Batch Gradient Descent

In Batch Gradient Descent, all the training data is taken into consideration to take a single step. We take the average of the gradients of all the training examples and then use that mean gradient to update our parameters-(weights, bias) : one epoch.

- Computes gradient using the whole Training sample
- Not suggested for huge training samples.
- Deterministic in nature.
- Gives optimal solution given sufficient time to converge.

Suppose our dataset has 5 million examples, then just to take one step the model will have to calculate the gradients of all the 5 million examples. This does not seem an efficient way.
To tackle this problem we have Stochastic Gradient Descent

Stochastic Gradient Descent

In Stochastic Gradient Descent (SGD), we consider just one example at a time to take a single step. We do the following steps in one epoch for SGD:

- Take an example
- Feed it to Neural Network
- Calculate it's gradient
- Use the gradient we calculated in step 3 to update the weights
- Repeat steps 1–4 for all the examples in training dataset

Mini-batch Gradient Descent

In this, Neither we use all the dataset all at once nor we use the single example at a time. We use a batch of a fixed number of training examples which is less than the actual dataset and call it a mini-batch. Doing this helps us achieve the advantages of both the former variants we saw. So, after creating the mini-batches of fixed size, we do the following steps in one epoch:

- Pick a mini-batch
- Feed it to Neural Network
- Calculate the mean gradient of the mini-batch
- Use the mean gradient we calculated in step 3 to update the weights
- Repeat steps 1–4 for the mini-batches we created

Batch Gradient Descent	Stochastic Gradient Descent (SGD)
Use **all** training samples for one forward pass and then adjust weights	Use **one** (randomly picked) sample for a forward pass and then adjust weights
Good for small training set	Good when training set is very big and we don't want too much computation