

Social Data Science: Text Data and Deep Learning

Week 2

From logistic regression to neural networks

Multilayer Perceptron

The simplest Neural Network I can think of...

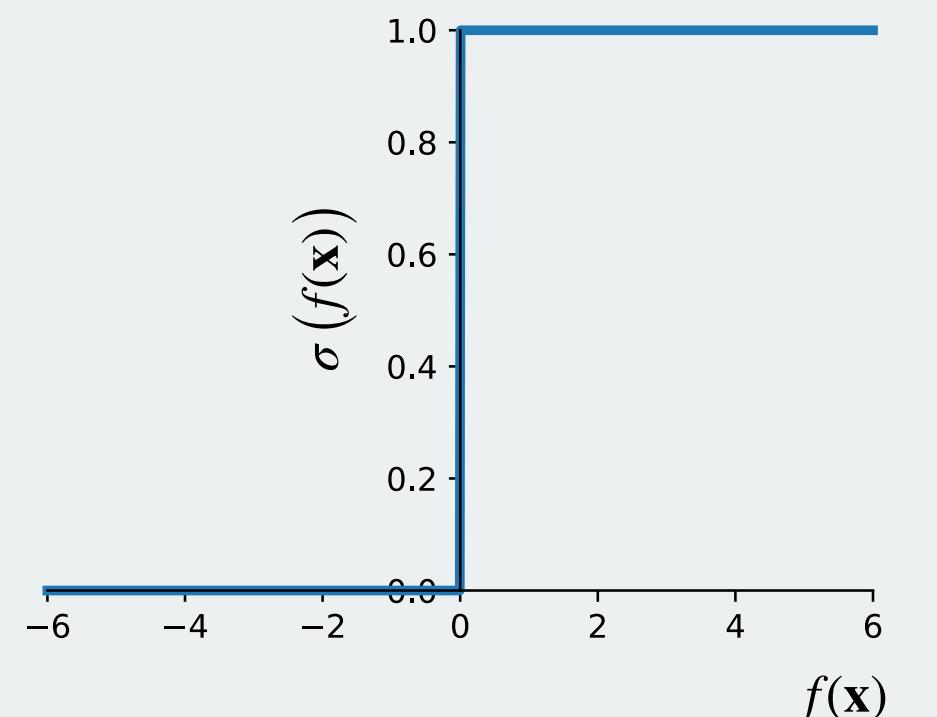
Linear regression

$$w_0 + x_0 w_1 + x_1 w_2 + x_2 w_3 = f(\mathbf{x})$$

Linear regression classifier

$$w_0 + x_0 w_1 + x_1 w_2 + x_2 w_3 = f(\mathbf{x})$$

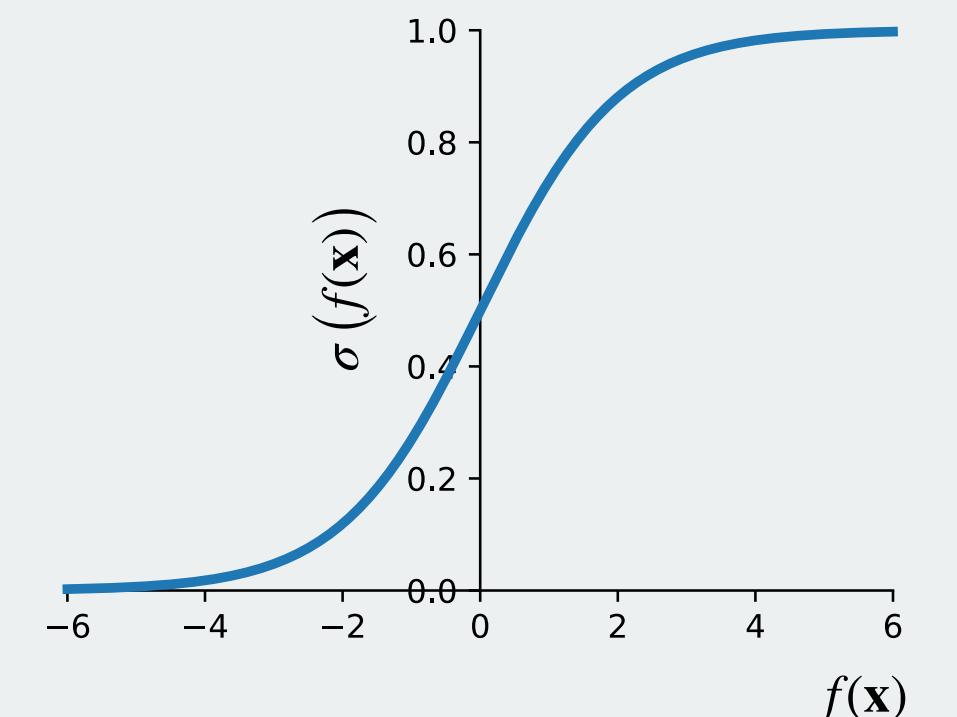
$$\sigma(f(\mathbf{x})) = \begin{cases} 1, & \text{if } f(\mathbf{x}) \geq 0 \\ 0, & \text{otherwise} \end{cases}$$



Linear regression classifier

$$w_0 + x_0 w_1 + x_1 w_2 + x_2 w_3 = f(\mathbf{x})$$

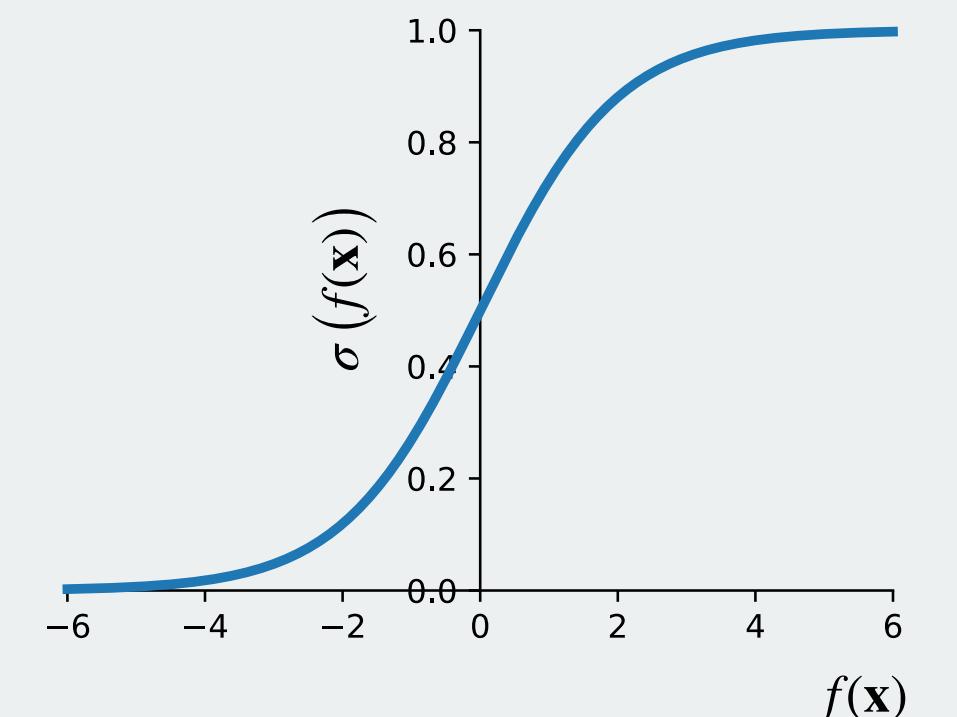
$$\sigma(f(\mathbf{x})) = \frac{1}{1 + \exp(-f(\mathbf{x}))}$$



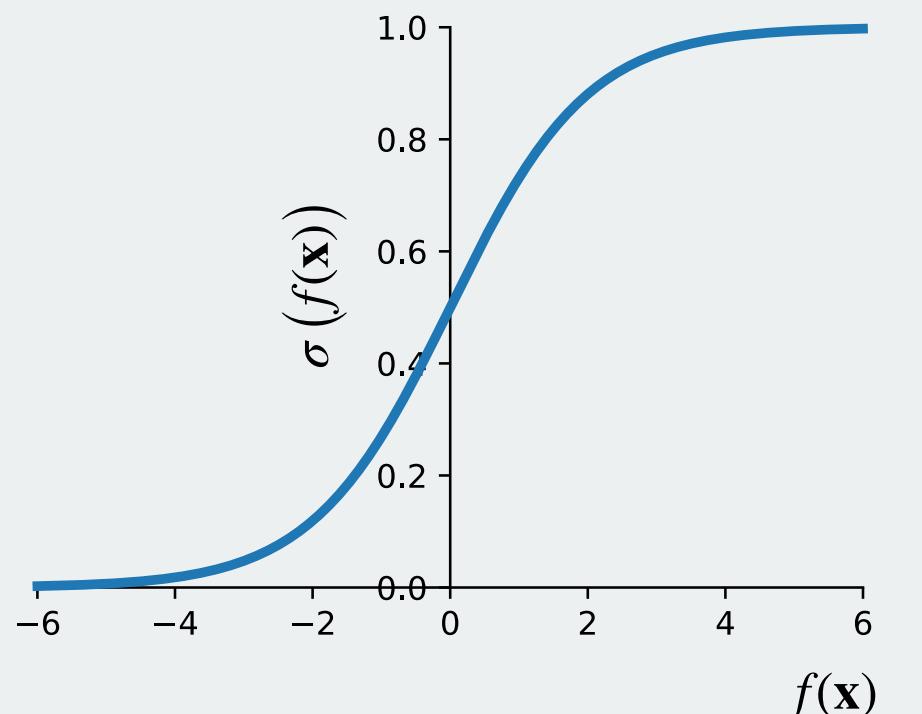
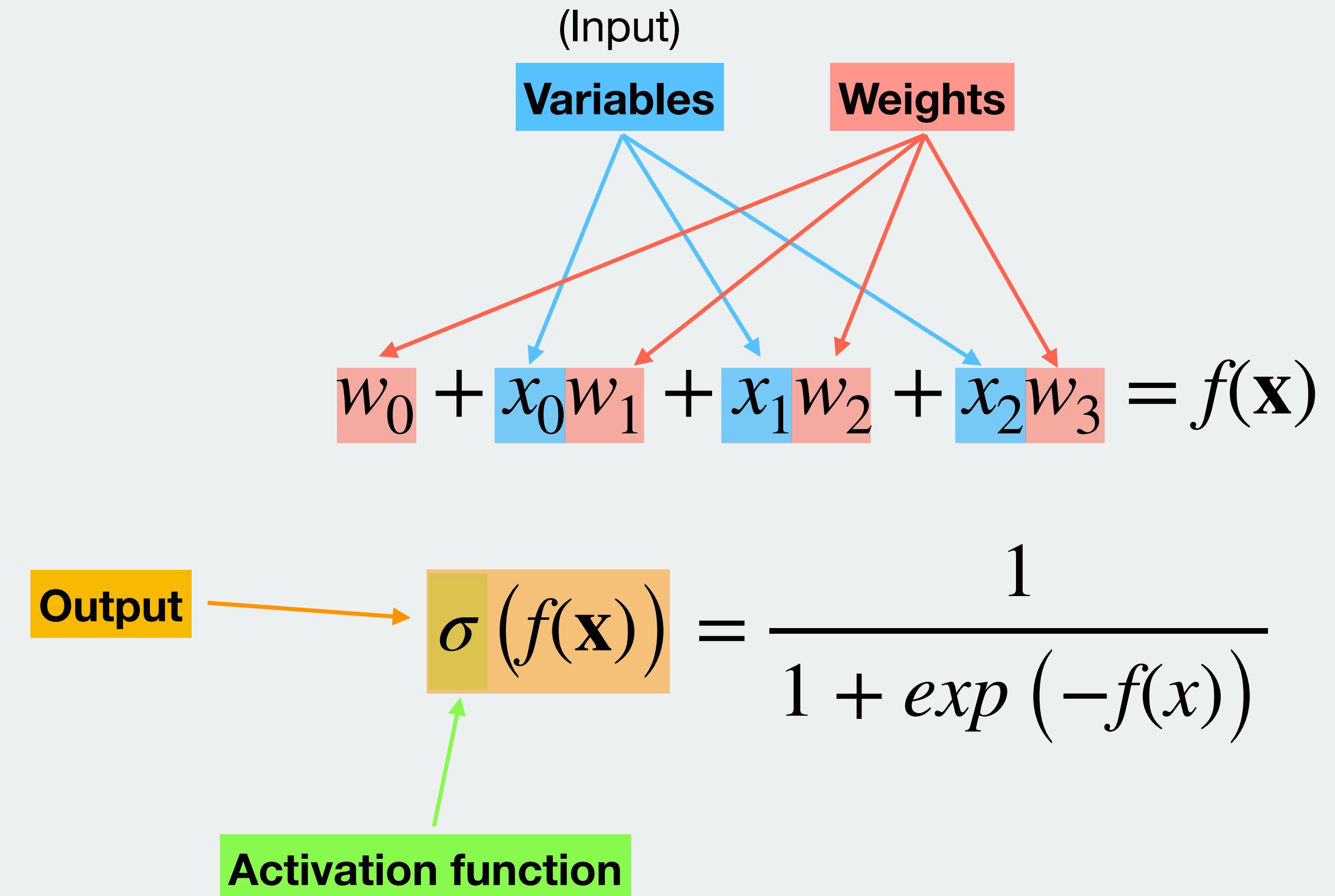
iLogistic regression! classifier

$$w_0 + x_0 w_1 + x_1 w_2 + x_2 w_3 = f(\mathbf{x})$$

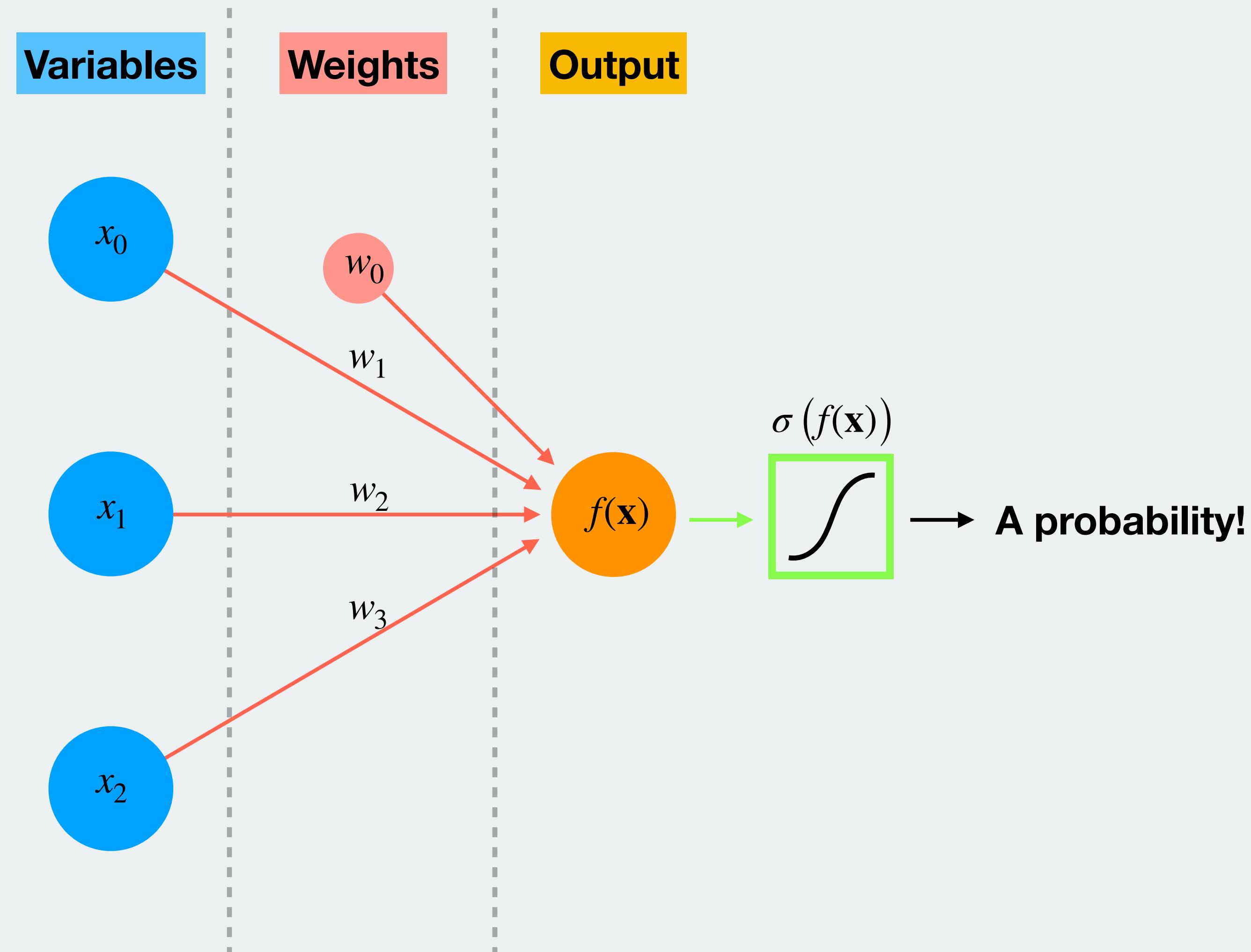
$$\sigma(f(\mathbf{x})) = \frac{1}{1 + \exp(-f(\mathbf{x}))}$$



Logistic regression classifier

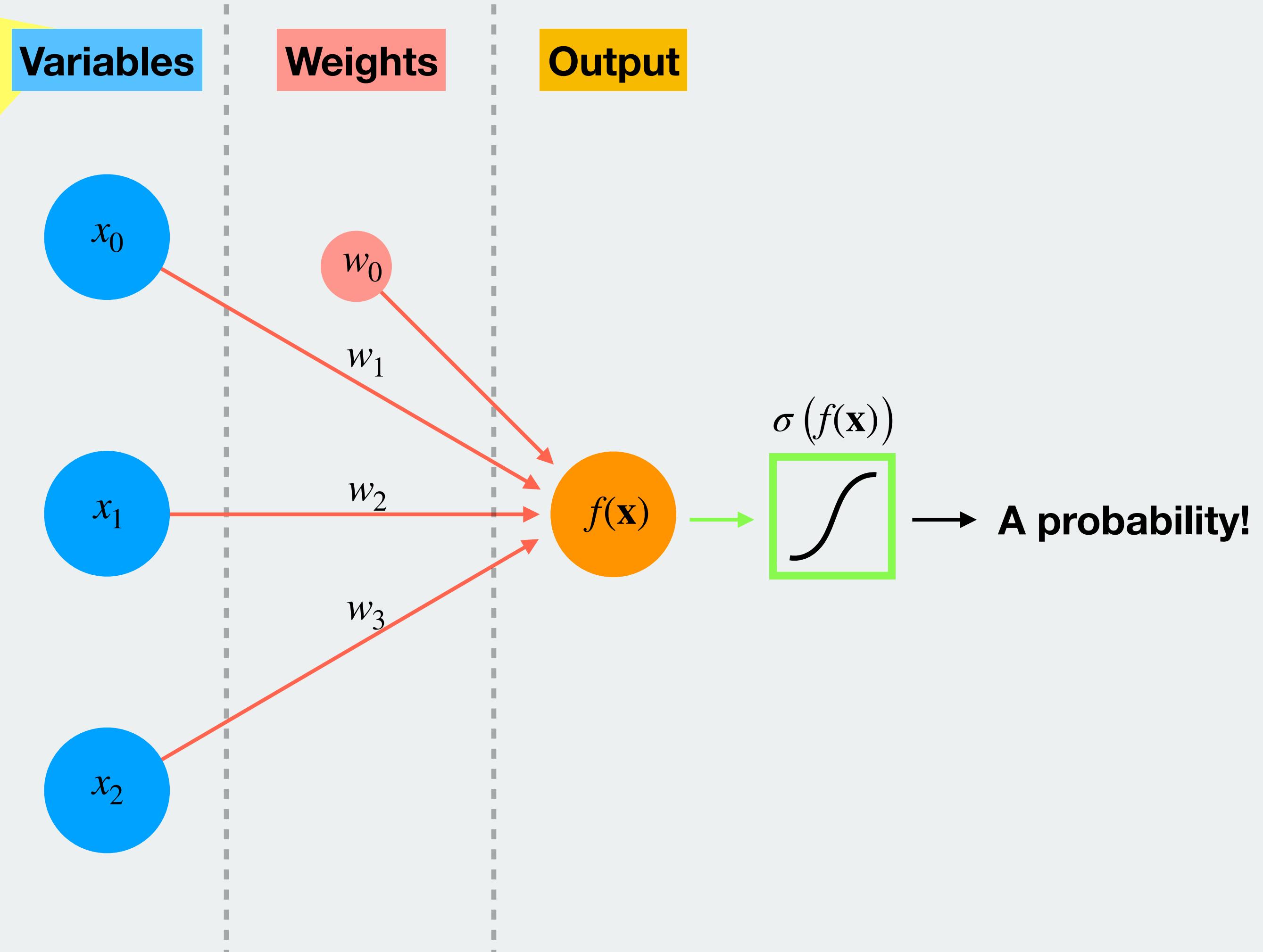


Logistic regression schematic illustration



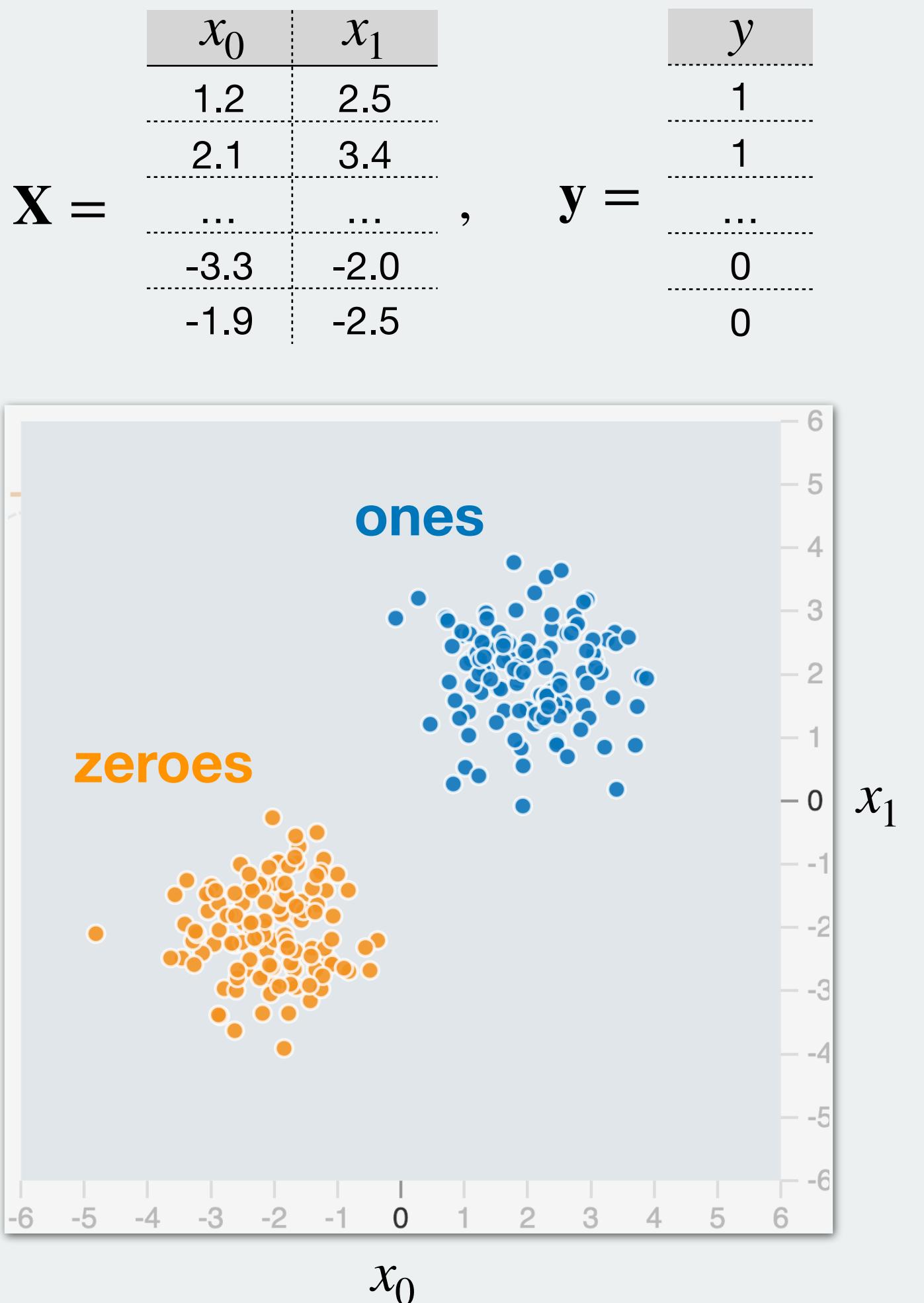
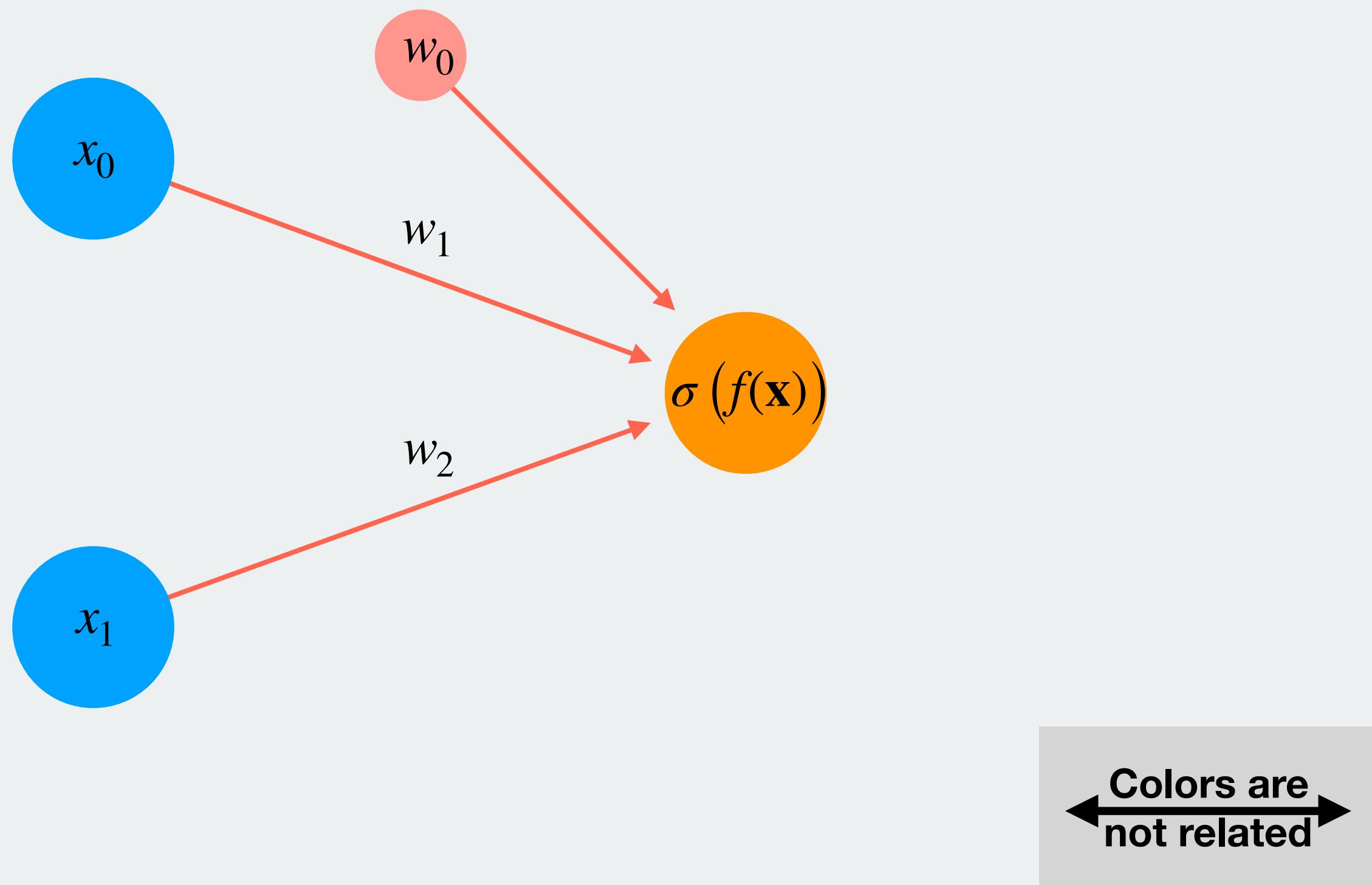
iPerceptron!

Invented by Frank Rosenblat in 1957



Logistic regression simple problem

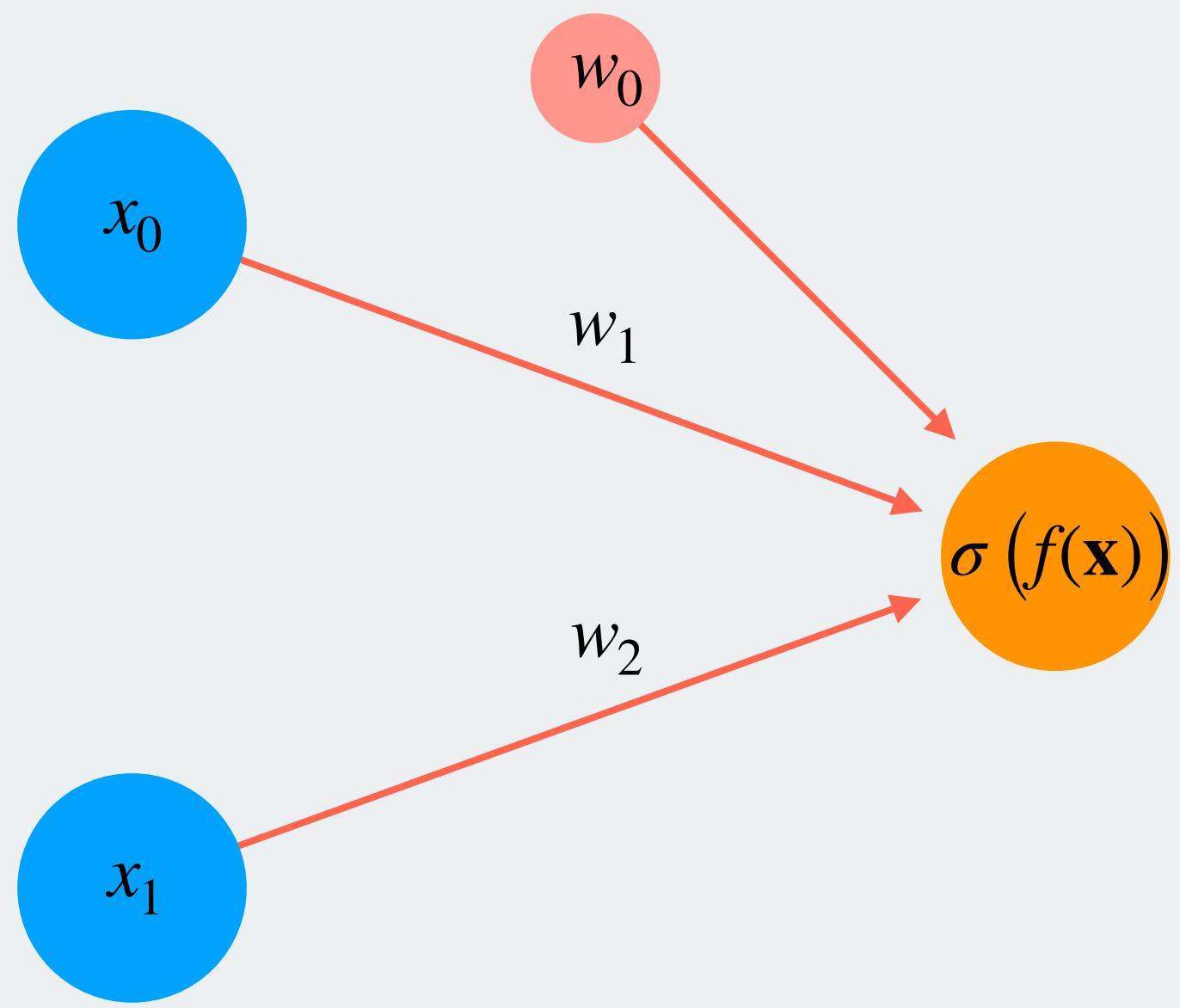
> Find values of $\{w_0, w_1, w_2\}$ that minimizes $\sum_n (\tilde{y}_n - y_n)^2$



Reminder: $w_0 + x_0 w_1 + x_1 w_2 = f(\mathbf{x})$

Logistic regression simple problem

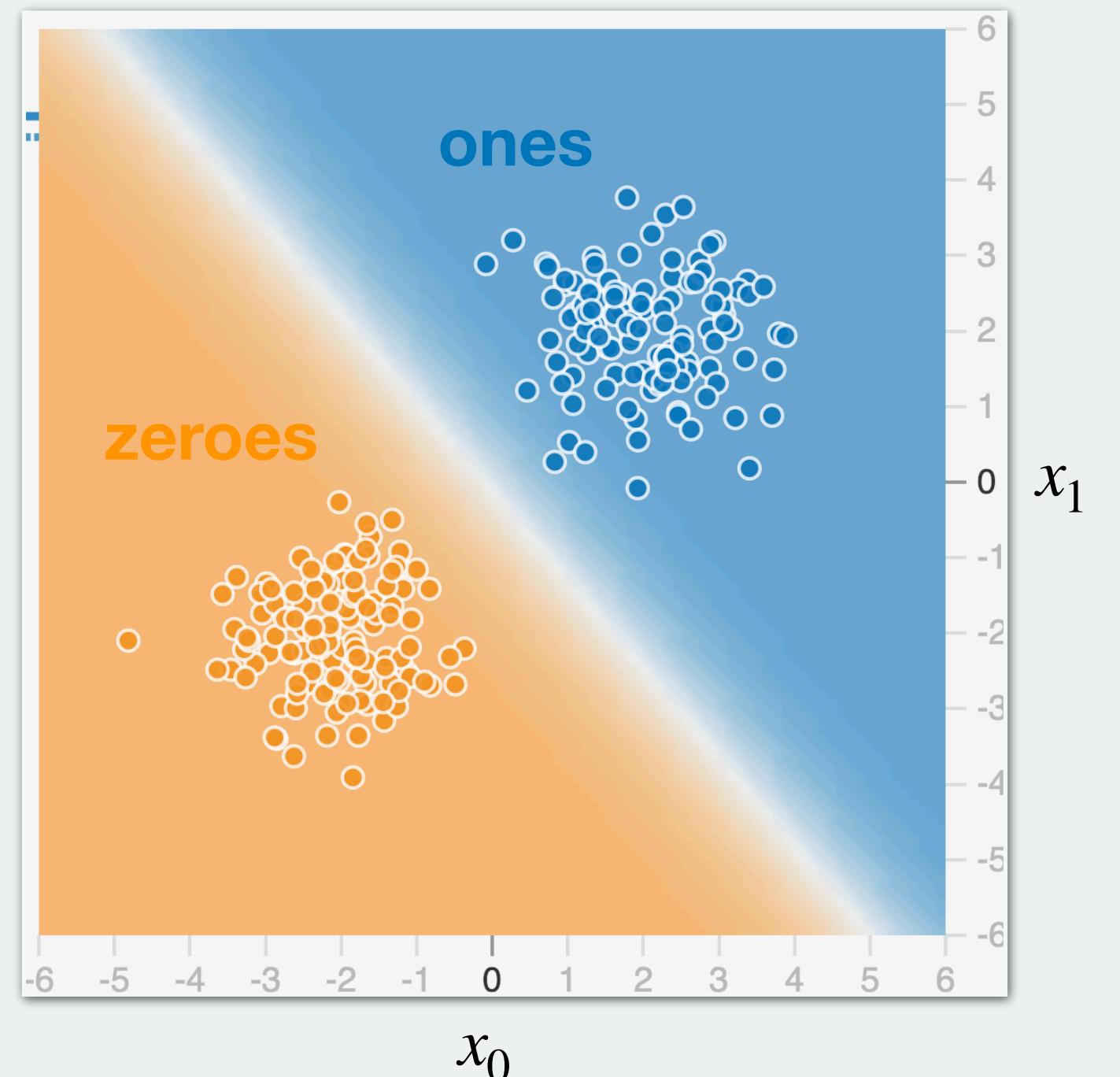
> Find values of $\{w_0, w_1, w_2\}$ that minimizes $\sum_n (\tilde{y}_n - y_n)^2$



x_0	x_1	\tilde{y}	y
1.2	2.5	1	1
2.1	3.4	1	1
...
-3.3	-2.0	0	0
-1.9	-2.5	0	0

$\mathbf{X} =$, $\sigma(f(\mathbf{X})) =$, $\mathbf{y} =$

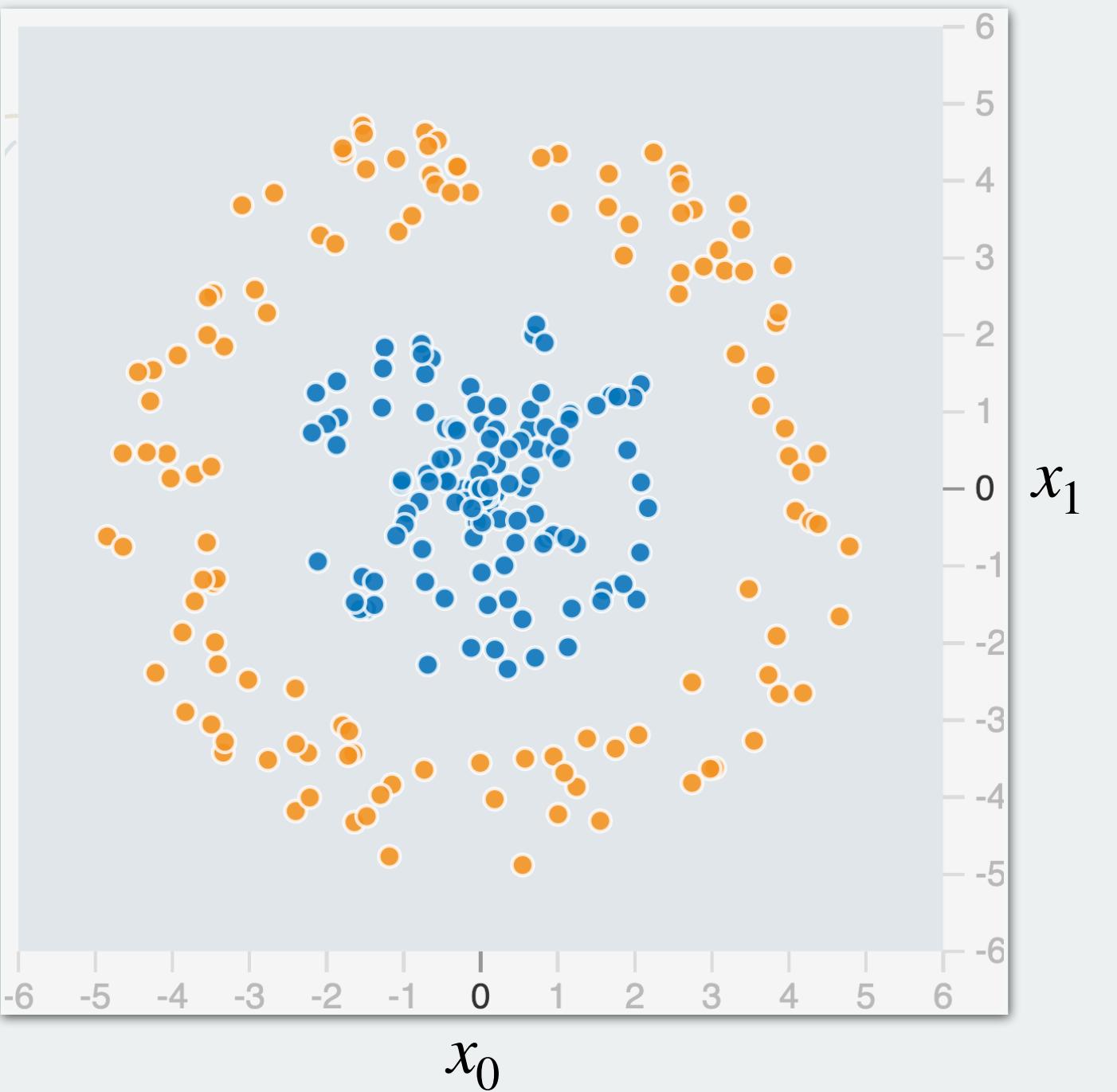
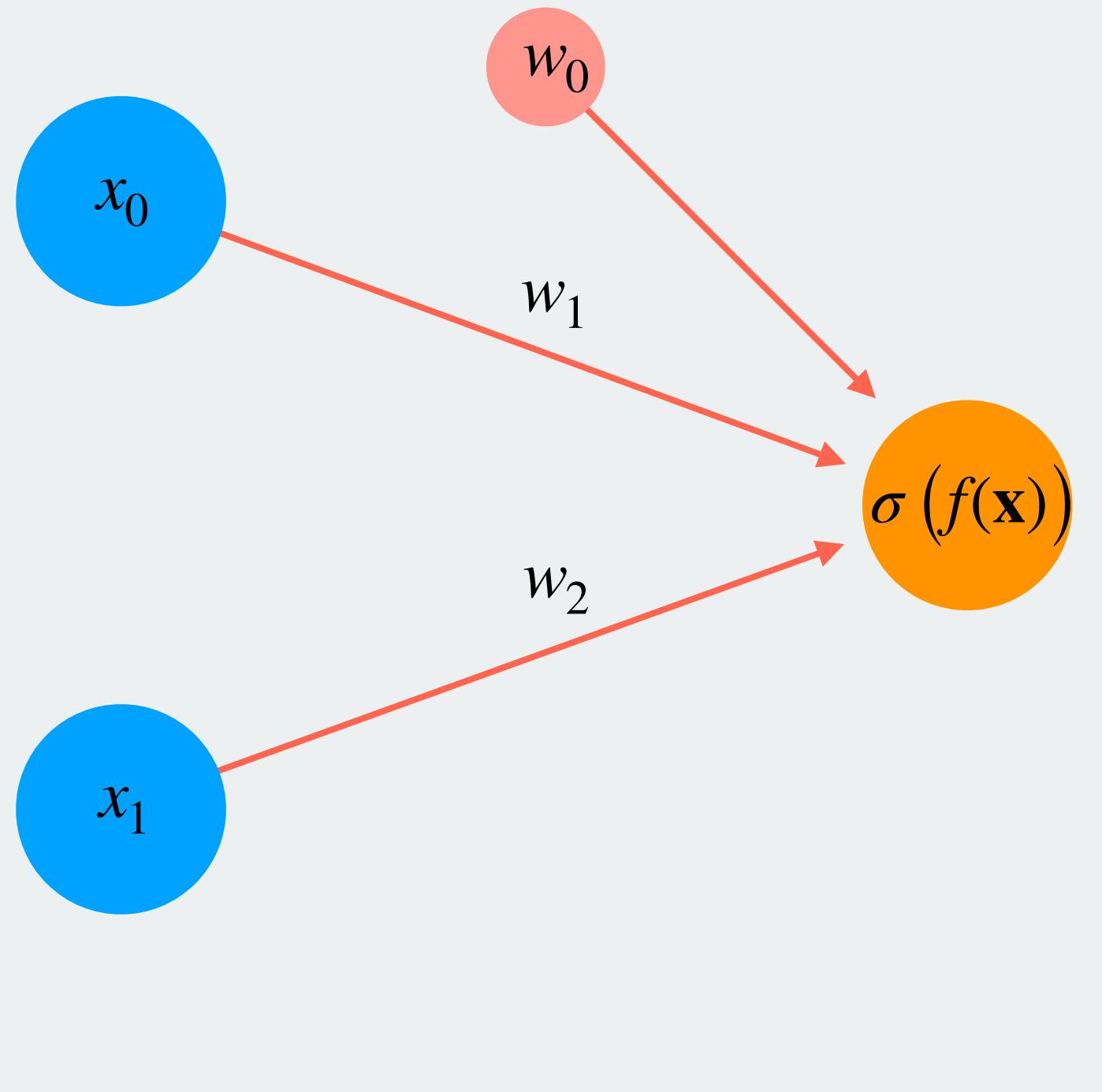
f has amazing
weights gives
perfect prediction



Reminder: $w_0 + x_0 w_1 + x_1 w_2 = f(\mathbf{x})$

Logistic regression not so simple problem

> Find values of $\{w_0, w_1, w_2\}$ that minimizes $\sum_n (\tilde{y}_n - y_n)^2$

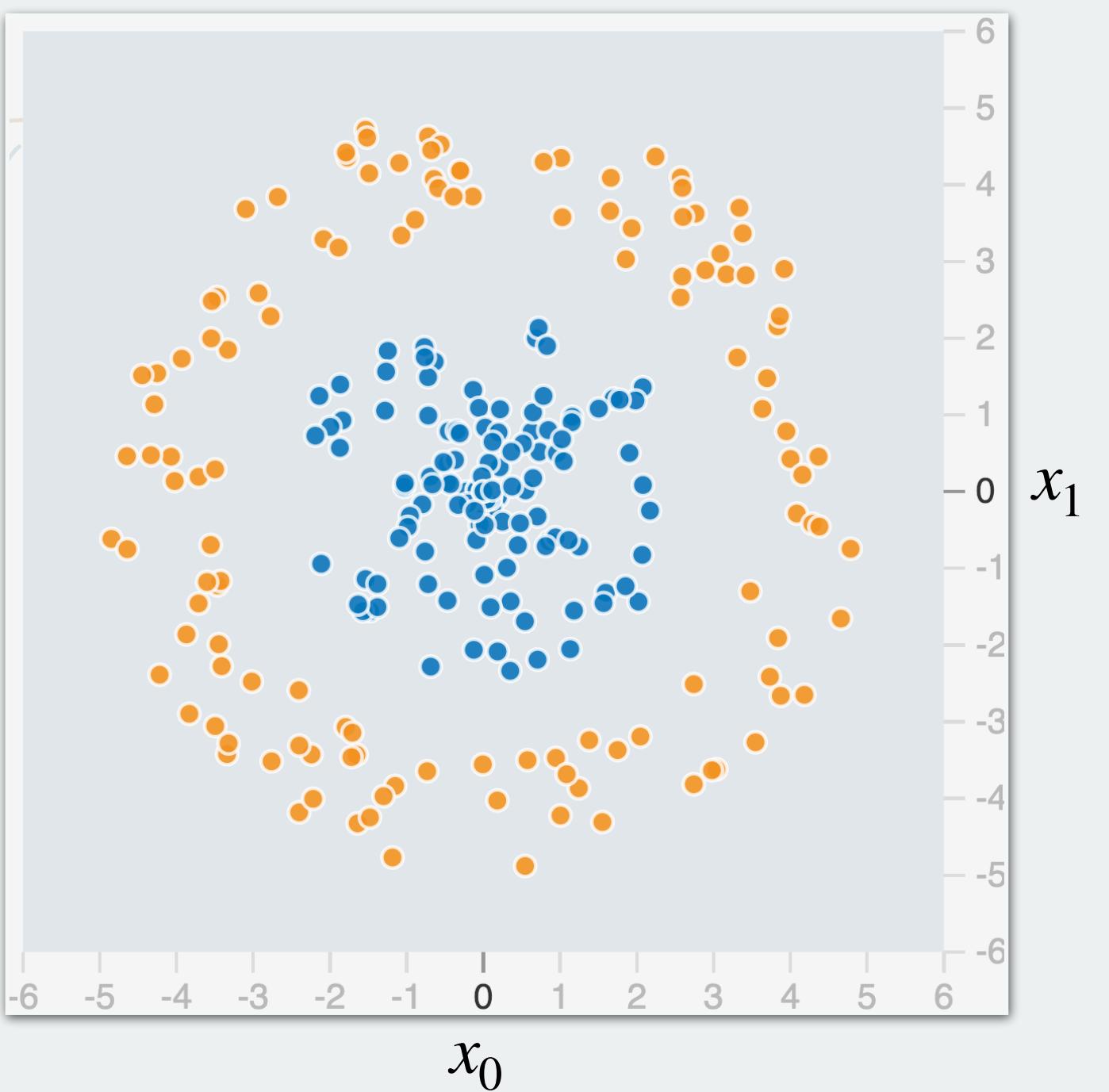
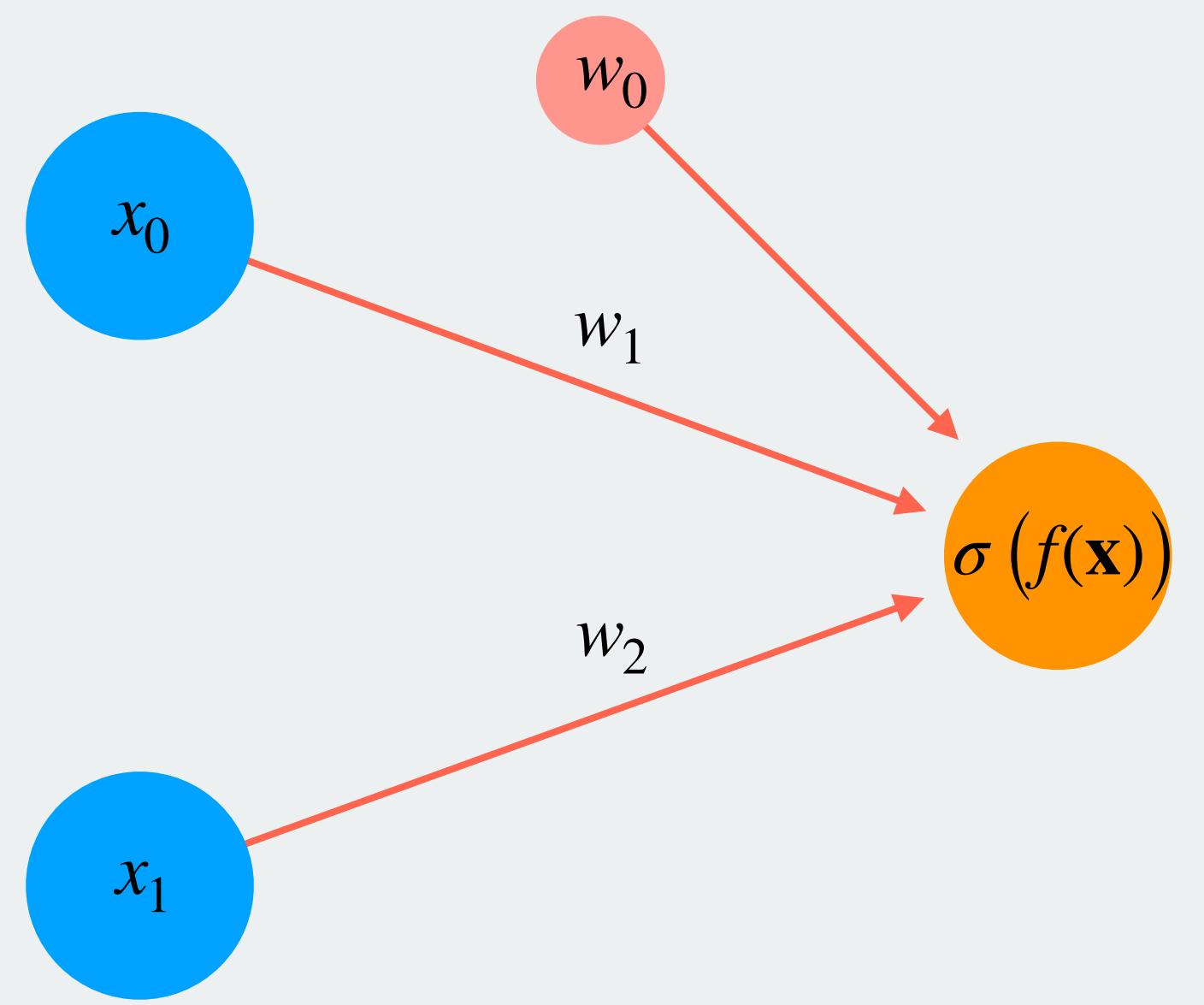


Reminder: $w_0 + x_0 w_1 + x_1 w_2 = f(\mathbf{x})$

Logistic regression not so simple problem

> Find values of $\{w_0, w_1, w_2\}$ that minimizes $\sum_n (\tilde{y}_n - y_n)^2$

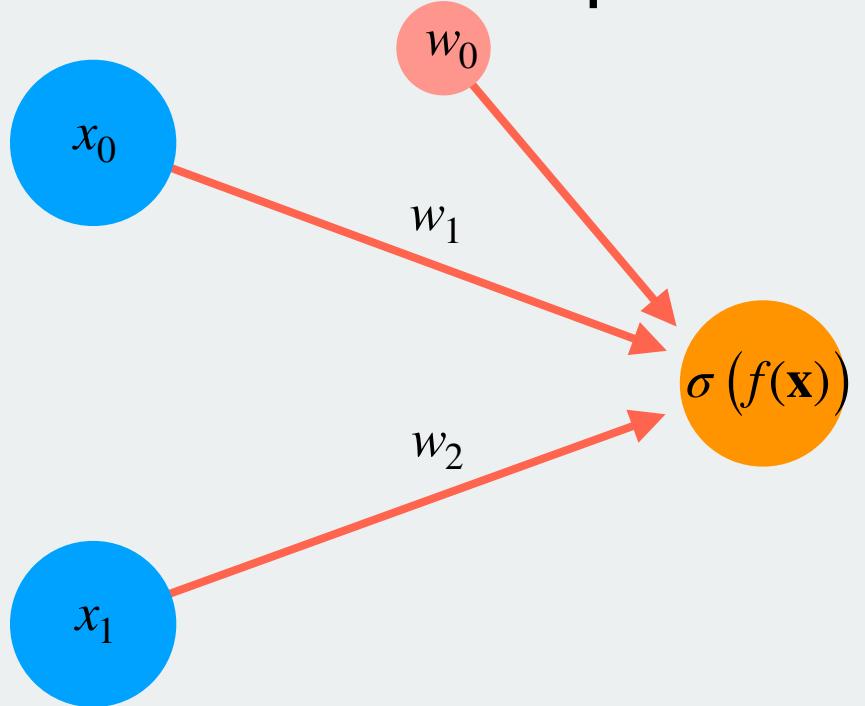
Model too simple



Reminder: $w_0 + x_0w_1 + x_1w_2 = f(\mathbf{x})$

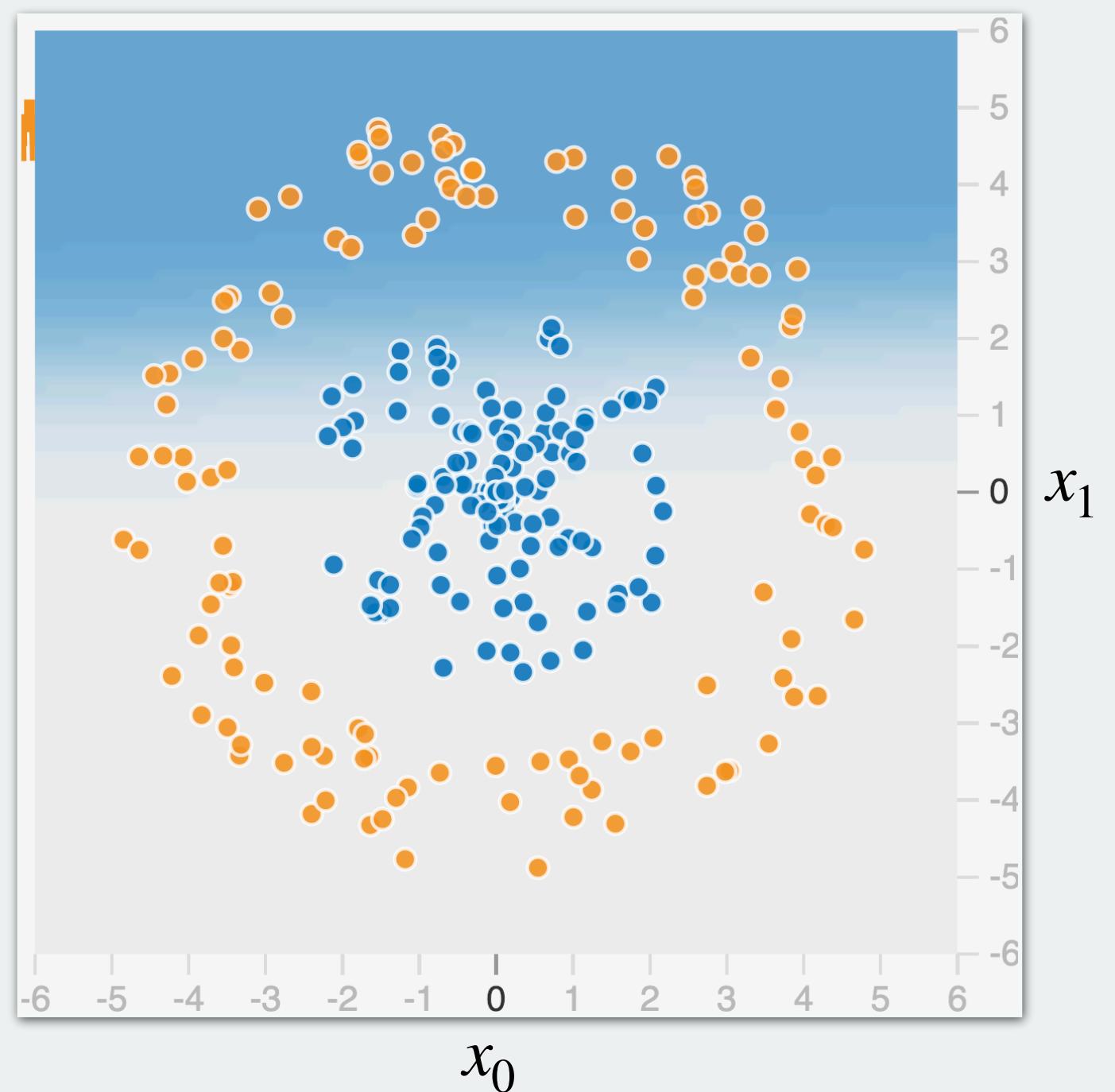
Logistic regression not so simple problem

> Solution: Break problem into subproblems



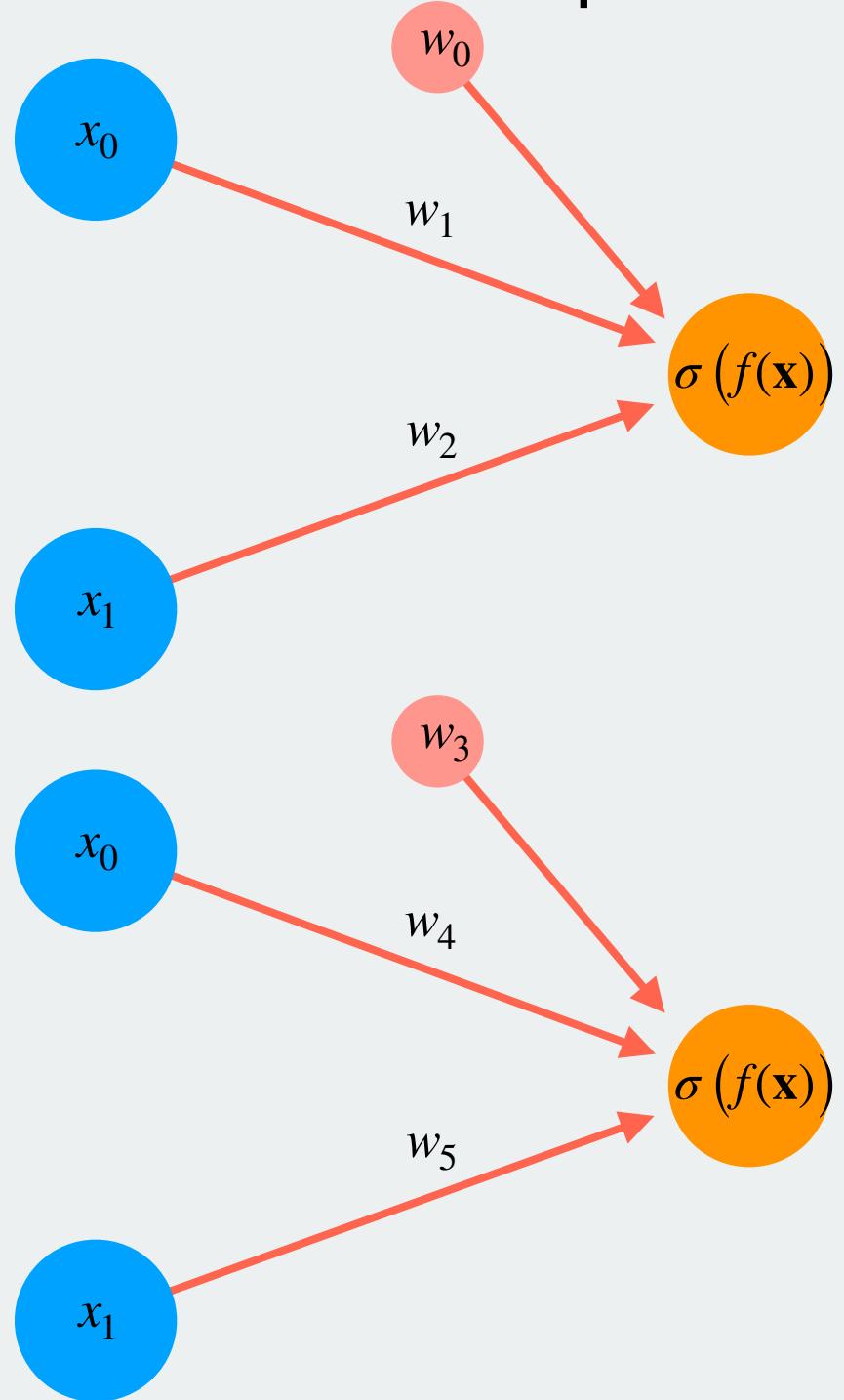
$$\sigma(w_0 + x_0 w_1 + x_1 w_2) = z_0(\mathbf{x})$$

$$\mathbf{X} = \begin{array}{|c|c|}\hline x_0 & x_1 \\ \hline \end{array} \begin{array}{|c|c|}\hline 0.5 & 1.5 \\ \hline 2.3 & -1.7 \\ \hline \dots & \dots \\ \hline 4.2 & -0.2 \\ \hline -1.9 & 2.3 \\ \hline \end{array}, \quad z_0(\mathbf{X}) = \begin{array}{|c|}\hline z_0 \\ \hline \end{array} \begin{array}{|c|}\hline 0.3 \\ \hline 0.25 \\ \hline \dots \\ \hline 0.79 \\ \hline 0.34 \\ \hline \end{array}$$



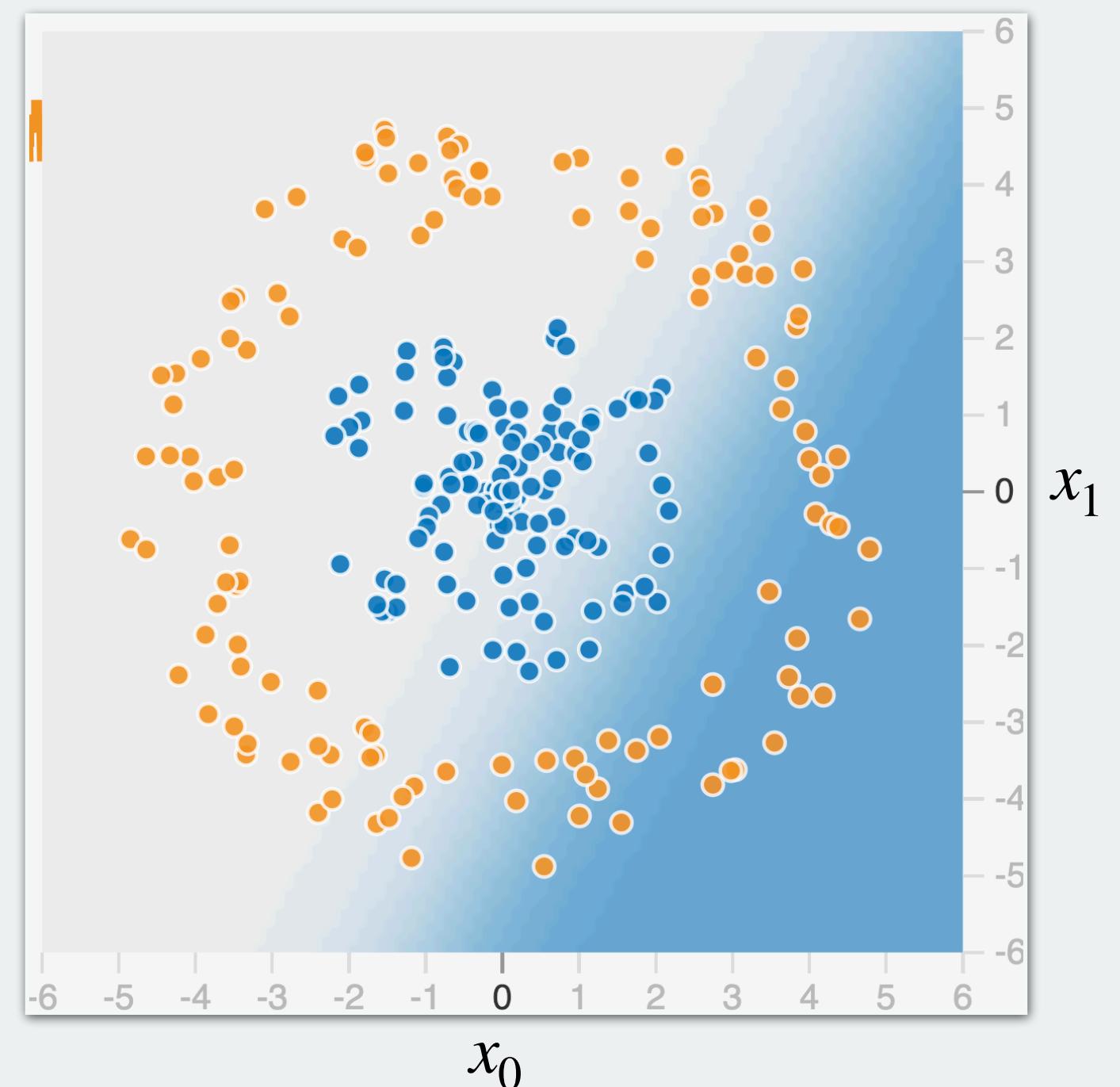
Logistic regression not so simple problem

> Solution: Break problem into subproblems



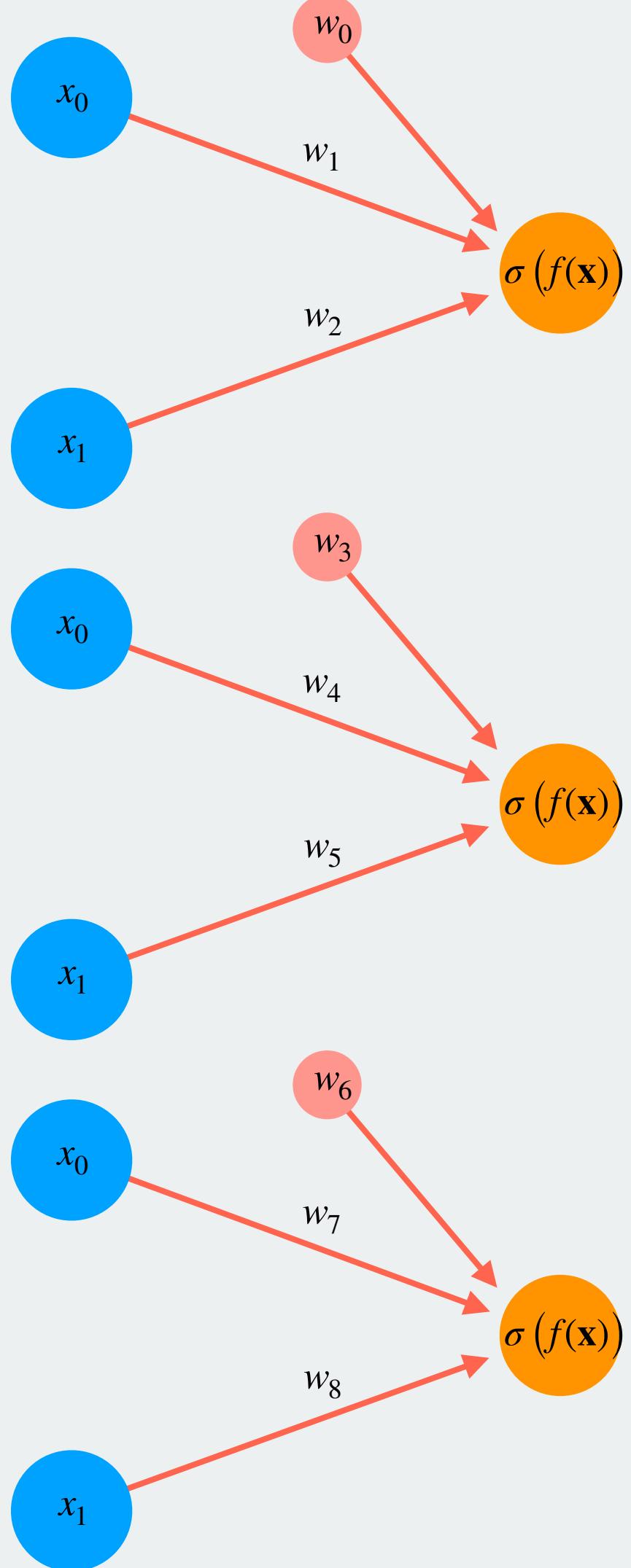
$$\sigma(w_0 + x_0 w_1 + x_1 w_2) = z_0(\mathbf{x})$$

$$\sigma(w_3 + x_0 w_4 + x_1 w_5) = z_1(\mathbf{x})$$



Logistic regression not so simple problem

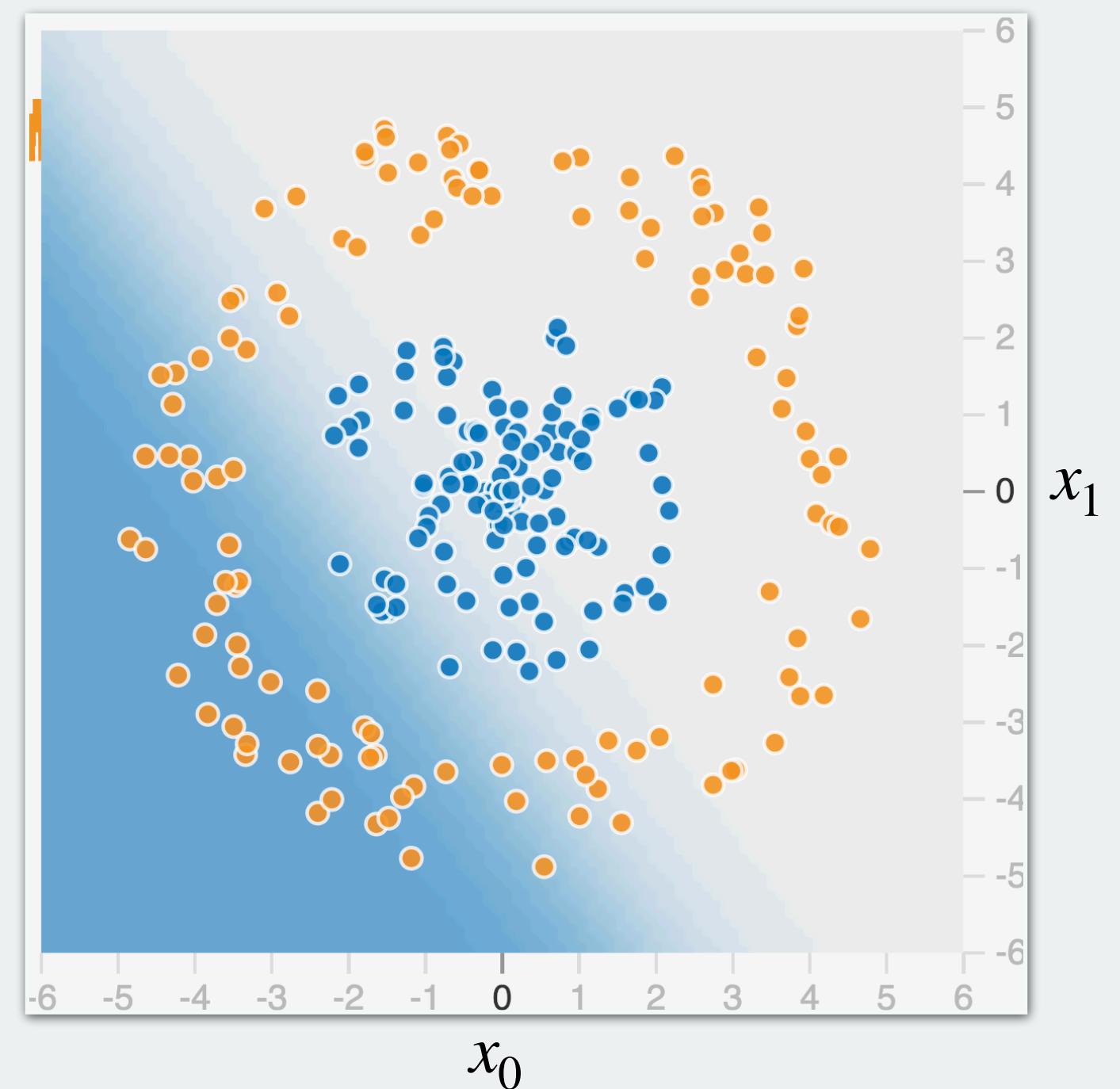
> Solution: Break problem into subproblems



$$\sigma(w_0 + x_0w_1 + x_1w_2) = z_0(\mathbf{x})$$

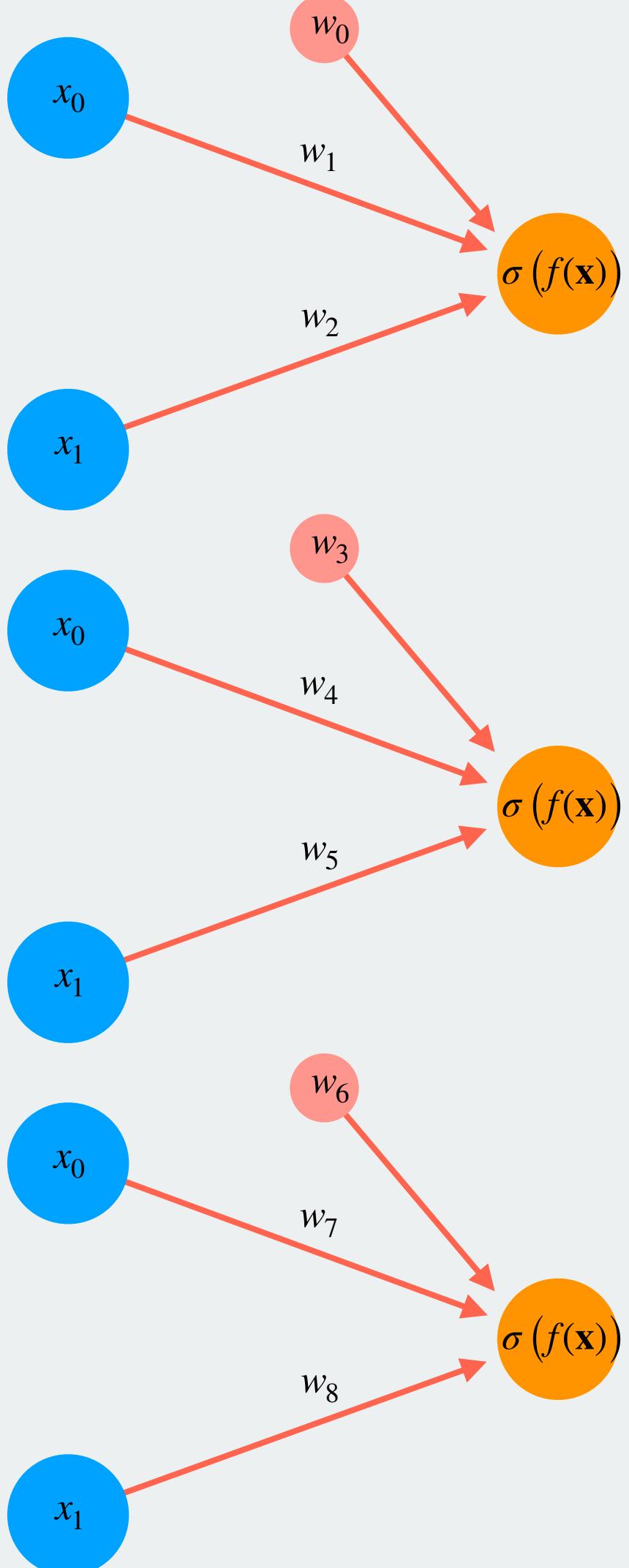
$$\sigma(w_3 + x_0w_4 + x_1w_5) = z_1(\mathbf{x})$$

$$\sigma(w_6 + x_0w_7 + x_1w_8) = z_2(\mathbf{x})$$



Logistic regression not so simple problem

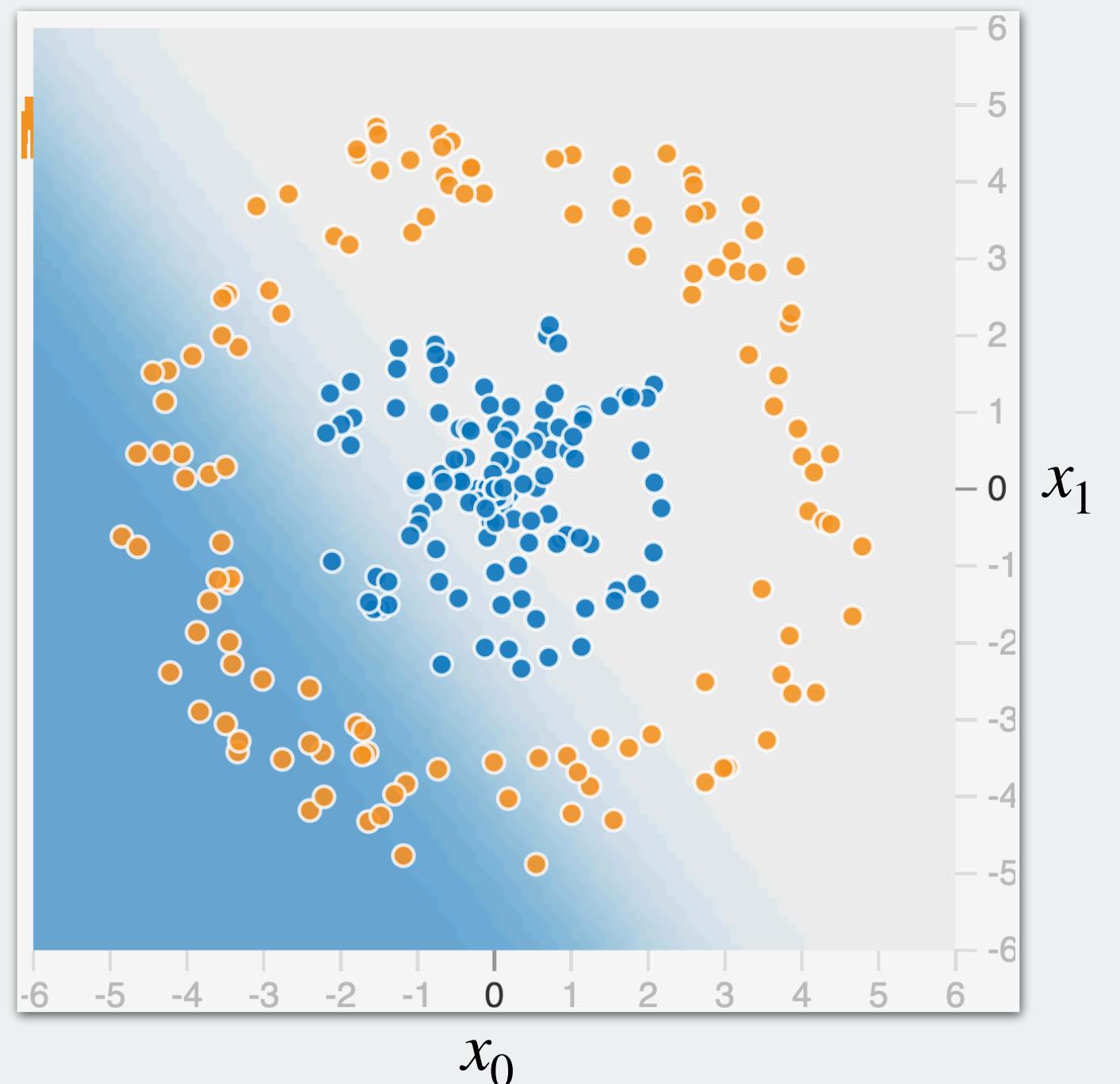
> Solution: Break problem into subproblems



$$\sigma(w_0 + x_0 w_1 + x_1 w_2) = z_0(\mathbf{x})$$

$$\sigma(w_3 + x_0 w_4 + x_1 w_5) = z_1(\mathbf{x})$$

$$\sigma(w_6 + x_0 w_7 + x_1 w_8) = z_2(\mathbf{x})$$



New problem: Given \mathbf{Z} , predict \mathbf{y}

z_0	z_1	z_2
0.3	0.75	0.78
0.25	0.1	0.95
...	...	
0.79	0.99	0.3
0.34	0.6	0.1

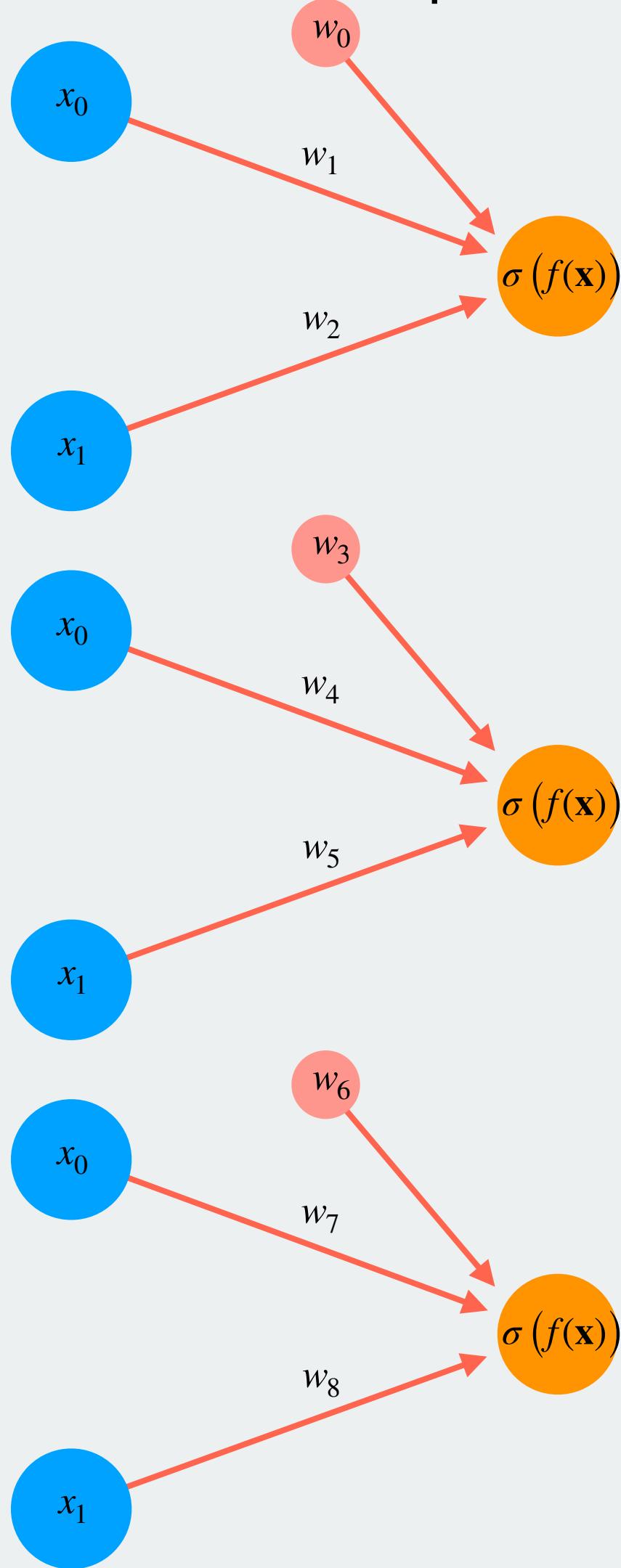
$\mathbf{Z} =$

y
0
1
...
1
0

$\mathbf{y} =$

Logistic regression not so simple problem

> Solution: Break problem into subproblems



$$\sigma(w_0 + x_0w_1 + x_1w_2) = z_0(\mathbf{x})$$

$$\sigma(w_3 + x_0w_4 + x_1w_5) = z_1(\mathbf{x})$$

$$\sigma(w_6 + x_0w_7 + x_1w_8) = z_2(\mathbf{x})$$

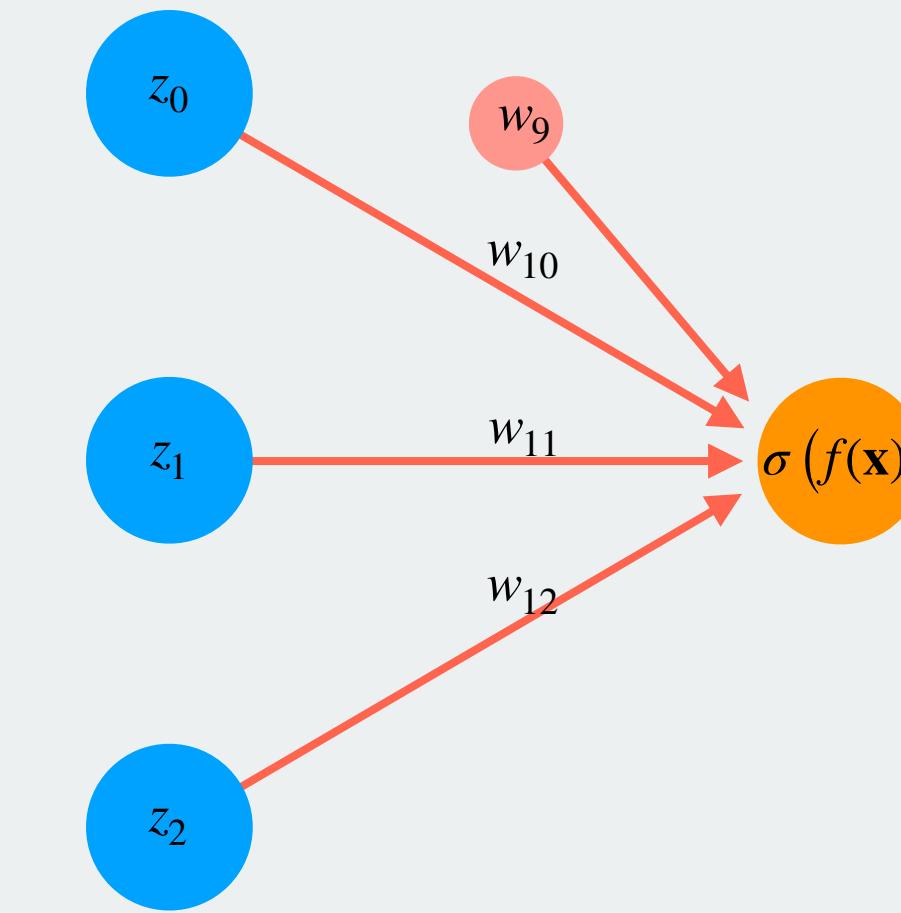
New problem: Given \mathbf{Z} , predict \mathbf{y}

z_0	z_1	z_2	y
0.3	0.75	0.78	0
0.25	0.1	0.95	1
...
0.79	0.99	0.3	1
0.34	0.6	0.1	0

$\mathbf{Z} =$

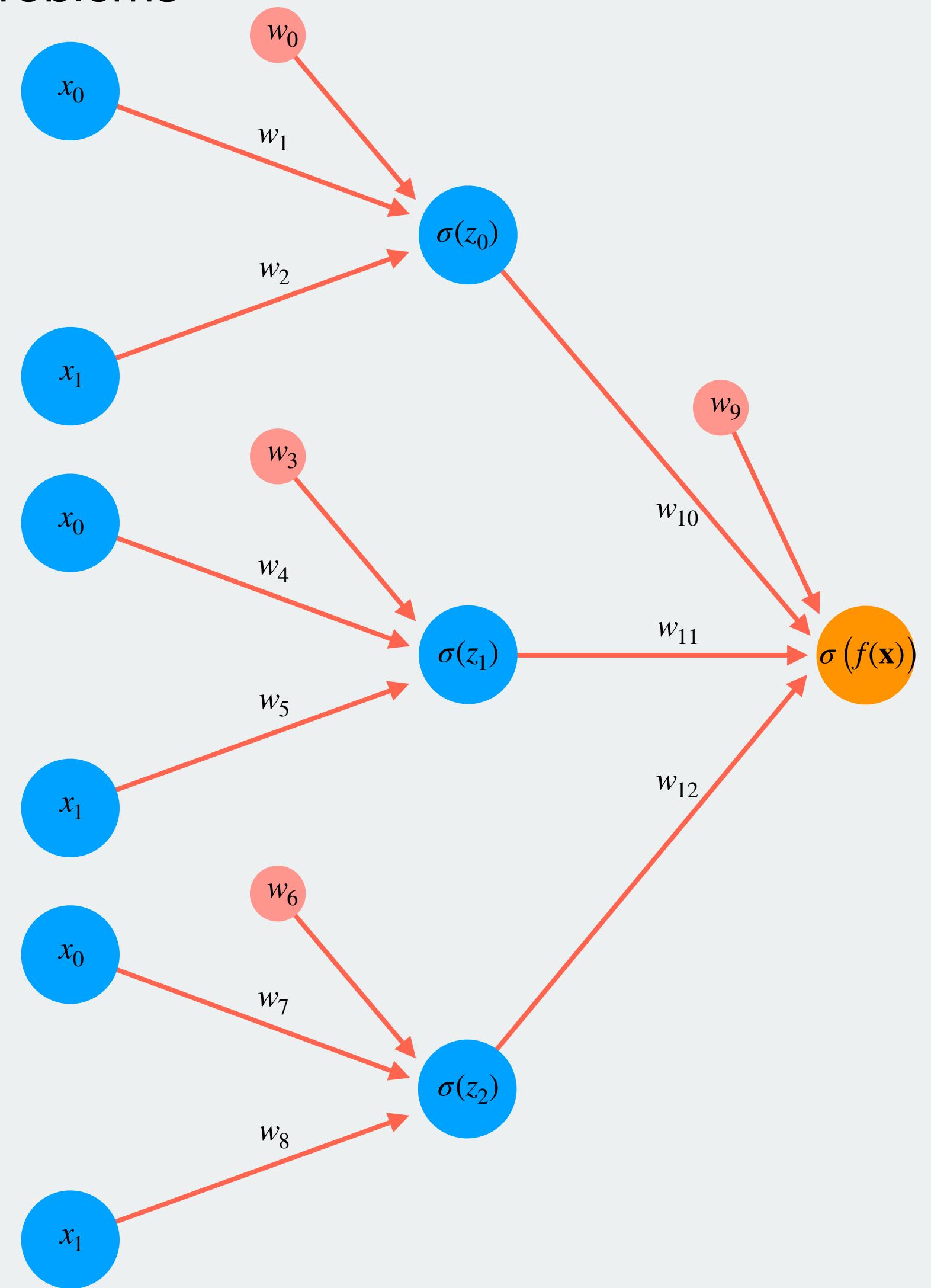
$\mathbf{y} =$

Solution: Why not use a logistic regression?



Logistic regression not so simple problem

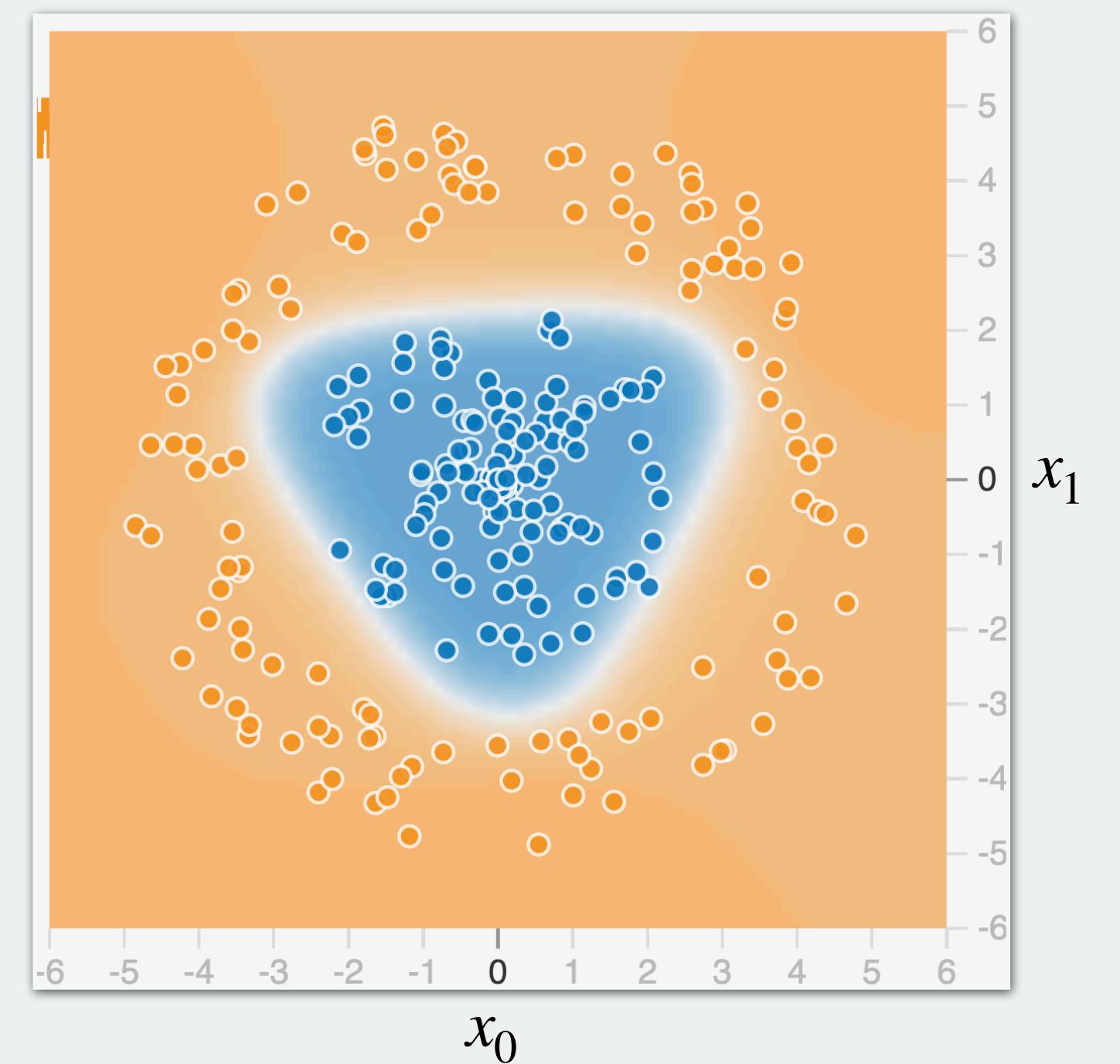
> Solution: Connect subproblems



Mathematical form

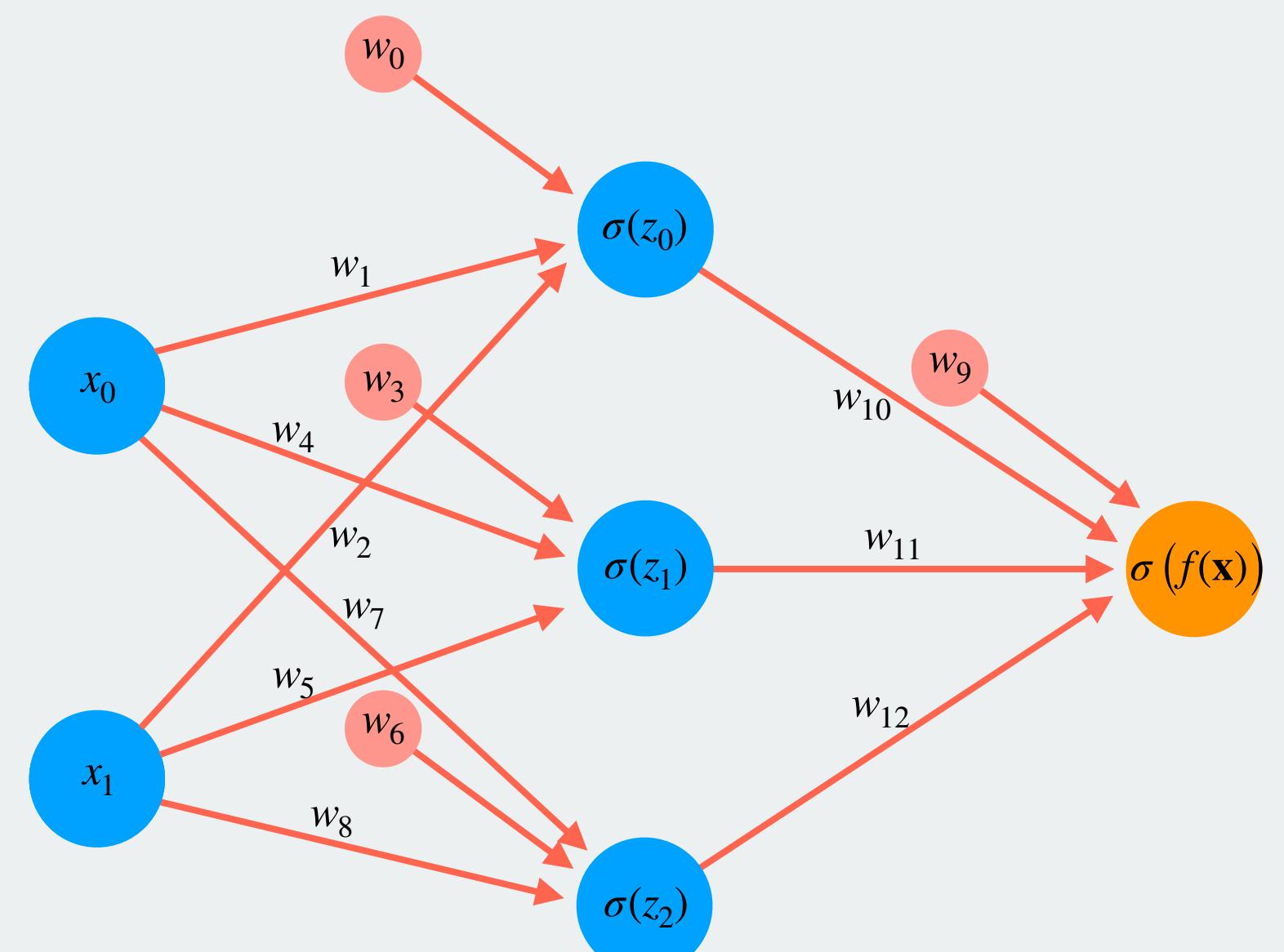
$$\begin{aligned}\sigma(w_9 + \sigma(w_0 + x_0 w_1 + x_1 w_2) w_{10}) + \\ \sigma(w_3 + x_0 w_4 + x_1 w_5) w_{11} + \\ \sigma(w_6 + x_0 w_7 + x_1 w_8) w_{12}) = \sigma(f(\mathbf{x}))\end{aligned}$$

Solution



Logistic regression not so simple problem

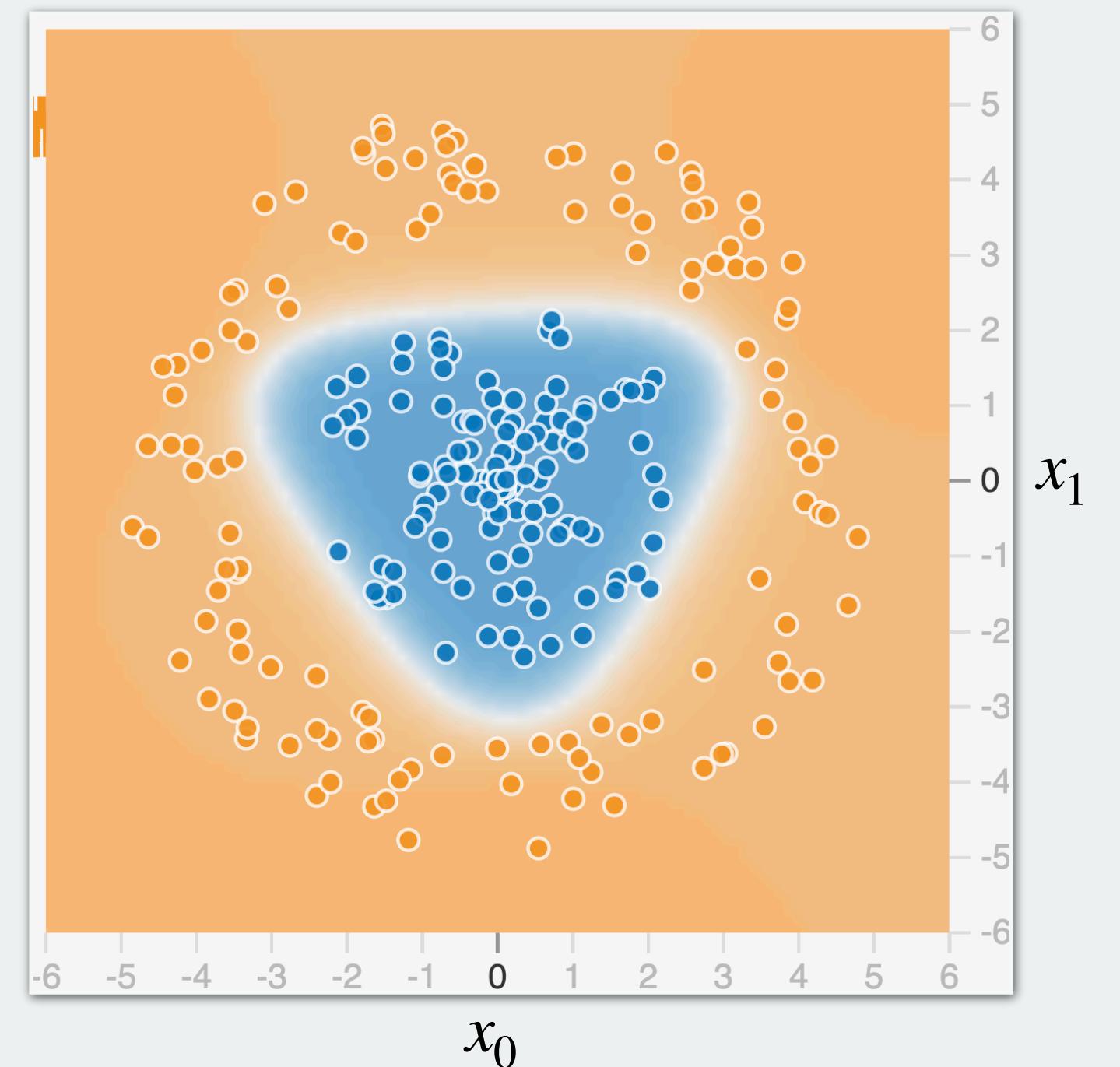
> Solution: Connect subproblems



Mathematical form

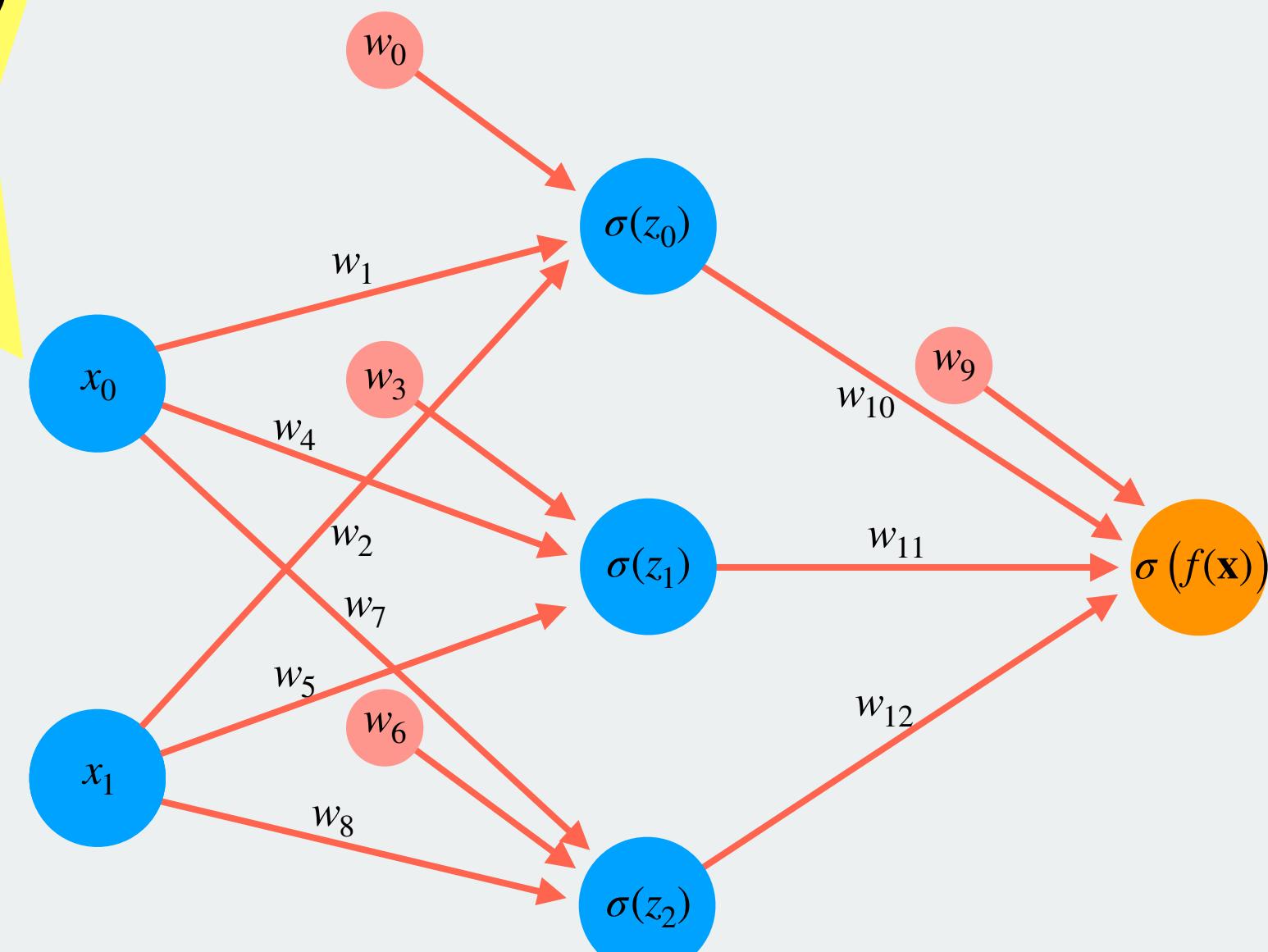
$$\begin{aligned}\sigma(w_9 + \sigma(w_0 + x_0 w_1 + x_1 w_2) w_{10}) + \\ \sigma(w_3 + x_0 w_4 + x_1 w_5) w_{11} + \\ \sigma(w_6 + x_0 w_7 + x_1 w_8) w_{12}) = \sigma(f(\mathbf{x}))\end{aligned}$$

Solution



Logistic regression is not so simple problem

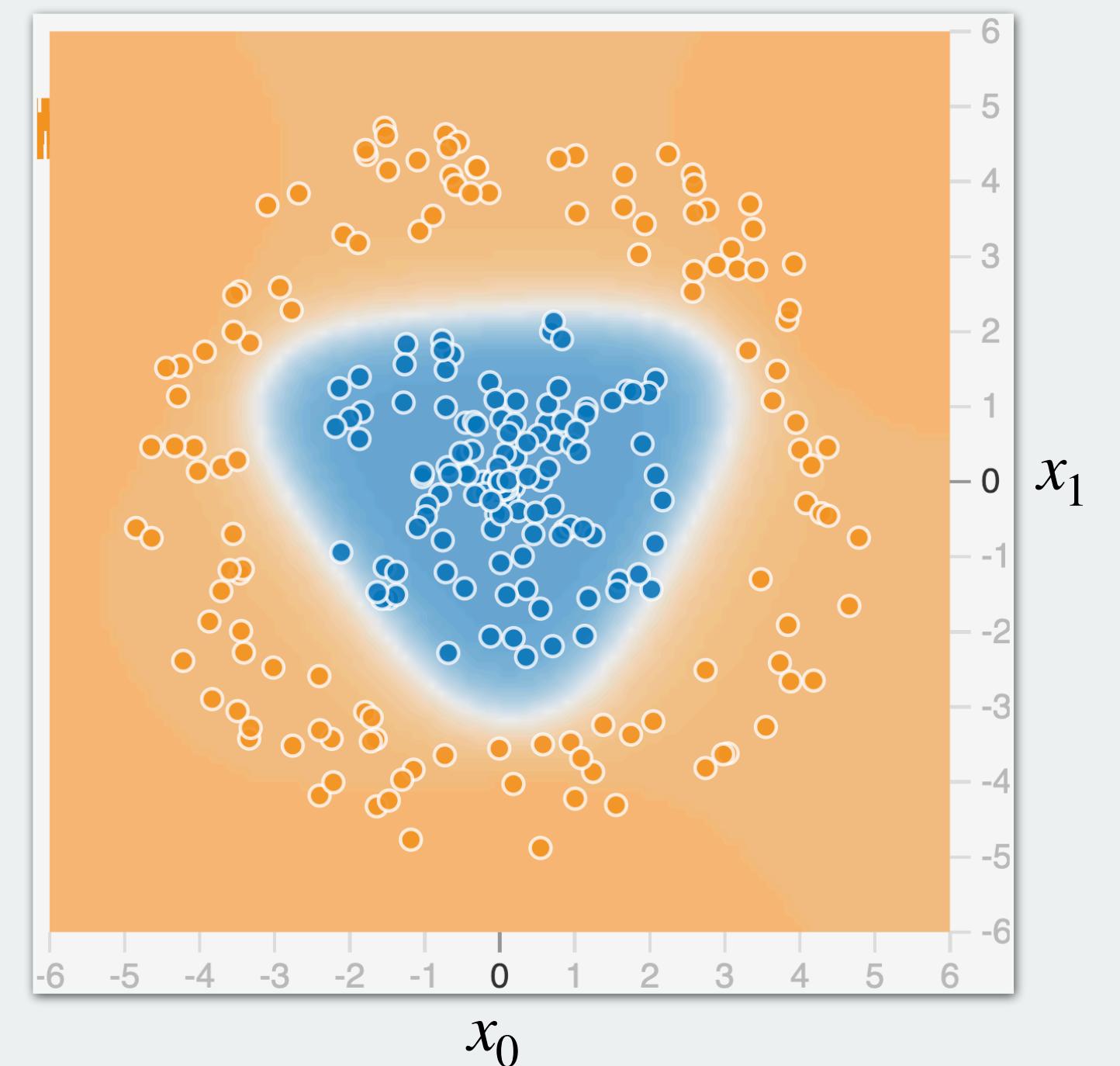
iMultilayer perceptron! classifier



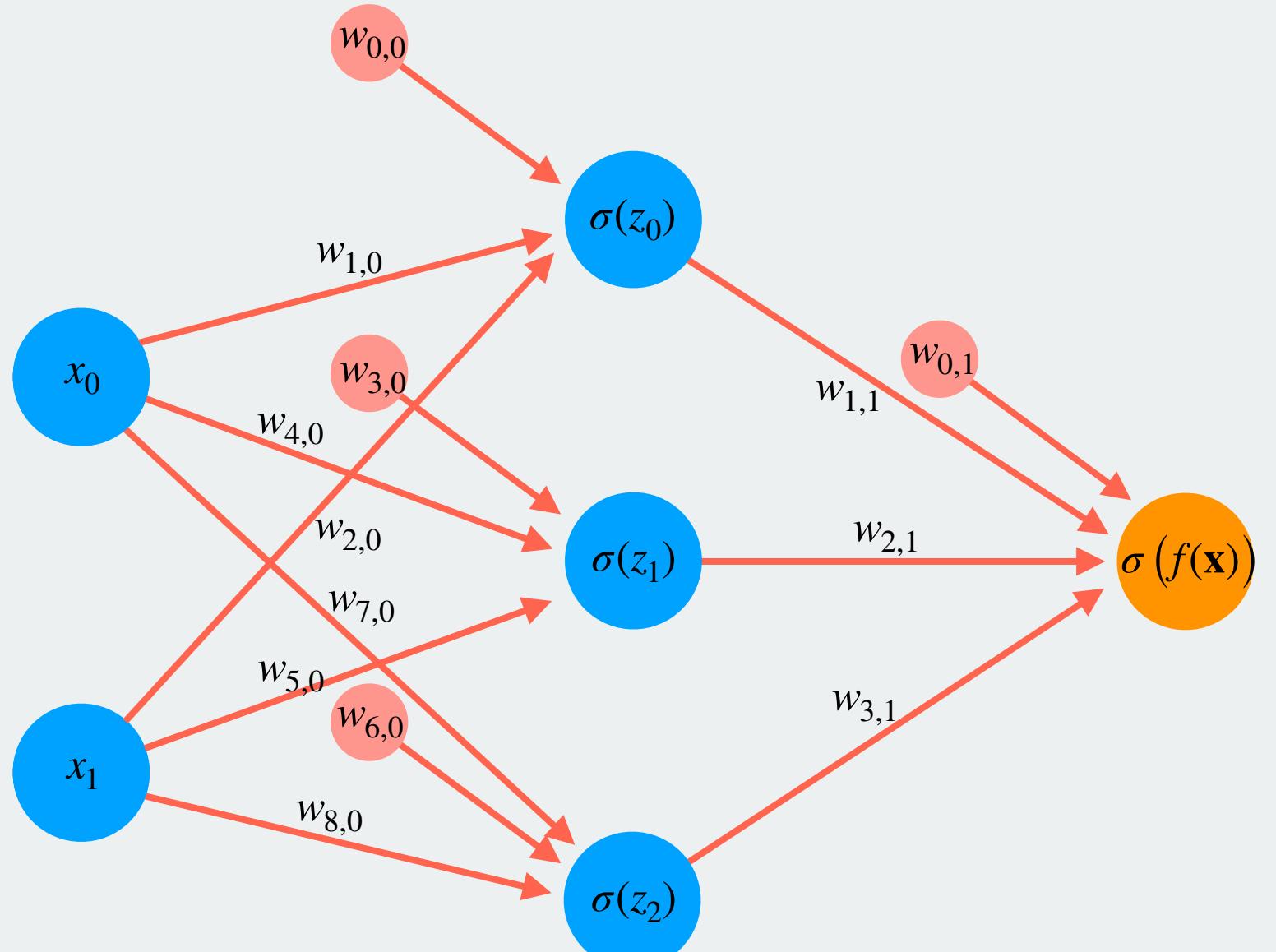
Mathematical form

$$\begin{aligned}\sigma(w_9 + \sigma(w_0 + x_0 w_1 + x_1 w_2)w_{10} + \\ \sigma(w_3 + x_0 w_4 + x_1 w_5)w_{11} + \\ \sigma(w_6 + x_0 w_7 + x_1 w_8)w_{12}) = \sigma(f(\mathbf{x}))\end{aligned}$$

Solution



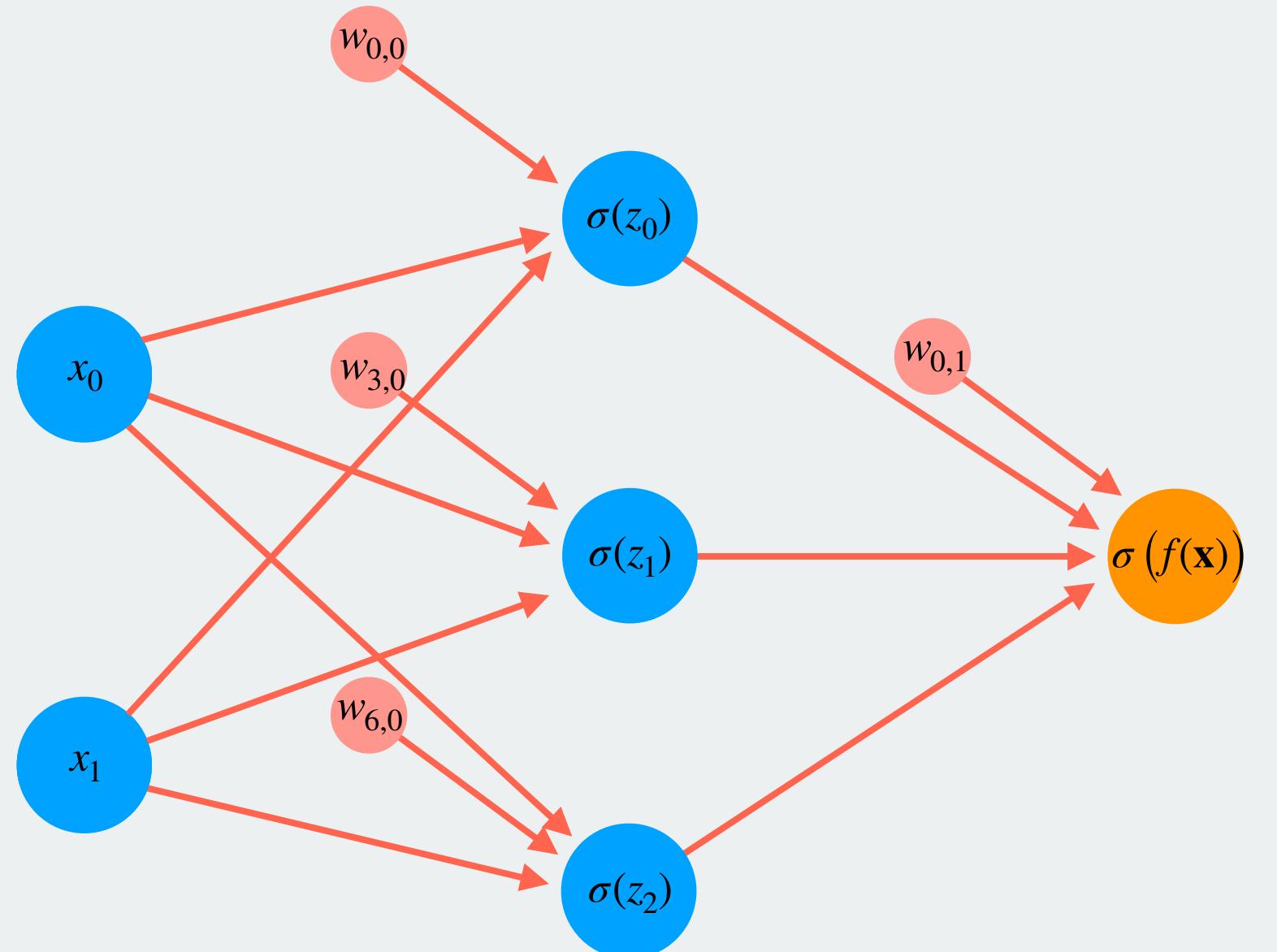
Multilayer perceptron – the math



Mathematical form

$$\begin{aligned}\sigma(w_{0,1} + \sigma(w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0}) w_{1,1} &+ \\ \sigma(w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0}) w_{2,1} &+ \\ \sigma(w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0}) w_{3,1}) &= \sigma(f(\mathbf{x}))\end{aligned}$$

Multilayer perceptron – the math



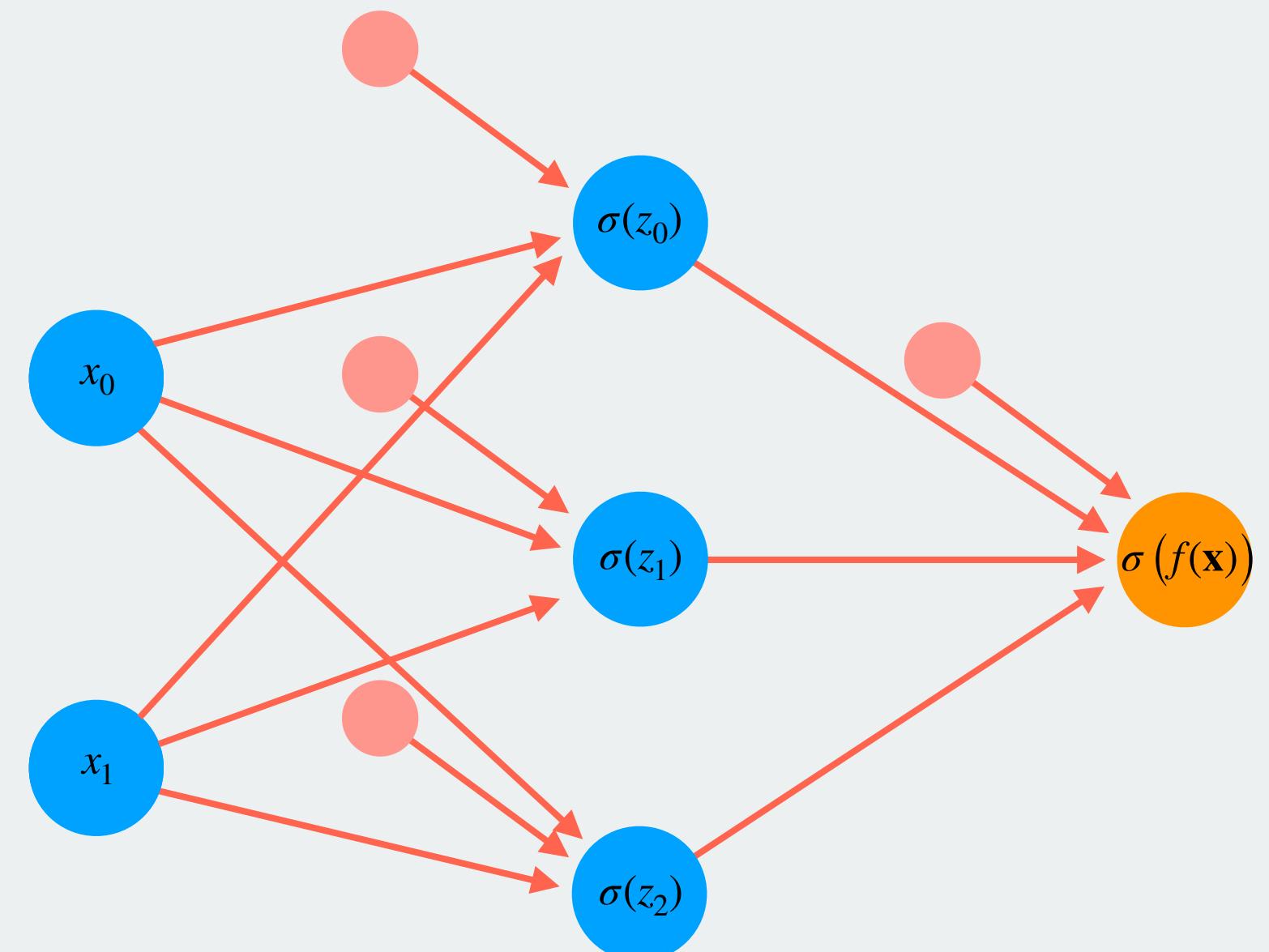
Mathematical form

$$\begin{aligned} & \sigma(w_{0,1} + \sigma(w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0}) w_{1,1}) + \\ & \sigma(w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0}) w_{2,1} + \\ & \sigma(w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0}) w_{3,1}) = \sigma(f(\mathbf{x})) \end{aligned}$$

$$\mathbf{W}_0 = \begin{bmatrix} w_{1,0} & w_{2,0} \\ w_{4,0} & w_{5,0} \\ w_{7,0} & w_{8,0} \end{bmatrix}$$

$$\mathbf{W}_1 = \begin{bmatrix} w_{1,1} & w_{2,1} & w_{3,1} \end{bmatrix}$$

Multilayer perceptron – the math

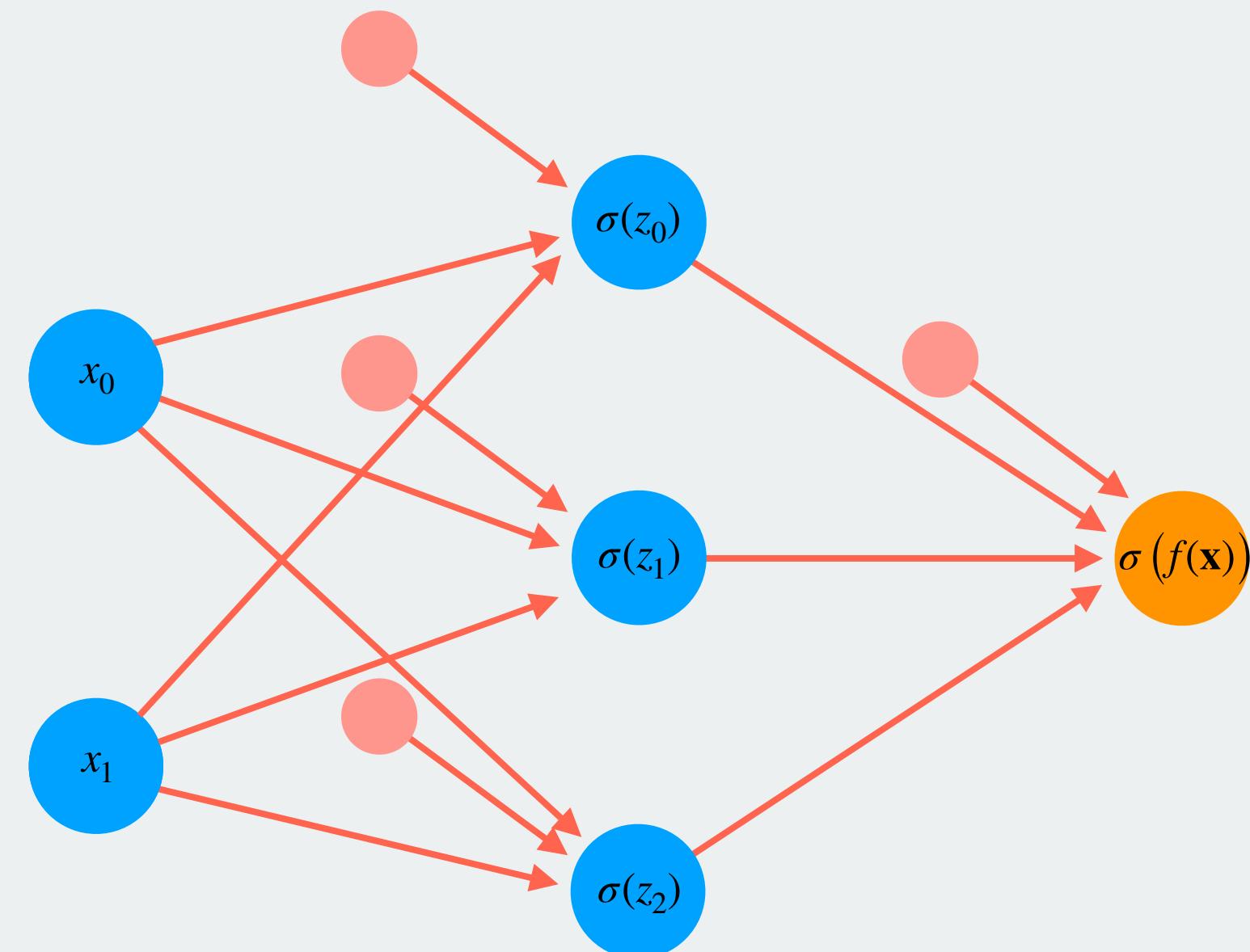


Mathematical form

$$\sigma(w_{0,1} + \sigma(w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} + \sigma(w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} + \sigma(w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1}) = \sigma(f(\mathbf{x}))$$

$$\mathbf{b}_0 = \begin{bmatrix} w_{0,0} \\ w_{3,0} \\ w_{6,0} \end{bmatrix} \quad \mathbf{W}_0 = \begin{bmatrix} w_{1,0} & w_{2,0} \\ w_{4,0} & w_{5,0} \\ w_{7,0} & w_{8,0} \end{bmatrix} \quad \mathbf{b}_1 = \begin{bmatrix} w_{0,1} \end{bmatrix} \quad \mathbf{W}_1 = \begin{bmatrix} w_{1,1} & w_{2,1} & w_{3,1} \end{bmatrix}$$

Multilayer perceptron – the math



Mathematical form

$$\sigma(w_{0,1} + \sigma(w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} + \sigma(w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} + \sigma(w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1}) = \sigma(f(x))$$

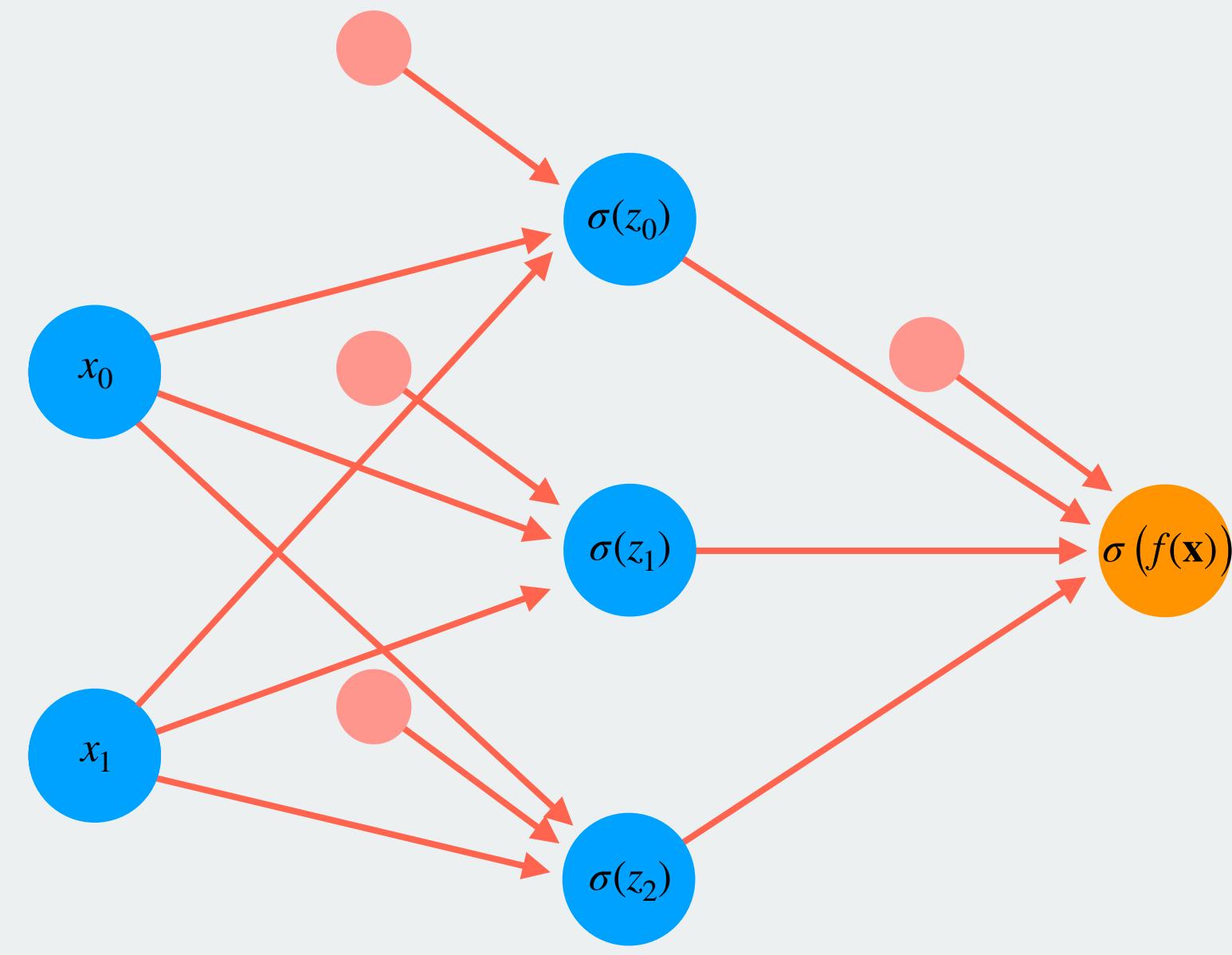
\Leftrightarrow

$$\sigma(\mathbf{b}_1 + \mathbf{W}_1 \sigma(\mathbf{b}_0 + \mathbf{W}_0 \mathbf{x})) = \sigma(f(x))$$

$$\mathbf{b}_0 = \begin{bmatrix} w_{0,0} \\ w_{3,0} \\ w_{6,0} \end{bmatrix} \quad \mathbf{W}_0 = \begin{bmatrix} w_{1,0} & w_{2,0} \\ w_{4,0} & w_{5,0} \\ w_{7,0} & w_{8,0} \end{bmatrix} \quad \mathbf{b}_1 = \begin{bmatrix} w_{0,1} \end{bmatrix} \quad \mathbf{W}_1 = \begin{bmatrix} w_{1,1} & w_{2,1} & w_{3,1} \end{bmatrix}$$

and $\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}$

Multilayer perceptron – the math



Mathematical form

$$\sigma(w_{0,1} + \sigma(w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} + \sigma(w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} + \sigma(w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1}) = \sigma(f(x))$$

\Leftrightarrow

$$\sigma(\mathbf{b}_1 + \mathbf{W}_1 \sigma(\mathbf{b}_0 + \mathbf{W}_0 \mathbf{x})) = \sigma(f(x))$$

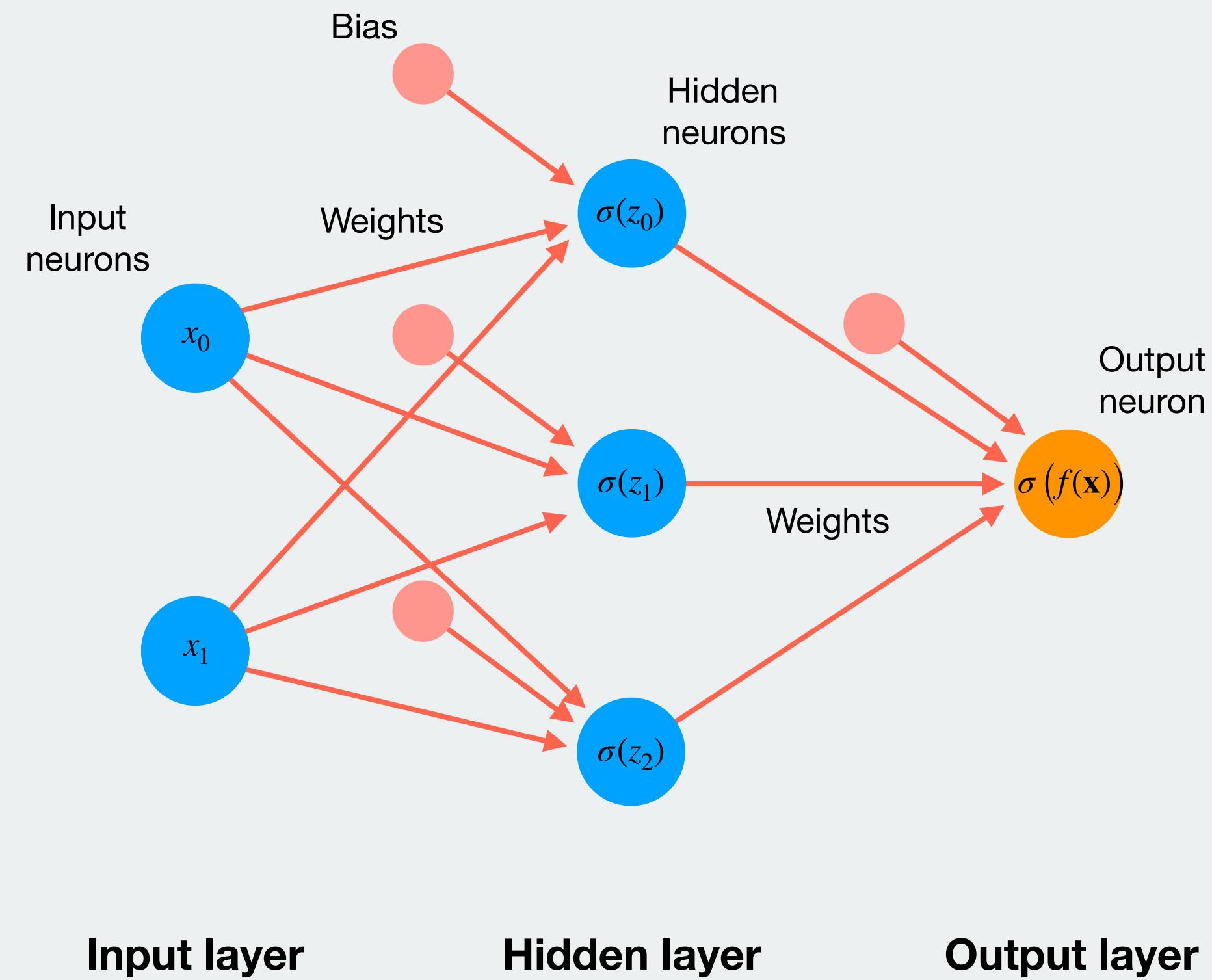
\Leftrightarrow

$$\sigma(\mathbf{b}_n + \mathbf{W}_n \mathbf{a}_{n-1}) = \mathbf{a}_n$$

$$\mathbf{b}_0 = \begin{bmatrix} w_{0,0} \\ w_{3,0} \\ w_{6,0} \end{bmatrix} \quad \mathbf{W}_0 = \begin{bmatrix} w_{1,0} & w_{2,0} \\ w_{4,0} & w_{5,0} \\ w_{7,0} & w_{8,0} \end{bmatrix} \quad \mathbf{b}_1 = \begin{bmatrix} w_{0,1} \end{bmatrix} \quad \mathbf{W}_1 = \begin{bmatrix} w_{1,1} & w_{2,1} & w_{3,1} \end{bmatrix}$$

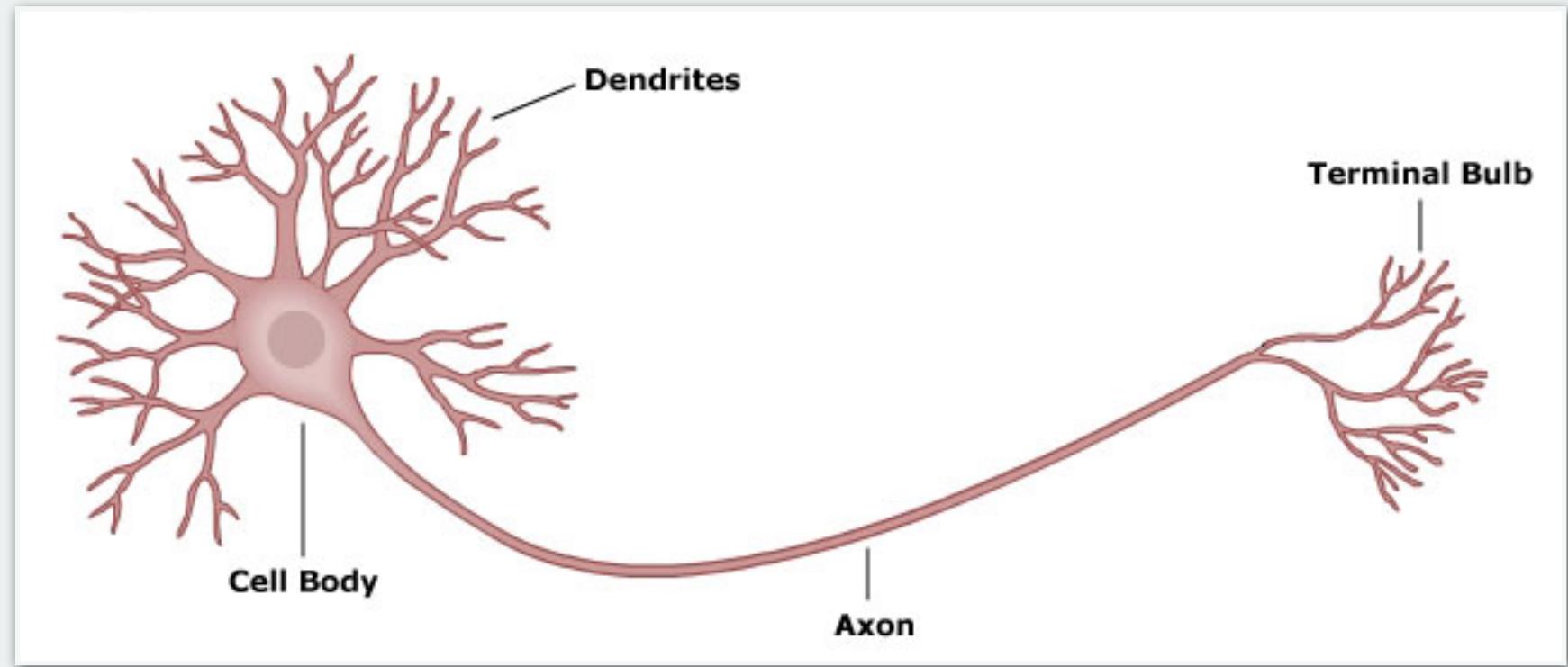
and $\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}$

Multilayer perceptron – the structure



Multilayer perceptron – why “Neural”?

Brain neuron



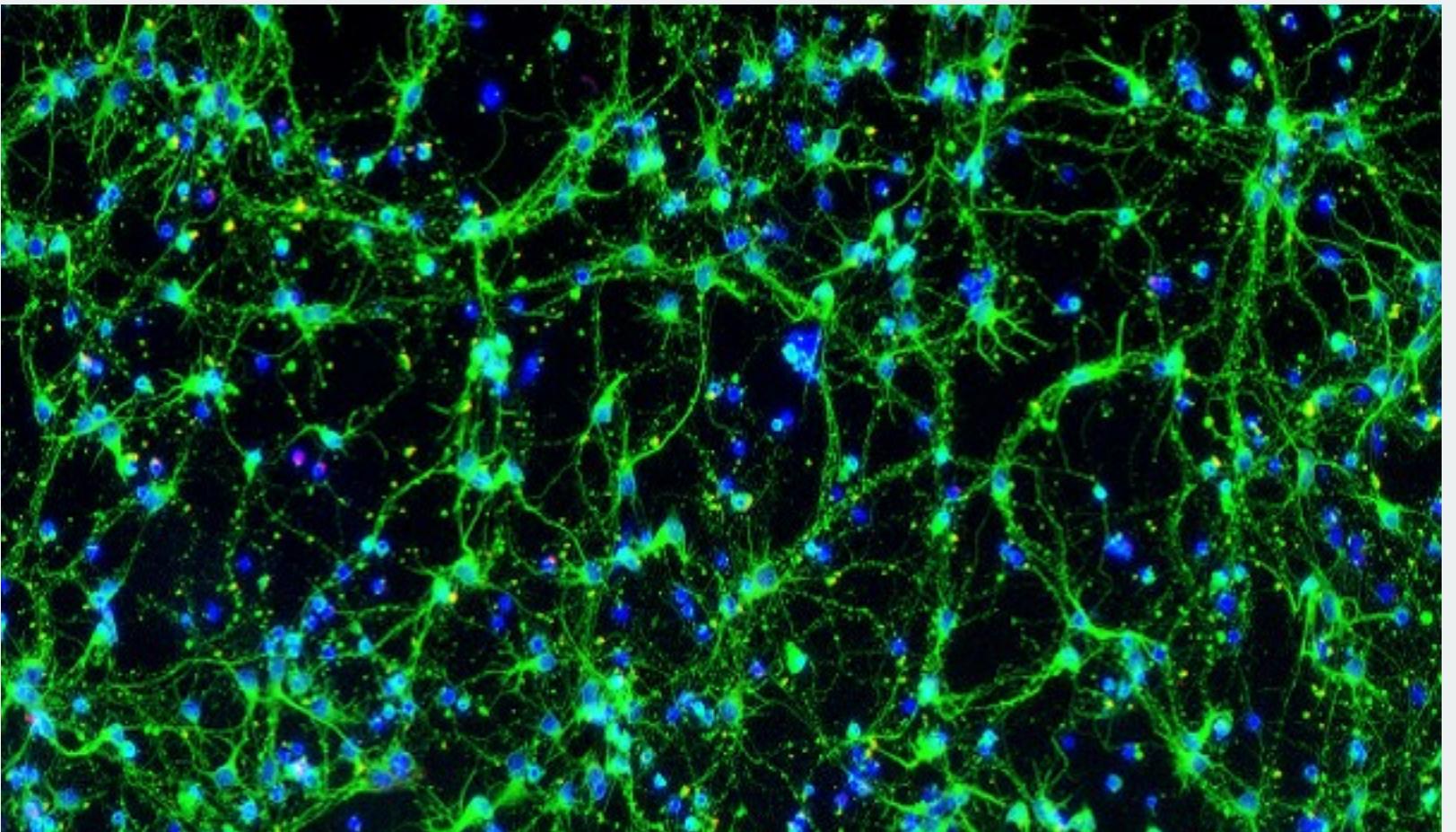
<https://neurofantastic.com/brain/2017/4/13/brain-computation-is-a-lot-more-analog-than-we-thought>

3D artist impression



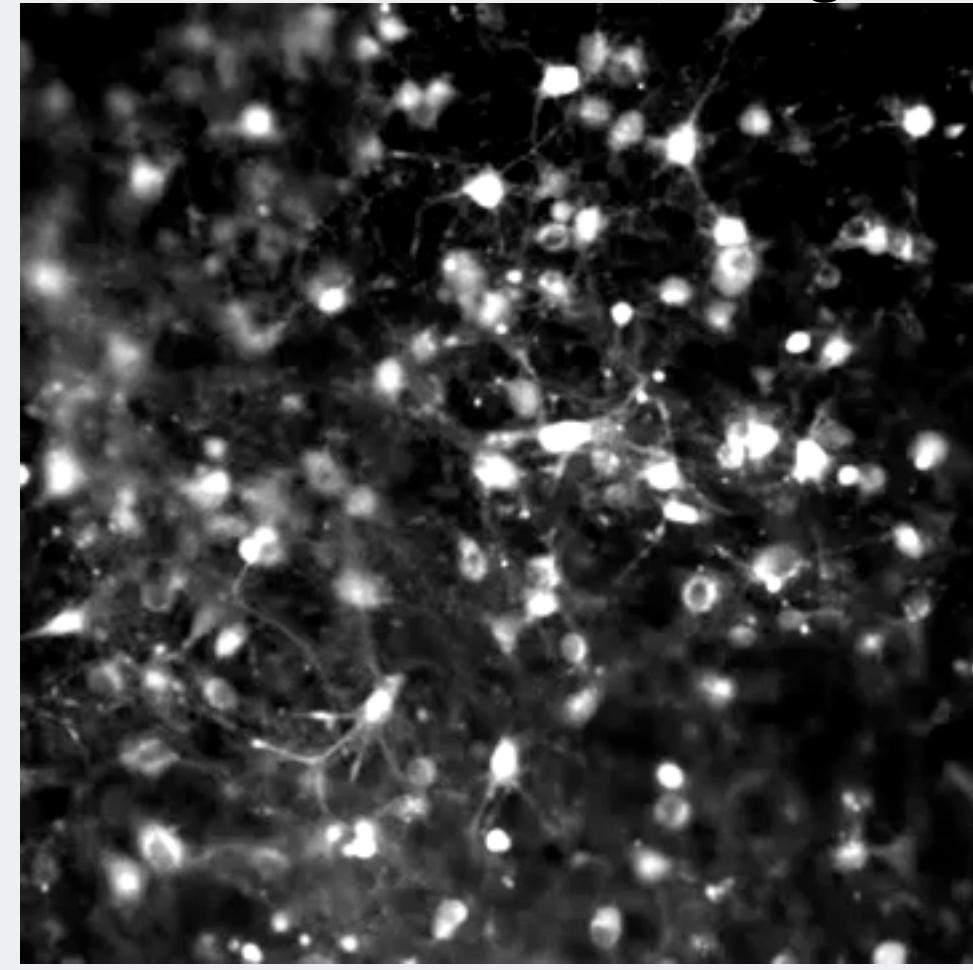
<https://alsnewstoday.com/2017/07/25/als-study-shows-how-excessive-dna-repetitions-trigger-neuron-deaths/>

Rat neuron image



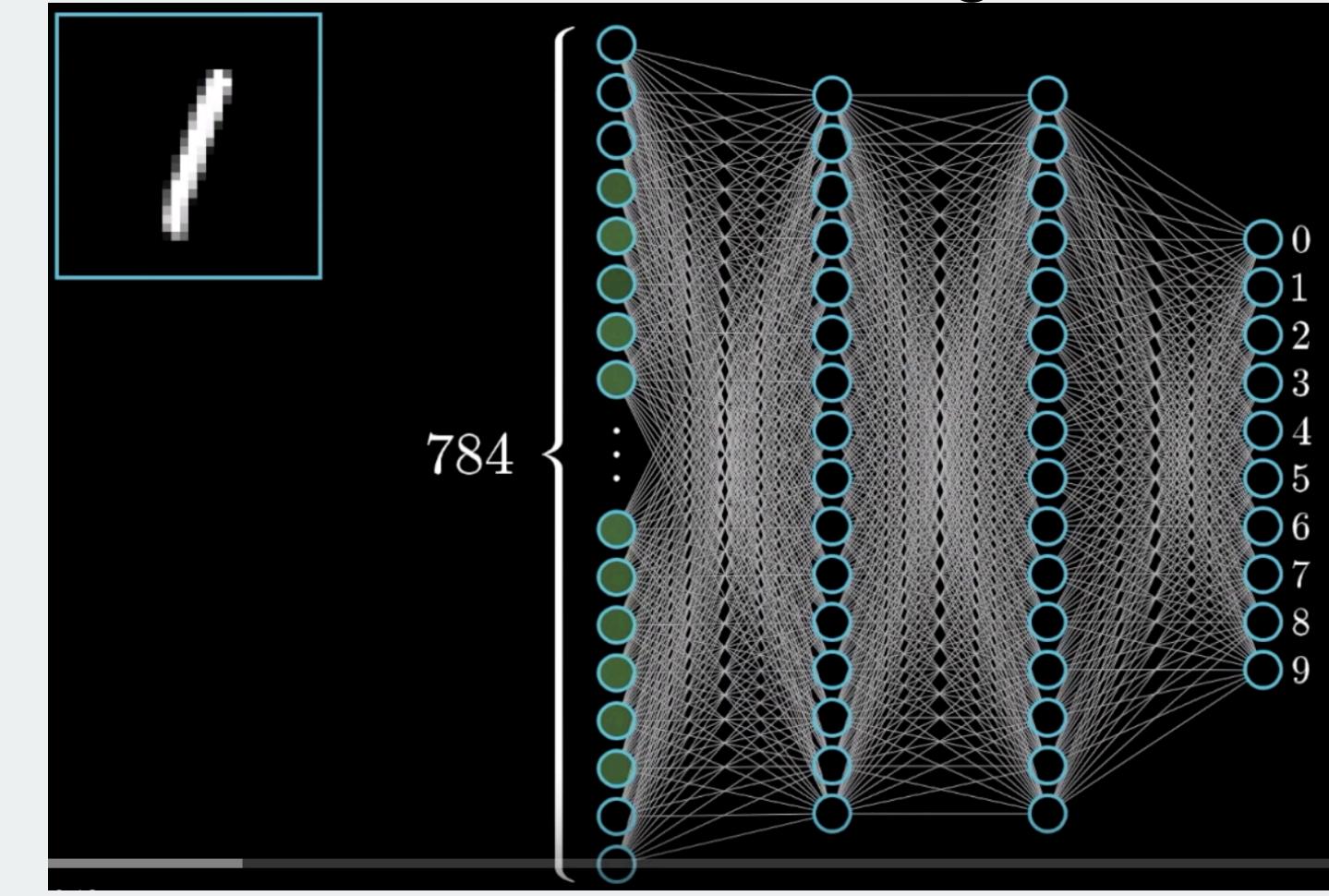
<https://www.cellapplications.com/expanded-neuron-offering>

Live rat neurons firing



<https://www.youtube.com/watch?v=yy994HpFudc>

Live artificial neurons firing



<https://www.youtube.com/watch?v=aircAruvnKk&t=1s>

Multilayer perceptron – why use non-linearities?

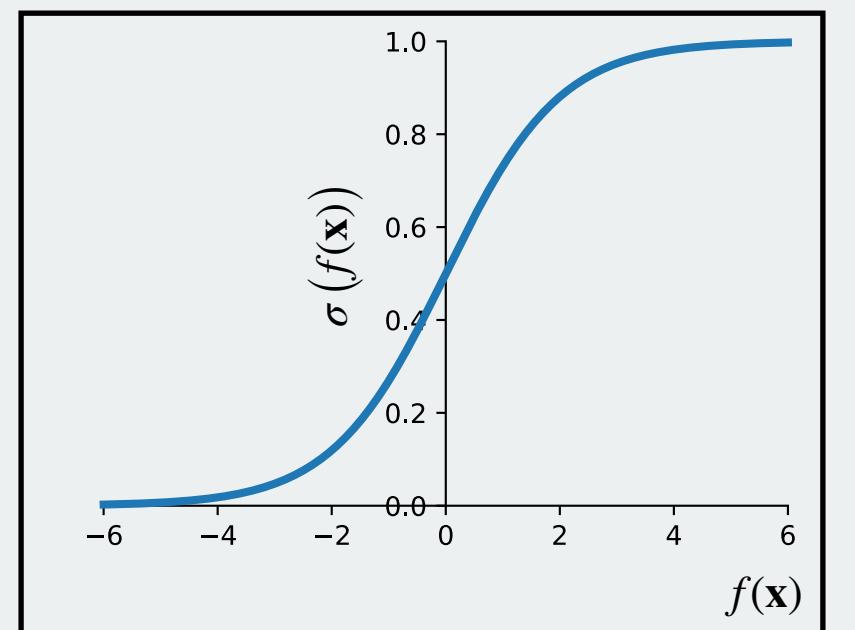
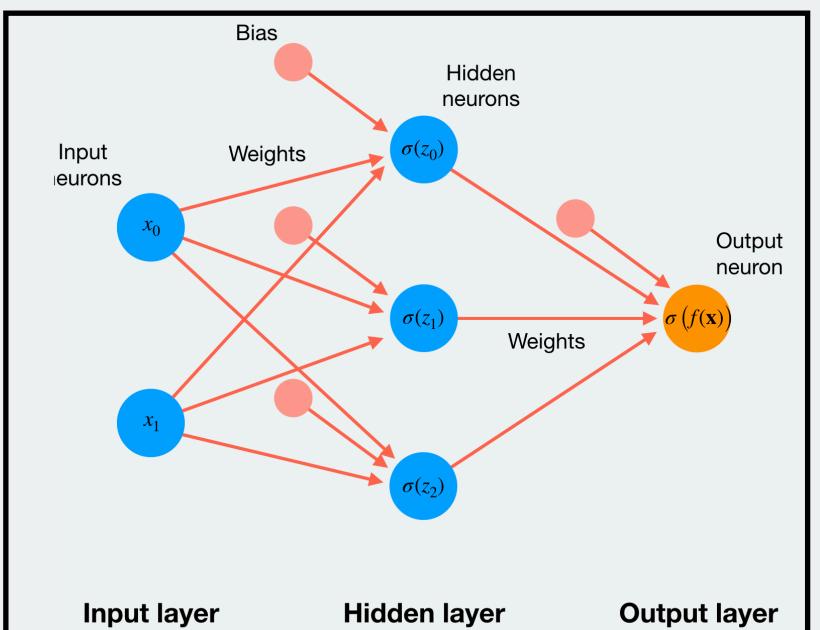
Mathematical form

$$\begin{aligned}\sigma(w_{0,1} + \sigma(w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} &+ \\ \sigma(w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} &+ \\ \sigma(w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1}) &= \sigma(f(x))\end{aligned}$$

Mathematical form without non-linearities

$$\begin{aligned}w_{0,1} + (w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} &+ \\ (w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} &+ \\ (w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1} &= f(x)\end{aligned}$$

Reminders



Multilayer perceptron – why use non-linearities?

Mathematical form

$$\begin{aligned} \sigma(w_{0,1} + \sigma(w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} &+ \\ \sigma(w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} &+ \\ \sigma(w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1}) &= \sigma(f(x)) \end{aligned}$$

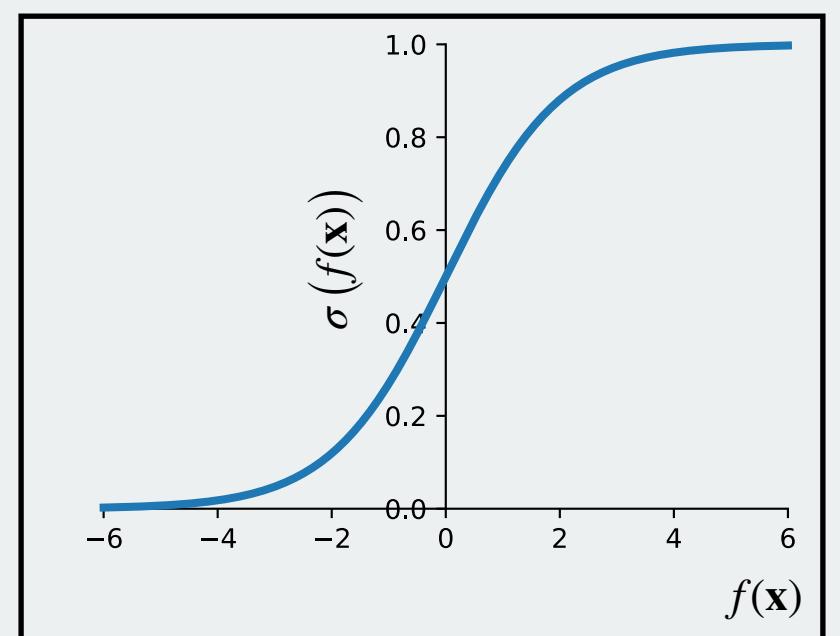
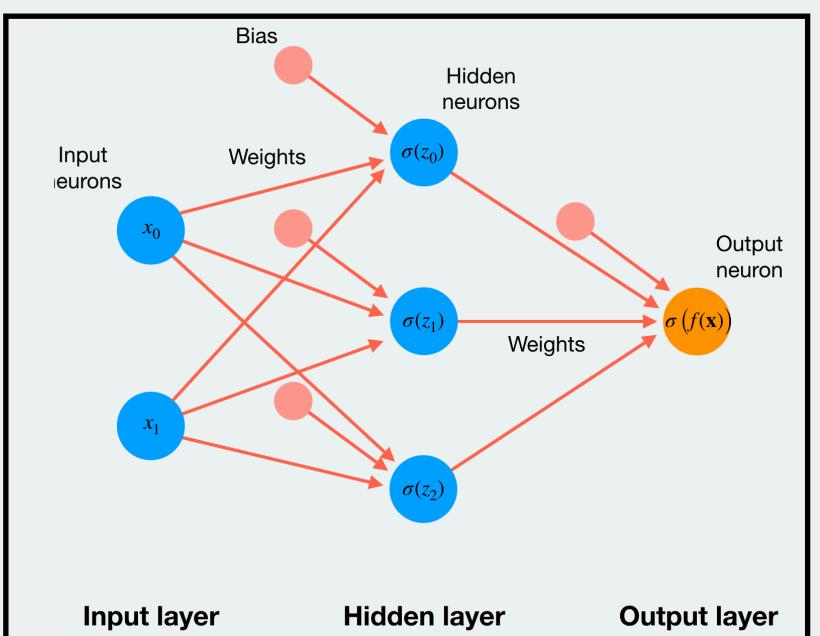
Mathematical form without non-linearities

$$\begin{aligned} w_{0,1} + (w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} &+ \\ (w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} &+ \\ (w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1} &= f(x) \end{aligned}$$

\Leftrightarrow

$$\begin{aligned} x_0(w_{1,0}w_{1,1} + w_{4,0}w_{2,1} + w_{7,0}w_{3,1}) &+ \\ x_1(w_{2,0}w_{1,1} + w_{5,0}w_{2,1} + w_{8,0}w_{3,1}) &+ \\ w_{0,1} + w_{0,0}w_{1,1} + w_{3,0}w_{2,1} + w_{6,0}w_{3,1} &= f(x) \end{aligned}$$

Reminders



Multilayer perceptron – why use non-linearities?

Mathematical form

$$\begin{aligned} \sigma(w_{0,1} + \sigma(w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} &+ \\ \sigma(w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} &+ \\ \sigma(w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1}) &= \sigma(f(x)) \end{aligned}$$

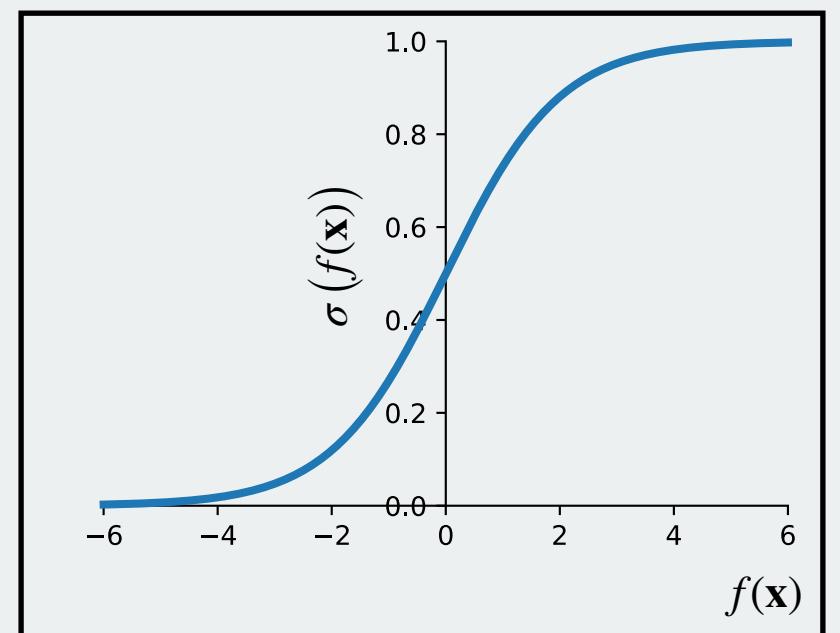
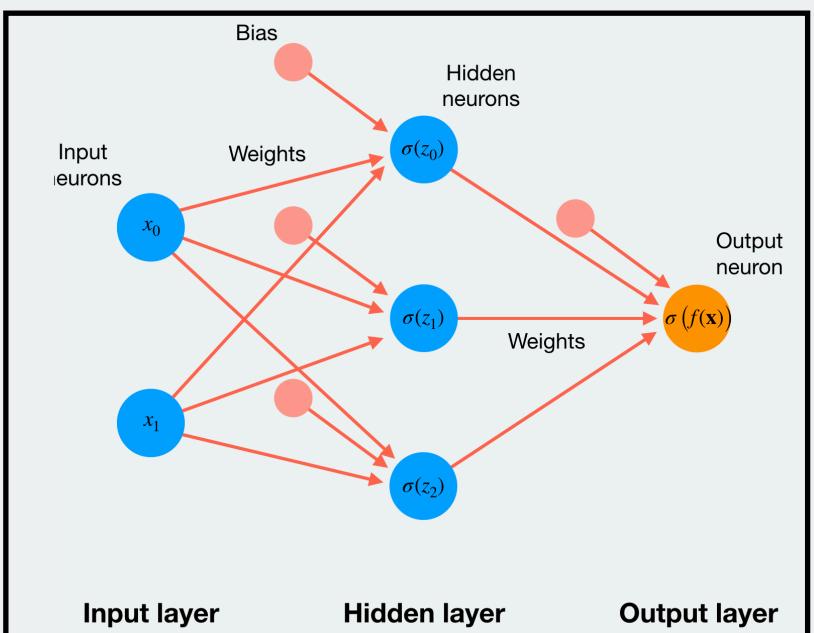
Mathematical form without non-linearities

$$\begin{aligned} w_{0,1} + (w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} &+ \\ (w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} &+ \\ (w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1} &= f(x) \end{aligned}$$

↔

$$\begin{aligned} x_0(w_{1,0}w_{1,1} + w_{4,0}w_{2,1} + w_{7,0}w_{3,1}) &+ \\ x_1(w_{2,0}w_{1,1} + w_{5,0}w_{2,1} + w_{8,0}w_{3,1}) &+ \\ w_{0,1} + w_{0,0}w_{1,1} + w_{3,0}w_{2,1} + w_{6,0}w_{3,1} &= f(x) \end{aligned}$$

Reminders



Multilayer perceptron – why use non-linearities?

Mathematical form

$$\begin{aligned} \sigma(w_{0,1} + \sigma(w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} &+ \\ \sigma(w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} &+ \\ \sigma(w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1}) &= \sigma(f(x)) \end{aligned}$$

Mathematical form without non-linearities

$$\begin{aligned} w_{0,1} + (w_{0,0} + x_0 w_{1,0} + x_1 w_{2,0})w_{1,1} &+ \\ (w_{3,0} + x_0 w_{4,0} + x_1 w_{5,0})w_{2,1} &+ \\ (w_{6,0} + x_0 w_{7,0} + x_1 w_{8,0})w_{3,1} &= f(x) \end{aligned}$$

\Leftrightarrow

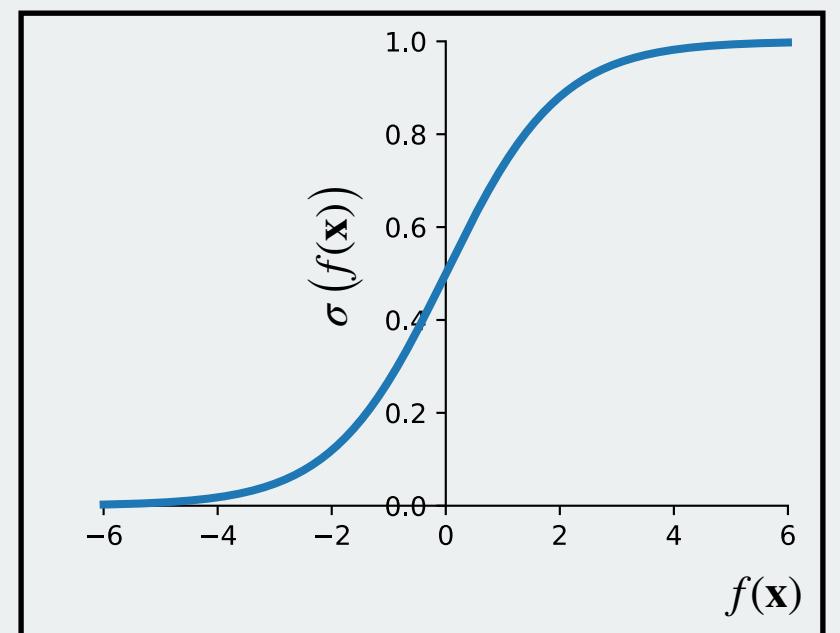
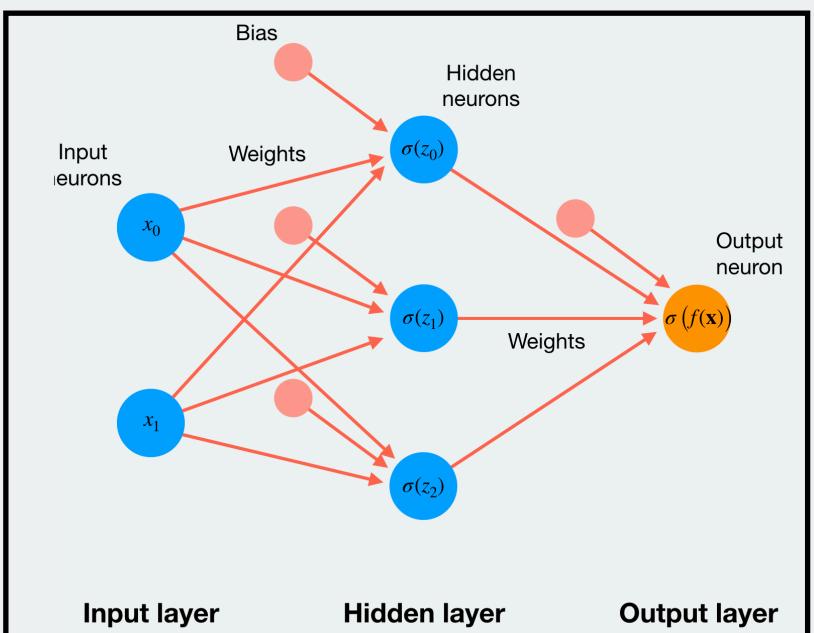
$$\begin{aligned} x_0(w_{1,0}w_{1,1} + w_{4,0}w_{2,1} + w_{7,0}w_{3,1}) &+ \\ x_1(w_{2,0}w_{1,1} + w_{5,0}w_{2,1} + w_{8,0}w_{3,1}) &+ \\ w_{0,1} + w_{0,0}w_{1,1} + w_{3,0}w_{2,1} + w_{6,0}w_{3,1} &= f(x) \end{aligned}$$

\Leftrightarrow

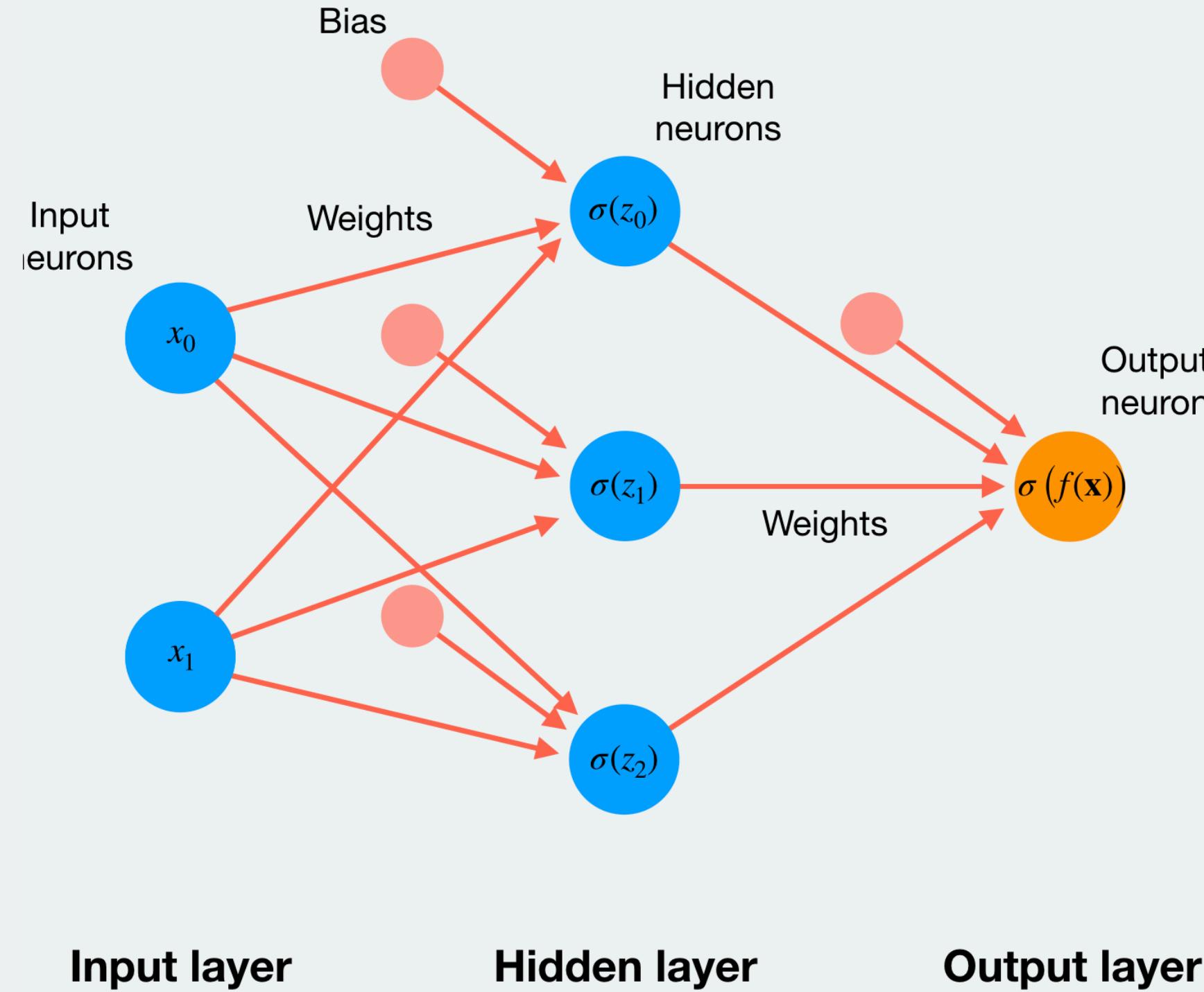
$$x_0w_a + x_1w_b + w_c = f(x)$$

omg! nested linear regression
is actually just linear regression

Reminders

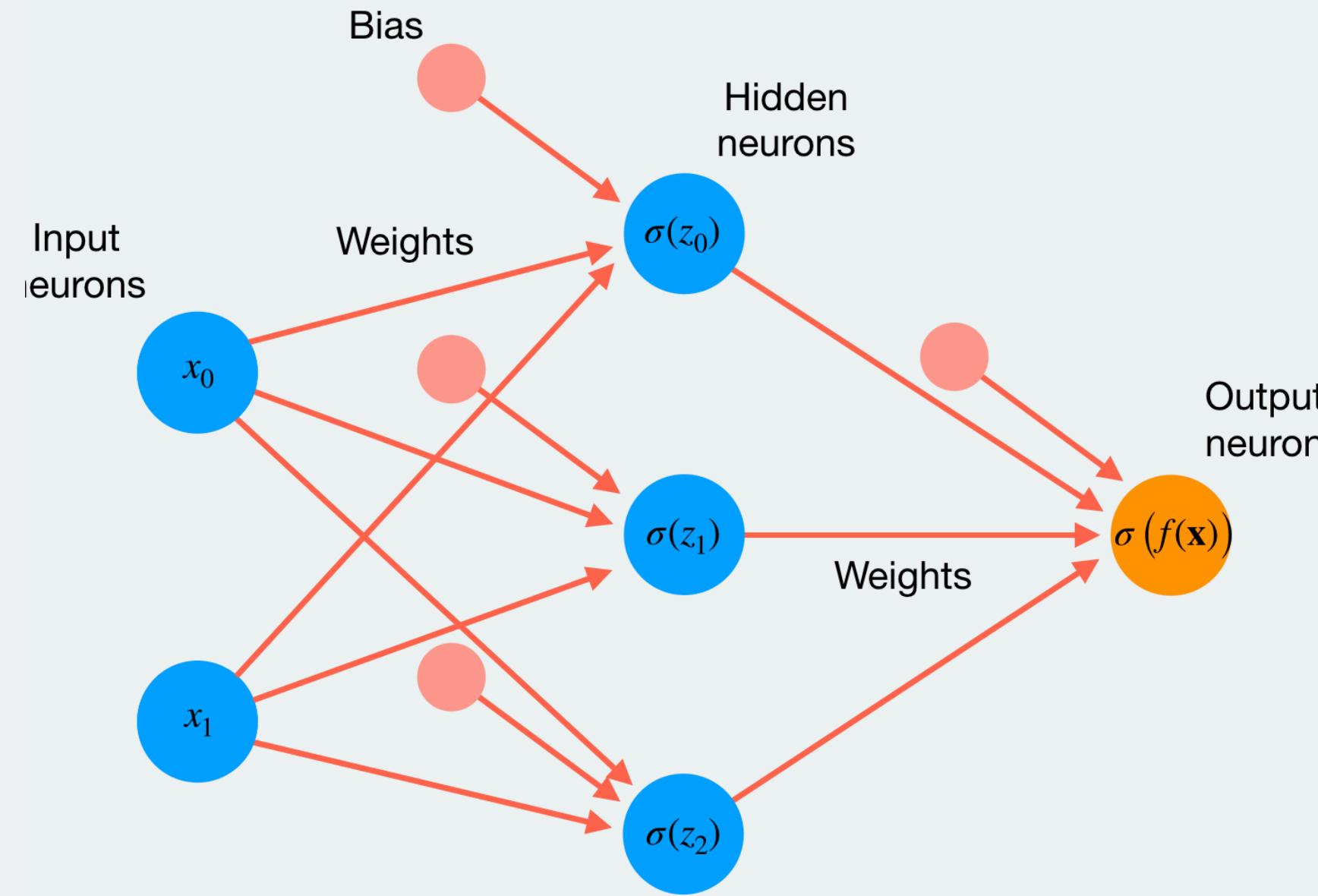


Multilayer perceptron – overview



- Forward-coupled logistic regressions
- Activation function **necessary**
- Number of *layers* refers to number of layers of weights (left, 2)
- Can be *deep*, i.e. have many *hidden* layers
- Dense/fully-connected layers
- Can have multiple output neurons
- More general name: **Feed forward neural network**

Multilayer perceptron – overview



- Forward-coupled logistic regressions
- Activation function **necessary**
- Number of *layers* refers to number of layers of weights (left, 2)
- Can be *deep*, i.e. have many *hidden layers*
- Dense/fully-connected layers
- Can have multiple output neurons
- More general name: **Feed forward neural network**

What I haven't covered yet

- Backpropagation
 - Gradient descent
 - Special architectures
- i.e. how neural nets learn / fit to data*

Exercises:

- Play with a live neural network in your browser
- Understand and explain (part of) a pure Python code

implementation of a neural network