HW 6

Colby Reichenbach

04/09/2024

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 (GD):

Update Step: $\theta = \theta - \eta \cdot \nabla_\theta J(\theta)$

Why? The gradient is computed using the dataset in its entirety. The parameters are updated in such a way that, on average, reduces the loss across all of the data points.

Stochastic Gradient Descent (SGD):

Update Step: $\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i)}, y^{(i)})$

Why? The gradient is computed for each data point, or for a small batch of data points. The parameters are updated to reduce the loss for those specific points.

Difference: In terms of the equation we see in SGD $J(\theta; x^{(i)}, y^{(i)})$, this shows compartmentalization of the dataset for computation, and in GD we don't see the dataset being broken up, meaning it takes the whole set in. This overall leads to GD being a slower, yet more stable computation. While SGD is a faster, but more variable computation.

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.*)

Substitution:

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

Distribute:

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

Simplify: *Note $\sum_{k=1}^{K} \frac{n_k}{n} = 1$

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

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.

The FedAve algorithm works in a way that model updates are computed locally from local data before being averaged and aggregated centrally. This second formulation breaks is then more intuitive because it breaks the formulation into two, the local computation, and the aggregate. The local update is the first equation, and that is what is substituted into the aggregate, which is the second equation. This makes sense because it mimics how FedAve works, that the local equations are updated then substituted into the aggregate in which those individually local updates are combined to update the central model.

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 an individual is free to act as they wish as long as their actions do not harm others. In order for this to be applicable to an entity, that entity must have a grounded agency, that is they must have an ability to carry out an action with an understanding and interpretation of its environment. Machine models lack understanding, intent and ethical reasoning in their decision making. Meaning they do not possess grounded agency. As a result, the harm principle is not directly applicable to machine learning models, as the models are not making autonomous decisions, but rather follow patterns from their training.

Despite the harm principle not being applicable to the machine learning model itself, the harm principle is applicable to the people behind the designs and development of the model. Models are often developed with an ethical guideline or framework that might limit user autonomy. For example, a model used for personalized advertising might violate a user's personal data wihtout consent. This can lead to harm through intrusive collection of data without proper consent, violating an individuals privacy.

In all, the harm principle is not directly applicable to machine learning models due to their lack of grounded agency. Consequently, the harm principle cannot be directly applied to these models, as they do not make decisions autonomously but are guided by their programming and training data. However, this principle remains relevant when considering the human input behind these models. Those who develop and design the models are responsible of ensuring these models do not cause harm, and adhere to ethical guidelines.