Gradient descent

Types of gradient descent

- There are three types of gradient descent learning algorithms:
  - batch gradient descent,
  - stochastic gradient descent
  - mini-batch gradient descent.

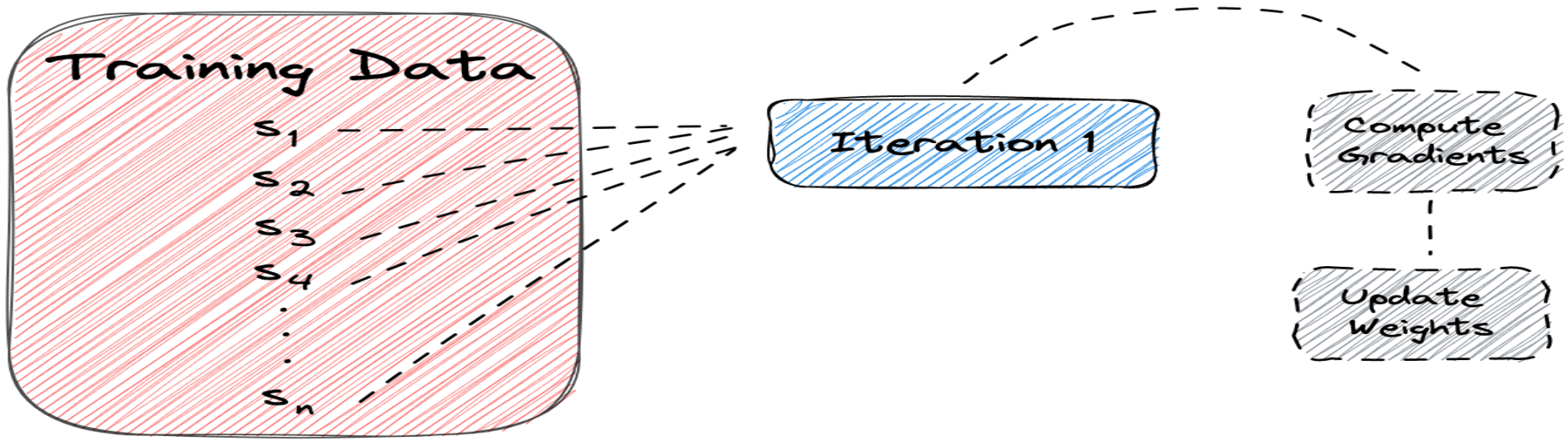
Batch gradient descent

Batch gradient descent sums the error for each point in a training set, updating the model only after all training examples have been evaluated. This process is referred to as a training epoch.

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$$J(\theta) \quad : -\frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i) x_i$$

Batch gradient descent



While this batching provides computation efficiency, it can still have a long processing time for large training datasets as it still needs to store all of the data in memory.

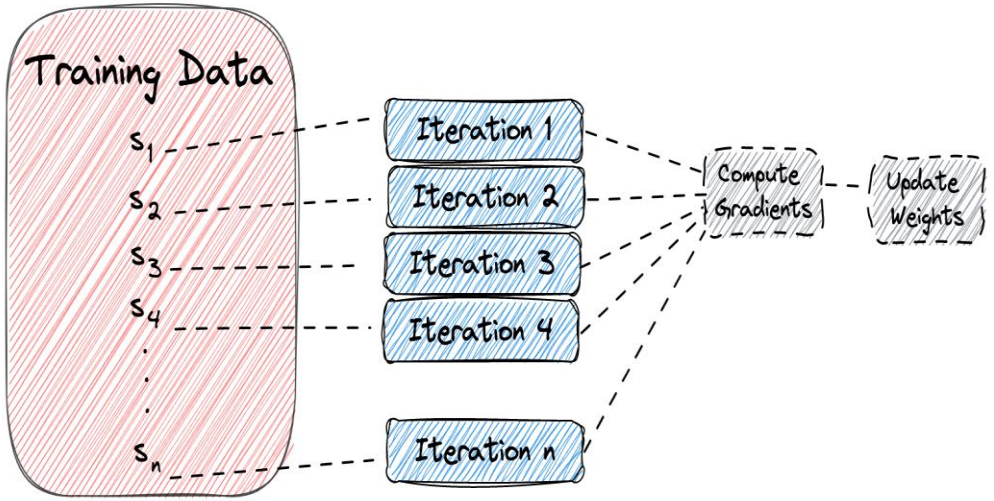
Batch gradient descent also usually produces a stable error gradient and convergence, but sometimes that convergence point isn't the most ideal, finding the local minimum versus the global one.

Stochastic gradient descent

- Stochastic gradient descent (SGD) runs a training epoch for each example within the dataset and it updates each training example's parameters one at a time.

$$for\ i\ in\ range\ (m):$$

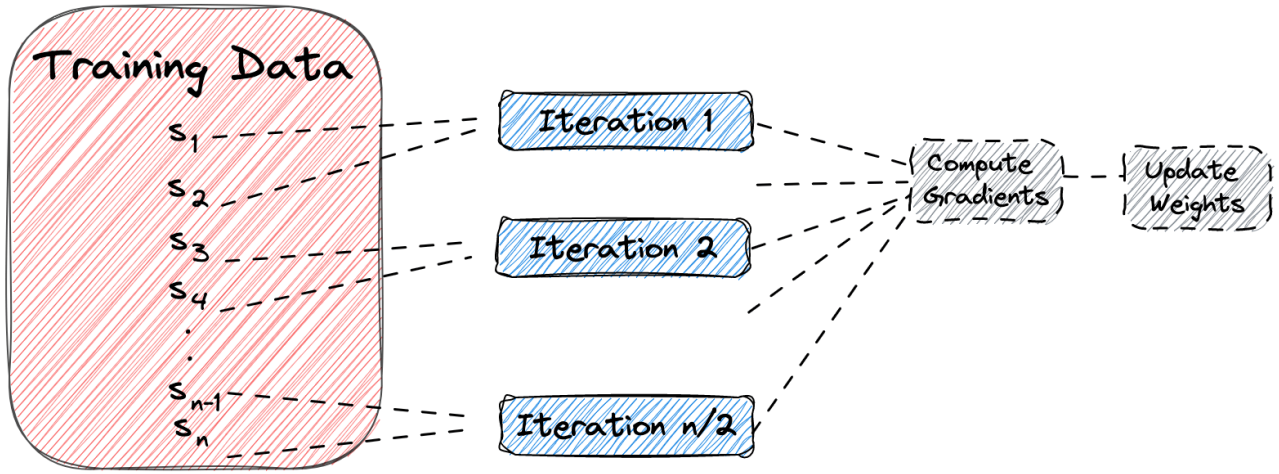
$$\theta_j = \theta_j - \alpha\ (y_i - \hat{y}_i)x_i$$



- Since you only need to hold one training example, they are easier to store in memory. While these frequent updates can offer more detail and speed, it can result in losses in computational efficiency when compared to batch gradient descent. Its frequent updates can result in noisy gradients, but this can also be helpful in escaping the local minimum and finding the global one.

Mini-batch gradient descent

- SGD in the long run, the method will never reach the global minimum since there will be a lot of fluctuations in the value of the loss function.
- **Mini-batch gradient descent is a combination of the previous methods where we use a group of samples called mini-batch in a single iteration of the training algorithm.** The mini-batch is a fixed number of training examples that is less than the actual dataset. So, in each iteration, we train the network on a different group of samples until all samples of the dataset are used.



simple example

- Let’s assume that we have a dataset with {n=2000} samples, and we want to train a deep learning model using gradient descent for 10 epochs and mini-batch size b=4:

- In batch gradient descent, we’ll update the network’s parameters (using all the data) 10 times which corresponds to 1 time for each epoch.
- In stochastic gradient descent, we’ll update the network’s parameters (using one sample each time) 2000*10=20000 times which corresponds to 2000 times for each epoch.
- In mini-batch gradient descent, we’ll update the network’s parameters (using b=4 samples each time) 2000/4*10=5000 times that corresponds to 2000/4=500 times for each epoch.

Difference between Batch Gradient Descent and Stochastic Gradient Descent

Batch Gradient Descent	Stochastic Gradient Descent
Computes gradient using the whole Training sample	Computes gradient using a single Training sample
Slow and computationally expensive algorithm	Faster and less computationally expensive than Batch GD
Not suggested for huge training samples.	Can be used for large training samples.
Deterministic in nature.	Stochastic in nature.
Gives optimal solution given sufficient time to converge.	Gives good solution but not optimal.
Can’t escape shallow local minima easily.	SGD can escape shallow local minima more easily.
Convergence is slow.	Reaches the convergence much faster.
It updates the model parameters only after processing the entire training set.	It updates the parameters after each individual data point.
The learning rate is fixed and cannot be changed during training.	The learning rate can be adjusted dynamically.
It typically converges to the global minimum for convex loss functions.	It may converge to a local minimum or saddle point.
It may suffer from overfitting if the model is too complex for the dataset.	It can help reduce overfitting by updating the model parameters more frequently.