Loss Functions and Optimizers

Loss Functions

The loss function in a neural network quantifies the difference between the expected outcome and the outcome produced by the model, i.e., the predicted outcome.

For regression problems, we usually use Mean Squared Error as the loss function:

$$L(data, \theta) = \frac{1}{n}\sum_{i=1}^{n}\left(y^i - F\left(x^i; \theta\right)\right)^2$$

Where,

$L(data,\ \theta)$ is the loss function, y^i is the expected outcome, and $F\left(x^i,\ \theta\right)$ is the predicted outcome.

For binary classification problems, we can use the Cross-Entropy loss function:

$$L(data,\ \theta) = -\frac{1}{n}\left[\sum_{i,\ y^i=1}\log F\left(x^i;\ \theta\right) + \sum_{j,\ y^j=0}\log\left(1 - F\left(x^j;\ \theta\right)\right)\right]$$

We prefer a model which makes the least error, hence to get a good model, we have to minimize the loss function. But how do we minimize the loss function? The answer is **Optimizers**.

What are Optimizers?

Optimizers are algorithms or methods used to change the parameters of the neural network such as weights and learning rate to reduce the loss.

Gradient Descent

Gradient Descent is an optimization algorithm that is used for updating the parameters namely, weights and biases in neural networks such that the cost function is minimized. The mathematical representation of gradient descent can be given as:

$$w_{jk}^{t+1} \longleftarrow w_{jk}^{t} - \eta\,\frac{\partial}{\partial w_{jk}}\,L\left(x^i,\ y^i,\ \theta\right)$$

Where,

$$w_{jk}^{t+1} = New\,Weight$$

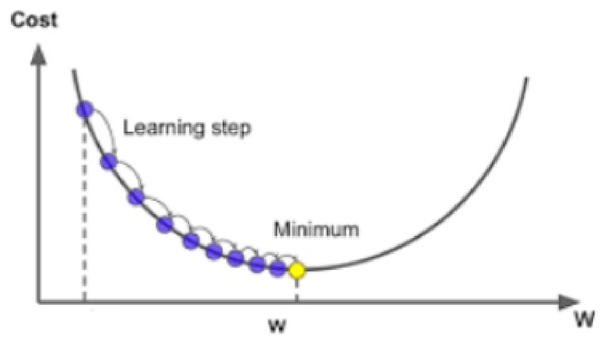
$$w_{jk}^{t} = Old\,Weight$$

$$\eta = Learning\ Rate$$

$$L\left(x^i,\ y^i,\ \theta\right) = Loss\ Function$$

Recall, in the previous pre-reads, we saw that the gradient/derivative of a function f(x) represents the slope/rate of change w.r.t x.

Here, we are calculating the gradient/derivative of loss function w.r.t the weights which is the rate of change in loss with respect to weight. We multiply the derivative by the learning rate to move in the curve.



Gradient descent sums the error for each point in a training set and **updates the weights only after all training examples have been evaluated**. This process is referred to as a **training epoch**. We repeat this process of calculating loss and updating weights until the loss function is minimized.

Stochastic Gradient Descent

Stochastic Gradient Descent(SGD) works the same as gradient descent. The only difference between Stochastic Gradient Descent and Gradient Descent is that in Gradient Descent we updated the weights after evaluating all training examples but in Stochastic Gradient Descent, we **update the weights after evaluating the error of each training data point**.

Batch Stochastic Gradient Descent

This algorithm also works in a similar way as the previous two algorithms. The only difference between Batch Stochastic Gradient Descent and Stochastic Gradient Descent is that in Stochastic Gradient Descent we update the weights after evaluating each training data but in Batch Stochastic Gradient Descent, we **evaluate a subset of data points** before updating the weights.

Previous

Next