus converge to a global extreme. Its formula was represented by $\theta_{i+1} = \theta_i - \alpha\nabla f(\theta_i, X_i', Y_i')$.

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

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

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

Recognize that the sum of $\frac{n_k}{n}$ over all clients equals 1: $\sum_{k=1}^{K}\frac{n_k}{n} = 1/n\sum_{k=1}^{K}n_k = 1/n * n = 1$
$w_t\sum_{k=1}^{K}\frac{n_k}{n} = w_t * 1 = w_t\ (\omega_t)(\sum_{k=1}^{K}\frac{n_k}{n}) - \eta\sum_{k=1}^{K}(\frac{n_k}{n})\nabla F_k(\omega_t) = (\omega_t) - \eta\sum_{k=1}^{K}(\frac{n_k}{n})\nabla F_k(\omega_t) = \omega_{t+1}$

# 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 uses operational steps taken in federated learning: each client independently computes a local update using its own subset(`\omega_{t+1} = \omega_t - \eta \sum_{k=1}^K{\frac{n_k}{n}\nabla F_k(\omega_t)}}`), and then these locally updated models are aggregated by the server to form the new global model($\omega_{t+1} = \sum_{k=1}^{K}\frac{n_k}{n}\omega_{t+1}^k$). / # Prove that randomized-response differential privacy is $\epsilon$-differentially private.

The randomized response mechanism is $\epsilon$-differentially private because, with the appropriate choice of p, it ensures that the probability of any output does not differ by more than a factor of $e^{\epsilon}$ between any two possible inputs. Consider the randomized response mechanism, where the input D and the output S are both binary variables, taking values in {0, 1}. Assume the true input is D = 1. The randomized response mechanism works as follows:

- With probability p, report the true value D.
- With probability 1-p, report the opposite value.

We need to show that the randomized response mechanism is $\epsilon$-differentially private for an appropriate choice of p.

Consider the case S = 1 :

$\frac{P[A(1)=1]}{P[A(0)=1]} = \frac{P[Output=1|Input=1]}{P[Output=1|Input=0]} = \frac{p}{1-p}$

To achieve $\epsilon$-differential privacy, we require: $\frac{P[A(1)=1]}{P[A(0)=1]} \leq e^{\epsilon}$

Thus: $\frac{p}{1-p} \leq e^{\epsilon}$

Similarly, Consider the case S = 0 :

$\frac{P[A(1)=0]}{P[A(0)=0]} = \frac{P[Output=0|Input=0]}{P[Output=0|Input=1]} = \frac{p}{1-p}$

Thus: $\frac{p}{1-p} \leq e^{\epsilon}$

Again, for $\epsilon$-differential privacy: $\frac{P[A(1)=0]}{P[A(0)=0} \leq e^{\epsilon}$

Thus, for the randomized response mechanism to satisfy $\epsilon$-differential privacy, we need:

This implies that the ratio of probabilities between any two possible outputs (whether S = 0 or S = 1) is bounded by $e^{\epsilon}$, ensuring differential privacy.

# 4

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 states that individuals should be free to act however they wish unless their actions cause harm to others. It provides a moral basis for limiting individual liberty only when those actions pose a potential harm to others.Machine learning models can inadvertently cause harm through biased predictions, unfair discrimination, privacy breaches, or perpetuation of stereotypes. For instance, an algorithm might discriminate against certain racial or gender groups due to biased training data, directly impacting individuals' opportunities for employment or access to financial services. However, these models do not possess true agency in the philosophical sense—they lack intent, understanding, and independent decision-making. Their "actions" are fundamentally derived from statistical correlations and optimizations defined by the data they were trained on and the objectives programmed by humans.Despite this lack of true agency, machine learning models do have significant practical influence that can limit the autonomy of individuals. When users defer to the outputs of these algorithms, they might unknowingly be subjected to biases present in the data or make harmful decisions. Developers and institutions should be careful when deploying these models, and putting in place safeguards to protect individual autonomy and minimize harm.*