CSCI 1470/2470
Spring 2022

Ritambhara Singh

February 02, 2024
Wednesday

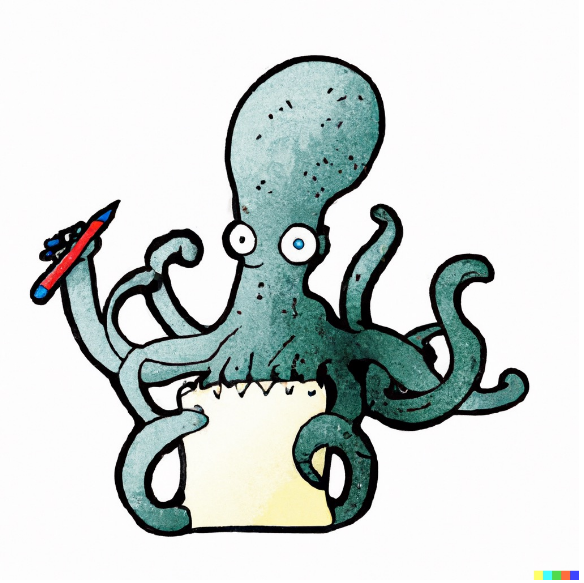
Optimization and Backpropagation

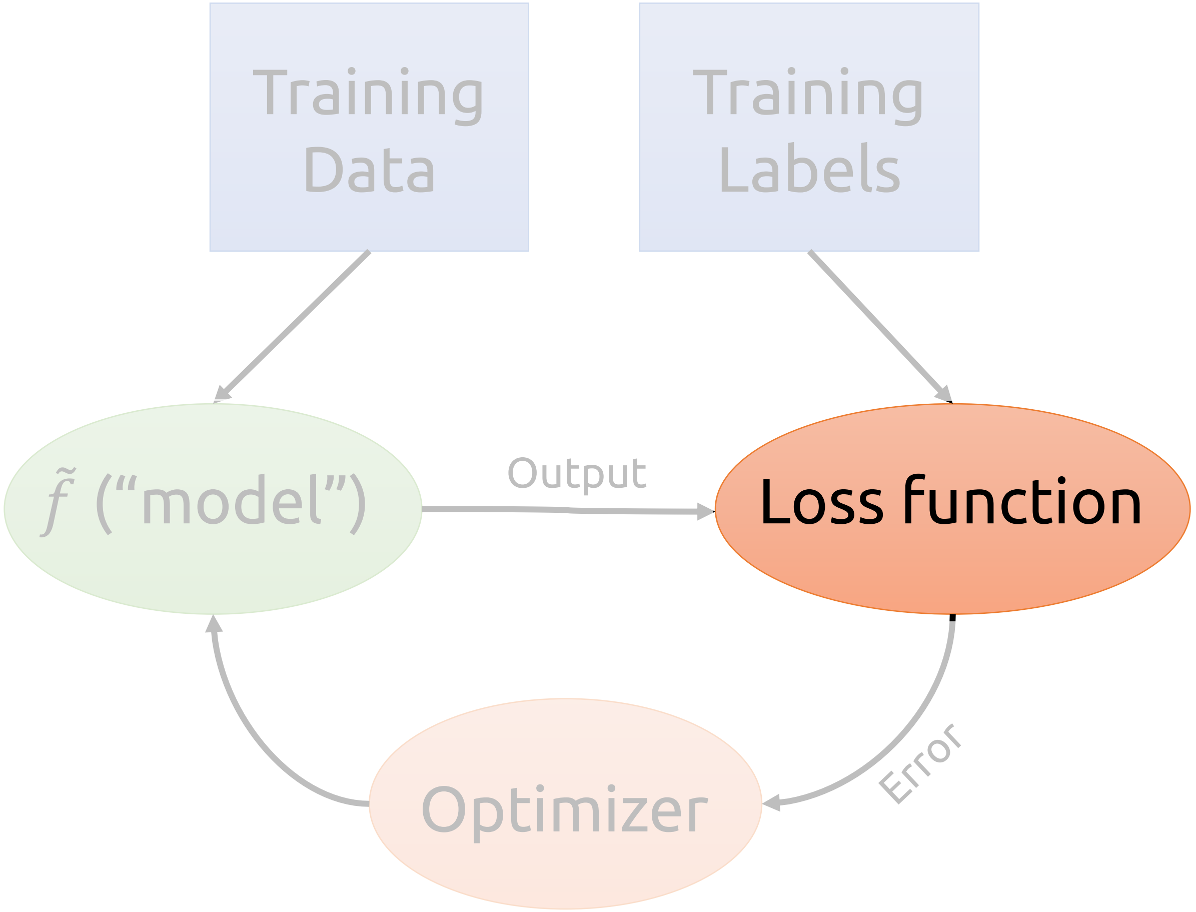
Deep Learning

ChatGPT prompt “minimalist landscape painting of a deep underwater scene with a blue tang fish in the bottom right corner”

Recap: A critical ingredient for our new approach: **Loss functions**

A function L which measures how “wrong” a network is





Mean Squared Error (MSE)

Average squared residual (residual: difference between predicted and true value)

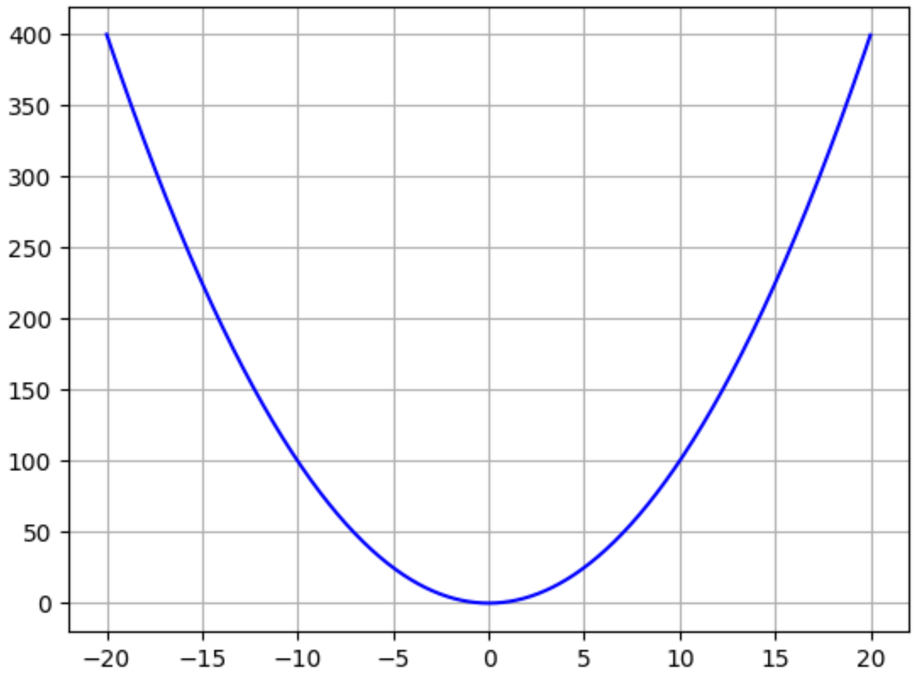
Decreasing the MSE = the model has less error = data points fall closer to the regression line

$$MSE = \frac{\sum_{k=1}^{n}(y^k - \hat{y}^k)^2}{n}$$

$y^k :$ *true output value*

$\hat{y}^k:$ *predicted output value*

$n:$ *number of samples*



Courtesy: https://statisticsbyjim.com/regression/mean-squared-error-mse/

Cross Entropy Loss (for Binary classification)

$y = true\ label\ of\ class\ (0\ or\ 1)$

$p = predicted\ probability\ of\ class\ 1$

y = 1, p = 0.9

$$-(\mathrm{y} \log(p) + (1-y)\log(1-p))$$

y = 0, p = 0.9

y = 1, p = 0.001

Some examples:

$\log(0.9) = -0.04$

$\log(0.5) = -0.3$

$\log(0.001) = -3$

We get this probability by using a Sigmoid function

y = 0, p = 0.001

Cross Entropy Loss (for Multi-class classification)

$$-\sum_{j=1}^{m} y_j \log(p_j)$$

p	Classes (m)	**y**
0.3	“0”	0
0.2	“1”	0
0.5	“2”	1

We can get these probabilities by using a Softmax function

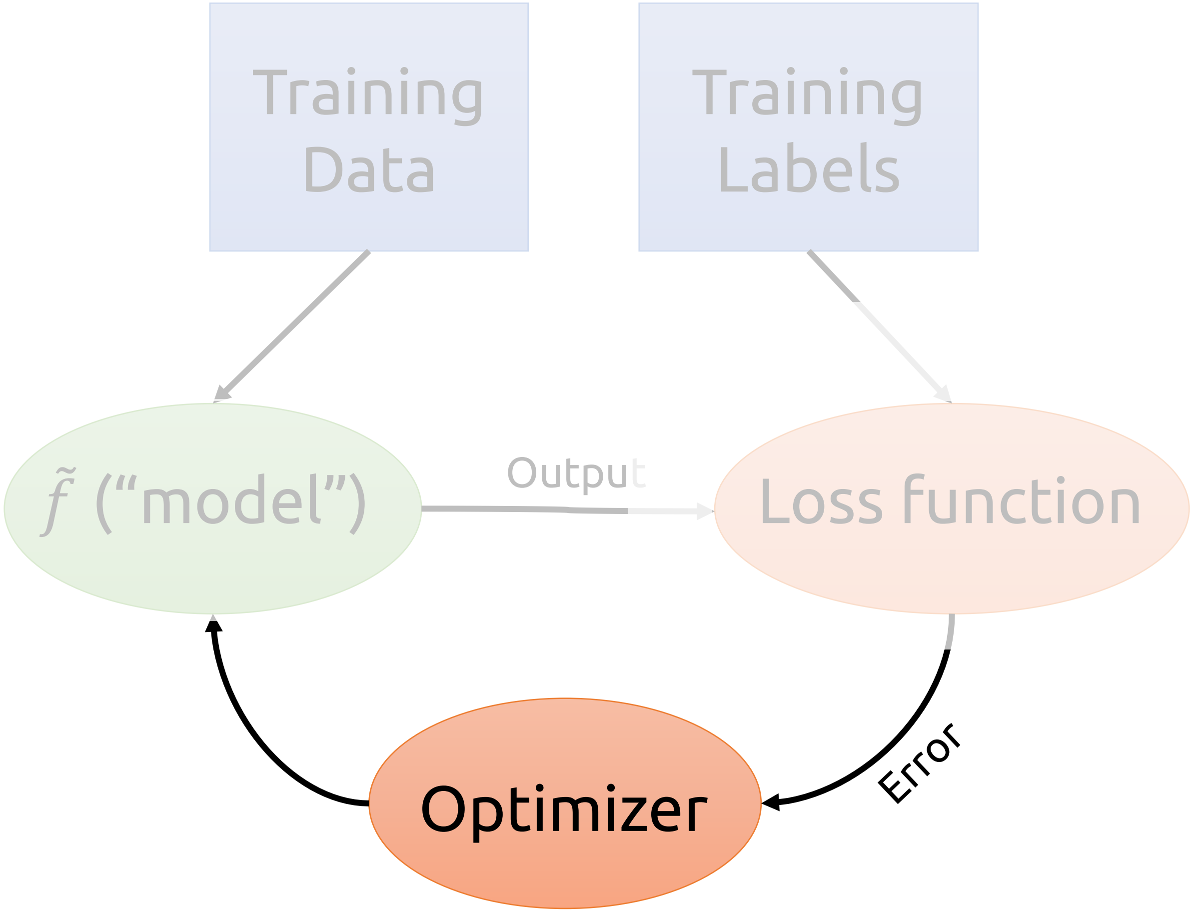
Some examples:

$\log(0.9) = -0.04$

$\log(0.5) = -0.3$

$\log(0.001) = -3$

Next critical ingredient for our new approach: Optimizer



Today’s goal – learn about the optimizer

(1) What does it mean to optimize?

(2) Gradient descent for linear regression

(3) Start building a neural network

(4) Calculating gradients for composite functions (Chain rule)

What does it mean to optimize?

“Optimization” comes from the same root as “optimal”, which means *best*. When you optimize something, you are “making it best”.

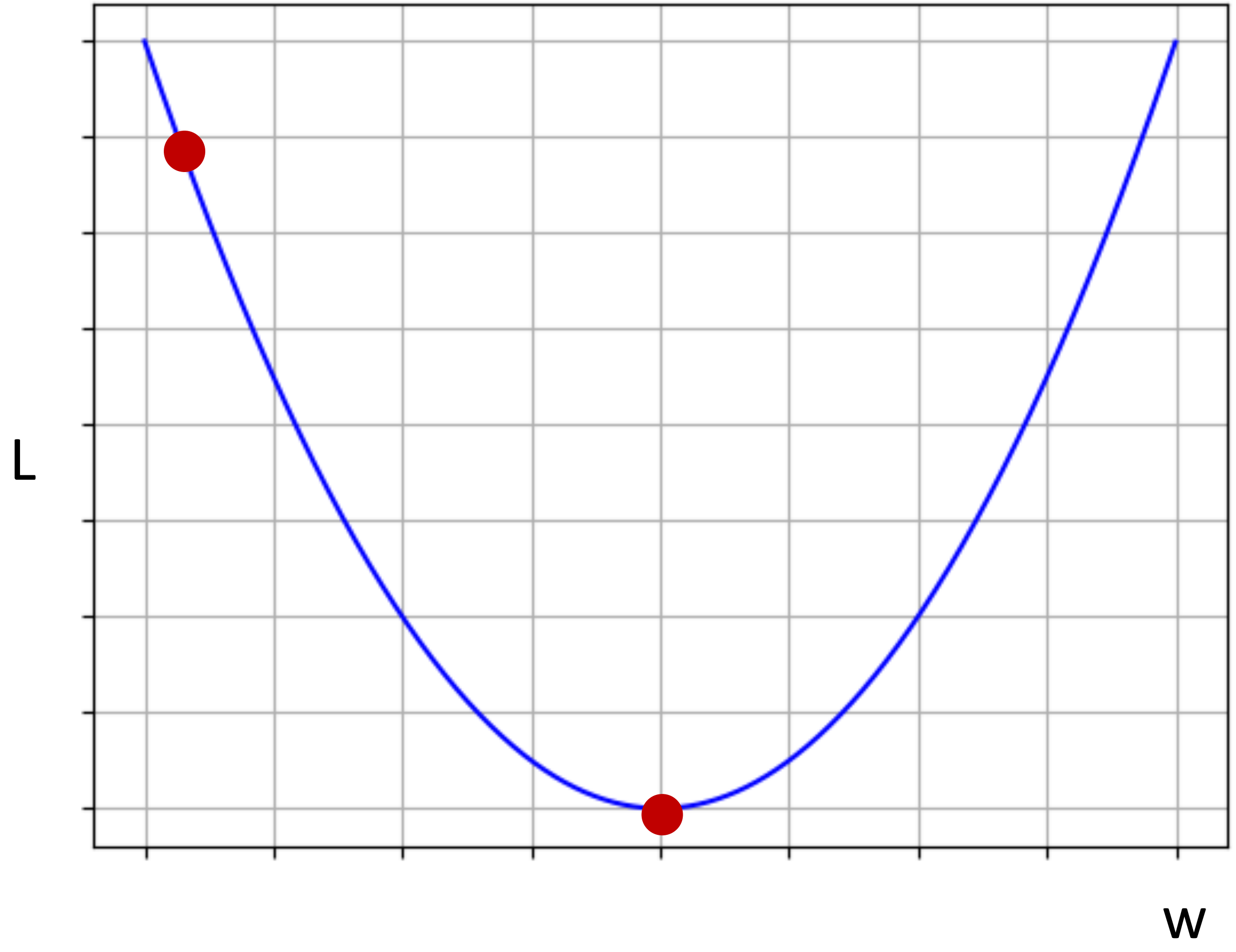
For our case, we want to minimize the loss function to get the “best” model!

https://web.stanford.edu/group/sisl/k12/optimization/MO-unit1-pdfs/1.1optimization.pdf

What does it mean to optimize?

1. Calculate the parameter update values

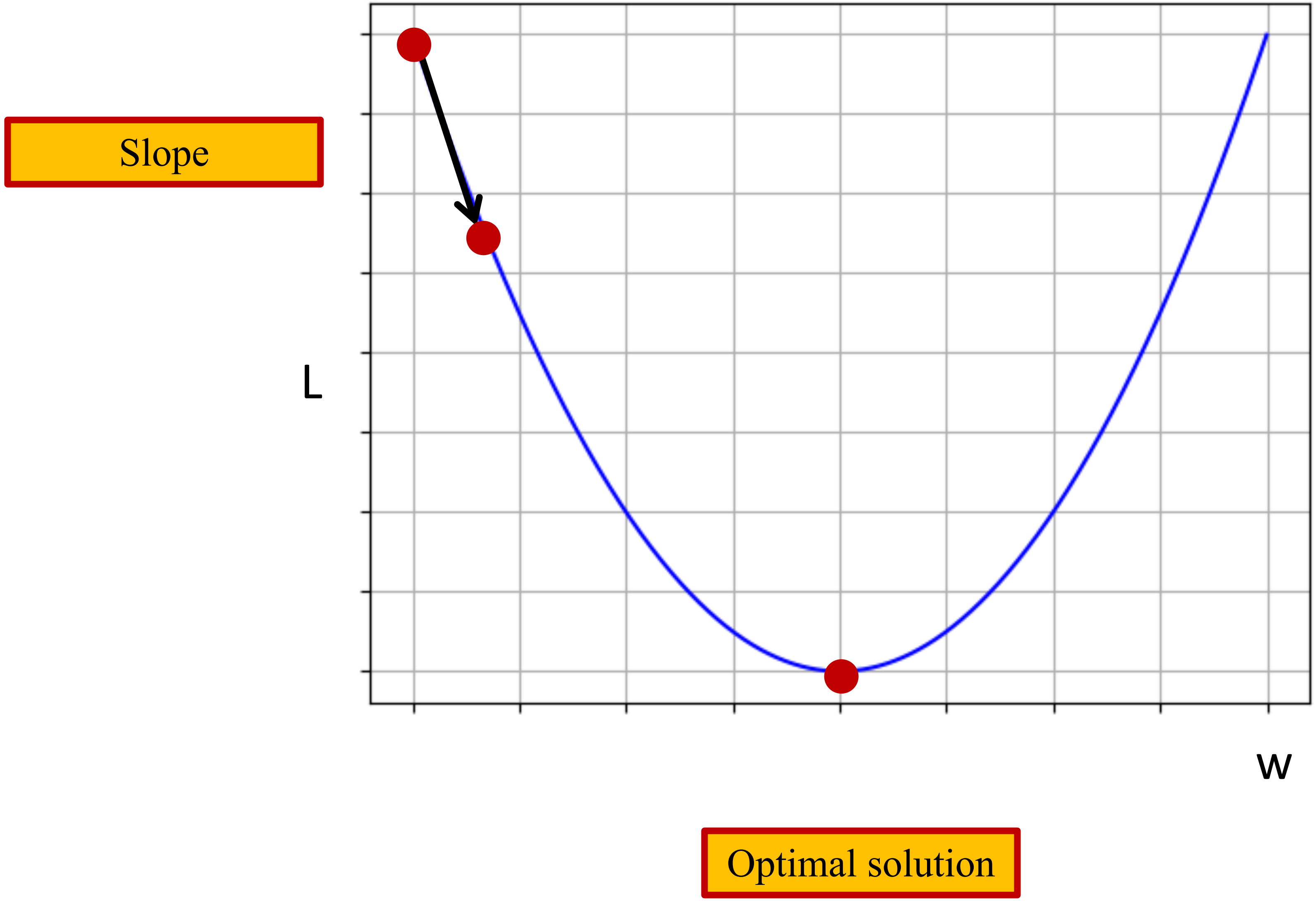
2. Update the parameters



Optimal solution

Gradient (measuring the change)

Calculating partial derivative of the Loss with respect to the weights/parameters



Courtesy: https://statisticsbyjim.com/regression/mean-squared-error-mse/

Vector Calculus Recap

- Partial derivative: the derivative of a multivariable function with respect to one of its variables

Vector Calculus Recap

- Partial derivative: the derivative of a multivariable function with respect to one of its variables
- Example: $f(x, w, b) = wx + b$
- The partial derivative of f with respect to w is $\frac{\partial f}{\partial w}$

Vector Calculus Recap

- Partial derivative: the derivative of a multivariable function with respect to one of its variables
- Example: $f(x, w, b) = wx + b$
- The partial derivative of f with respect to w is $\frac{\partial f}{\partial w}$
- How to compute? -- treat all other variables as constants and differentiate

$$\frac{\partial f}{\partial w} =$$

Vector Calculus Recap

- Partial derivative: the derivative of a multivariable function with respect to one of its variables
- Example: $f(x, w, b) = wx + b$
- The partial derivative of f with respect to w is $\frac{\partial f}{\partial w}$
- How to compute? -- treat all other variables as constants and differentiate

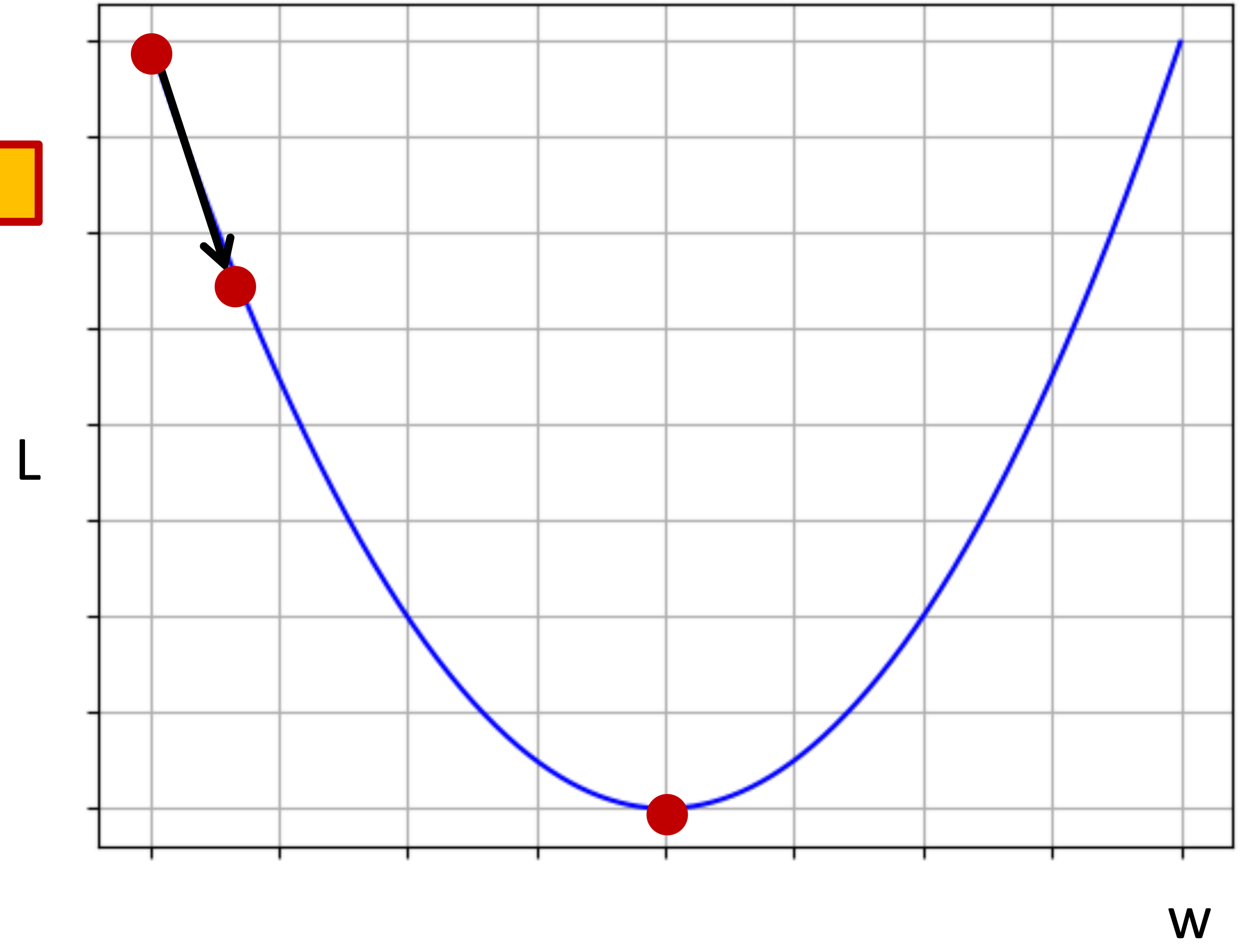
$$\frac{\partial f}{\partial w} = \frac{\partial}{\partial w}(wx + \mathrm{b}\ ) = \frac{\partial}{\partial w}(wx) + \frac{\partial}{\partial w}(b) = x + 0 = x$$

Gradient Descent

$$\Delta w = -\alpha \cdot \frac{\partial L}{\partial w}$$

Learning rate

Slope



Optimal solution

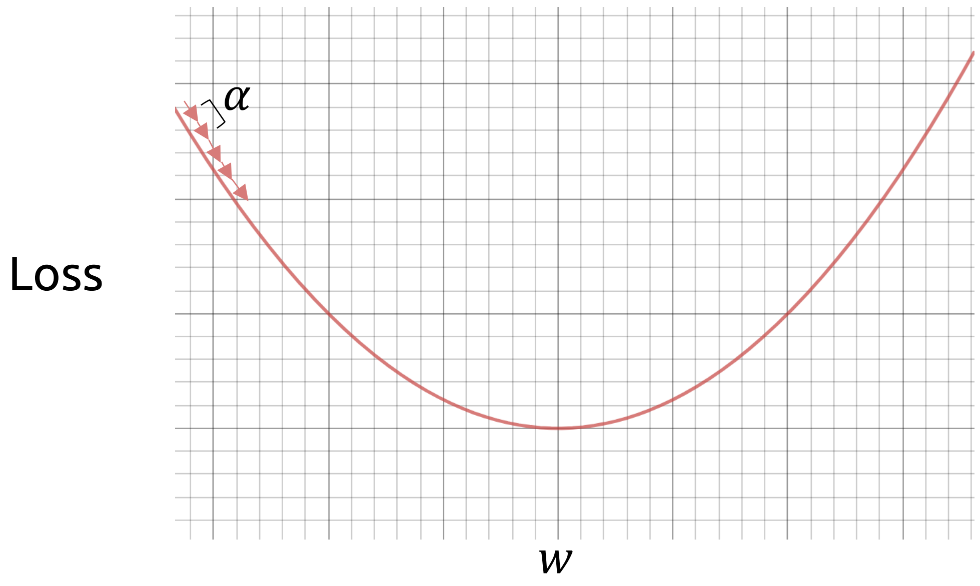
Courtesy: https://statisticsbyjim.com/regression/mean-squared-error-mse/

Impact of Learning Rate

$$\Delta w = -\alpha \cdot \frac{\partial L}{\partial w}$$

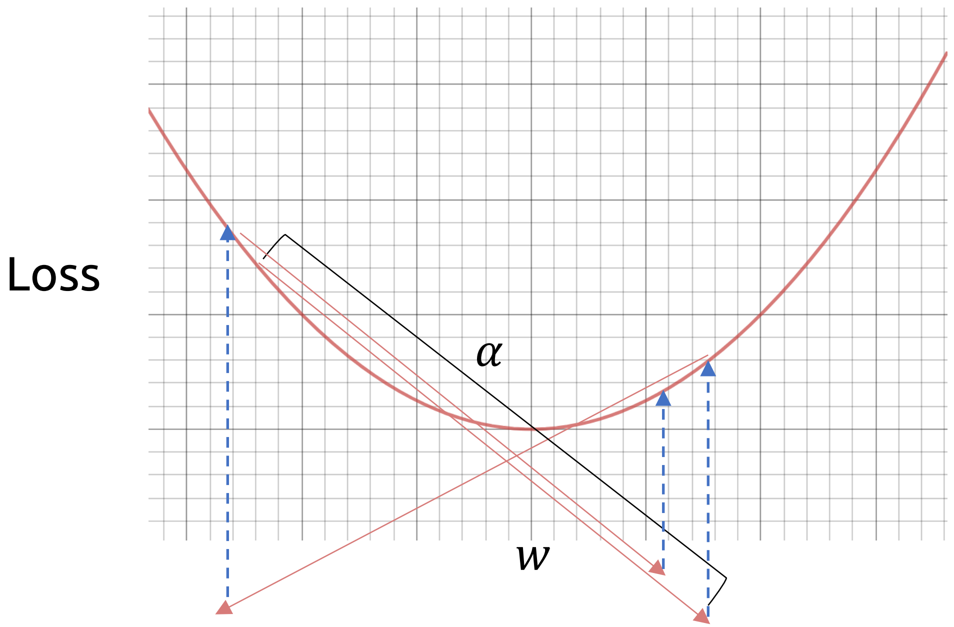
Learning rate too small?
Slow Convergence

$\alpha = 10^{-8}$



Learning rate too big?
Instability (“overshooting”)

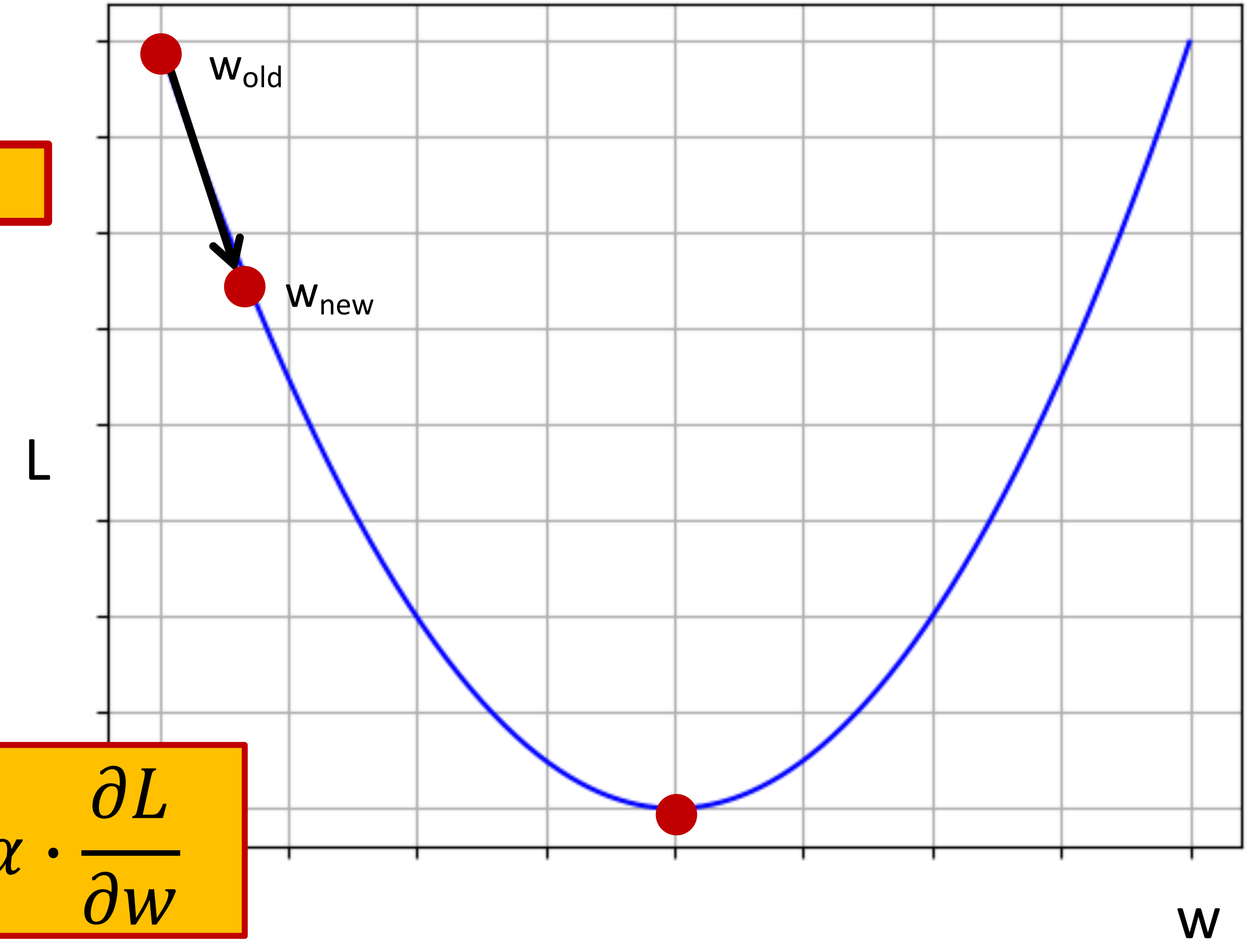
$\alpha = 10^{-1}$



Gradient Descent (updating parameters)

$$\Delta w = -\alpha \cdot \frac{\partial L}{\partial w}$$

Learning rate



Slope

$$w_{new} = w_{old} - \alpha \cdot \frac{\partial L}{\partial w}$$

Optimal solution

Courtesy: https://statisticsbyjim.com/regression/mean-squared-error-mse/

Recap: Mean Squared Error (MSE)

Average squared residual (residual: difference between predicted and true value)

Decreasing the MSE = the model has less error = data points fall closer to the regression line

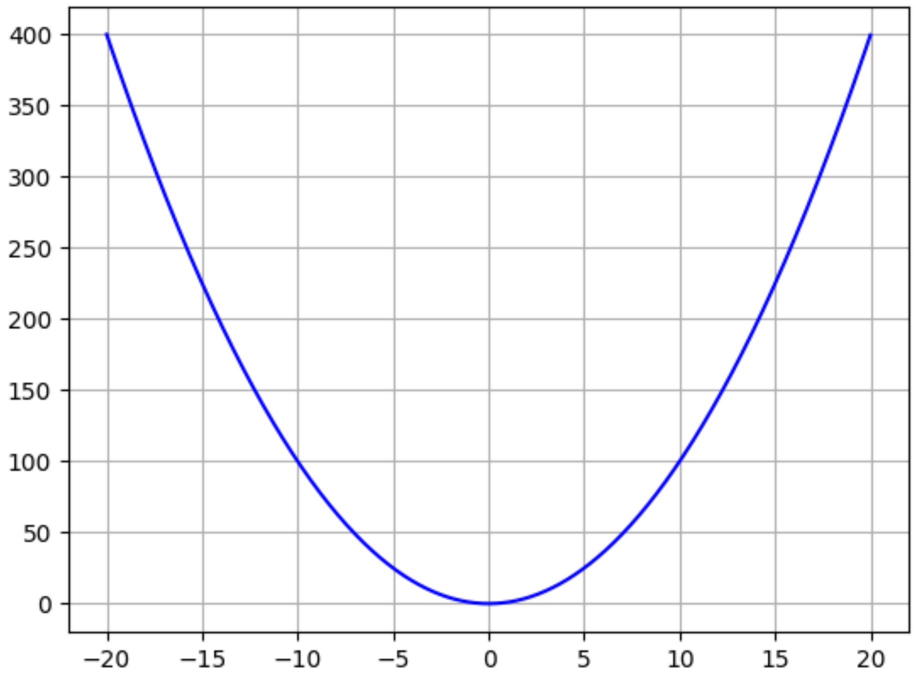
$$MSE = \frac{\sum_{k=1}^{n} (y^k - \hat{y}^k)^2}{n}$$

y^k : *true output value*

$\hat{y}^k$: *predicted output value*

n: *number of samples*

What could be the purpose of squaring the distance?



Courtesy: https://statisticsbyjim.com/regression/mean-squared-error-mse/

Gradient Descent of MSE (1 sample)

$$\Delta w = -\alpha \cdot \frac{\partial L}{\partial w}$$

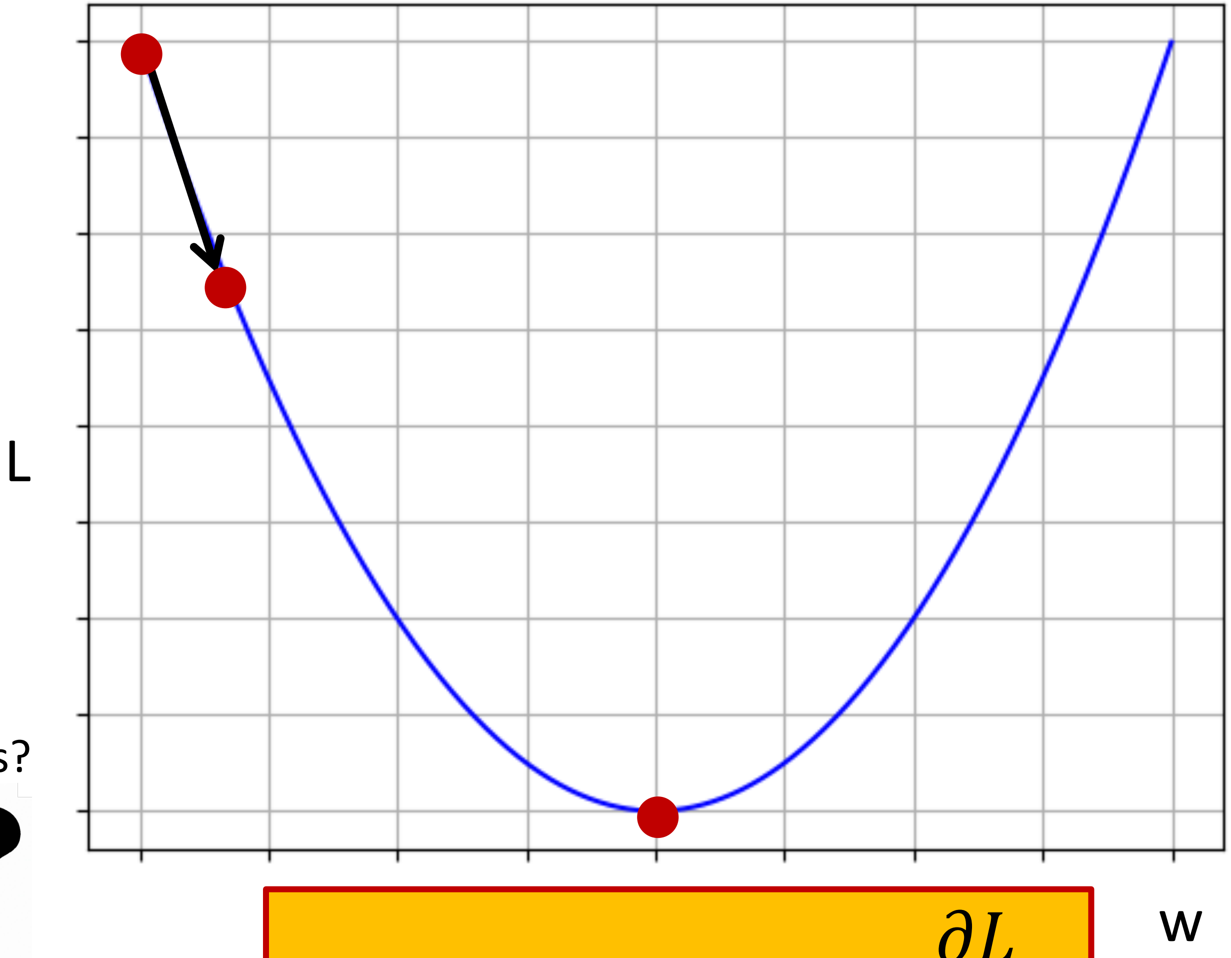
$$L = (y - \hat{y})^2$$

$$= (y - f(x))^2$$

$$= y^2 + f(x)^2 - 2yf(x)$$

$$= y^2 + (wx + b)^2 - 2y(wx + b)$$

$$= y^2 + w^2x^2 + b^2 + 2wxb - 2ywx - 2yb$$



$$\frac{\partial L}{\partial w} = ?$$

$$\frac{\partial L}{\partial w} = 2wx^2 + 2xb - 2yx$$

$$\frac{\partial L}{\partial w} = 2x(wx + b - y)$$

Any questions?

$$\boxed{w_{new} = w_{old} - \alpha \cdot \frac{\partial L}{\partial w}}$$

Convex functions

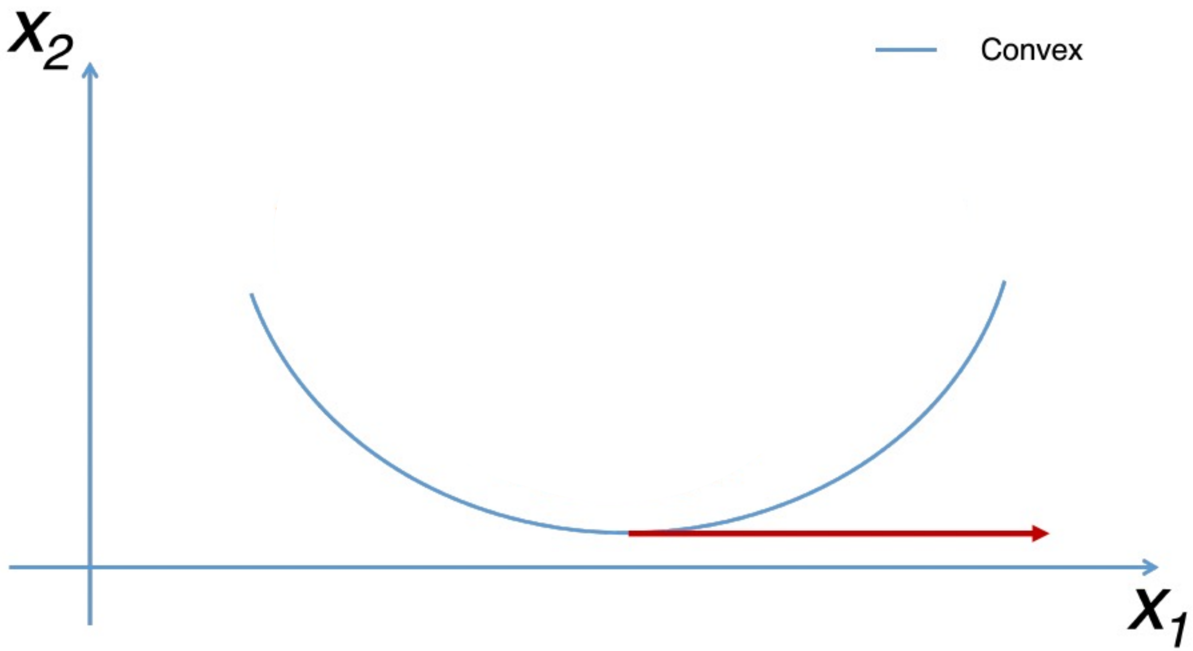


Figure: https://fmin.xyz/docs/theory/Convex_function/

Convex and Non convex functions

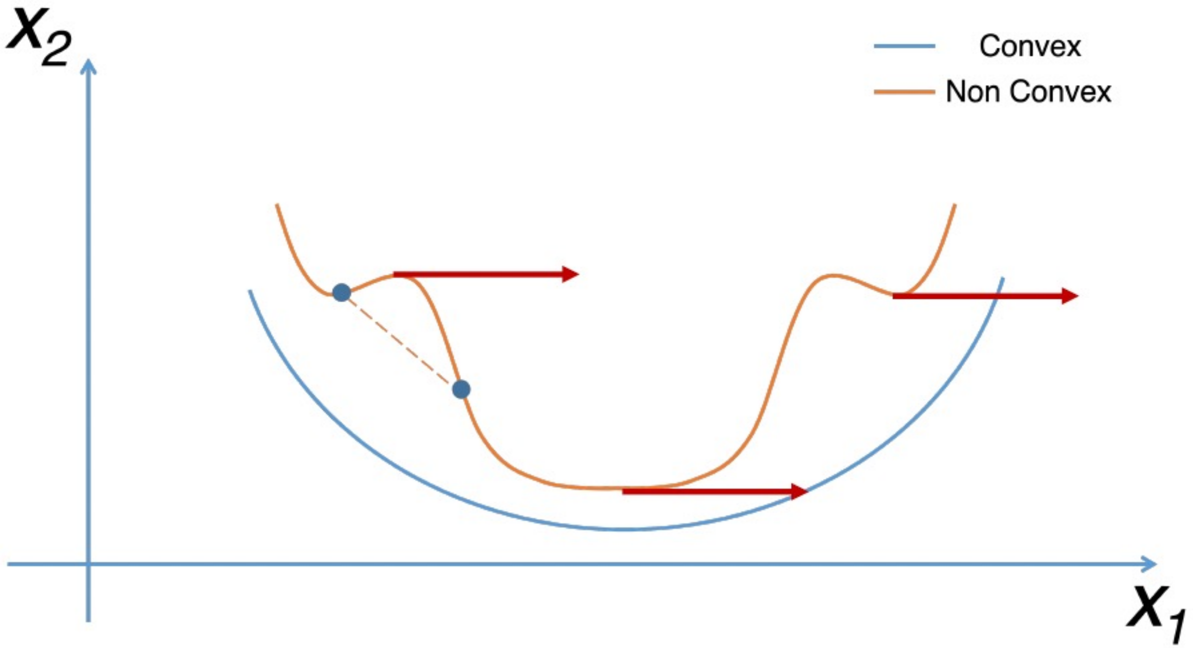
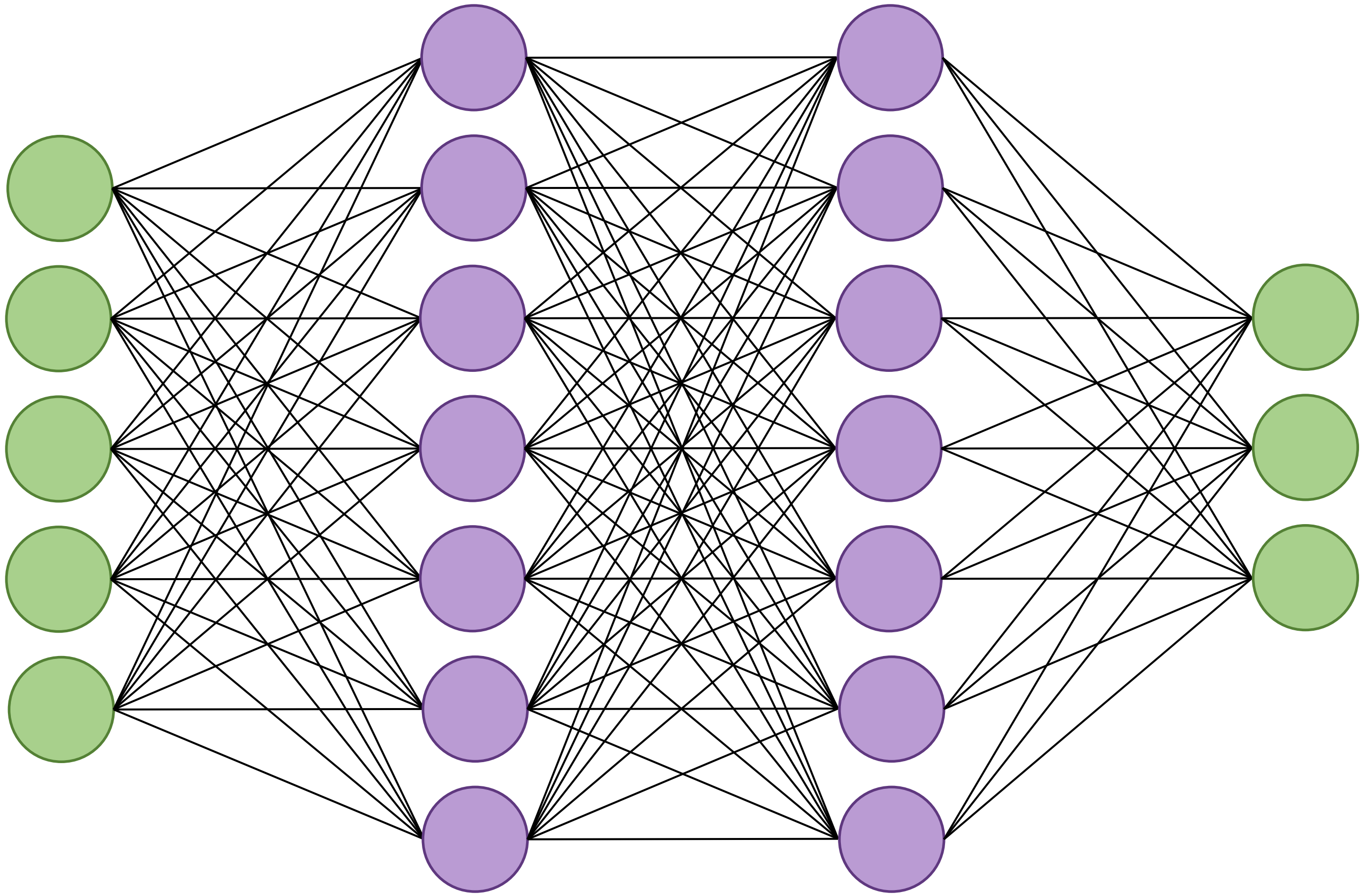


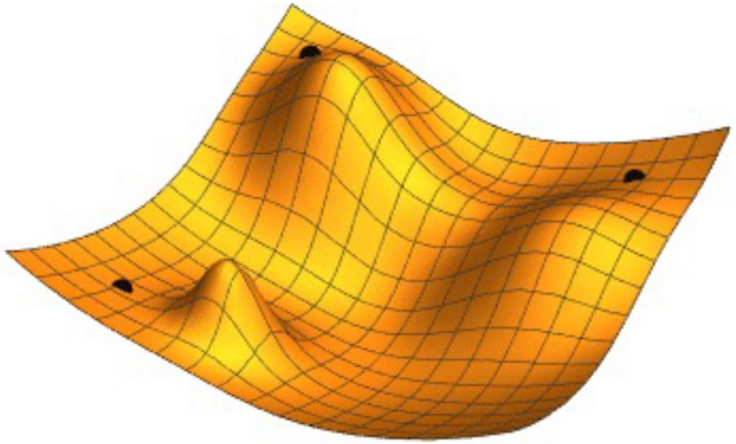
Figure: https://fmin.xyz/docs/theory/Convex_function/

Why do we care about non-convex functions?

Why we care about non-convex functions?



A Multi-Layered Neural Net



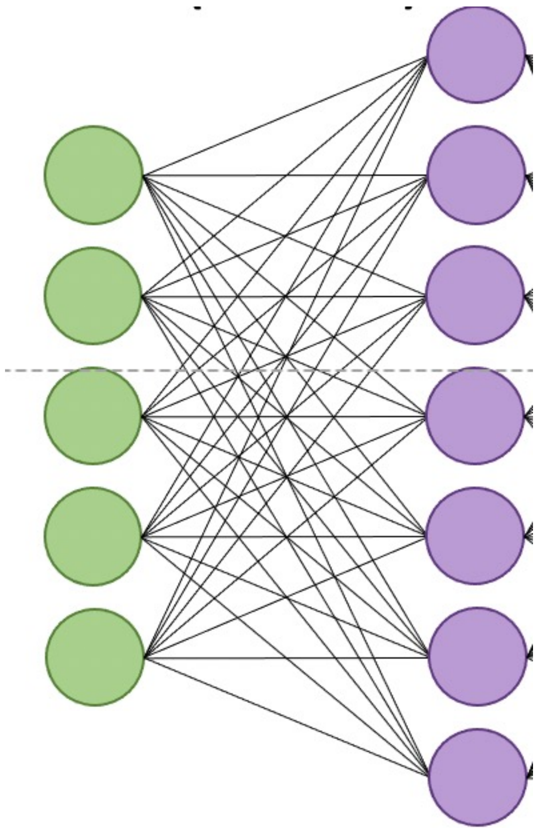
Gradient descent can help the neural net learn!

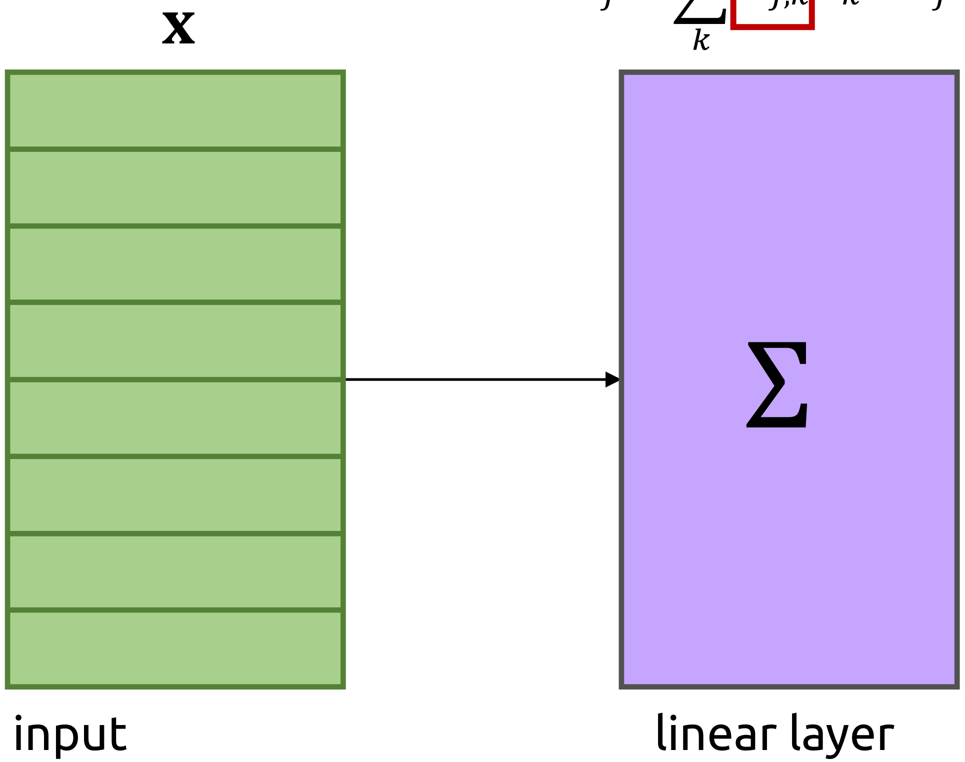
Let’s start building our neural network model

- This is a simplified view of our model with an input and a linear layer

$$l_j = \sum_k \boxed{W_{j,k}}\, x_k + b_j$$

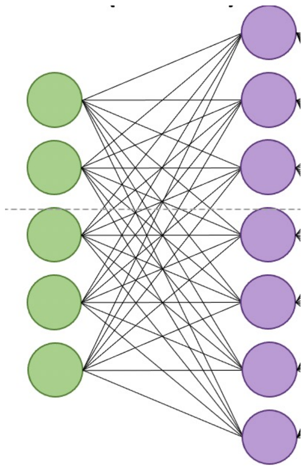
Product of weight matrix with input vector

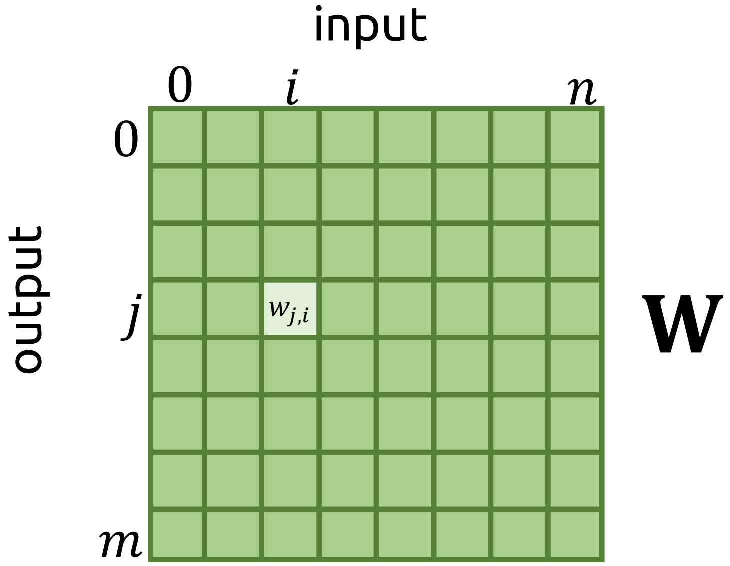




Our Weight Matrix

- We have an input vector of size n and an output vector of size m, so our weights matrix $\mathbf{W}$ is of dimensionality $m \times n$
- $w_{j,i}$ is the j^{th} row and the i^{th} column of our matrix, or the weight multiplied by the i^{th} index of the input which is used to create the j^{th} index in the output





Our Weight Matrix [Example]

$x = [x_1 \;\; x_2]$

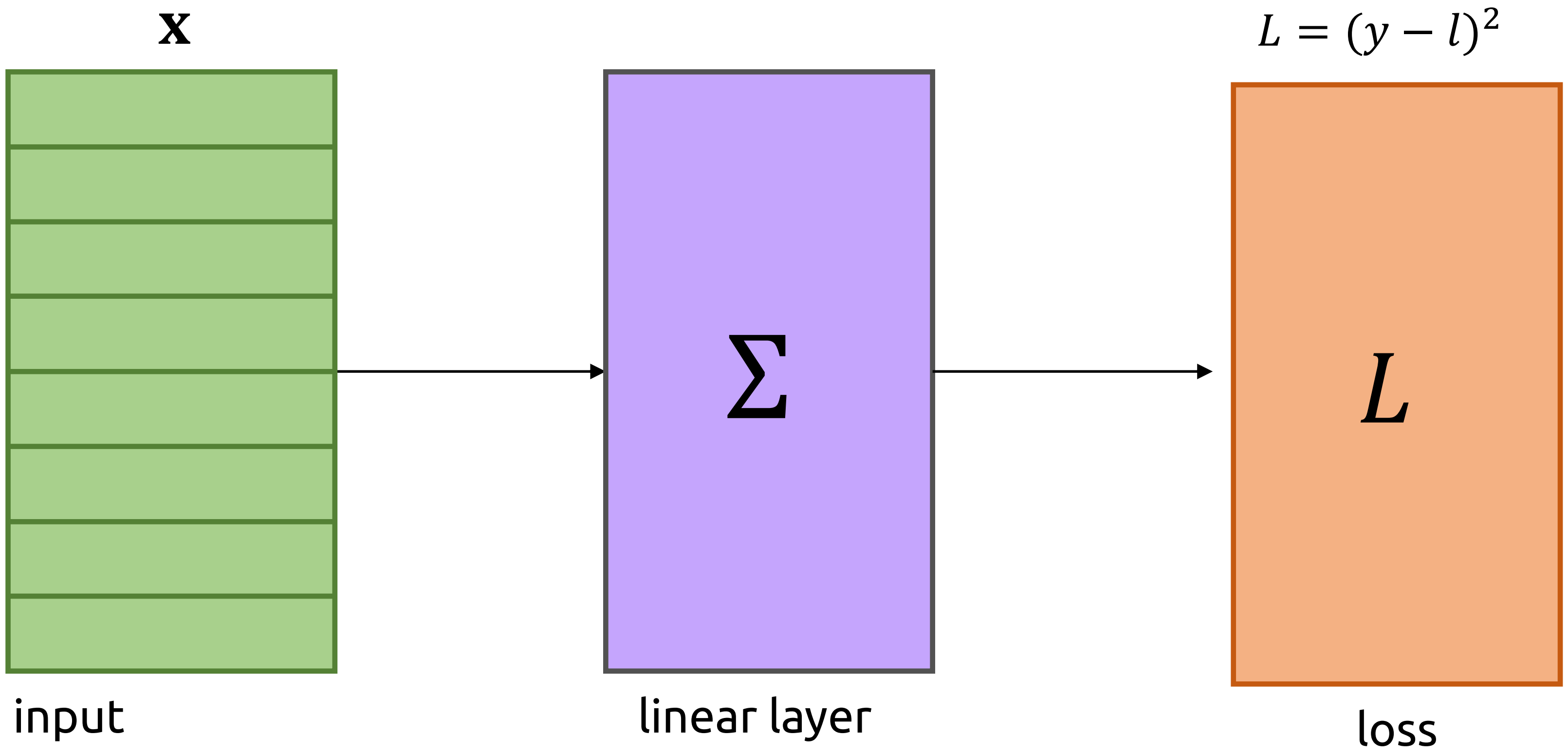
Any questions?

Adding MSE Loss to Our Network

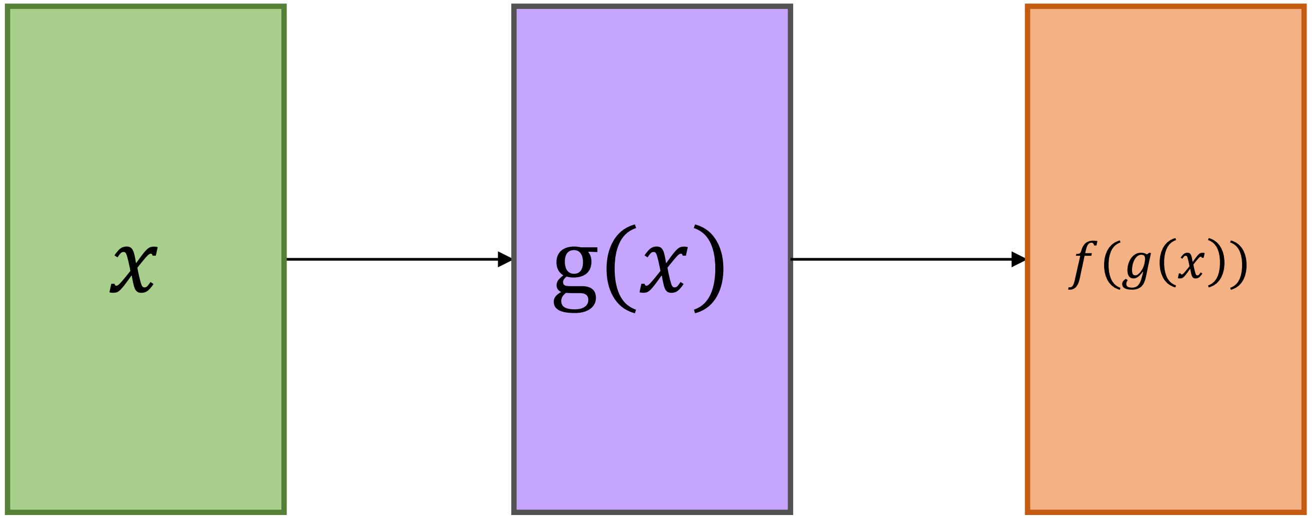
j=1

$$l_j = \sum_k w_{j,k}\, x_k + b_j$$

$$l \quad = \boldsymbol{w}.\boldsymbol{x} \quad + \boldsymbol{b}$$



Looking at composite function!



Using gradient descent to update parameters

- Recall the parameter update for Gradient Descent: $\Delta w = -\alpha \cdot \frac{\partial L}{\partial w}$
- L is a composition of a series of functions (linear layers, loss layer, maybe more...)
- How do we compute the derivative of a composition of functions?
  - Hint: think back to your calculus classes...

Chain rule

If f and g are both differentiable and $F(x)$ is the composite function defined by $F(x) = f(g(x))$ then F is differentiable and F' is given by the product

$$F'(x) = \underbrace{f'(g(x))}_{\text{Differentiate outer function}}\ \underbrace{g'(x)}_{\text{Differentiate inner function}}$$

Courtesy: https://www.onlinemathlearning.com/chain-rule.html

Applying Chain rule [Example]

$$f(x) = x^2 \qquad \mathrm{g}(x) = (2x^2 + 1)$$

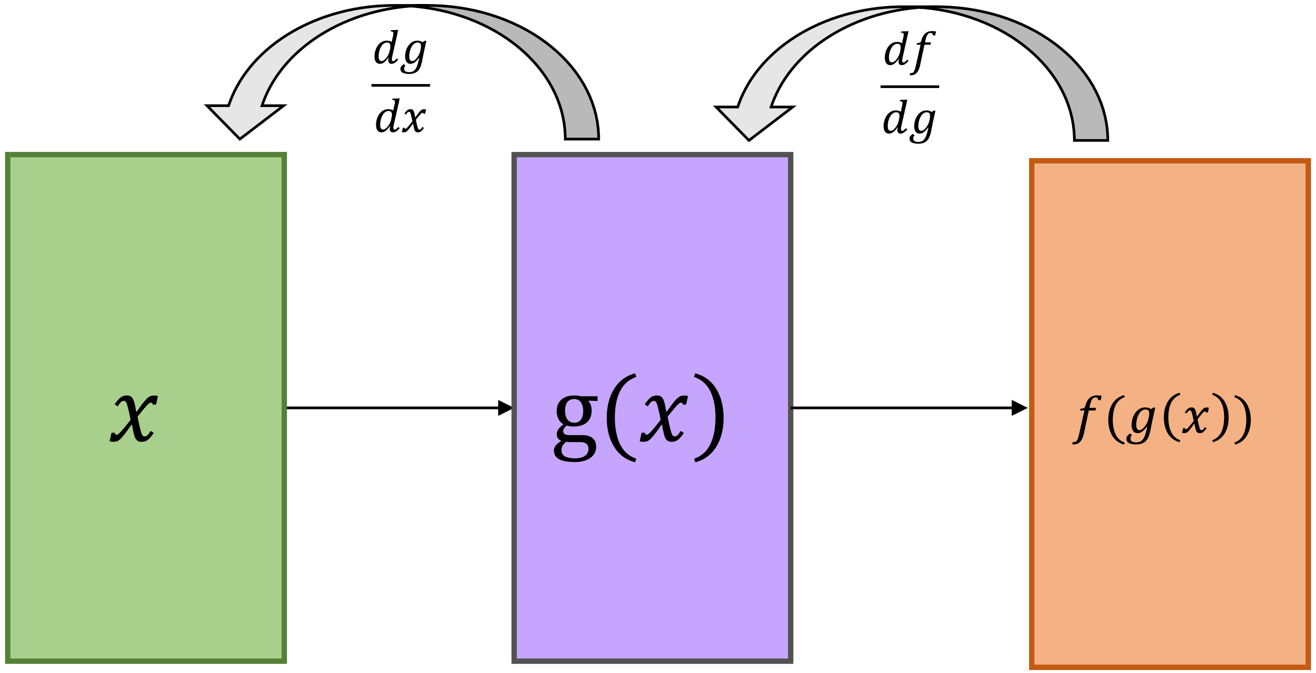
$$F(x) = f(g(x))$$

$$F(x) = (2x^2 + 1)^2$$

The Chain Rule (for Differentiation)

- Given arbitrary function: $f\big(g(x)\big) \Rightarrow \frac{df}{dx} = \frac{df}{dg} \cdot \frac{dg}{dx}$

Each layer computes the gradients with respect to it's variables and passes the result backwards



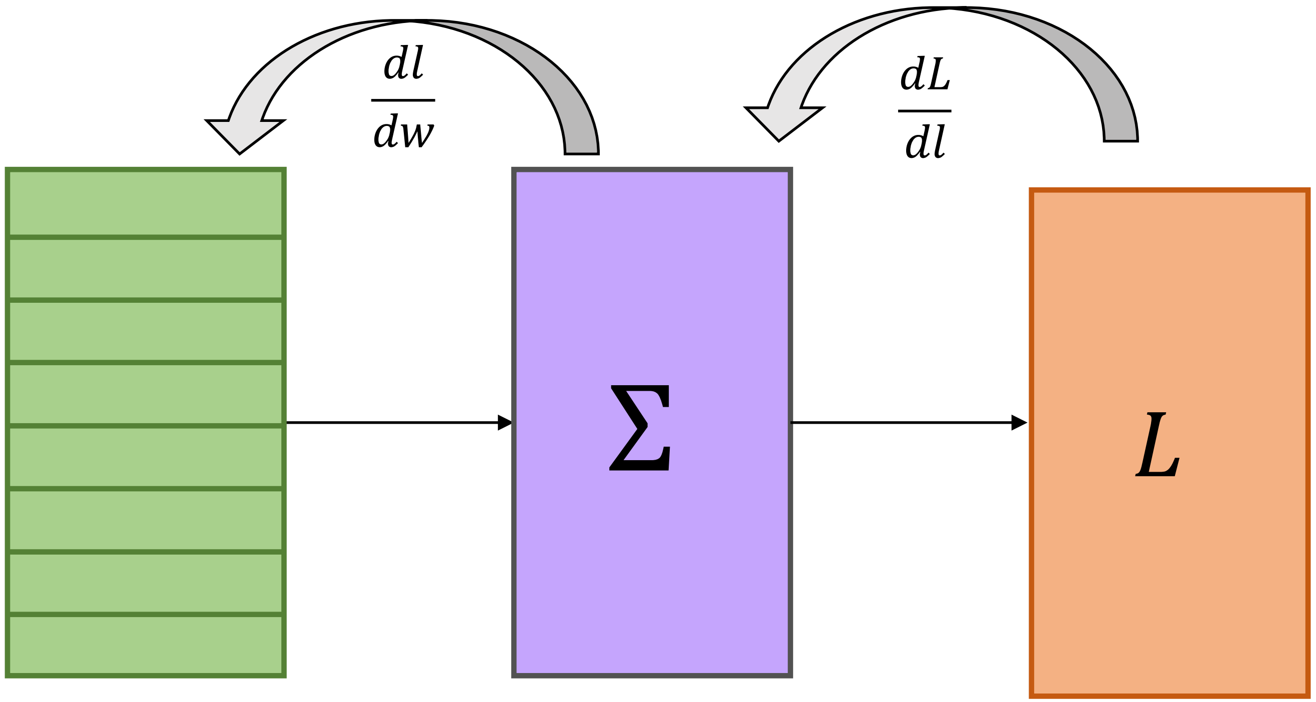
Backpropagation

(or backward pass)

The Chain Rule in Our Network

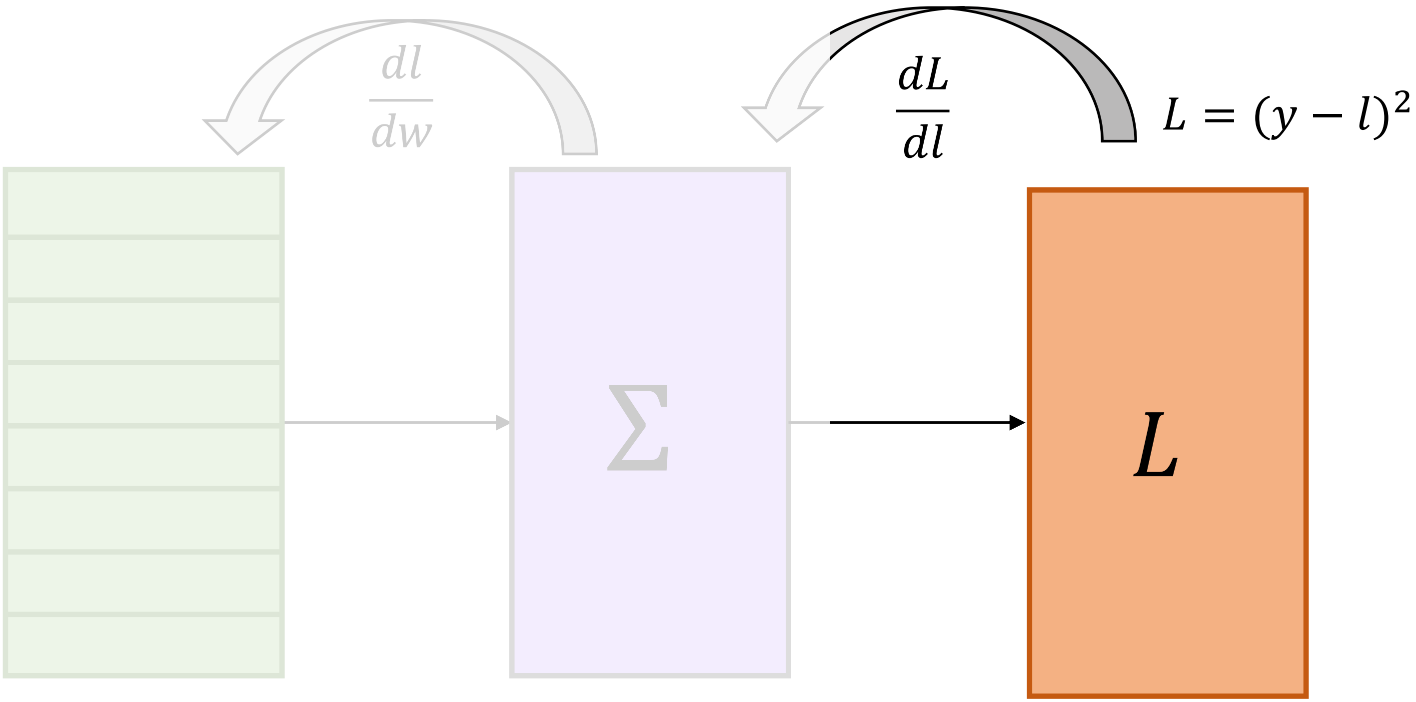
Remember – We calculate the gradient of L with respect to the parameters for learning them using gradient descent!

- Here's our function: $L(l(w)) \Rightarrow \frac{dL}{dw} = \frac{dL}{dl} \cdot \frac{dl}{dw}$



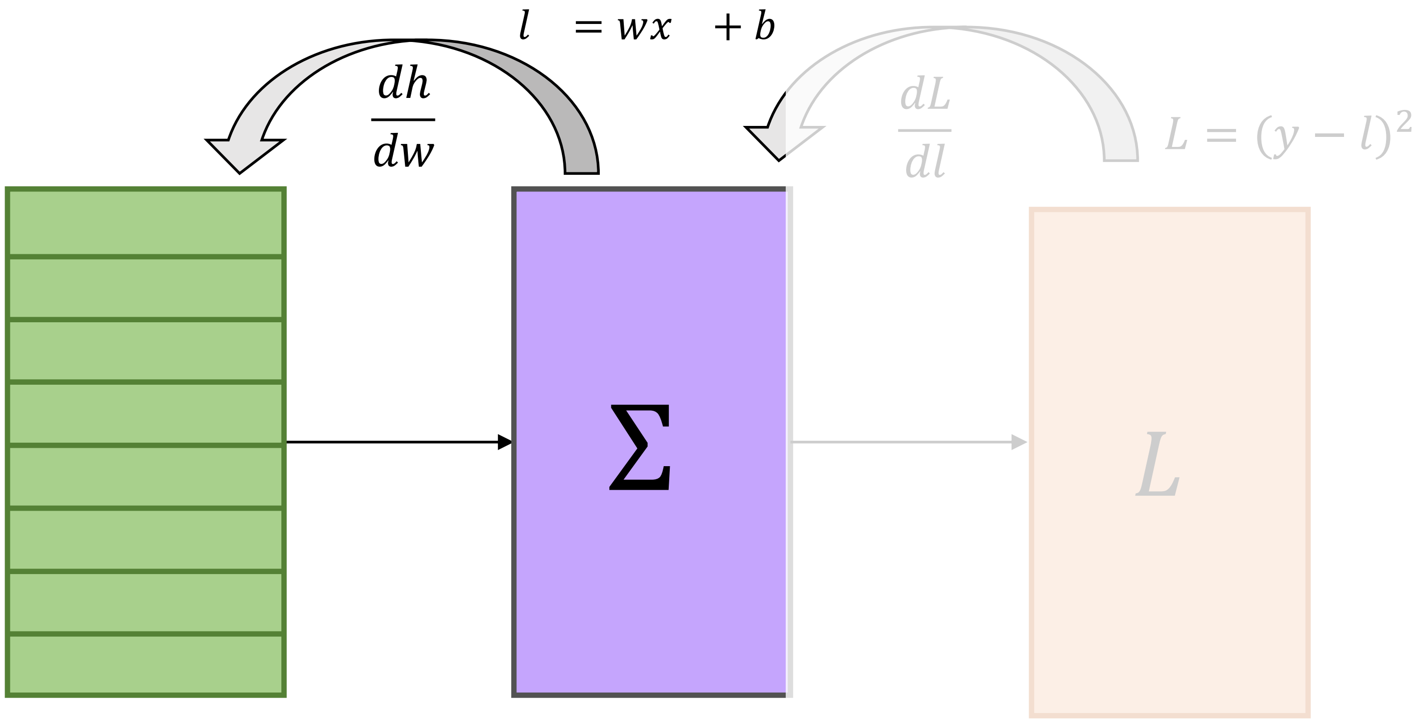
Derivative of loss layer

- $\frac{dL}{dl} = \frac{d(y-l)^2}{dl}$



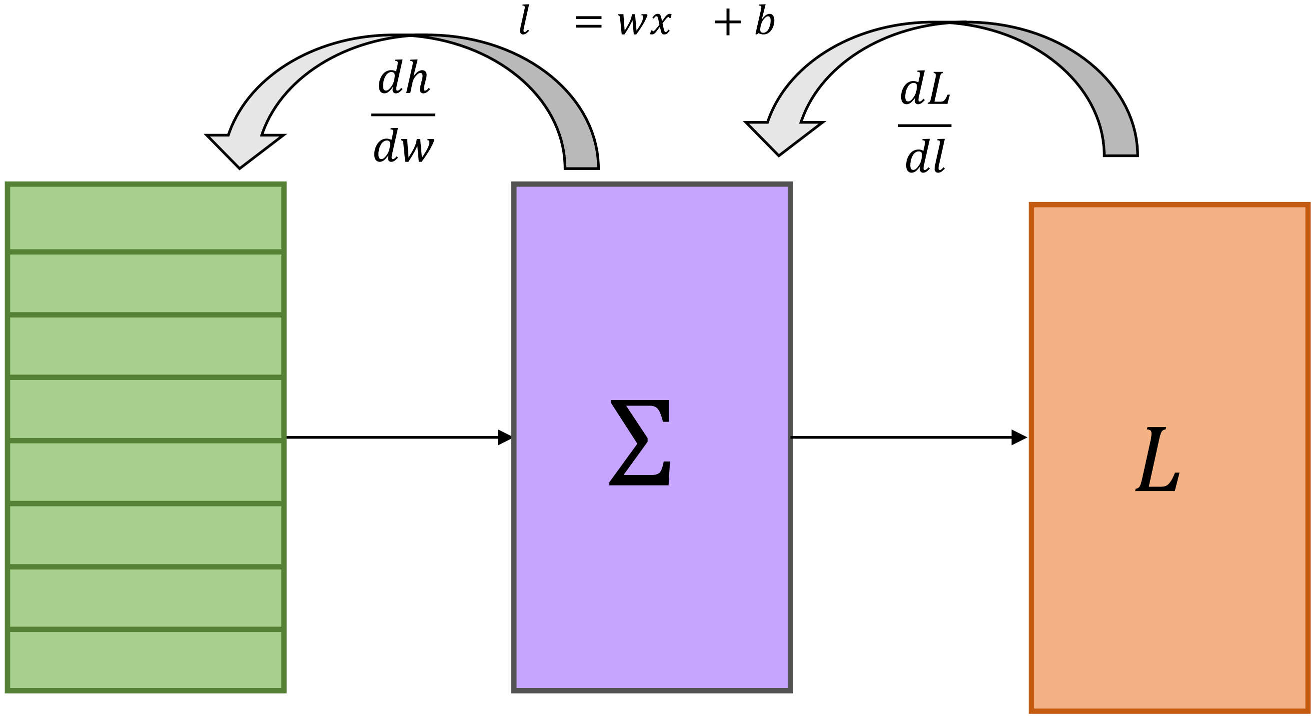
Derivative of linear layer

- $\frac{dl}{dw} = \frac{d(wx+b)}{dw}$



Putting it all together

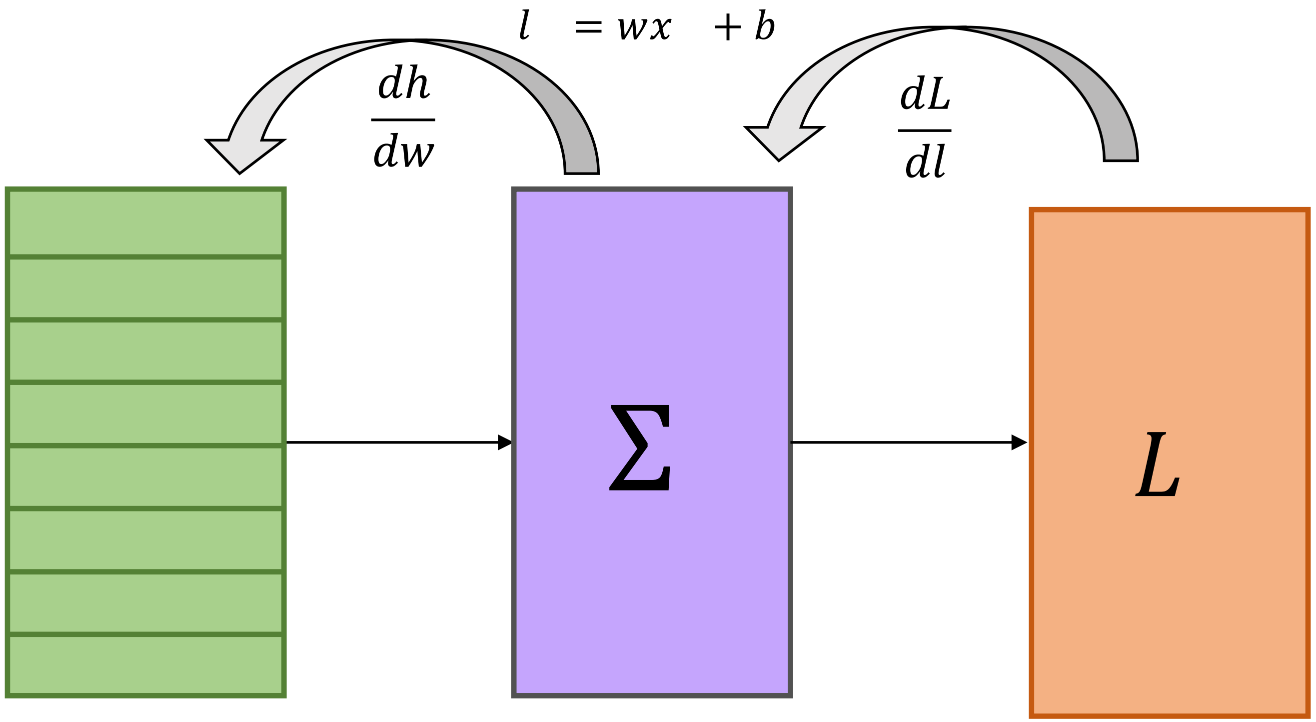
- $\frac{dL}{dw} = \frac{dL}{dl} \cdot \frac{dl}{dw} =$



Putting it all together

Have we seen this before?

- $\frac{dL}{dw} = \frac{dL}{dl} \cdot \frac{dl}{dw} = -2(\mathrm{y} - \mathrm{l}).\mathrm{x} = -2\mathrm{x}(\mathrm{y} - \mathrm{wx} - \mathrm{b}) = 2\mathrm{x}(\mathrm{wx} + \mathrm{b} - \mathrm{y})$



Gradient Descent of MSE (1 sample)

$$\Delta w = -\alpha \cdot \frac{\partial L}{\partial w}$$

$$L = (y - \hat{y})^2$$

$$= (y - f(x))^2$$

$$= y^2 + f(x)^2 - 2yf(x)$$

$$= y^2 + (wx + b)^2 - 2y(wx + b)$$

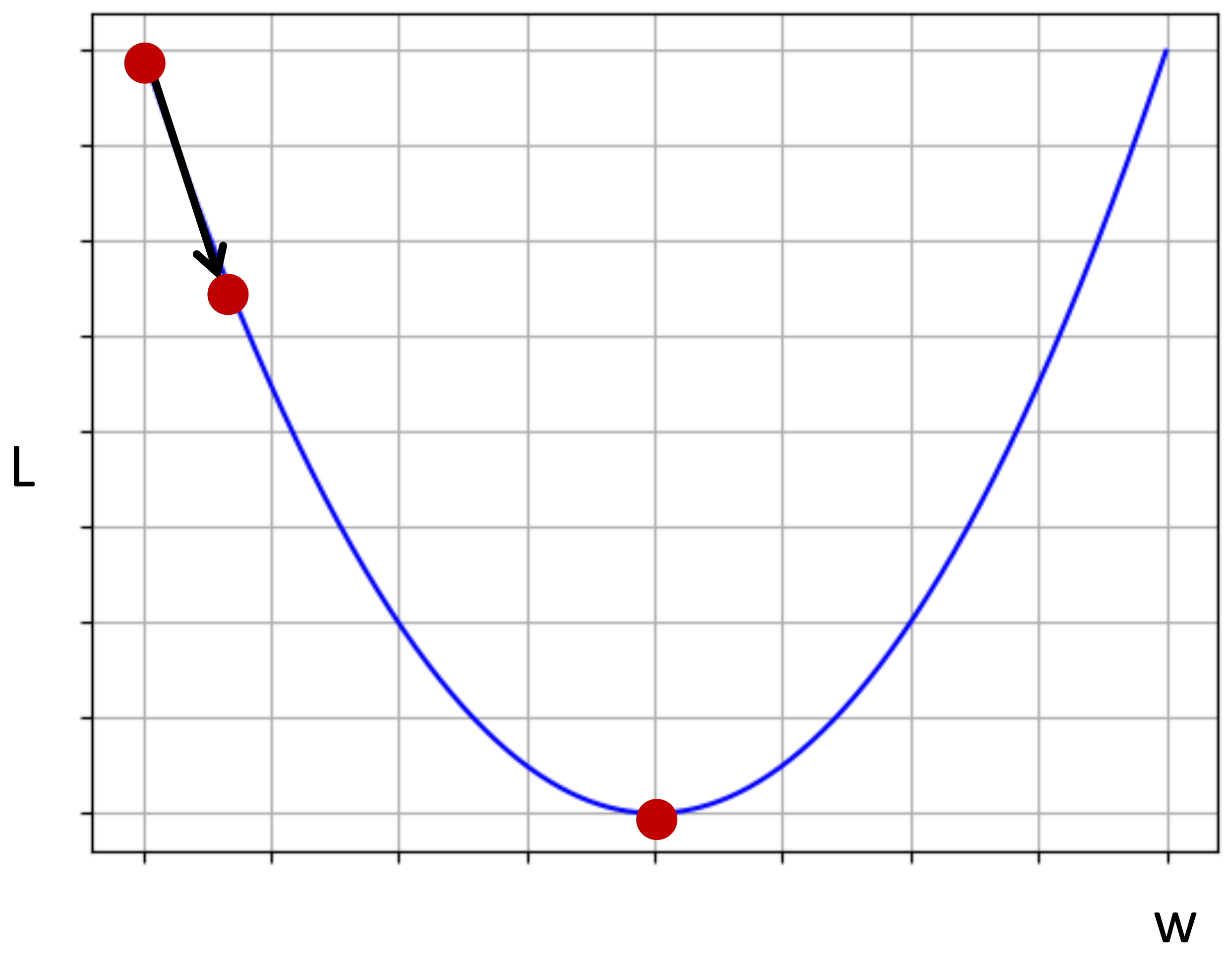
$$= y^2 + (wx + b)^2 - 2y(wx + b)$$

$$= y^2 + (wx + b)^2 - 2y(wx + b)$$

$$= y^2 + w^2x^2 + b^2 + 2wxb - 2ywx - 2yb$$

$$\frac{\partial L}{\partial w} = 2wx^2 + 2xb - 2yx$$

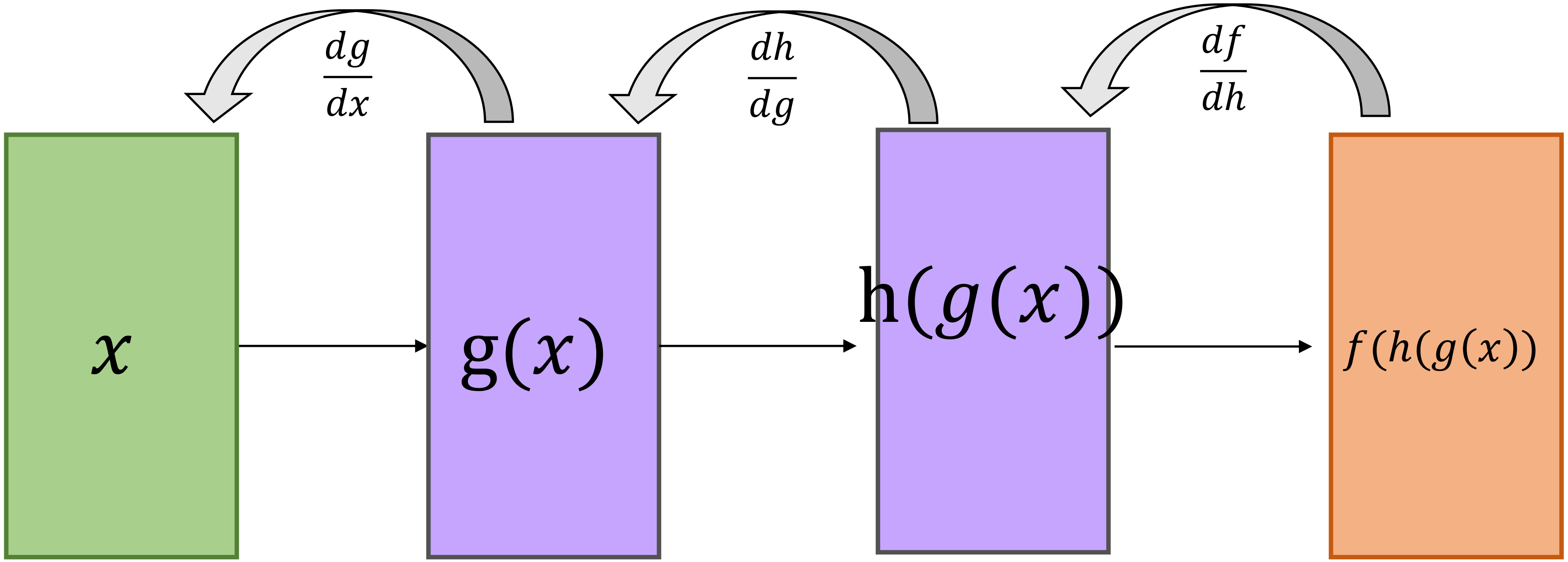
$$\boxed{\frac{\partial L}{\partial w} = 2x(wx + b - y)}$$



Adding more layers!

Can we add any function?

- $f\big(h(g(x))\big) \Rightarrow \frac{df}{dx} = \frac{df}{dh} \cdot \frac{dh}{dg} \cdot \frac{dg}{dx}$



Any questions?

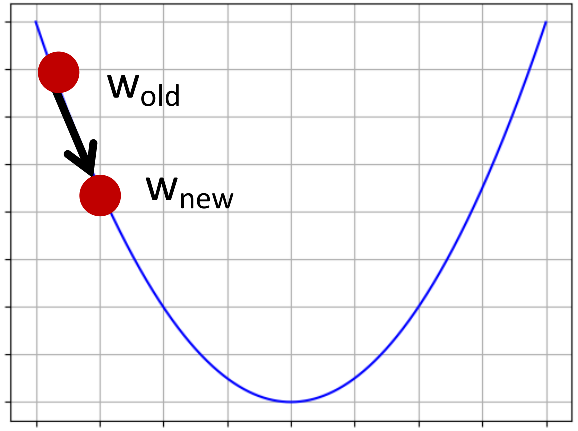
Few more important points: Backpropagation

- The process of calculating gradients of functions via chain rule in a neural network
- Is a part of and **NOT the whole learning algorithm**
- Can be calculated with respect to any variable of choice
- For learning in neural networks we calculate gradients with respect to the weights

Recap

Optimization

- Calculating gradients
- Gradient Descent for MSE Loss
- Convex and Non convex functions



Building a neural network

- Simple model with linear layer
- Adding loss layer (regression)
- Chain rule to calculate gradients (Backpropagation)