Linear Regression and Intro to Neural Networks

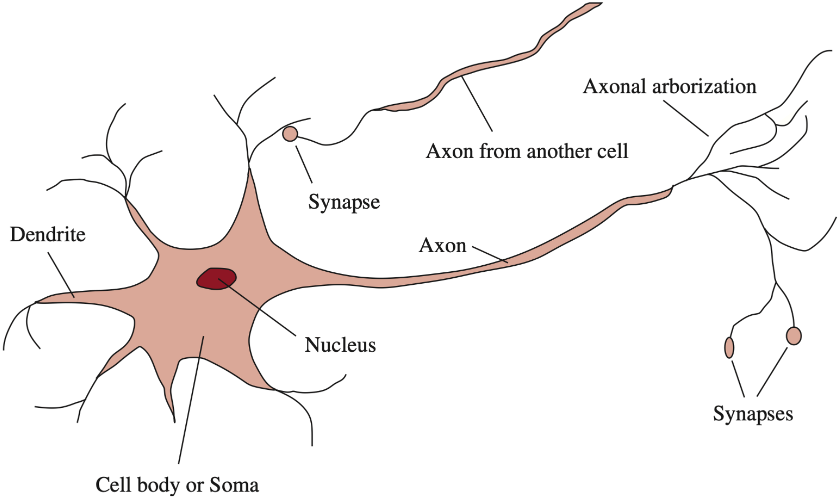
COM 214: Introduction to Artificial Intelligence

Sasha Fedchin[1,2]

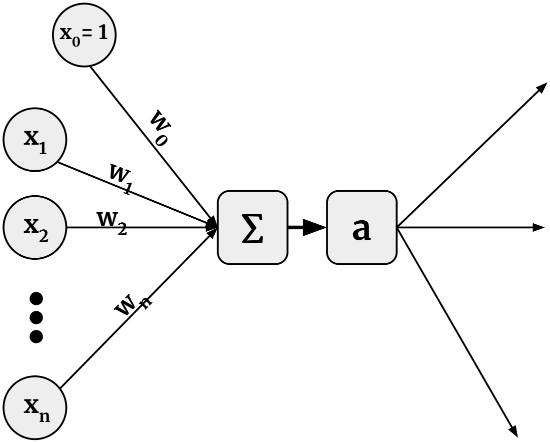
[1]Department of Software Engineering
American University of Central Asia

[2]Department of Computer Science
Tufts University

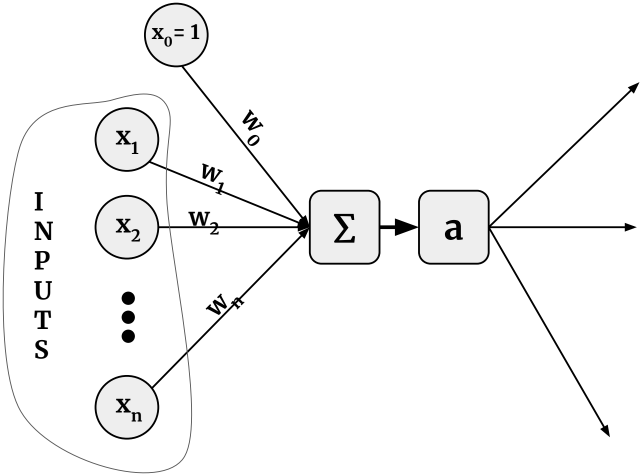
A biological neuron



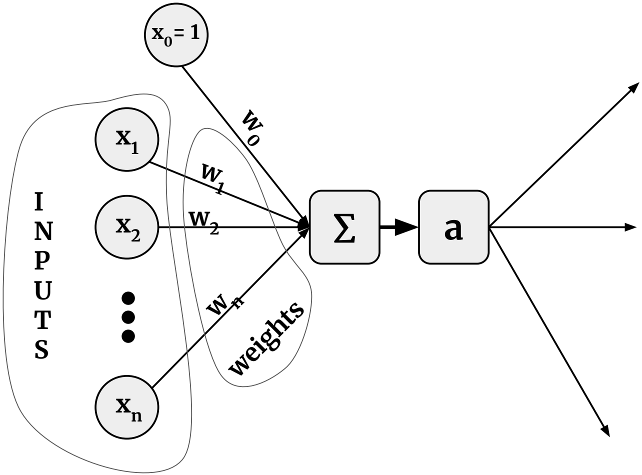
An Artificial Neuron: Close-up View



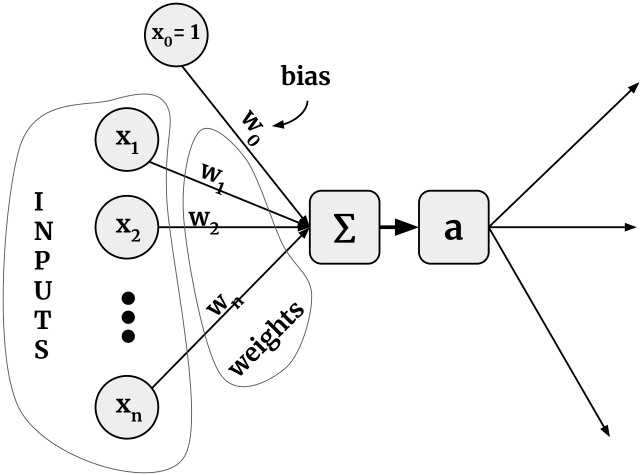
An Artificial Neuron: Close-up View



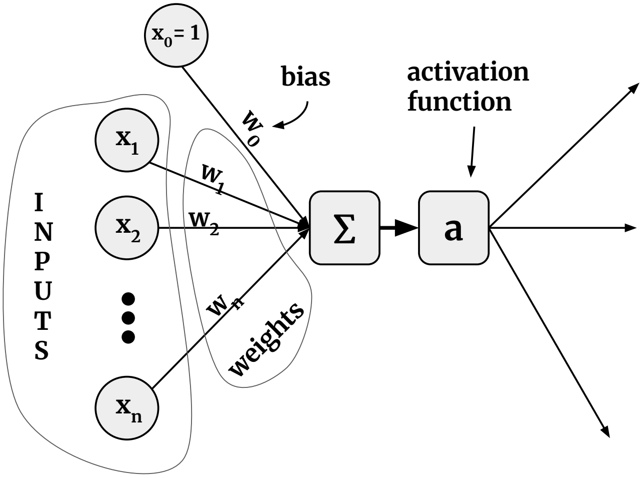
An Artificial Neuron: Close-up View



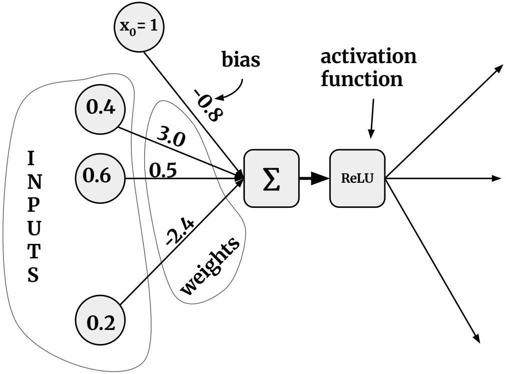
An Artificial Neuron: Close-up View



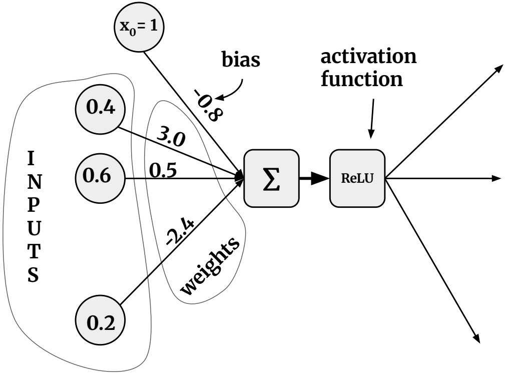
An Artificial Neuron: Close-up View



Artificial Neuron: Example



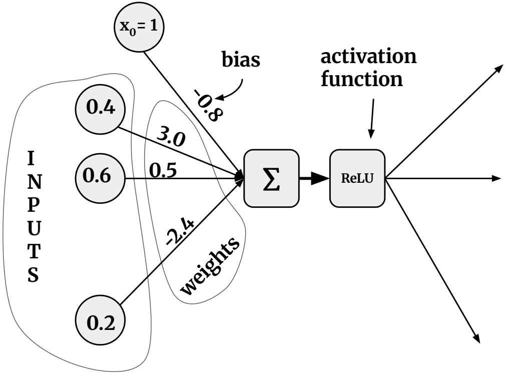
Artificial Neuron: Example



- ReLU = ``rectified linear unit"

$$\mathrm{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

Artificial Neuron: Example

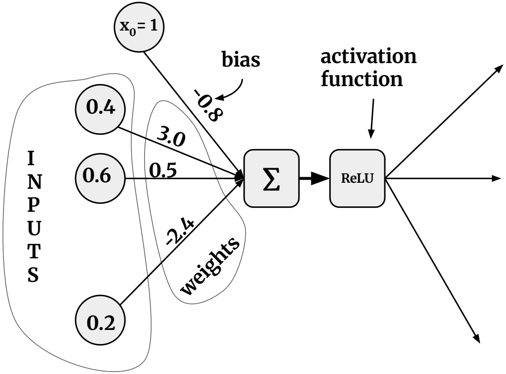


- ReLU = ``rectified linear unit"

$$\mathrm{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

- $f(\langle x_1, x_2, x_3 \rangle) =$
$\mathrm{ReLU}(\sum_{0 \leq i \leq 3} x_i \times w_i) =$

Artificial Neuron: Example

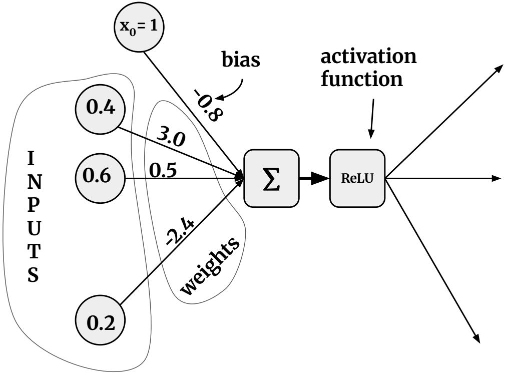


- ReLU = ``rectified linear unit"

$$\mathrm{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

- $f(\langle x_1, x_2, x_3 \rangle) =$
$\mathrm{ReLU}(\sum_{0 \leq i \leq 3} x_i \times w_i) =$
$\mathrm{ReLU}(1 \times -0.8 + 0.4 \times 3.0 +$
$+ 0.6 \times 0.5 + 0.2 \times -2.4) =$

Artificial Neuron: Example

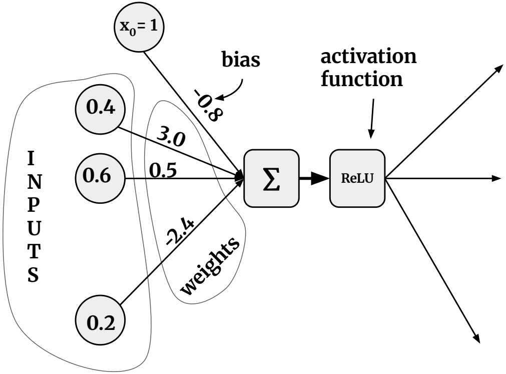


- ReLU = ``rectified linear unit"

$$\mathrm{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

- $f(\langle x_1, x_2, x_3 \rangle) =$
$\mathrm{ReLU}(\sum_{0 \leq i \leq 3} x_i \times w_i) =$
$\mathrm{ReLU}(1 \times -0.8 + 0.4 \times 3.0 +$
$+ 0.6 \times 0.5 + 0.2 \times -2.4) =$
$\mathrm{ReLU}(-0.8 + 1.2 + 0.3 - 0.48) =$

Artificial Neuron: Example

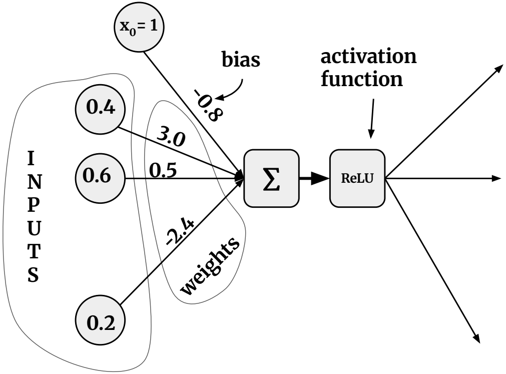


- ReLU = ``rectified linear unit"

$$\text{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

- $f(\langle x_1, x_2, x_3 \rangle) =$
  $\text{ReLU}(\sum_{0 \leq i \leq 3} x_i \times w_i) =$
  $\text{ReLU}(1 \times -0.8 + 0.4 \times 3.0 +$
  $+ 0.6 \times 0.5 + 0.2 \times -2.4) =$
  $\text{ReLU}(-0.8 + 1.2 + 0.3 - 0.48) =$
  $\text{ReLU}(0.22) =$

Artificial Neuron: Example

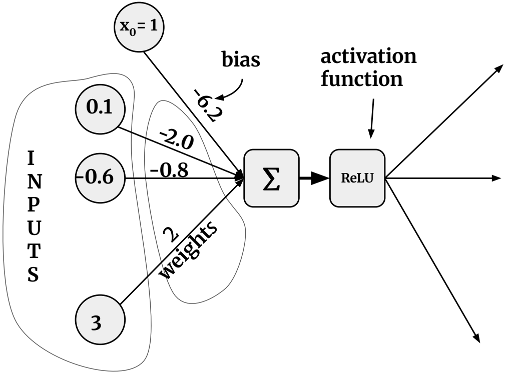


- ReLU = ``rectified linear unit"

$$\mathrm{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

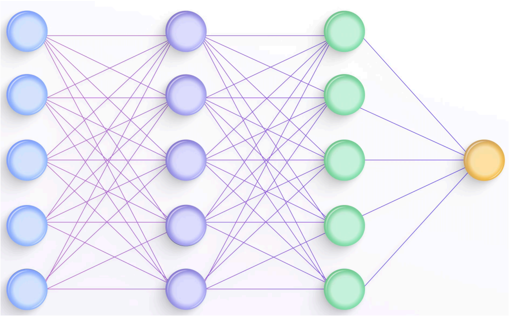
- $f(\langle x_1, x_2, x_3 \rangle) =$
$\mathrm{ReLU}(\sum_{0 \leq i \leq 3} x_i \times w_i) =$
$\mathrm{ReLU}(1 \times -0.8 + 0.4 \times 3.0 +$
$+ 0.6 \times 0.5 + 0.2 \times -2.4) =$
$\mathrm{ReLU}(-0.8 + 1.2 + 0.3 - 0.48) =$
$\mathrm{ReLU}(0.22) =$
0.22

Your Turn!

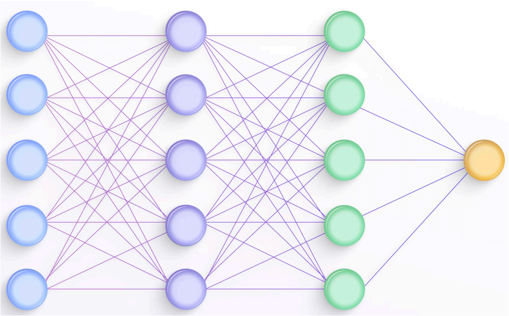


$x_0 = 1$
bias
activation function
-6.2
0.1
-2.0
-0.6
-0.8
3
2
weights
INPUTS
Σ
ReLU

How do Neural Networks Differ?

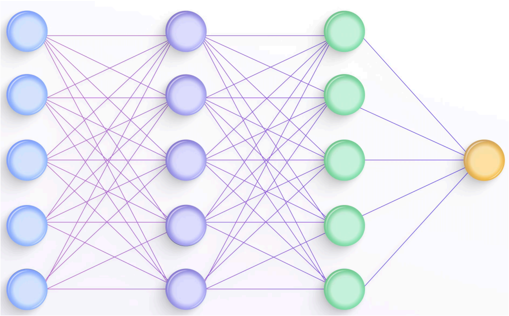


How do Neural Networks Differ?



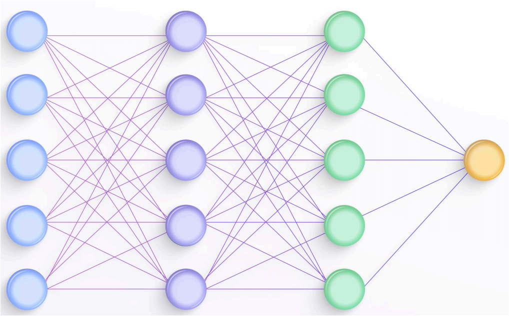
- Architecture: how the nodes are arranged

How do Neural Networks Differ?



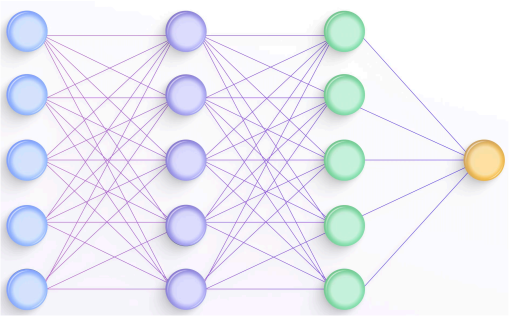
- Architecture: how the nodes are arranged
- Activation function: how the output of a neuron is computed

How do Neural Networks Differ?



- Architecture: how the nodes are arranged
- Activation function: how the output of a neuron is computed
- Input/Output representation

How do Neural Networks Differ?



- Architecture: how the nodes are arranged
- Activation function: how the output of a neuron is computed
- Input/Output representation
- Training methods

Biological vs Artificial Neural Networks

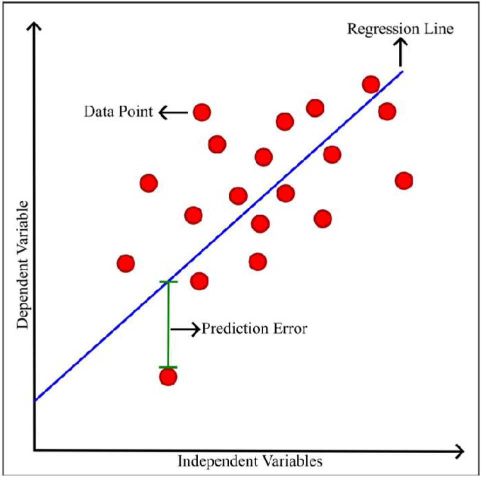
Biological:

- Neuron: an excitable cell.
- A neuron sends an electrochemical pulse when a sufficient voltage change occurs.
- Biological Neural Networks: collection of neurons along some pathway through the brain.

Artificial:

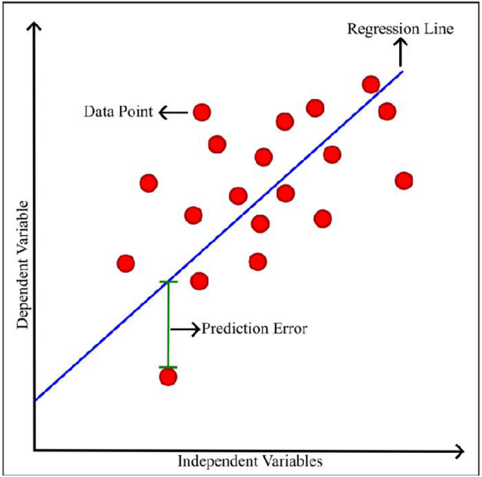
- Neuron: node in a graph.
- Weight: multiplier on each edge.
- Activation Function: nonlinear threshold function which allows neuron to ``fire" when the input value is sufficiently high.
- Artificial Neural Network: collection of neurons into a DAG, which define some differentiable function.

Linear Regression



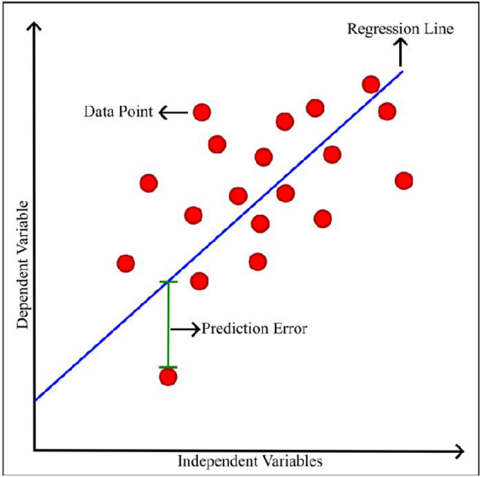
- Section 19.6.1 in your textbook

Linear Regression



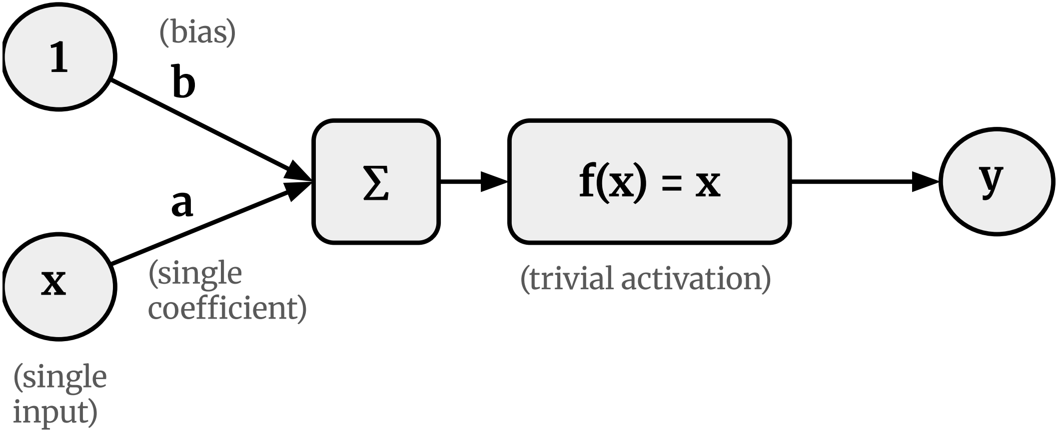
- Section 19.6.1 in your textbook
- With one independent variable, we want to learn a line $f(x) = ax + b$.

Linear Regression

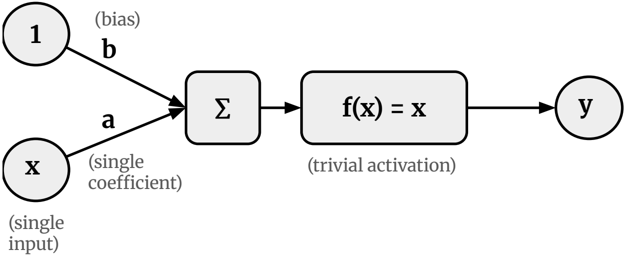


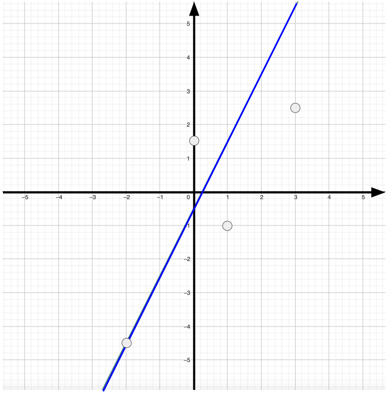
- Section 19.6.1 in your textbook
- With one independent variable, we want to learn a line $f(x) = ax + b$.
- We need to quantify the error ------ mean squared error (MSE) $= \frac{1}{n}\sum_{1\leq i\leq n}(y_i - f(x_i))^2$.

Linear Regression as a Neural Network



Linear Regression as a Neural Network

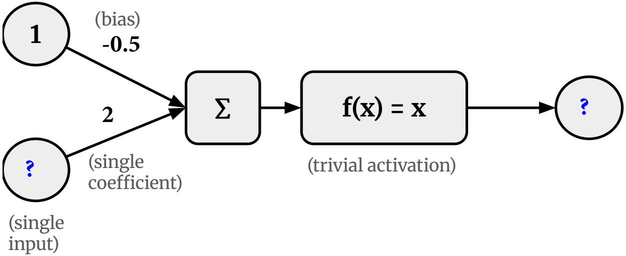


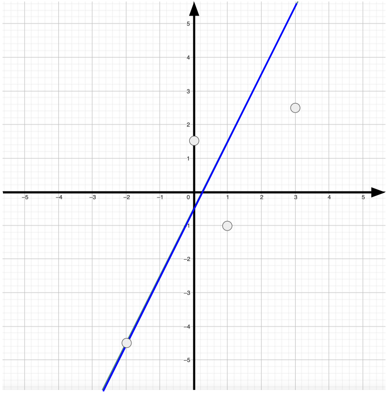


Squared Error $= \quad + \quad + \quad + \quad =$

Mean Squared Error $= \frac{?}{4}$

Linear Regression as a Neural Network

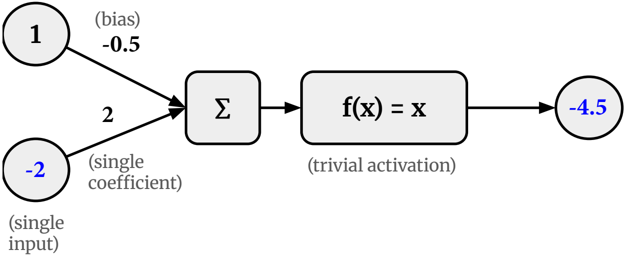


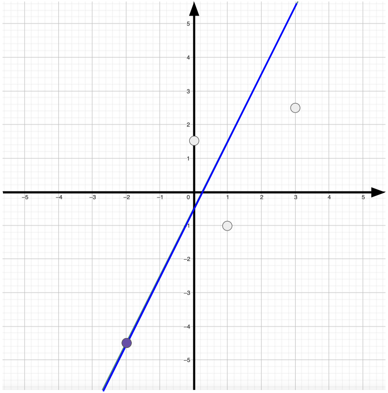


Squared Error $= \quad + \quad + \quad + \quad =$

Mean Squared Error $= \frac{?}{4}$

Linear Regression as a Neural Network

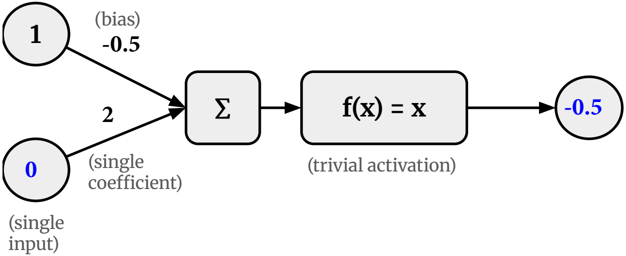


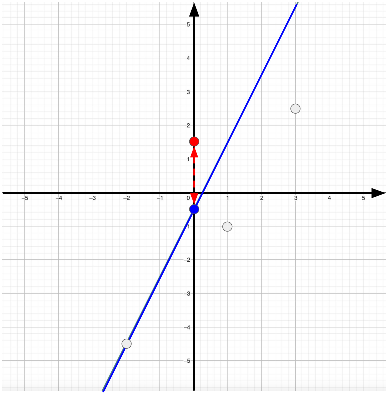


Squared Error $= 0 + \quad + \quad + \quad =$

Mean Squared Error $= \frac{?}{4}$

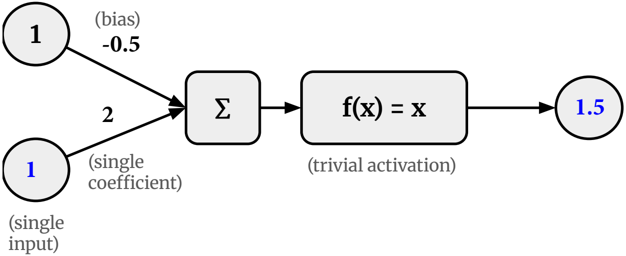
Linear Regression as a Neural Network

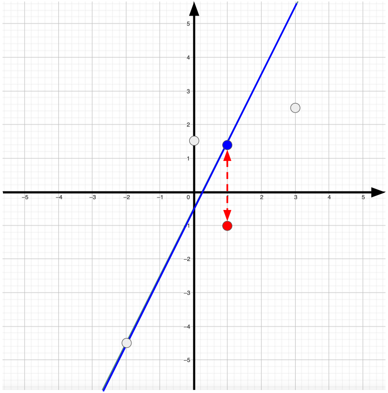




$$\text{Squared Error} = 0 + 2^2 + \qquad + \qquad =$$
$$\text{Mean Squared Error} = \frac{?}{4}$$

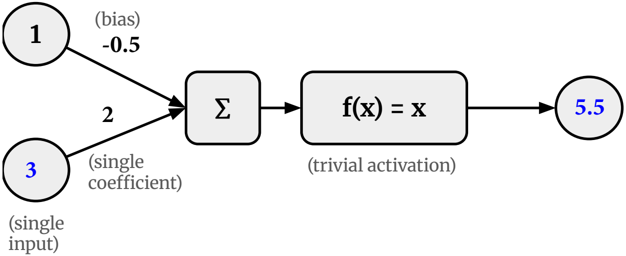
Linear Regression as a Neural Network

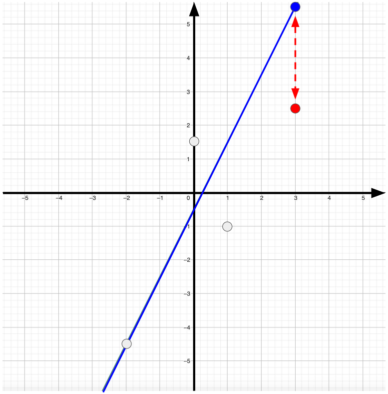




Squared Error $= 0 + 2^2 + 2.5^2 + \quad =$
Mean Squared Error $= \frac{?}{4}$

Linear Regression as a Neural Network

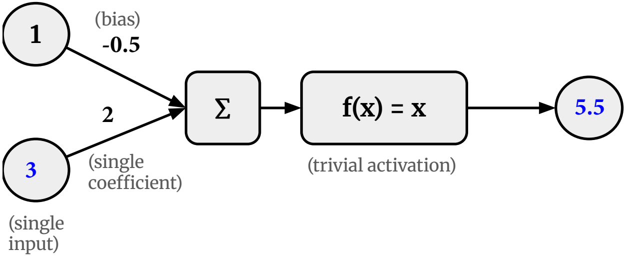


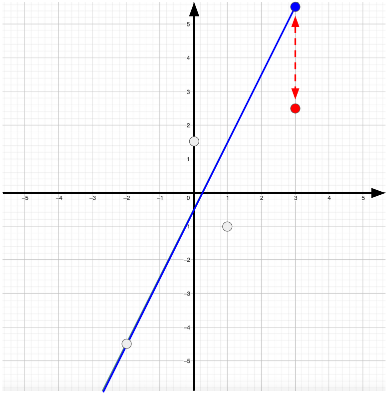


Squared Error $= 0 + 2^2 + 2.5^2 + 3^2 =$

Mean Squared Error $= \frac{?}{4}$

Linear Regression as a Neural Network





Squared Error $= 0 + 2^2 + 2.5^2 + 3^2 = 19.25$
Mean Squared Error $= \frac{19.25}{4} \approx 4.81$

Calculus Refresher: Useful Differetiation Rules

Calculus Refresher: Useful Differetiation Rules

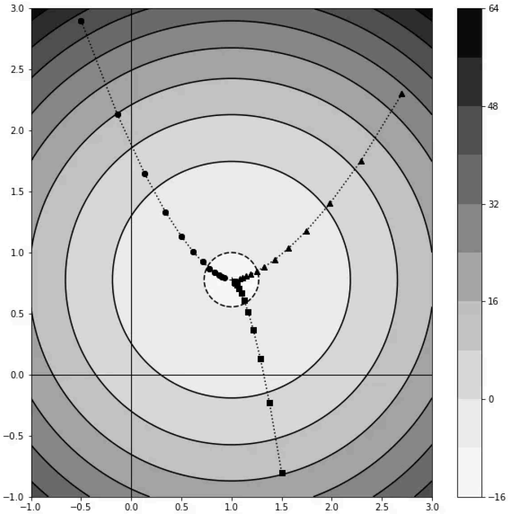
Name	Function	Derivative
Constant	c	0
Constant Multiplier	$cf(x)$	$cf'(x)$
Sum	$f(x)+g(x)$	$f'(x)+g'(x)$
Power	x^n	nx^{n-1}
Exponential (base e)	e^x	e^x
Exponential (base a)	a^x	$a^x ln(a)$
Logarithm	$ln(x)$	$1/x$
Product	$f(x)g(x)$	$f'(x)g(x)+f(x)g'(x)$
Quotient	$\frac{f(x)}{g(x)}$	$\frac{f'(x)g(x)-f(x)g'(x)}{[g(x)]^2}$
Chain	$f(g(x))$	$f'(g(x))\,g'(x)$

Calculus Refresher: Differentiation Example

Function	Derivative
c	0
$cf(x)$	$cf'(x)$
$f(x)+g(x)$	$f'(x)+g'(x)$
x^n	nx^{n-1}
e^x	e^x
a^x	$a^x ln(a)$
$ln(x)$	$1/x$
$f(x)g(x)$	$f'(x)g(x)+f(x)g'(x)$
$\frac{f(x)}{g(x)}$	$\frac{f'(x)g(x)-f(x)g'(x)}{[g(x)]^2}$
$f(g(x))$	$f'(g(x))\,g'(x)$

$$
\begin{aligned}
&\frac{d}{dx}\frac{1}{1+e^{-x}} = \\
&\frac{-1}{(1+e^{-x})^2}\frac{d}{dx}(1+e^{-x}) = \\
&\frac{-1}{(1+e^{-x})^2}\frac{d}{dx}(e^{-x}) = \\
&\frac{-1}{(1+e^{-x})^2}e^{-x}\frac{d}{dx}(-x) = \\
&\frac{e^{-x}}{(1+e^{-x})^2}
\end{aligned}
$$

Gradient Descent



- The gradient of a function wrt weights always points in the direction of maximal increase.
- To decrease the error, we need to subtract the gradient.

Gradient for Squared Error

We want to compute the gradient of the squared error

$$SE(a, b) = (y - f(x))^2 \quad \text{where} \quad f(x) = ax + b$$

Gradient for Squared Error

We want to compute the gradient of the squared error

$$SE(a, b) = (y - f(x))^2 \quad \text{where} \quad f(x) = ax + b$$

This means we want:

$$\nabla SE = \left\langle \frac{\partial SE}{\partial a}, \frac{\partial SE}{\partial b} \right\rangle$$

Gradient for Squared Error

We want to compute the gradient of the squared error

$$SE(a, b) = (y - f(x))^2 \quad \text{where} \quad f(x) = ax + b$$

This means we want:

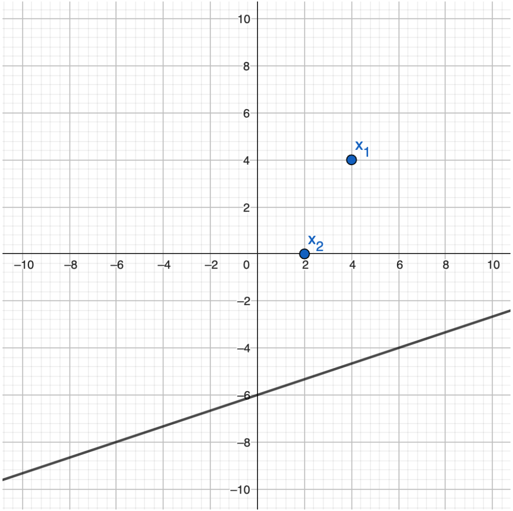
$$\nabla SE = \left\langle \frac{\partial SE}{\partial a}, \frac{\partial SE}{\partial b} \right\rangle$$

Apply the chain rule to each partial derivative:

$$\frac{\partial SE}{\partial a} = \frac{\partial}{\partial a}(y - ax - b)^2 = 2(y - ax - b)(-x) = -2x(y - f(x))$$

$$\frac{\partial SE}{\partial b} = \frac{\partial}{\partial b}(y - ax - b)^2 = 2(y - ax - b)(-1) = -2(y - f(x))$$

Gradient Descent: Example



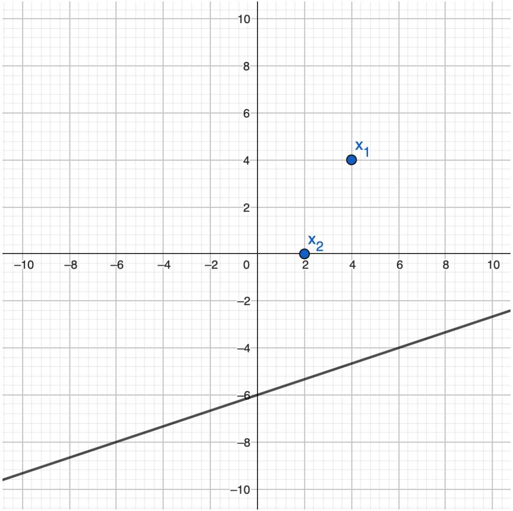
Initially we have:

$x_1 = 2,\ x_2 = 4,$

$y_1 = 0,\ y_2 = 4,$

$b = \quad ,\ a =$

Gradient Descent: Example



Initially we have:

$x_1 = 2,\ x_2 = 4,$

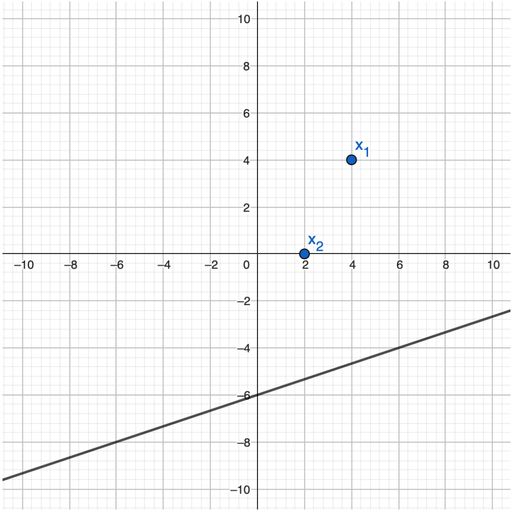
$y_1 = 0,\ y_2 = 4,$

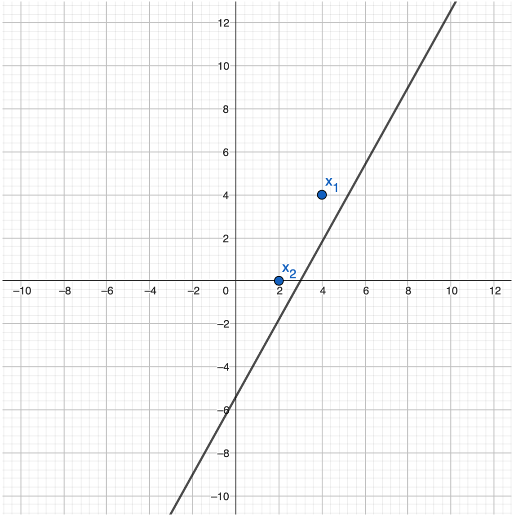
$b = -6,\ a = \frac{1}{3}$

Gradient Descent: Example

- Initially we have: $x_1 = 2$, $x_2 = 4$, $y_1 = 0$, $y_2 = 4$, $b =-6$, $a =\frac{1}{3}$
- So, our predictions, are:
  $f(x_1) = ax_1 + b = \frac{1}{3}2 - 6 = -\frac{16}{3}$
  $f(x_2) = ax_2 + b = \frac{1}{3}4 - 6 = -\frac{14}{3}$
- $\nabla SE = \left\langle \frac{\partial SE}{\partial a}, \frac{\partial SE}{\partial b} \right\rangle =$
  $\left\langle \frac{1}{2} \sum_i -2x_i(y_i - f(x_i)), \frac{1}{2} \sum_i -2(y_i - f(x_i)) \right\rangle =$
  $\langle \frac{1}{2}[-4(0 + \frac{16}{3}) - 8(4 + \frac{14}{3})], \frac{1}{2}[-2(0 + \frac{16}{3}) - 2(4 + \frac{14}{3})]\rangle =$
  $\langle -\frac{32}{3} - \frac{104}{3}, -\frac{16}{3} - \frac{26}{3} \rangle = \langle -\frac{136}{3}, -\frac{42}{3} \rangle$
- If we shift our weights by $\frac{1}{30}\nabla SE$, then $a = \frac{1}{3} + \frac{136}{90} \approx \frac{11}{6}$, and $b = -6 + \frac{42}{90} \approx -\frac{11}{2}$

Gradient Descent: Before vs After

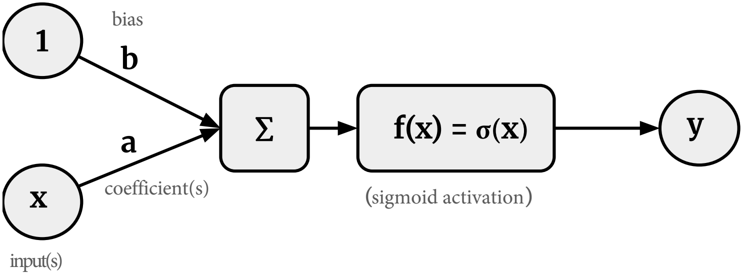




Logistic Regression

Same setup as Linear Regression but:

- Uses sigmoid activation function, $\sigma(z) = \frac{1}{1+e^{-z}}$
- Uses cross-entropy loss: $Loss(y, \hat{y}) = -y ln(\hat{y}) - (1-y) ln(1-\hat{y})$, where $y \in \{0, 1\}$ is the true label and $\hat{y} = \sigma(z) \in (0, 1)$ is the prediction



Logistic Regression

Same setup as Linear Regression but:

- Uses sigmoid activation function, $\sigma(z) = \frac{1}{1+e^{-z}}$
- Uses cross-entropy loss: $Loss(y, \hat{y}) = -y ln(\hat{y}) - (1 - y) ln(1 - \hat{y})$, where $y \in \{0, 1\}$ is the true label and $\hat{y} = \sigma(z) \in (0, 1)$ is the prediction

$\frac{d}{d\hat{y}} Loss(y, \hat{y}) = -\frac{y}{\hat{y}} + \frac{1-y}{1-\hat{y}}$

Logistic Regression

Same setup as Linear Regression but:

- Uses sigmoid activation function, $\sigma(z) = \frac{1}{1+e^{-z}}$
- Uses cross-entropy loss: $Loss(y, \hat{y}) = -y ln(\hat{y}) - (1-y) ln(1-\hat{y})$, where $y \in \{0,1\}$ is the true label and $\hat{y} = \sigma(z) \in (0,1)$ is the prediction

$\frac{d}{d\hat{y}} Loss(y, \hat{y}) = -\frac{y}{\hat{y}} + \frac{1-y}{1-\hat{y}}$

$\frac{d}{dz}\sigma(z) = \sigma(z)(1-\sigma(z))$

Logistic Regression

Same setup as Linear Regression but:

- Uses sigmoid activation function, $\sigma(z) = \frac{1}{1+e^{-z}}$
- Uses cross-entropy loss: $Loss(y, \hat{y}) = -yln(\hat{y}) - (1-y)ln(1-\hat{y})$, where $y \in \{0, 1\}$ is the true label and $\hat{y} = \sigma(z) \in (0, 1)$ is the prediction

$$\frac{d}{d\hat{y}} Loss(y, \hat{y}) = -\frac{y}{\hat{y}} + \frac{1-y}{1-\hat{y}}$$

$$\frac{d}{dz}\sigma(z) = \sigma(z)(1-\sigma(z))$$

$$\frac{d}{dz} Loss(y, \hat{y}) = \left(-\frac{y}{\sigma(z)} + \frac{1-y}{1-\sigma(z)}\right) \frac{d}{dz}\sigma(z) = y\sigma(z) - y + \sigma(z) - y\sigma(z) = \sigma(z) - y$$