Linear Regression, Neural Networks, etc.

Gradient Descent

- Many machine learning problems can be cast as optimization problems
  - Define a function that corresponds to learning error. (More on this later)
  - Minimize the function.
- One possible approach (maximization):
  1) take the derivative of the function: $f'(x)$
  2) *guess a value of x:* $\hat{x}$
  3) move $\hat{x}$ a little bit according to the derivative: $\hat{x} \leftarrow \hat{x} + \alpha f'(\hat{x})$
  4) goto 3, repeat.
- Example...

Partial Derivatives

- Derivative of a function of multiple variables, with all but the variable of interest held constant.

$$f(x, y) = x^2 + xy^2$$

$$f_x(x, y) = 2\mathrm{x} + y^2$$

OR

$$\frac{\partial f(x, y)}{\partial x} = 2\mathrm{x} + y^2$$

$$f_y(x, y) = 2\mathrm{xy}$$

OR

$$\frac{\partial f(x, y)}{\partial y} = 2\mathrm{xy}$$

Gradient

- The gradient is just the generalization of the derivative to multiple dimensions.

$$\nabla f(\boldsymbol{w}) = \begin{bmatrix} \frac{\partial f(\boldsymbol{w})}{\partial w_1} \\ \frac{\partial f(\boldsymbol{w})}{\partial w_2} \\ \vdots \\ \frac{\partial f(\boldsymbol{w})}{\partial w_n} \end{bmatrix}$$

- Gradient descent update:

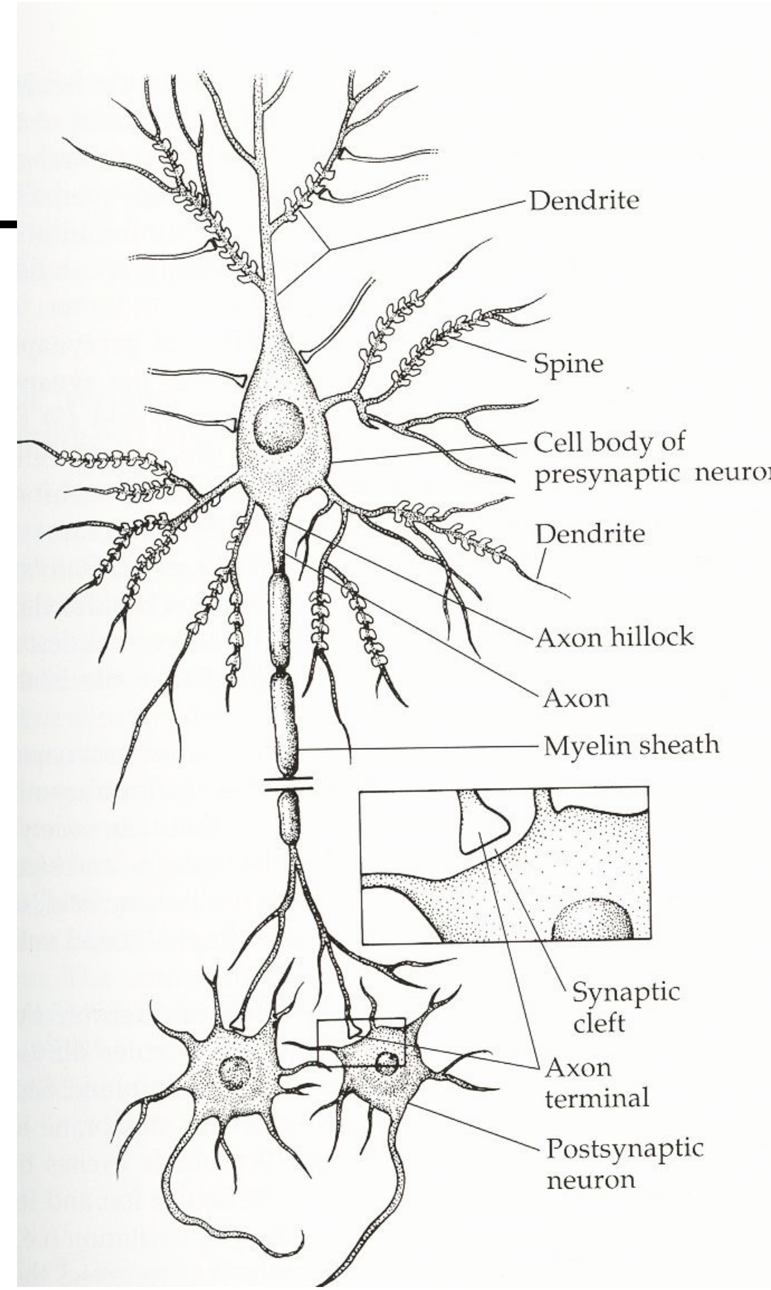
$$\hat{\boldsymbol{w}} \leftarrow \hat{\boldsymbol{w}} - \alpha \nabla f(\hat{\boldsymbol{w}})$$

The Brain

- The human brain weighs about three pounds.
- Has around 10^{11} neurons.
- About 10^{14} connections between neurons (synapses).

Neurons

- Neurons communicate using discrete electrical signals called “spikes” (or action potentials).
  - Spikes travel along axons.
  - Reach axon terminals.
  - Terminals release neurotransmitters.
  - Postsynaptic neurons respond by allowing current to flow in (or out).
  - If voltage crosses a threshold a spike is created



Neuroanatomy, Martin, 1996

Neuron Communication Factoids

- Typically neurons have many dendrites (inputs), but only a single axon (output).
- Dendrites tend to be short, while axons can be very long.
  - Dendrites passively transmit current.
  - Axons actively propagate signals.
- Maximum firing rate for neurons is about 1000 HZ.
- Different synapses have different strengths.
  - I.e. a spike may result in more or less current entering the cell.
- The strength of synapses changes over time.

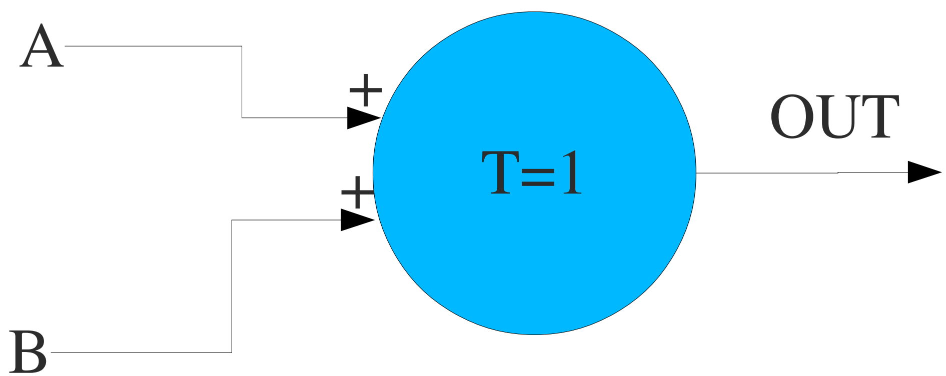
What Are Neurons Doing?

- By the early 1940's the gist of what individual neurons were doing was known.
- By the 1950's we knew pretty much everything from the last several slides.
- We knew what individual neurons were doing, but not how they work together to perform computation.
- What was needed was an abstract model of the neuron.
- The first plausible account came from McCulloch and Pitts in 1943.

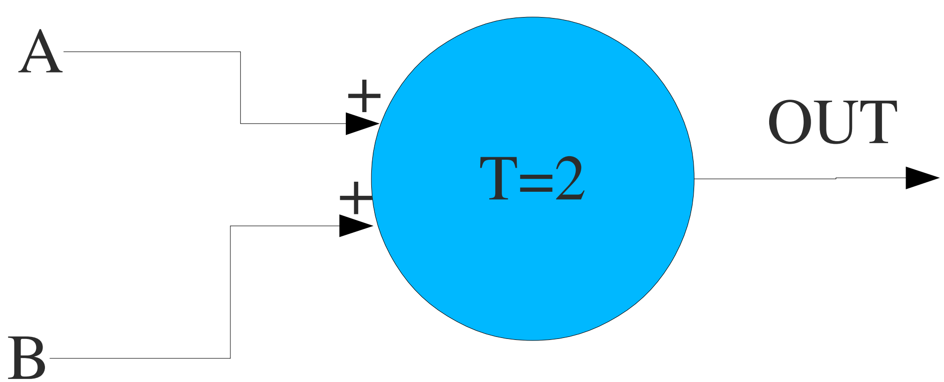
McCulloch Pitts Neurons

- Inputs to a neuron can be either active (spiking) or inactive (not spiking).
- Inputs may be excitatory, or inhibitory.
- Excitatory inputs are summed.
- If the sum exceeds some threshold, then the neuron becomes active.
- If any inhibitory input is active, then the cell does not fire.

McCulloch Pitts Examples



A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	1



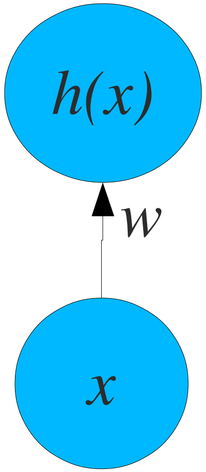
A	B	OUT
0	0	0
0	1	0
1	0	0
1	1	1

Great! Logic!

- Any logical proposition (or digital circuit) can be expressed as a network of McCulloch Pitts neurons.
- The brain is a digital computer!
- A nice idea, but ...
- An important piece is missing – how could these networks learn?

Linear Regression – The Neural View

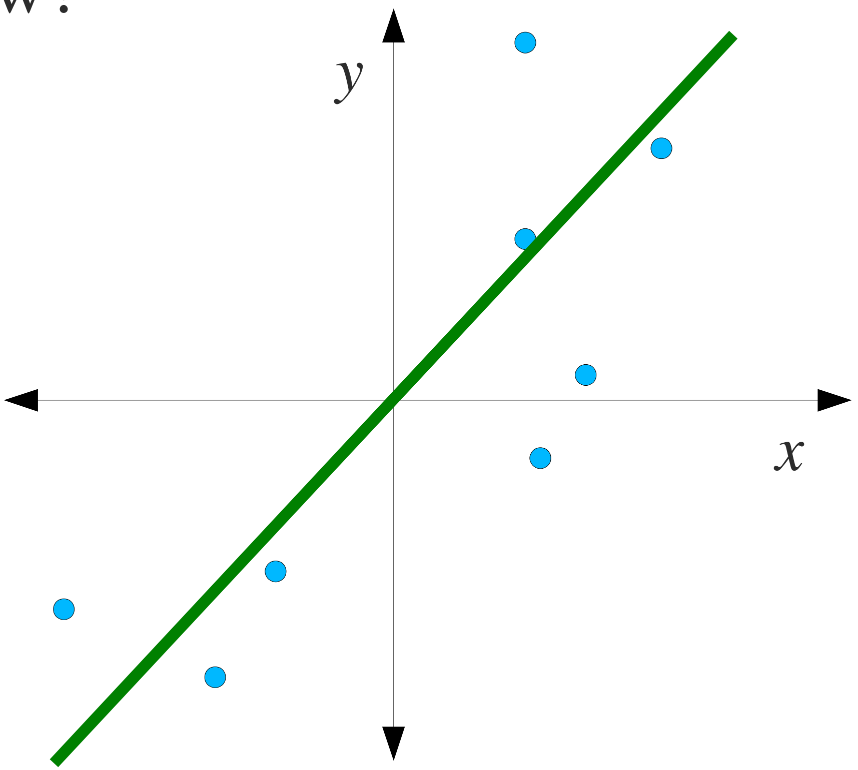
- input = x, desired output = y, weight = w.
- $h(x) = wx$



- We are given a set of inputs, and a corresponding set of outputs, and we need to choose w.
- What's going on geometrically?

Lines

- $h(x) = wx$ *is* the equation of a line with a y intercept of 0.
- What is the best value of w?
- How do we find it?

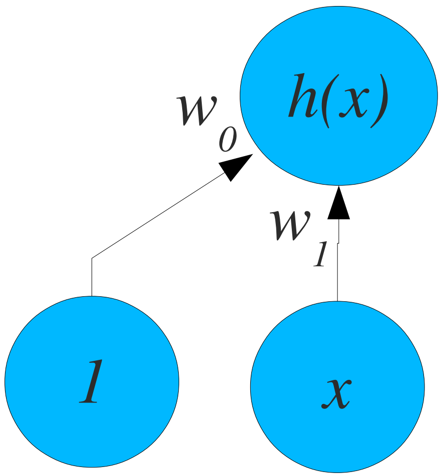


Bias Weights

- We need to use the general equation for a line:

$$h(x) = w_1 x + w_0$$

- This corresponds to a new neural network with one additional weight, and an input fixed at 1.



Error Metric

- Sum squared error (*y* is the desired output):

$$E=\sum_{j=1}^{N}\frac{1}{2}\left(y_j-h\left(x_j\right)\right)^2$$

$$=\sum_{j=1}^{N}\frac{1}{2}\left(y_j-\left(w_1 x_j+w_0\right)\right)^2$$

- The goal is to find a *w* that minimizes *E*. How?

The Wrong Way...

- Remember gradient descent?

$$\frac{\partial}{\partial w_1} \sum_j \frac{1}{2}\left(y_j - (w_1 x_j + w_0)\right)^2 =$$

$$\sum_j \left(y_j - (w_1 x_j + w_0)\right) \frac{\partial}{\partial w_1}\left(y_j - (w_1 x_j + w_0)\right) = \quad -\sum_j \left(y_j - (w_1 x_j + w_0)\right) x_j$$

$$= -\sum_j \left(y_j - h(x_j)\right) x_j$$

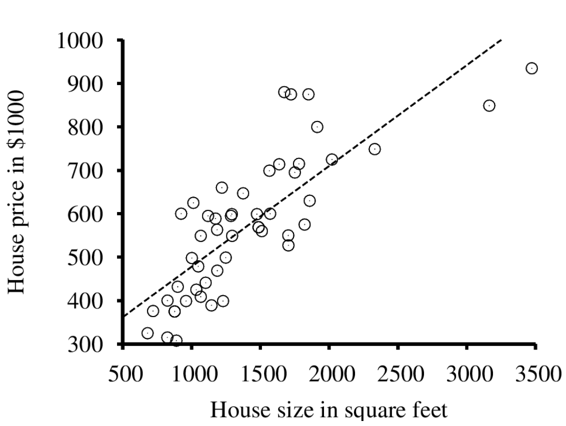
$$\frac{\partial}{\partial w_0} \sum_j \frac{1}{2}\left(y_j - (w_1 x_j + w_0)\right)^2 = \quad -\sum_j \left(y_j - h(x_j)\right)$$

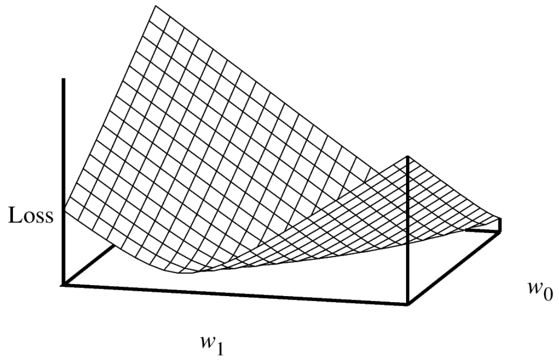
Update Rules...

$$w_1 \leftarrow w_1 + \alpha \sum_j \left(y_j - h\left(x_j \right) \right) x_j$$

$$w_0 \leftarrow w_0 + \alpha \sum_j \left(y_j - h\left(x_j \right) \right)$$

Visualizing Error





The Right Way...

- Set the partial derivatives to 0, and solve for w's:

$$\frac{\partial}{\partial w_0} \sum_{j=1}^{N} \frac{1}{2}\left(y_j - \left(w_1 x_j + w_0\right)\right)^2 = 0$$

$$\frac{\partial}{\partial w_1} \sum_{j=1}^{N} \frac{1}{2}\left(y_j - \left(w_1 x_j + w_0\right)\right)^2 = 0$$

- Result:

$$w_0 = \frac{\sum y_j - w_1 \left(\sum x_j\right)}{N} \qquad w_1 = \frac{N\left(\sum x_j y_j\right) - \left(\sum x_j\right)\left(\sum y_j\right)}{N\left(\sum x_j^2\right) - \left(\sum x_j\right)^2}$$

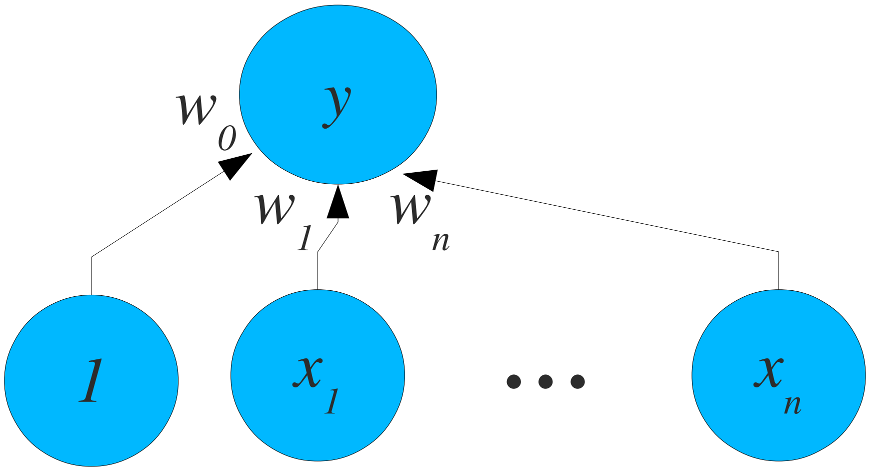
Multivariate Linear Regression

- Multi-dimensional input vectors:

$$y = w_0 + w_1 x_1 + \ldots + w_n x_n$$

- Or:

$$y = \boldsymbol{w}^T \boldsymbol{x}$$



Gradient Descent for MVLR

- Error for the multi-dimensional case:

$$E = \sum_j \frac{1}{2}\left(y_j - \boldsymbol{w}^T \boldsymbol{x}_j\right)^2$$

$$\frac{\partial E}{\partial w_i} = \sum_j \left(y_j - \boldsymbol{w}^T \boldsymbol{x}_j\right)\left(-x_{j,i}\right)$$

$$= -\sum_j \left(y_j - \boldsymbol{w}^T \boldsymbol{x}\right) x_{j,i}$$

- The new update rule: $w_i \leftarrow w_i + \alpha \sum_j \left(y_j - \boldsymbol{w}^T \boldsymbol{x}\right) x_{j,i}$

- Vector version: $\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha \sum_j \left(y_j - \boldsymbol{w}^T \boldsymbol{x}\right) \boldsymbol{x}_j$

Analytical Solution

$$\boldsymbol{w} = \left(X^T X\right)^{-1} X^T y$$

- Where X is a matrix with one input per row, y the vector of target values.

Notice that we get Polynomial Regression for Free

$$y = w_1 x^2 + w_2 x + w_0$$

In-Class Exercise...

- Here is the least squares error function for a single point:

$$E = \frac{1}{2}\left(y - \boldsymbol{w}^T \boldsymbol{x}\right)^2$$

- Why not try un-squared error (L1 error)?

$$E = \left|y - \boldsymbol{w}^T \boldsymbol{x}\right|$$

$$= \left|y - \left(w_0 x_0 + w_1 x_1 + \ldots + w_n x_n\right)\right|$$

- Your task... Develop a gradient descent learning rule for this new objective function. (helpful to remember that: $\frac{d}{dx}|u| = \frac{u \times u'}{|u|}$)

$$w_i \leftarrow w_i + \alpha\, ?\, ?$$

In-Class Exercise...

$$\frac{d}{dx}|u| = \frac{u \times u'}{|u|}$$

$$E = \left| y - \boldsymbol{w}^T \boldsymbol{x} \right|$$

$$\frac{\partial}{\partial w_i} \left| y - \left(w_0 x_0 + w_1 x_1 + \ldots + w_n x_n \right) \right| =$$

$$\frac{\left(y - \left(w_0 x_0 + w_1 x_1 + \ldots + w_n x_n \right) \right)}{\left| y - \left(w_0 x_0 + w_1 x_1 + \ldots w_n x_n \right) \right|} \frac{\partial}{\partial w_i} \left(y - \left(w_0 x_0 + w_1 x_1 + \ldots + w_n x_n \right) \right)$$

$$= -x_i \times sign(E)$$

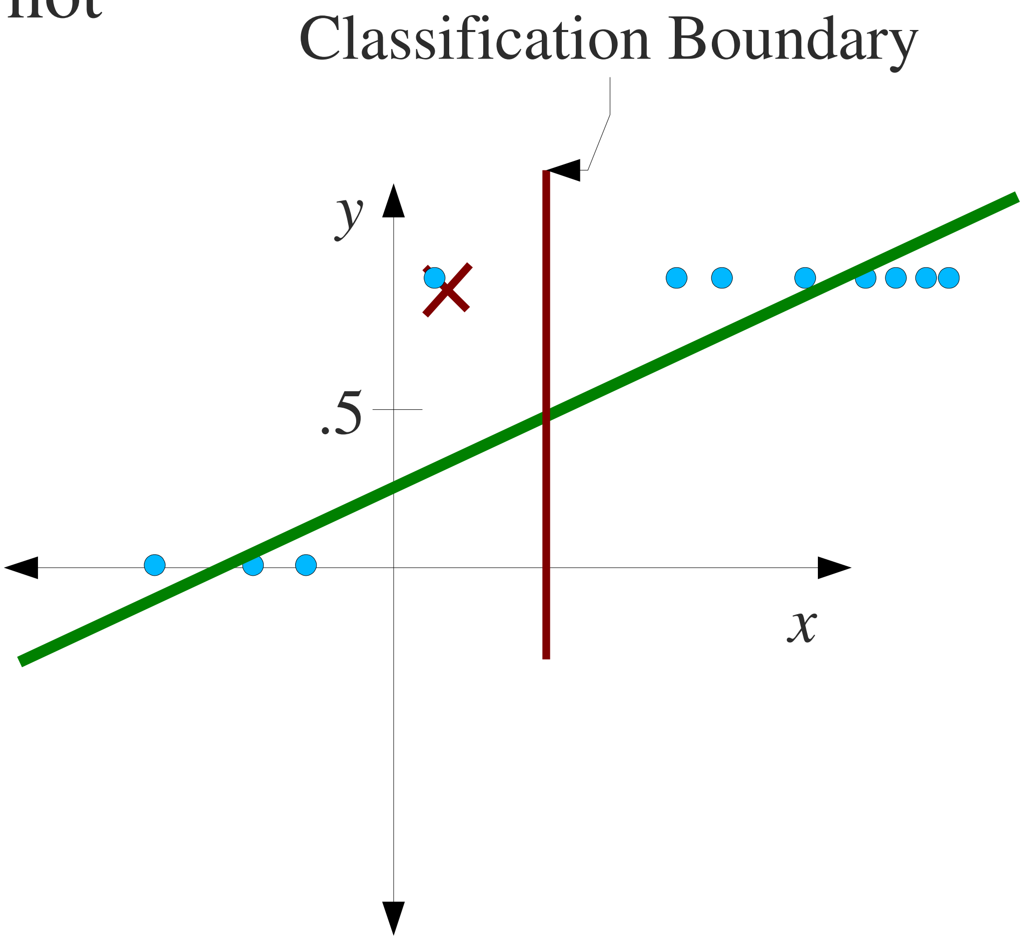
$$w_i \leftarrow w_i + \alpha \times x_i \times sign(E)$$

Regression vs. Classification

- Now we have the machinery to fit a line (plane, hyperplane) to a set of data points - regression.
- What about classification?
- First thought:
  - For each data point $\boldsymbol{x}$, set the value of y to be 0 or 1, depending on the class
  - Use linear regression to fit the data.
  - During classification assume class 0 if $y < .5$, assume class 1 if $y >= .5$.

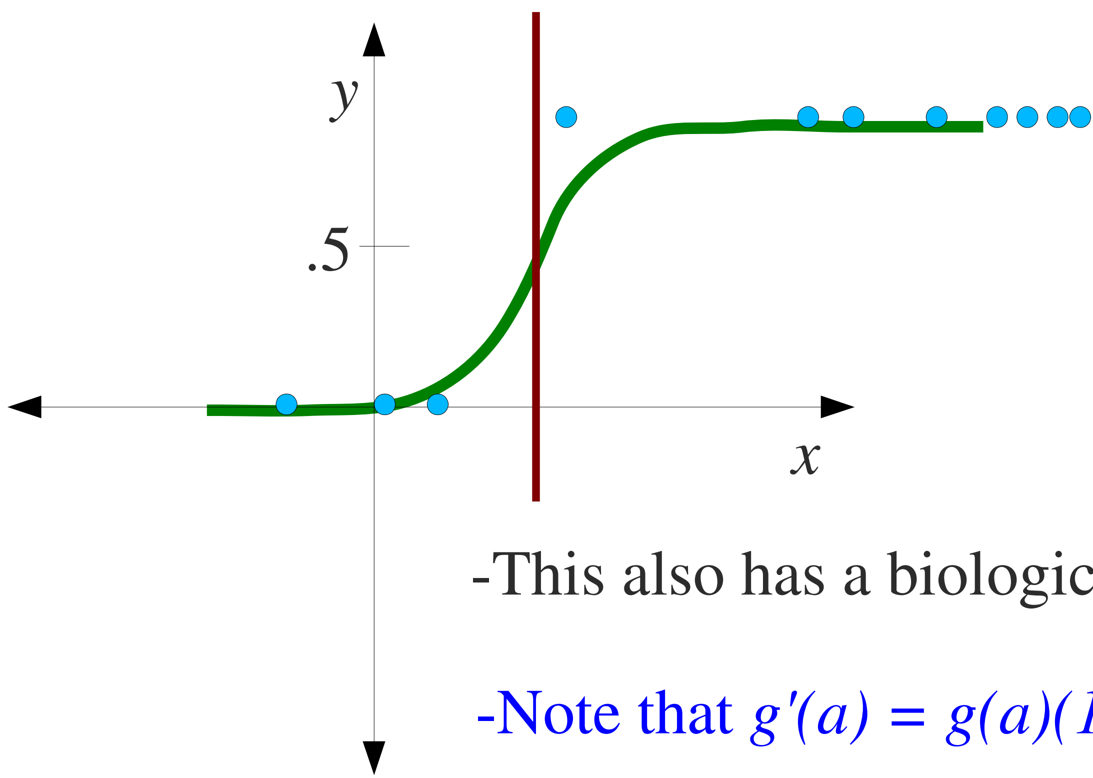
Classification Example

- The least squares fit does not necessarily lead to good classification.



Apply a Sigmoid to the Output

- Let's apply a squashing function to the output of the network: $y = g(\boldsymbol{w}^T \boldsymbol{x})$, where $g(a) = \frac{1}{(1+e^{-a})}$



-This also has a biological motivation

-Note that $g'(a) = g(a)(1-g(a))$

The New Update Rule...

- The partial derivative:

$$E = \frac{1}{2}\left(y - g\left(\boldsymbol{w}^T \boldsymbol{x}\right)\right)^2$$

$$\frac{\partial E}{\partial w_i} = \left(y - g\left(\boldsymbol{w}^T \boldsymbol{x}\right)\right)\frac{\partial}{\partial w_i}\left(\left(y - g\left(\boldsymbol{w}^T \boldsymbol{x}\right)\right)\right)$$

$$= -\left(y - g\left(\boldsymbol{w}^T \boldsymbol{x}\right)\right) g'\left(\boldsymbol{w}^T \boldsymbol{x}\right) x_i$$

- The new update rule: $w_i \leftarrow w_i + \alpha\left(y - g\left(\boldsymbol{w}^T \boldsymbol{x}\right)\right) g'\left(\boldsymbol{w}^T \boldsymbol{x}\right) x_i$

- Vector version: $\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha\left(y - g\left(\boldsymbol{w}^T \boldsymbol{x}\right)\right) g'\left(\boldsymbol{w}^T \boldsymbol{x}\right) \boldsymbol{x}$