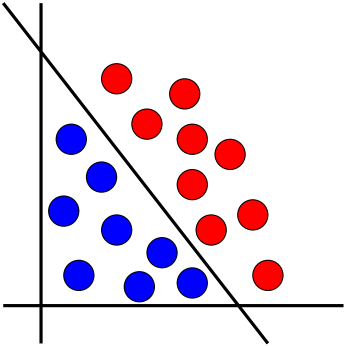
Neural Networks

STAT 641

Perceptron and logistic regression classifiers have linear decision boundaries.



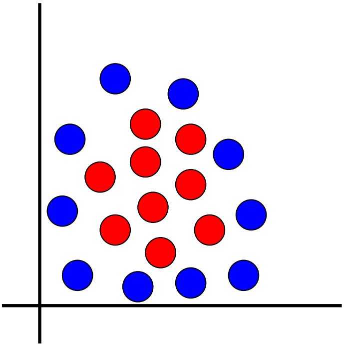


Figure: Linearly separable, not linearly separable

Perceptron, logistic regression **can**[1] learn:

$$f : \big\{(0,0),(1,0),(1,1),(0,1)\big\} \longrightarrow \{0,1\}$$

$$f(0,0) = 1, \quad f(1,0) = f(1,1) = f(0,1) = 0$$

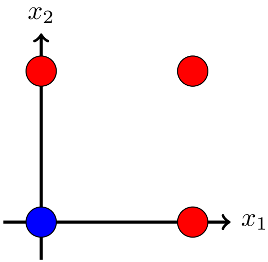


Figure: Graph of f

[1]Maximum likelihood estimates for the parameters don't exist.

Perceptron, logistic regression **can**[1] learn:

$$f : \{(0,0),(1,0),(1,1),(0,1)\} \longrightarrow \{0,1\}$$

$$f(0,0) = 1, \quad f(1,0) = f(1,1) = f(0,1) = 0$$

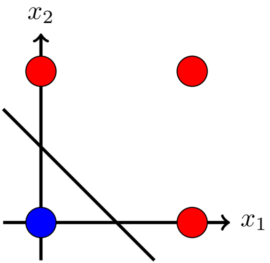
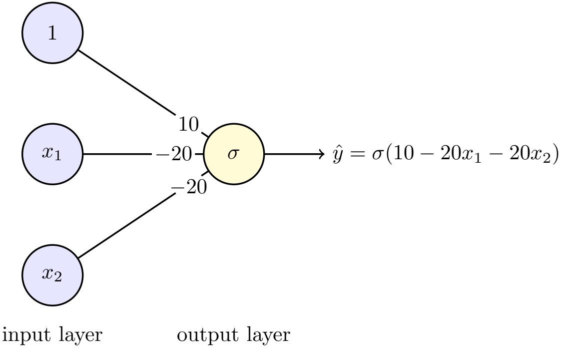
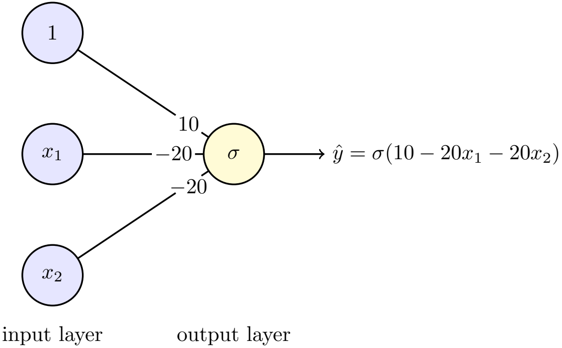


Figure: Graph of f

[1] Maximum likelihood estimates for the parameters don't exist.



x_1	x_2	$\widehat{y}$
0	0	
1	0	
0	1	
1	1	



x_1	x_2	$\widehat{y}$
0	0	$\sigma(10) \approx 1$
1	0	$\sigma(-10) \approx 0$
0	1	$\sigma(-10) \approx 0$
1	1	$\sigma(-30) \approx 0$

Perceptron, logistic regression **can't** learn...

$$f : \{(0,0),(1,0),(1,1),(0,1)\} \longrightarrow \{0,1\}$$

$$f(0,0) = f(1,1) = 1, \quad f(1,0) = f(0,1) = 0$$

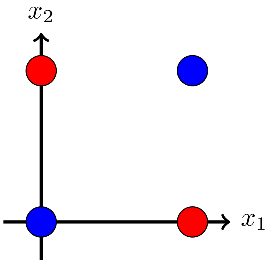
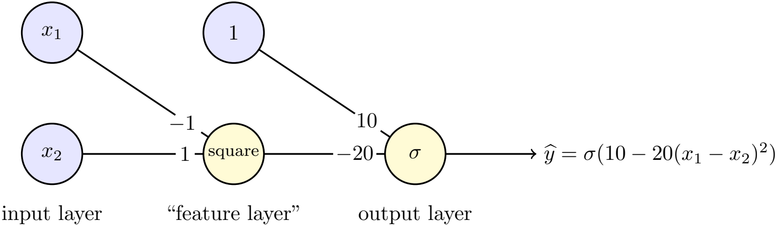


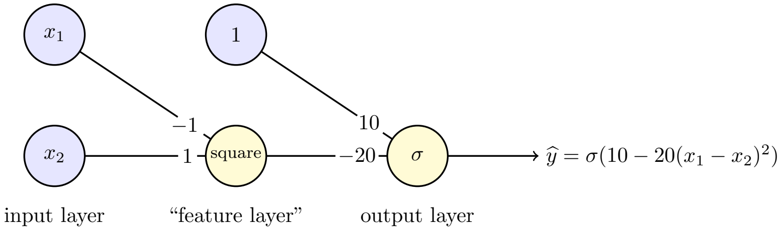
Figure: Graph of f

... unless we introduce new features.



x_1	x_2	$\widehat{y}$
0	0	
1	0	
0	1	
1	1	

... unless we introduce new features.



x_1	x_2	$\widehat{y}$
0	0	$\sigma(10) \approx 1$
1	0	$\sigma(-10) \approx 0$
0	1	$\sigma(-10) \approx 0$
1	1	$\sigma(10) \approx 1$

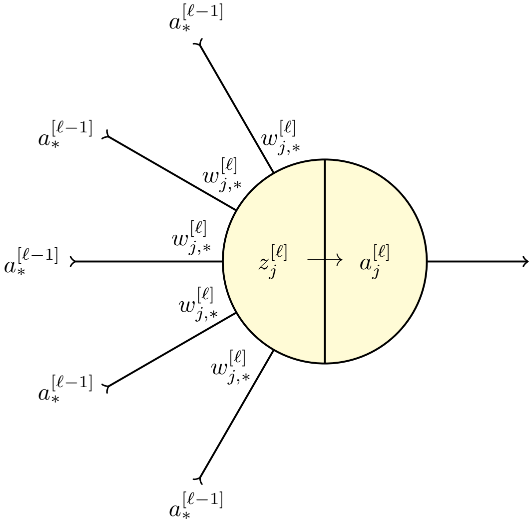
What is a neural network?

A neural network consists of **neurons** or **units**. Neurons have **inputs**, **outputs**, and **activations**. Connections between these neurons have **weights**. Neurons are organized into **layers**. **Hidden layers** are sandwiched between an **input layer** and an **output layer**.

Linear regression, logistic regression, and perceptron classification are all neural networks, degenerate in the sense that they have no hidden layers.

depth of network (number of layers):	L
number of neurons in layer ℓ:	p_ℓ
activation (output) of neuron k in layer ℓ:	$a_k^{[\ell]}$
bias of neuron k in layer ℓ:	$b_k^{[\ell]}$
weight of the connection between neuron i in layer $\ell - 1$ and neuron j in layer ℓ:	$w_{ji}^{[\ell]}$
activation function in layer ℓ:	h_ℓ

$$a_i^{[\ell+1]} = h\left(z_i^{[\ell+1]}\right), \quad \text{where} \quad z_i^{[\ell+1]} = b_i^{[\ell]} + \sum_{j=1}^{p_\ell} w_{ij}^{[\ell]} a_j^{[\ell]}$$



$$z_j^{[\ell]} = \sum_i w_{j,i}^{[\ell]} a_i^{[\ell-1]}$$

$$a_j^{[\ell]} = A_\ell\big(z_j^{[\ell]}\big)$$

Figure: A hidden or output unit

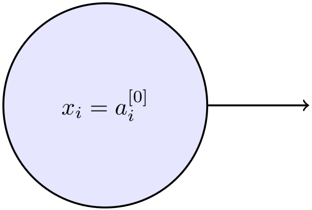


Figure: An input unit

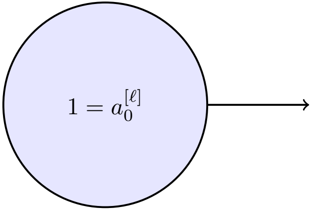


Figure: A bias unit

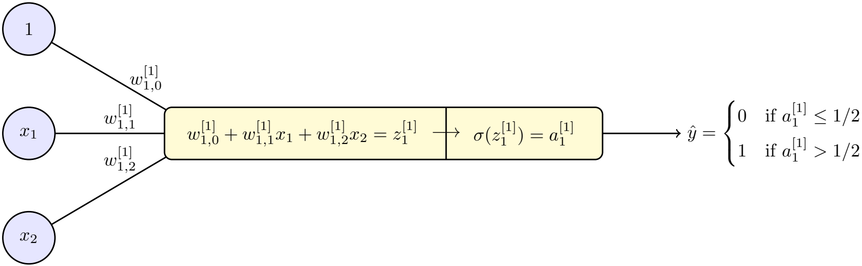


Figure: Logistic regression

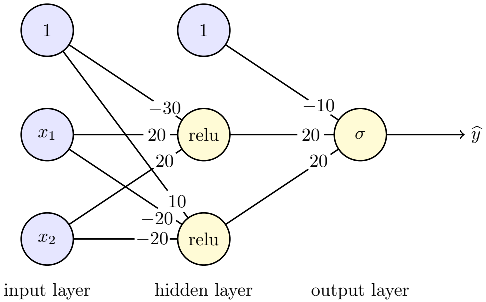


Figure: Learning XNOR

x_1	x_2	$z_1^{[1]}$	$z_2^{[1]}$	$a_1^{[1]}$	$a_2^{[1]}$	$z_1^{[2]}$	$a_1^{[2]}$	$\widehat{y}$
0	0	−30	10	0	10	190	1.00	1
0	1	−10	−10	0	0	−10	0.00	0
1	0	−10	−10	0	0	−10	0.00	0
1	1	−10	−10	10	0	190	1.00	1