REDUCING LOSS

Gradient Descent: Given loss function, take gradient and apply step in direction of negative gradient to get new model.

How big a step? Learning rate (step size) is one of the alg's hyperparameters. (knobs that programmers tweak)

Weight initialization: For convex problems, applying Grad. Descent allows us to find the optimum from anywhere. Non-convex problems need to consider initial values carefully.

Efficiency — can skip recomputing gradient over entire dataset:

- Stochastic GD — just one example per step
- Mini-batch GD — subset of 10–1000 examples, grad. averaged over batch.

Iterative approach: Start with $\vec{w} = 0$ (e.g.), apply loss fn, apply GD, update $\vec{w}$. Repeat until model converges: loss fn stops changing (or very slowly).

★ The variables in the loss function are the weights and bias! i.e. w and b. So GD allows us to directly update them.