Guided Tour of Machine Learning in Finance

Week 2 - Lesson 1 - part 5: Gradient Descent Optimization

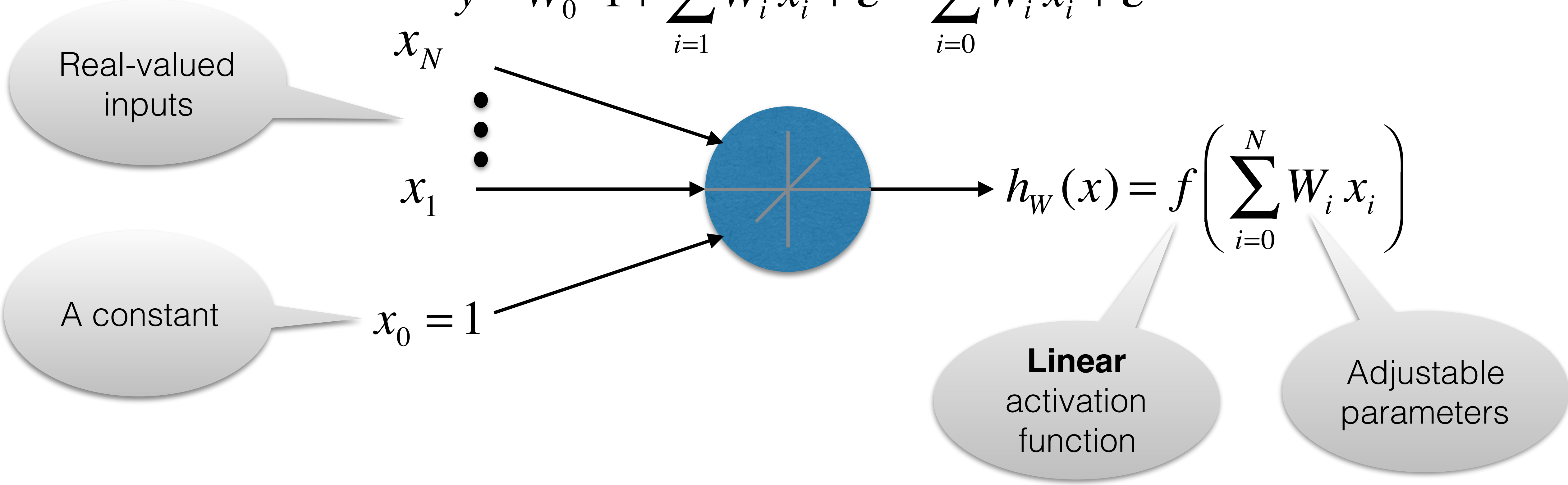
Igor Halperin

NYU Tandon School of Engineering, 2017

Linear Regression as a one-neuron calculation

- **Inputs**: real-valued numbers
- **Output:** a real-valued number
- Can be thought of as a node computing a **linear function** of inputs
- Weights can be tuned to fit the data

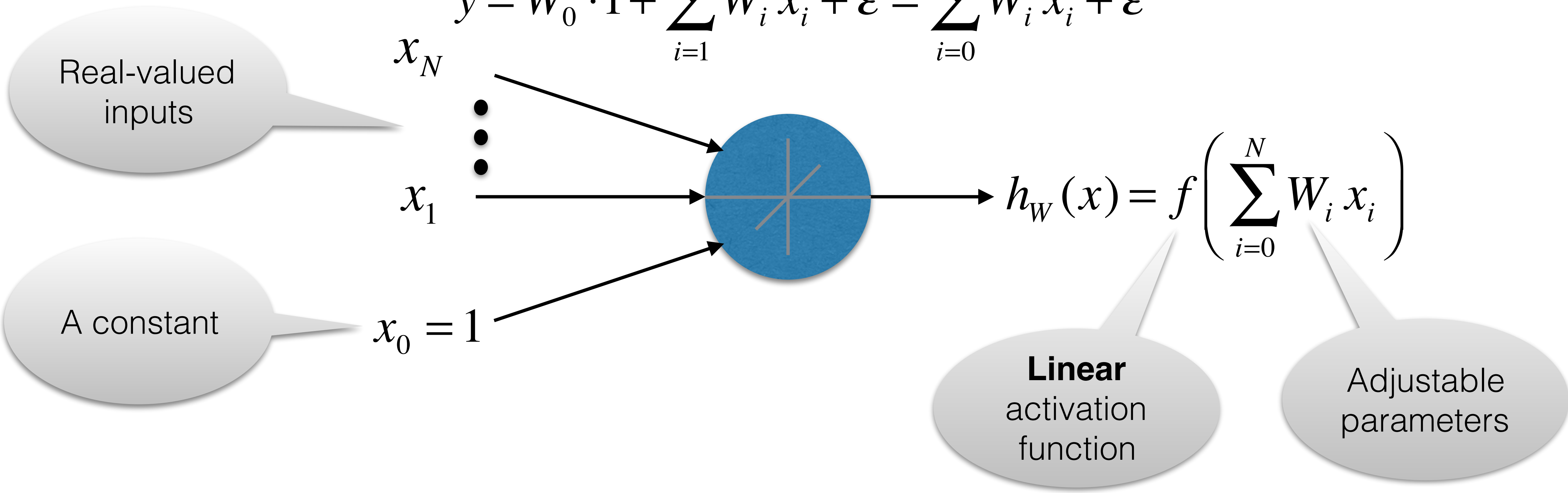
$$y = W_0 \cdot 1 + \sum_{i=1}^{N} W_i x_i + \varepsilon = \sum_{i=0}^{N} W_i x_i + \varepsilon$$



Linear Regression as a one-neuron calculation

- **Inputs**: real-valued numbers
- **Output:** a real-valued number
- Can be thought of as a node computing a **linear function** of inputs
- Weights can be tuned to fit the data

$$y = W_0 \cdot 1 + \sum_{i=1}^{N} W_i x_i + \varepsilon = \sum_{i=0}^{N} W_i x_i + \varepsilon$$



- For Linear Regression, there is an **analytical (one-step) solution** via the normal equation

Analytical learning in Linear Regression

Performance measure P : mean square error (**MSE**) MSE_{test} :

$$\mathrm{MSE}_{test} = \frac{1}{N_{test}} \sum_{n=1}^{N_{test}} \left(\hat{y}_i^{test} - y_i^{test} \right)^2 = \frac{1}{N_{test}} \left\| \hat{\mathbf{Y}}^{test} - \mathbf{Y}^{test} \right\|_2^2 = \frac{1}{N_{test}} \left\| \mathbf{X}^{test}\, \mathbf{W} - \mathbf{Y}^{test} \right\|_2^2$$

Parameters $\mathbf{W} \in \mathbb{R}^D$ are found by minimizing MSE_{train}

$$\mathrm{MSE}_{train} = \frac{1}{N_{train}} \sum_{n=1}^{N_{train}} \left(\hat{y}_i^{train} - y_i^{train} \right)^2 = \frac{1}{N_{train}} \left\| \hat{\mathbf{Y}}^{train} - \mathbf{Y}^{train} \right\|_2^2 = \frac{1}{N_{train}} \left\| \mathbf{X}^{train}\, \mathbf{W} - \mathbf{Y}^{train} \right\|_2^2$$

For optimal weights $\mathbf{W}$, the gradient of MSE_{train} should be 0:

$$\nabla_W \mathrm{MSE}_{train} = \nabla_W \frac{1}{N_{train}} \left\| \hat{\mathbf{Y}}^{train} - \mathbf{Y}^{train} \right\|_2^2 = \frac{1}{N_{train}} \nabla_W \left\| \mathbf{X}^{train}\, \mathbf{W} - \mathbf{Y}^{train} \right\|_2^2 = 0$$

$$\Rightarrow \quad \boxed{\mathbf{W} = \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{y}}$$

Normal equation

Analytical learning in Linear Regression

Performance measure P : mean square error (**MSE**) MSE_{test} :

$$\mathrm{MSE}_{test} = \frac{1}{N_{test}} \sum_{n=1}^{N_{test}} \left(\hat{y}_i^{test} - y_i^{test} \right)^2 = \frac{1}{N_{test}} \left\| \hat{\mathbf{Y}}^{test} - \mathbf{Y}^{test} \right\|_2^2 = \frac{1}{N_{test}} \left\| \mathbf{X}^{test}\, \mathbf{W} - \mathbf{Y}^{test} \right\|_2^2$$

Parameters $\mathbf{W} \in \mathbb{R}^D$ are found by minimizing MSE_{train}

$$\mathrm{MSE}_{train} = \frac{1}{N_{train}} \sum_{n=1}^{N_{train}} \left(\hat{y}_i^{train} - y_i^{train} \right)^2 = \frac{1}{N_{train}} \left\| \hat{\mathbf{Y}}^{train} - \mathbf{Y}^{train} \right\|_2^2 = \frac{1}{N_{train}} \left\| \mathbf{X}^{train}\, \mathbf{W} - \mathbf{Y}^{train} \right\|_2^2$$

For optimal weights $\mathbf{W}$, the gradient of MSE_{train} should be 0:

$$\nabla_W \mathrm{MSE}_{train} = \nabla_W \frac{1}{N_{train}} \left\| \hat{\mathbf{Y}}^{train} - \mathbf{Y}^{train} \right\|_2^2 = \frac{1}{N_{train}} \nabla_W \left\| \mathbf{X}^{train}\, \mathbf{W} - \mathbf{Y}^{train} \right\|_2^2 = 0$$

$$\Rightarrow \boxed{\mathbf{W} = \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{y}}$$

Normal equation

Only works analytically for linear regression

Analytical learning in Linear Regression

Performance measure P : mean square error (**MSE**) MSE_{test} :

$$\mathrm{MSE}_{test} = \frac{1}{N_{test}} \sum_{n=1}^{N_{test}} \left(\hat{y}_i^{test} - y_i^{test} \right)^2 = \frac{1}{N_{test}} \left\| \hat{\mathbf{Y}}^{test} - \mathbf{Y}^{test} \right\|_2^2 = \frac{1}{N_{test}} \left\| \mathbf{X}^{test} \mathbf{W} - \mathbf{Y}^{test} \right\|_2^2$$

Parameters $\mathbf{W} \in \mathbb{R}^D$ are found by minimizing MSE_{train}

$$\mathrm{MSE}_{train} = \frac{1}{N_{train}} \sum_{n=1}^{N_{train}} \left(\hat{y}_i^{train} - y_i^{train} \right)^2 = \frac{1}{N_{train}} \left\| \hat{\mathbf{Y}}^{train} - \mathbf{Y}^{train} \right\|_2^2 = \frac{1}{N_{train}} \left\| \mathbf{X}^{train} \mathbf{W} - \mathbf{Y}^{train} \right\|_2^2$$

For optimal weights $\mathbf{W}$, the gradient of MSE_{train} should be 0:

$$\nabla_W \mathrm{MSE}_{train} = \nabla_W \frac{1}{N_{train}} \left\| \hat{\mathbf{Y}}^{train} - \mathbf{Y}^{train} \right\|_2^2 = \frac{1}{N_{train}} \nabla_W \left\| \mathbf{X}^{train} \mathbf{W} - \mathbf{Y}^{train} \right\|_2^2 = 0$$

$$\Rightarrow \boxed{\mathbf{W} = \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{y}}$$

Normal equation

Only works analytically for linear regression

Complexity $O\left(N^3\right)$ to $O\left(N^{2.4}\right)$

Gradient Descent for Linear Regression

Performance measure P : mean square error (**MSE**) MSE_{test} :

$$\mathrm{MSE}_{test} = \frac{1}{N_{test}} \sum_{n=1}^{N_{test}} \left(\hat{y}_i^{test} - y_i^{test} \right)^2 = \frac{1}{N_{test}} \left\| \hat{\mathbf{Y}}^{test} - \mathbf{Y}^{test} \right\|_2^2 = \frac{1}{N_{test}} \left\| \mathbf{X}^{test} \mathbf{W} - \mathbf{Y}^{test} \right\|_2^2$$

Parameters $\mathbf{W} \in \mathbb{R}^D$ are found by minimizing MSE_{train}

$$\mathrm{MSE}_{train} = \frac{1}{N_{train}} \sum_{n=1}^{N_{train}} \left(\hat{y}_i^{train} - y_i^{train} \right)^2 = \frac{1}{N_{train}} \left\| \hat{\mathbf{Y}}^{train} - \mathbf{Y}^{train} \right\|_2^2 = \frac{1}{N_{train}} \left\| \mathbf{X}^{train} \mathbf{W} - \mathbf{Y}^{train} \right\|_2^2$$

For optimal weights $\mathbf{W}$, the gradient of MSE_{train} should be 0:

$$\nabla_W \mathrm{MSE}_{train} = \nabla_W \frac{1}{N_{train}} \left\| \hat{\mathbf{Y}}^{train} - \mathbf{Y}^{train} \right\|_2^2 = \boxed{\frac{1}{N_{train}} \nabla_W \left\| \mathbf{X}^{train} \mathbf{W} - \mathbf{Y}^{train} \right\|_2^2 = 0}$$

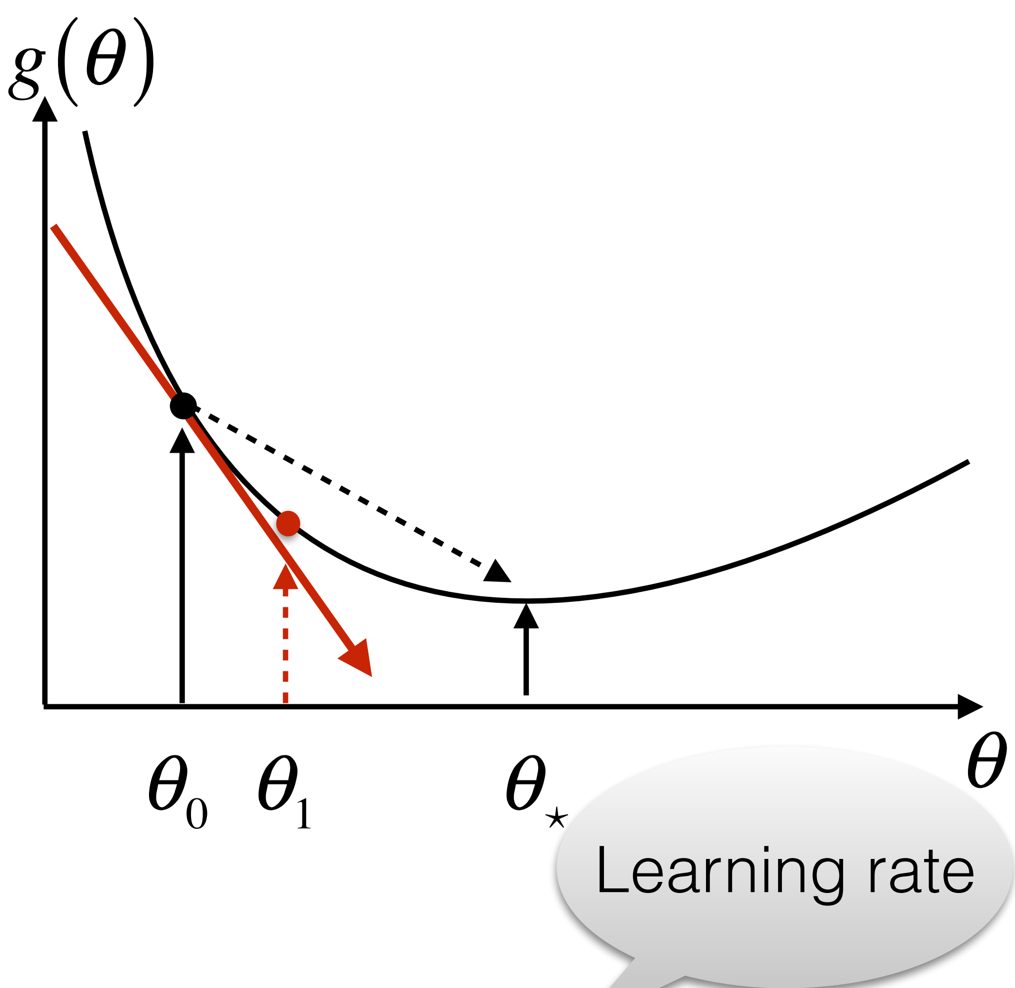
$$\Rightarrow \qquad \mathbf{W} = \cancel{\left(\mathbf{X}^T \mathbf{X} \right)^{-1}} \mathbf{X}^T \mathbf{y}$$

Only works analytically for linear regression

Complexity $O\left(N^3\right)$ to $O\left(N^{2.4}\right)$

Gradient Descent

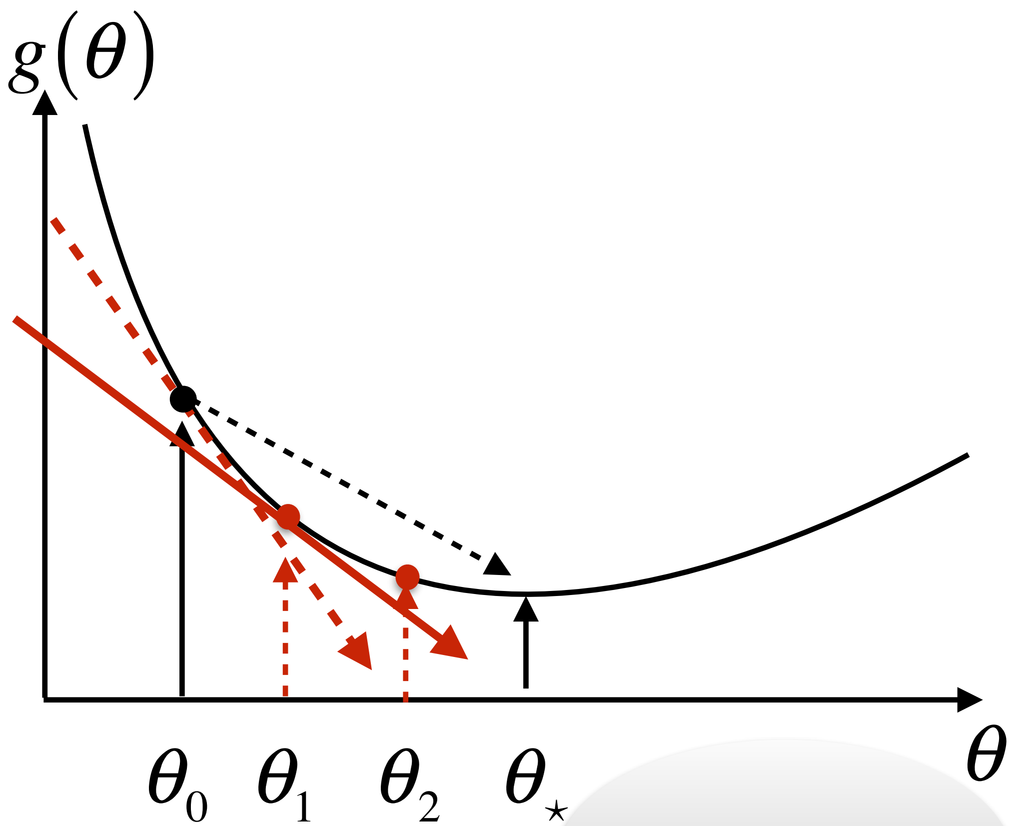
Optimize weights $\mathbf{W}$ **gradually** by moving against the gradients of MSE_{train}



$$\theta_1 = \theta_0 - \eta g_\theta^{'}(\theta_0)$$

Gradient Descent

Optimize weights $\mathbf{W}$ **gradually** by moving against the gradients of MSE_{train}

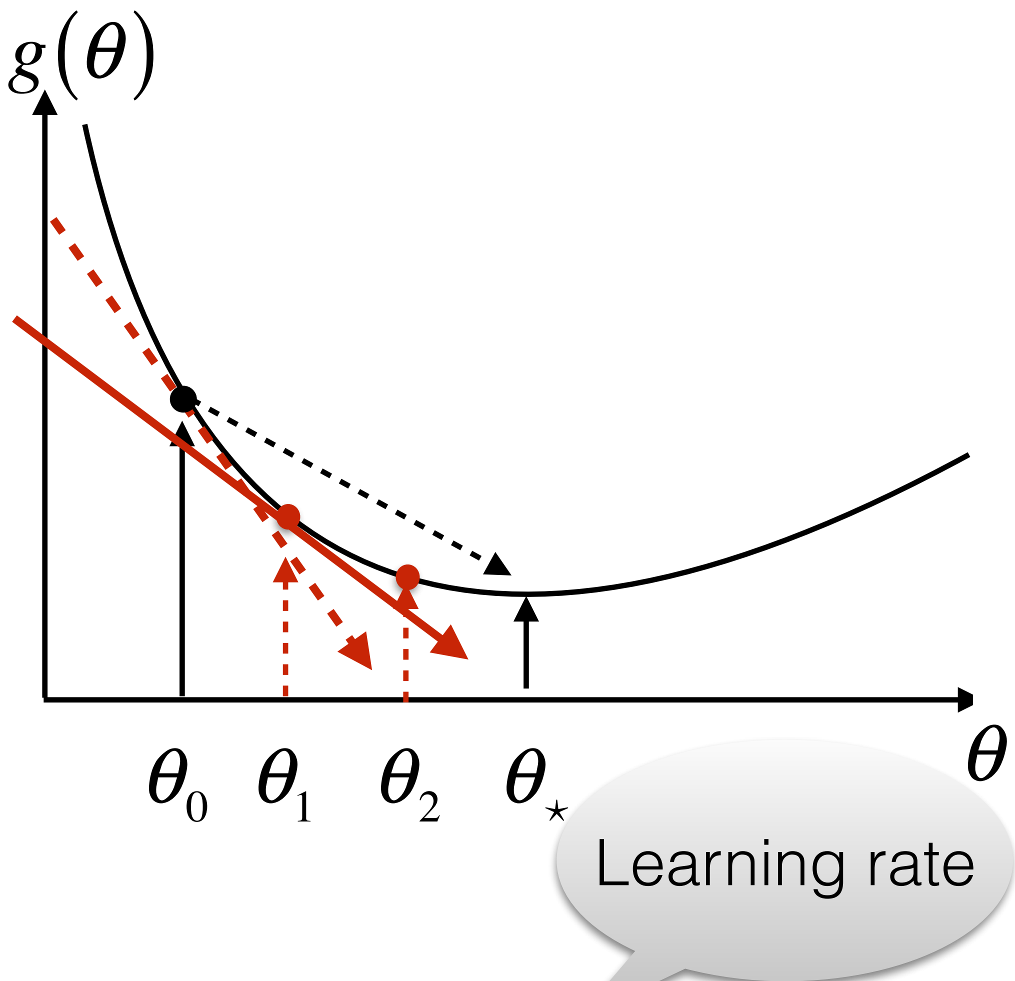


Learning rate

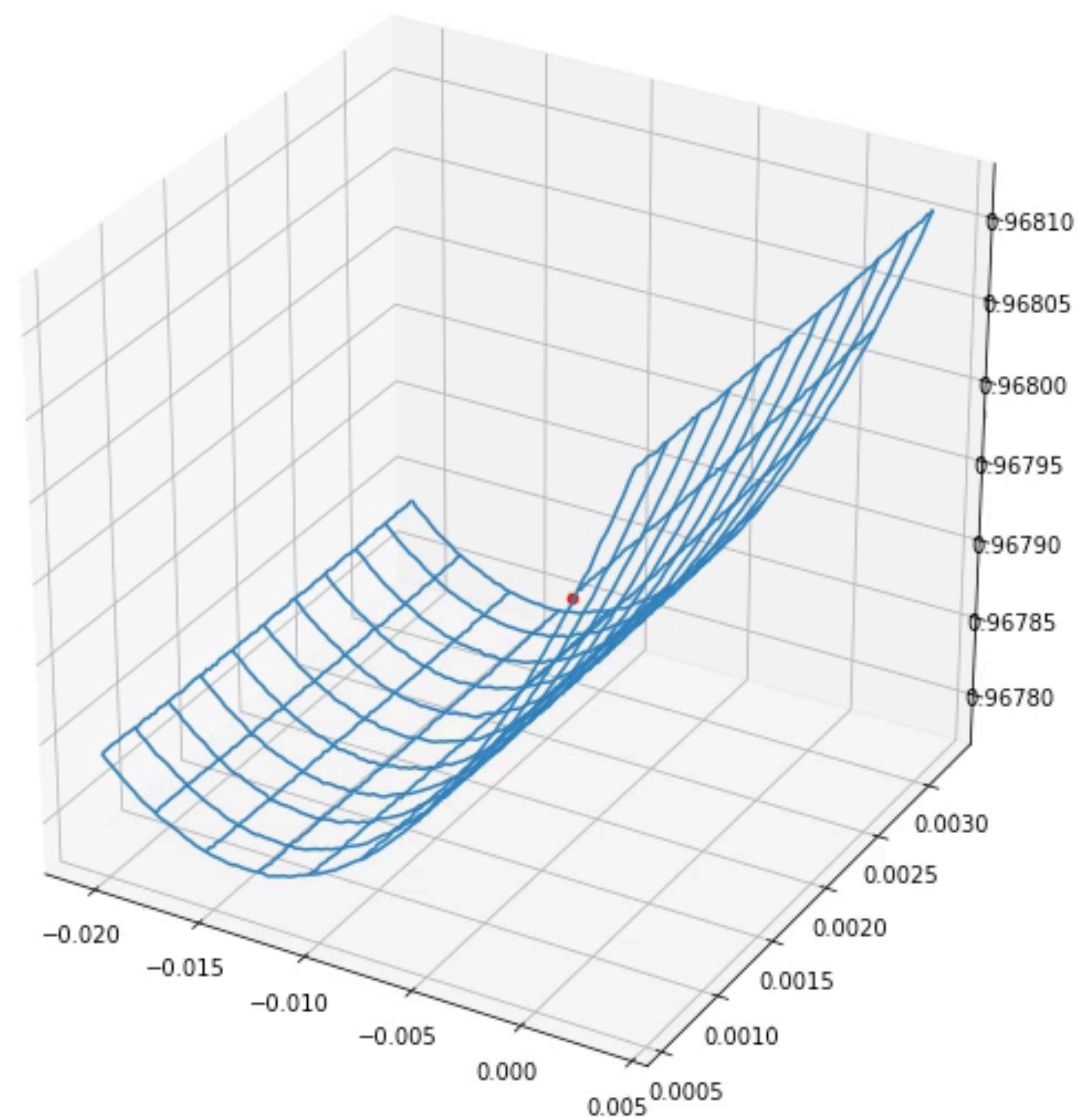
$$\theta_1 = \theta_0 - \eta g_\theta^{'}(\theta_0)$$
$$\theta_2 = \theta_1 - \eta g_\theta^{'}(\theta_1)$$
$$\vdots$$

Gradient Descent

Optimize weights $\mathbf{W}$ **gradually** by moving against the gradients of MSE_{train}

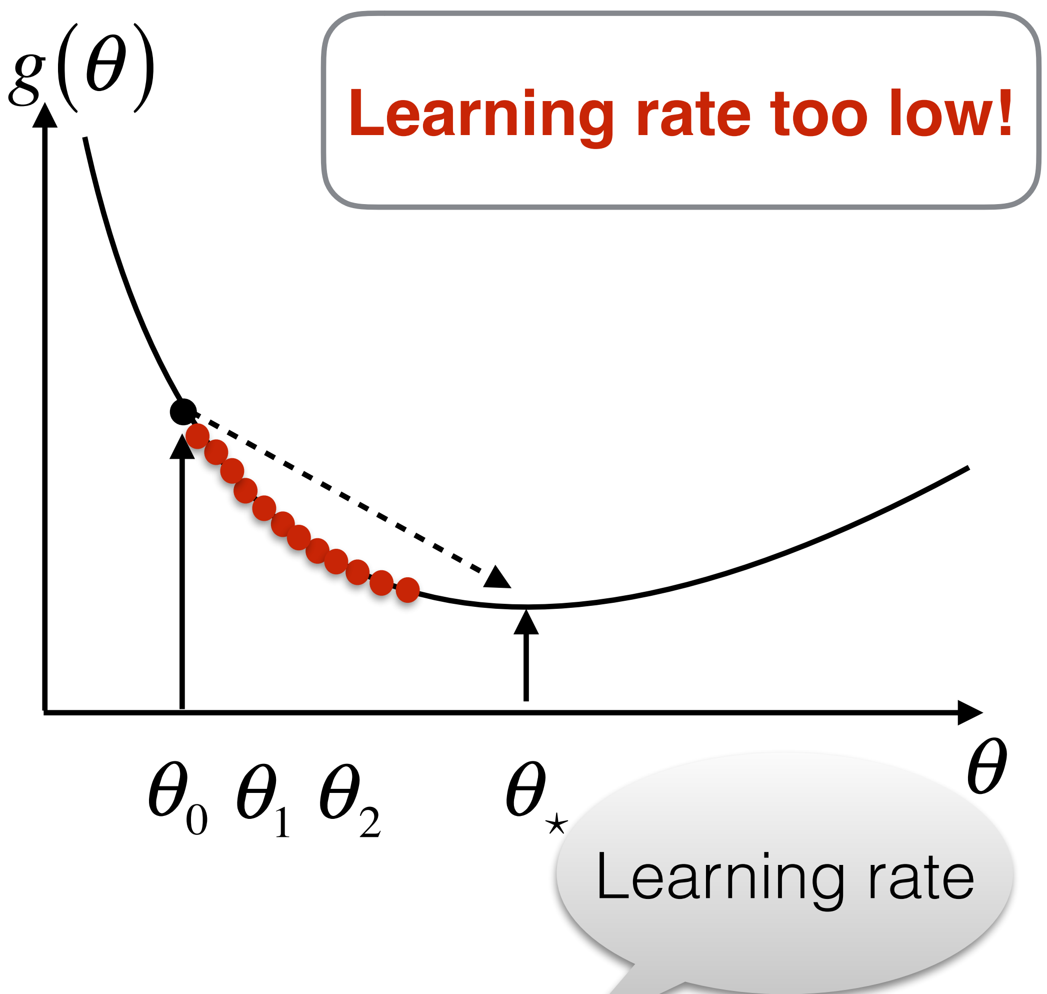


$$\theta_1 = \theta_0 - \eta g_\theta^{'}(\theta_0)$$
$$\theta_2 = \theta_1 - \eta g_\theta^{'}(\theta_1)$$
$$\vdots$$



Choice of learning rate

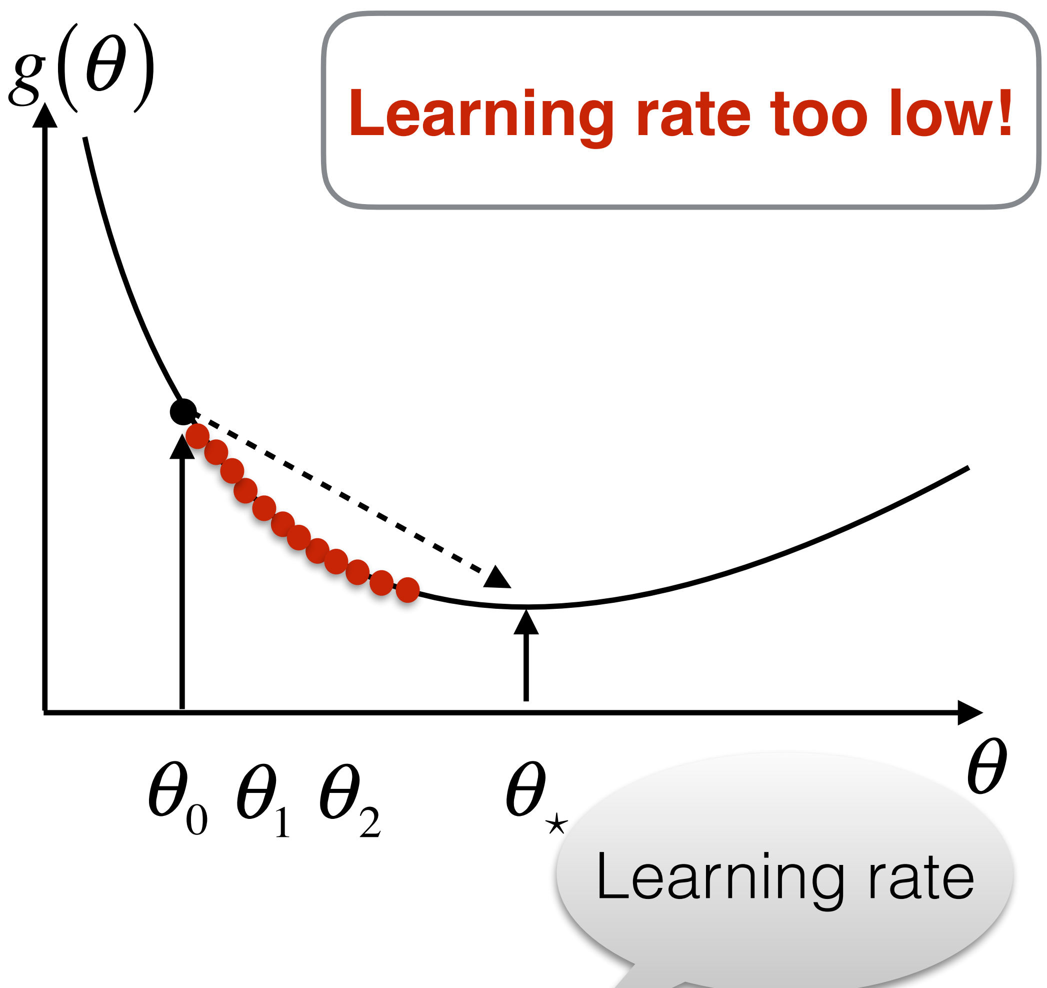
Optimize weights $\mathbf{W}$ **gradually** by moving against the gradients of MSE_{train}

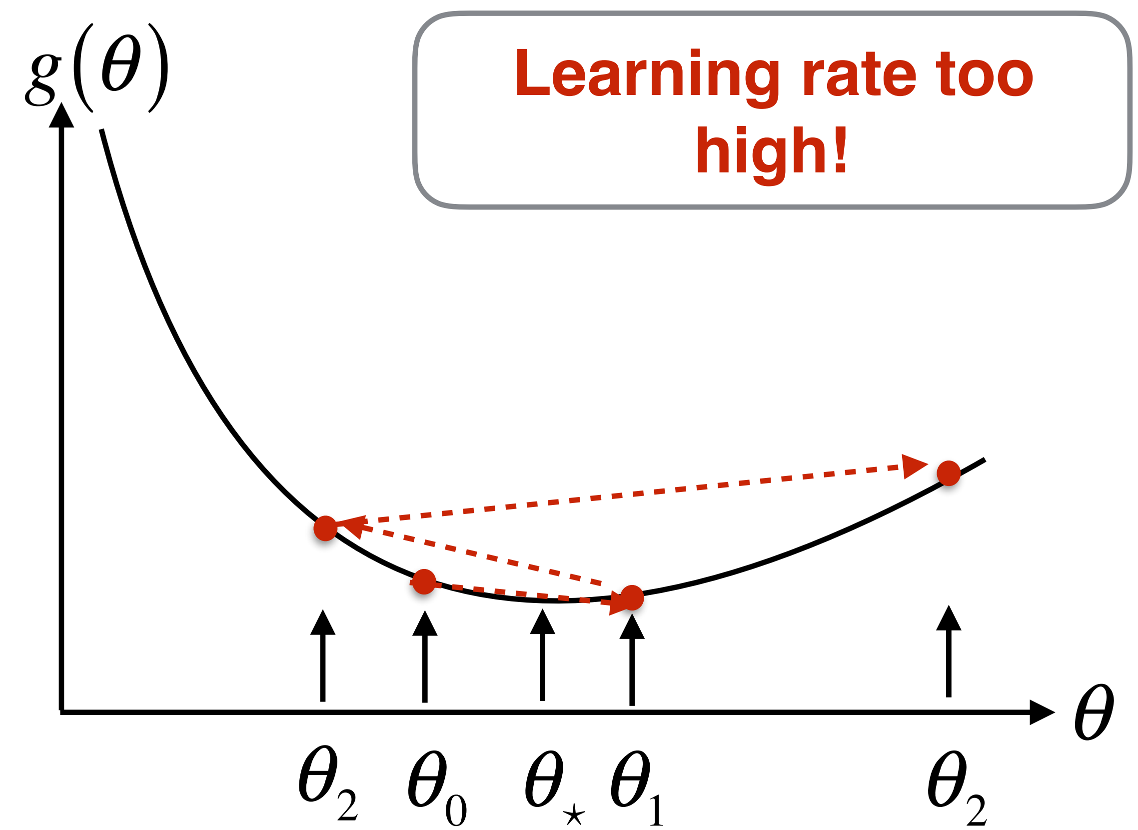


$$\theta_1 = \theta_0 - \eta g_\theta^{'}(\theta_0)$$
$$\theta_2 = \theta_1 - \eta g_\theta^{'}(\theta_1)$$
$$\vdots$$

Choice of learning rate

Optimize weights $\mathbf{W}$ **gradually** by moving against the gradients of MSE_{train}





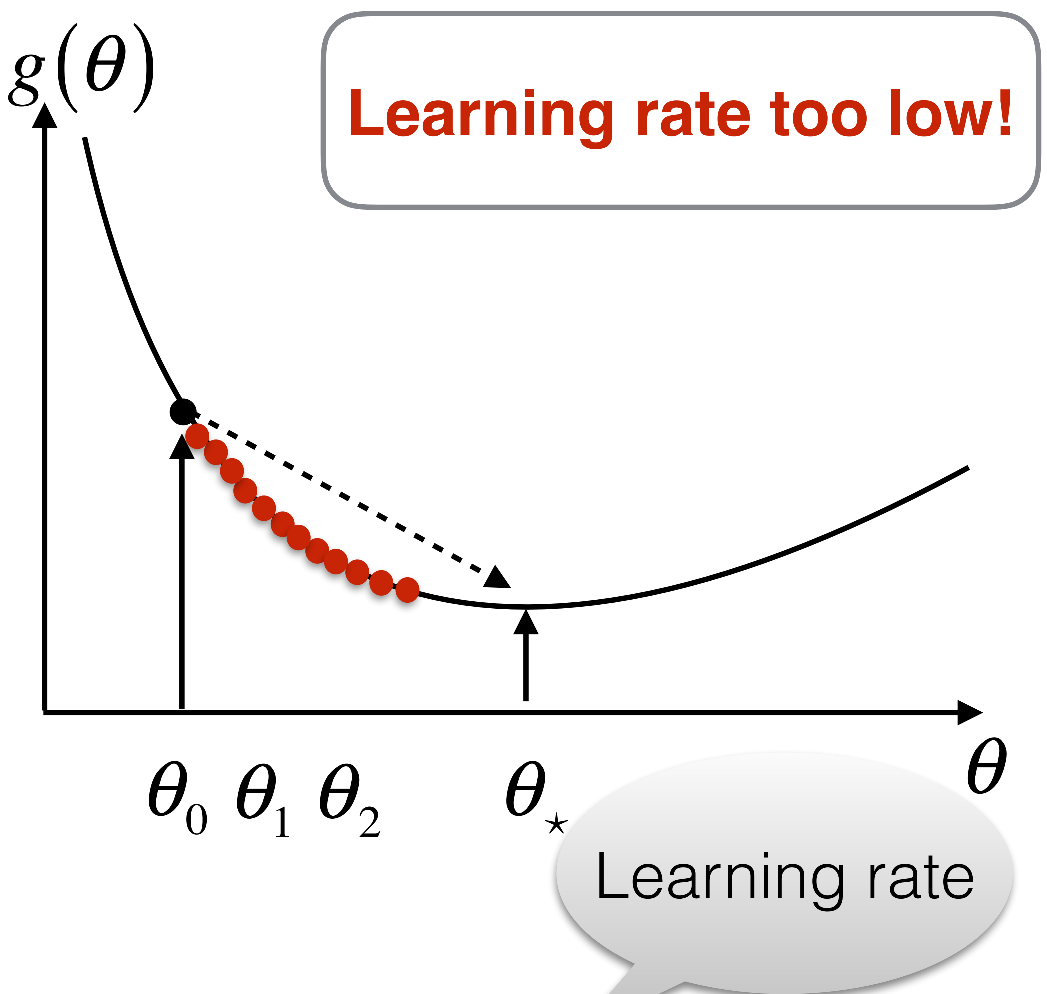
$$\theta_1 = \theta_0 - \eta g_\theta^{'}(\theta_0)$$

$$\theta_2 = \theta_1 - \eta g_\theta^{'}(\theta_1)$$

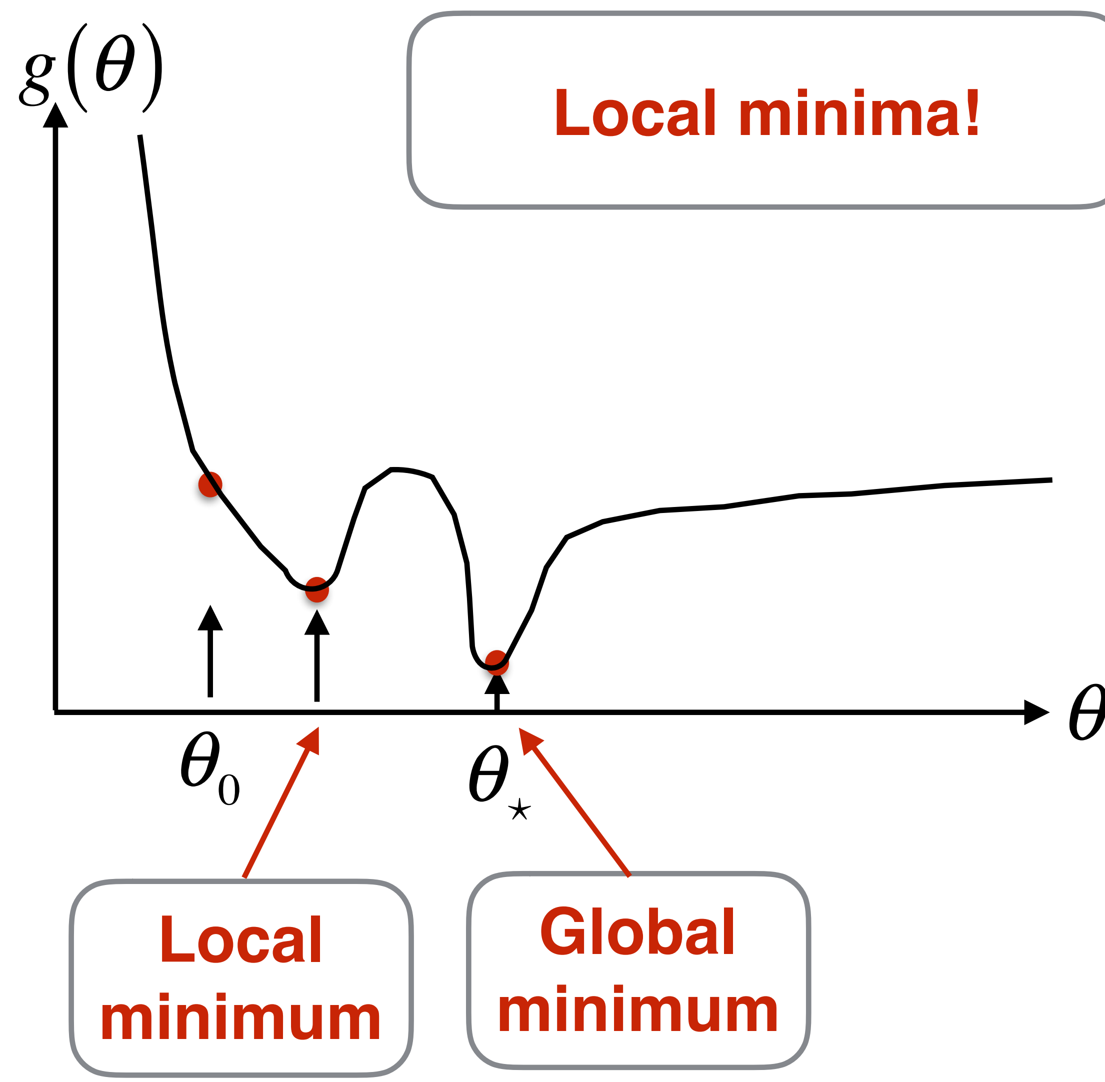
$$\vdots$$

Choice of learning rate

Optimize weights $\mathbf{W}$ **gradually** by moving against the gradients of MSE_{train}

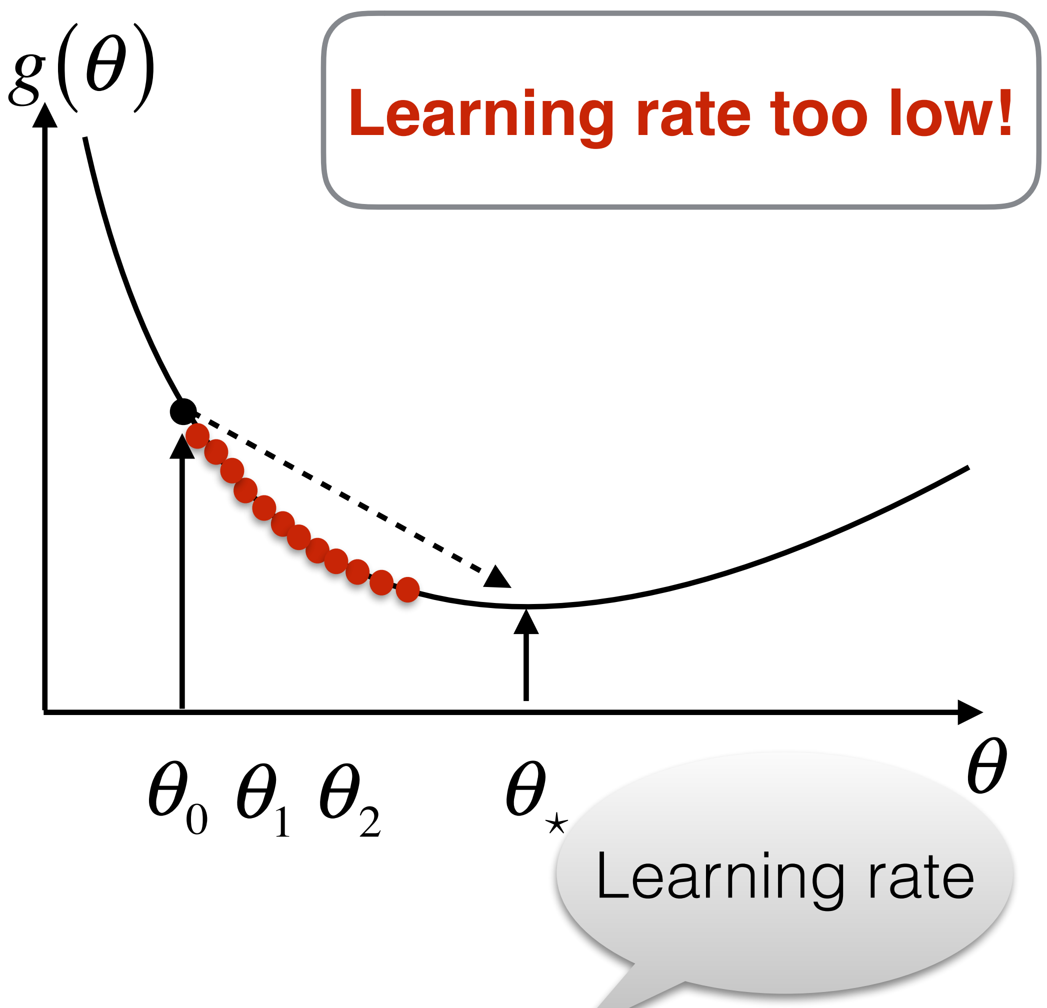


$$\theta_1 = \theta_0 - \eta g_\theta^{'}(\theta_0)$$
$$\theta_2 = \theta_1 - \eta g_\theta^{'}(\theta_1)$$
$$\vdots$$

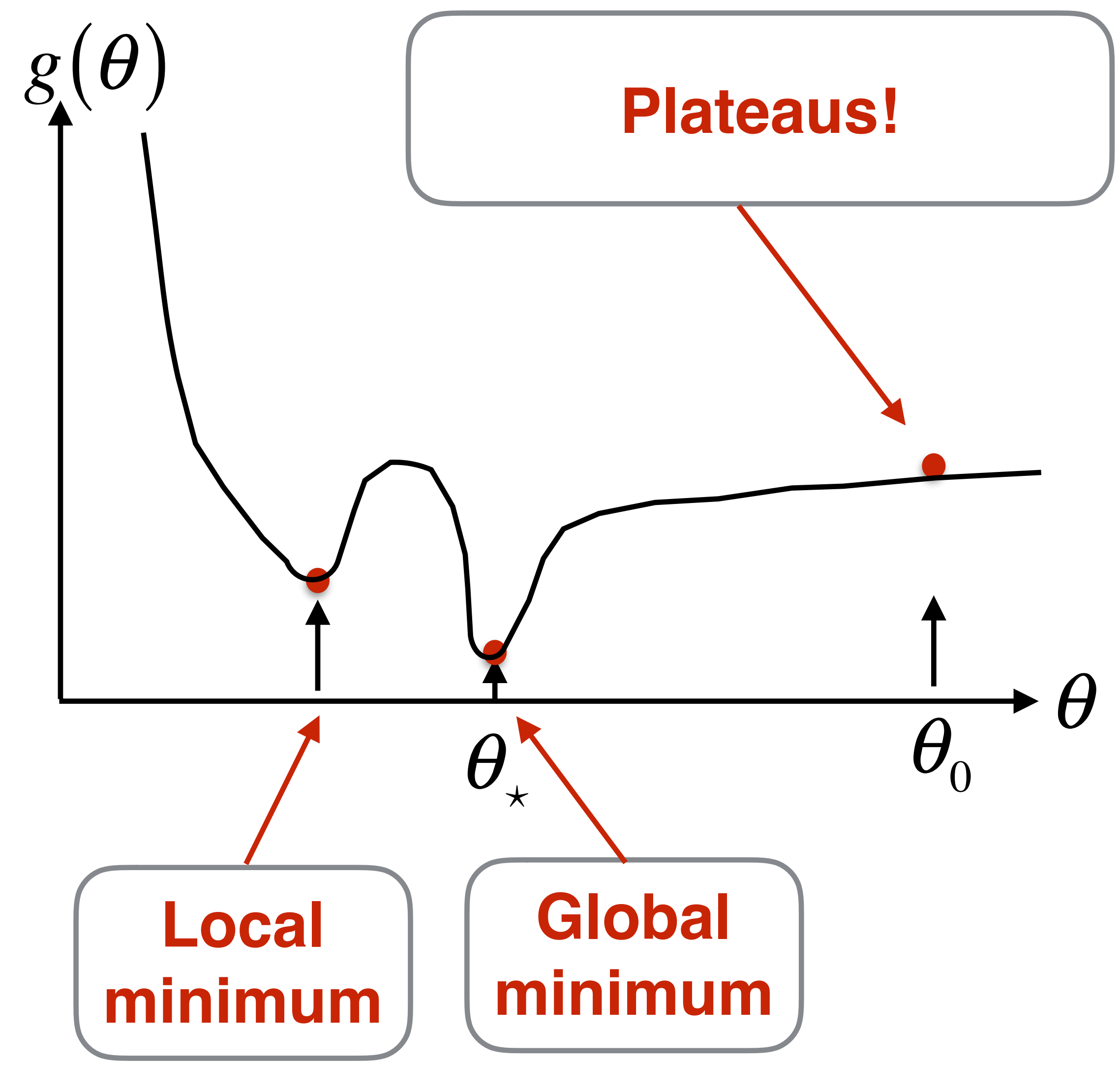


Choice of learning rate

Optimize weights $\mathbf{W}$ **gradually** by moving against the gradients of MSE_{train}



$$\theta_1 = \theta_0 - \eta g_\theta^{'}(\theta_0)$$
$$\theta_2 = \theta_1 - \eta g_\theta^{'}(\theta_1)$$
$$\vdots$$



Control question

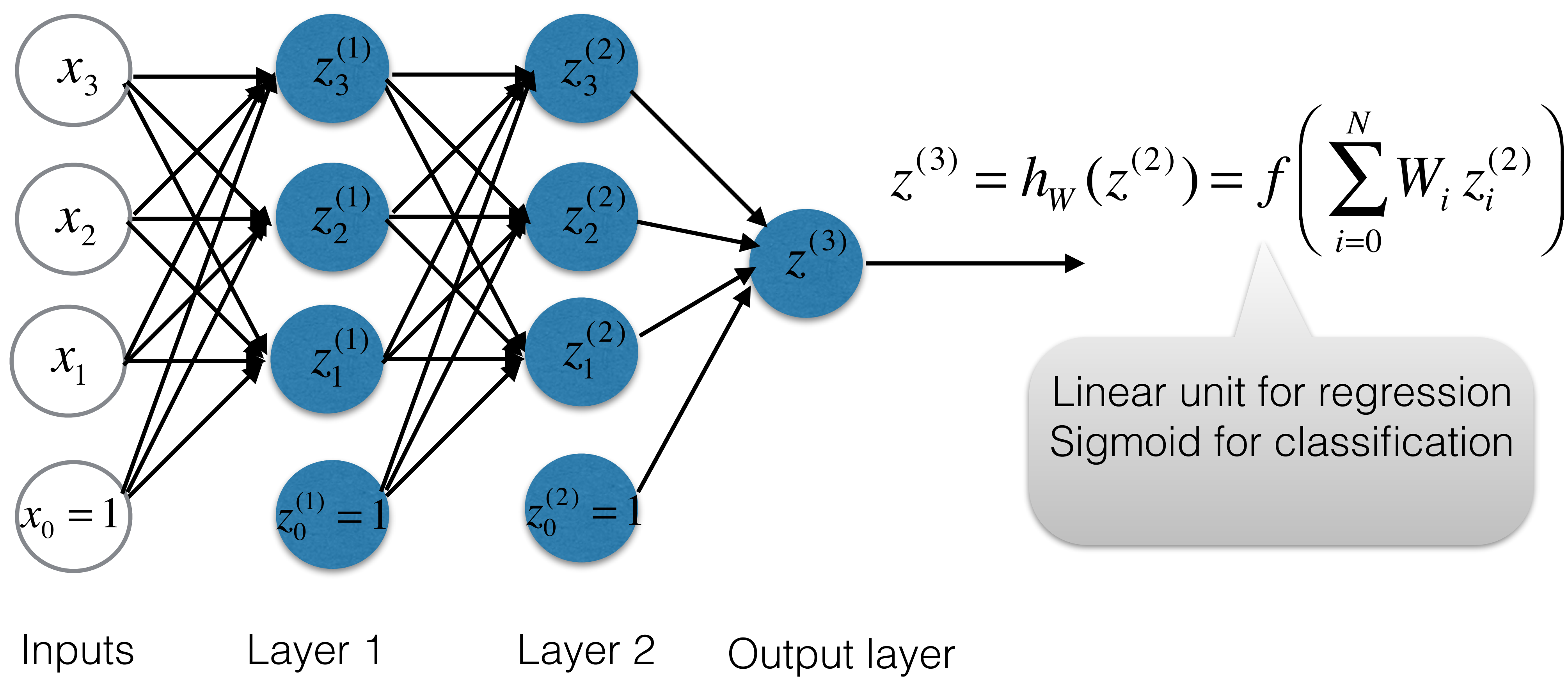
Q: Select all correct statements:

1. Gradient Descent always leads to a unique solution starting from any initial point, no matter what the objective function is.
2. Gradient Descent has one free parameter called the learning rate.
3. A good choice of the learning rate is important: if the learning rate is too small, it takes long for the algorithm to converge, but it if it too high, the algorithm may diverge.
4. Making the learning rate variable (larger initially in the training, and smaller as the training progresses) may accelerate ML algorithms.
5. The fastest way to find the optimal learning rate for a ML algorithm is to simply add it as one more model parameter and optimize over it, along with other model parameters, in the process of minimization of the train error.

Correct answers: 2, 3, 4.

Gradient Descent for Neural Networks

- Gradient descent can be apply not only to one neuron, but to a neural network too:



- Training by backpropagation = reverse-mode autodiff