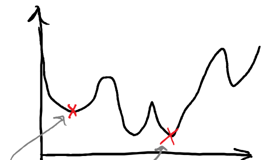
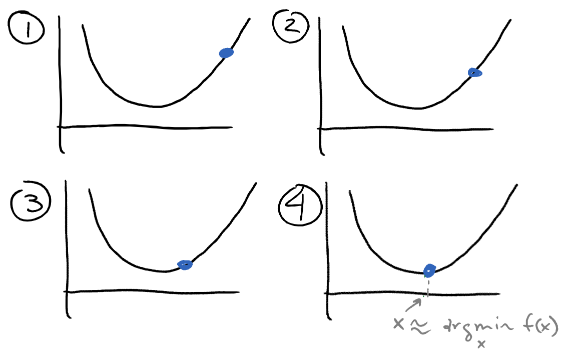
Gradient Descent finds a local minima of non-convex functions.



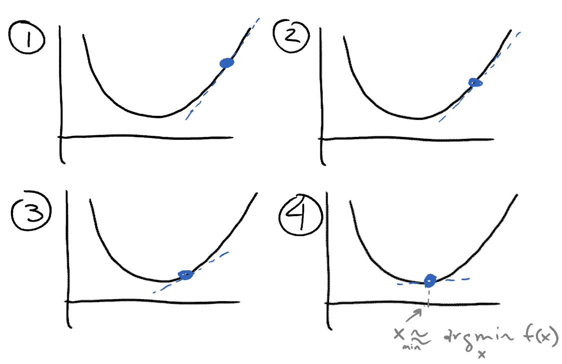
No guarantee which minima will be found...

Understanding training "tricks" can help you find good minima.

Minima is found iteratively:



G.D. moves the "guess" in the direction of the derivative of f(x):



1. $$x'_{min} = x_{min} - \alpha \left. \frac{df(x)}{dx} \right|_{x_{min}}$$

"Derivative of f(x) with respect to x, evaluated at some specific variable x_{min}."

2. $$x''_{min} = x'_{min} - \alpha \left. \frac{df(x)}{dx} \right|_{x'_{min}}$$

3. $$x'''_{min} = x''_{min} - \alpha \frac{df(x)}{dx} \boxed{\Big|_{x''_{min}}}$$

This part is often omitted. You have to assume it is implied.

stepsize ("hyperparameter")

Example

$f(x) = x^2 + 10$

$\frac{df(x)}{dx} = 2x$

initial guess: $x_{min} = 5$

$$x'_{min} = x_{min} - \alpha \left. \frac{df(x)}{dx} \right|_{x_{min}}$$

$$x'_{min} = 5 - .1 * 2x \Big|_5 = 5 - .1 \times 10$$

my choice, but will impact results.

$$= 5 - 1 = 4$$

$$x''_{min} = 4 - .1 * 2x \Big|_4$$

$= 4 - .8 = 3.2$

$\vdots$

With classification and regression we typically have very large models that we must differentiate (if we are using g.d.)

Ex:

$$f_\theta(\mathbf{x}) = x_1\theta_1 + x_2\theta_2 + \dots x_m\theta_m$$

This is a multivariate linear model which is parameterized by θ.

> **Note**
>
> Now we treat X as something we cannot control. It is like a constant. In the AI context x is observed from the environment. But we can control θ. We will eventually find θ that best maps $f_\theta(x) \to y$.

Now can we apply g.d. directly to $f_\theta(x)$? No. This will find parameters θ which give us the most-negative output.

> Toy example: $f(x) = 5x$, $\frac{df(x)}{dx} = 5$
>
> $$x'_{min} = x_{min} - \alpha 5$$
>
> $$\vdots$$
>
> $$x'_{min} \to -\infty$$

We <u>always</u> apply G.D. to a loss function. For regression a common loss is the Mean Squared Error (MSE). The loss between $\hat{y} = f_\theta(x)$, which is our model prediction and the true value is written as:

$$\underbrace{L(\hat{y}, y)}_{\hat{y}:\ \text{vector of predictions},\ y:\ \text{vector of labels}} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} (\hat{y}^{(i)} - y^{(i)})^2$$

where

$\hat{y}^{(i)} = f_\theta(x^{(i)})$ is the prediction

$x^{(i)}$ is one of the training set's feature vectors.

$y^{(i)}$ is the correct label matching $x^{(i)}$

N is the number of feature vectors and labels in the training set.

Note that the loss function takes as input our model. It is a function of a function

$$L(\hat{y}, y) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} (\hat{y}^{(i)} - y^{(i)})^2$$

We only care about optimizing our loss function. Will use g.d. for that

∇_θ = gradient

∇ is vector of partial derivatives

Ex:

$$f_\theta(x) = x_1\theta_1 + x_2\theta_2 + x_3\theta_3$$

$$\boxed{\nabla_\theta f_\theta(x) = \begin{bmatrix} \frac{\partial f_\theta(x)}{\partial \theta_1} \\ \frac{\partial f_\theta(x)}{\partial \theta_2} \\ \frac{\partial f_\theta(x)}{\partial \theta_3} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}$$

Optimizing a large model requires taking gradient of loss

$$\theta' = \theta - \alpha \nabla_\theta L(\hat{y}, y)\Big|_{x,y}$$

$$\Downarrow$$

$$\theta_1' = \theta_1 - \alpha \frac{\partial L(\hat{y}, y)}{\partial \theta_1}$$

Dropping this notation, because it is implied.

$$\theta_2' = \theta_2 - \alpha \frac{\partial L(\hat{y}, y)}{\partial \theta_2}$$

$$\vdots$$

$$\theta_m' = \theta_m - \alpha \frac{\partial L(\hat{y}, y)}{\partial \theta_m}$$

Example

If $\hat{y}^{(i)} = f_\theta(x^{(i)}) = \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \theta_3 x_3^{(i)}$

And $L(\hat{y}, y) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} \left(\hat{y}^{(i)} - y^{(i)}\right)^2$

$$\nabla_\theta L(\hat{y}, y) = \nabla_\theta \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} \left(\hat{y}^{(i)} - y^{(i)}\right)^2$$

$$= \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \frac{1}{2} \left(\hat{y}^{(i)} - y^{(i)}\right)^2$$

$$= \frac{1}{N} \sum_{i=1}^{N} \cancel{\frac{1}{2}}\, \cancel{2} \left(\hat{y}^{(i)} - y^{(i)}\right) \nabla_\theta \left[\hat{y}^{(i)} - y^{(i)}\right]$$

$$= \frac{1}{N} \sum_{i=1}^{N} \left(\hat{y}^{(i)} - y^{(i)}\right) \left[\nabla_\theta \hat{y}^{(i)} - \cancel{\nabla_\theta y^{(i)}}\right]$$

$$= \frac{1}{N} \sum_{i=1}^{N} \underbrace{\left(\hat{y}^{(i)} - y^{(i)}\right)}_{\text{scalar}} \underbrace{\nabla_\theta \hat{y}^{(i)}}_{\text{vector}} \Rightarrow \text{sum of vectors}$$

$$\nabla_\theta \hat{y}^{(i)} = \nabla_\theta \sum_{j=1}^{m} \theta_j x_j^{(i)}$$

$$= \begin{bmatrix} \frac{\partial}{\partial \theta_1} \sum_{j=1}^{m} \theta_j x_j^{(i)} \\ \frac{\partial}{\partial \theta_2} \sum_{j=1}^{m} \theta_j x_j^{(i)} \\ \frac{\partial}{\partial \theta_3} \sum_{j=1}^{m} \theta_j x_j^{(i)} \end{bmatrix} = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \\ x_3^{(i)} \end{bmatrix}$$