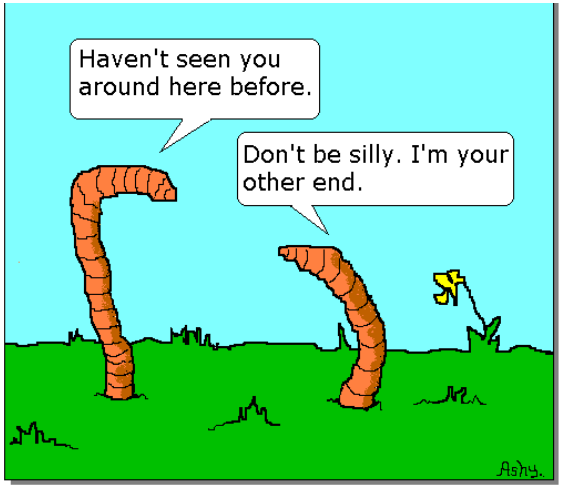
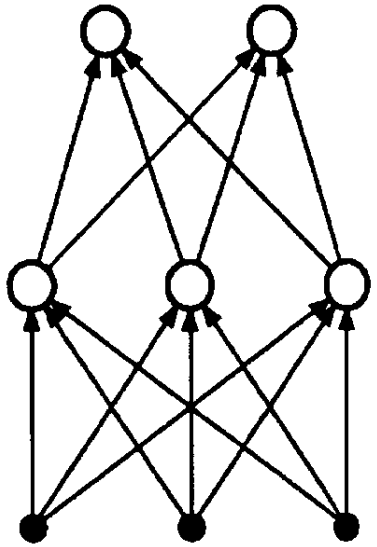
CSE 473

Lecture 27
(Chapter 18)

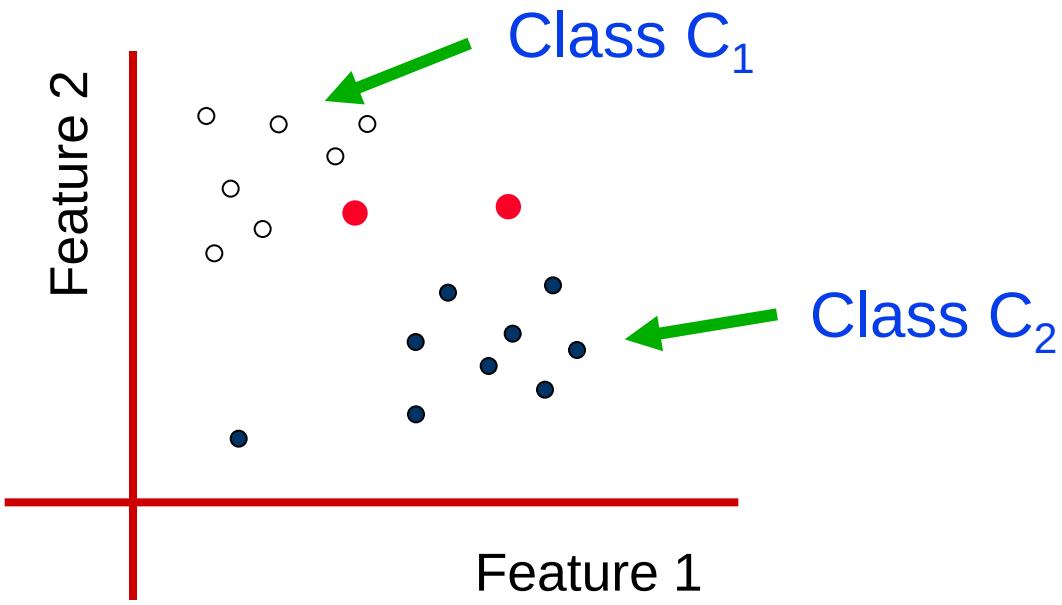
Nearest Neighbors and Neural Networks





http://dnowicki123.tripod.com/sitebuildercontent/sitebuilderpictures/worm_cartoon.gif

Recall: Binary Classification



How do we classify the new red data points?

K-Nearest Neighbors

Idea:

- "Do as your neighbors do!"
- Classify a new data-point according to a *majority vote* of your k nearest neighbors

How do you measure "near"?

x discrete (e.g., strings): Hamming distance

$d(x_1,x_2)$ = # features on which x_1 and x_2 differ

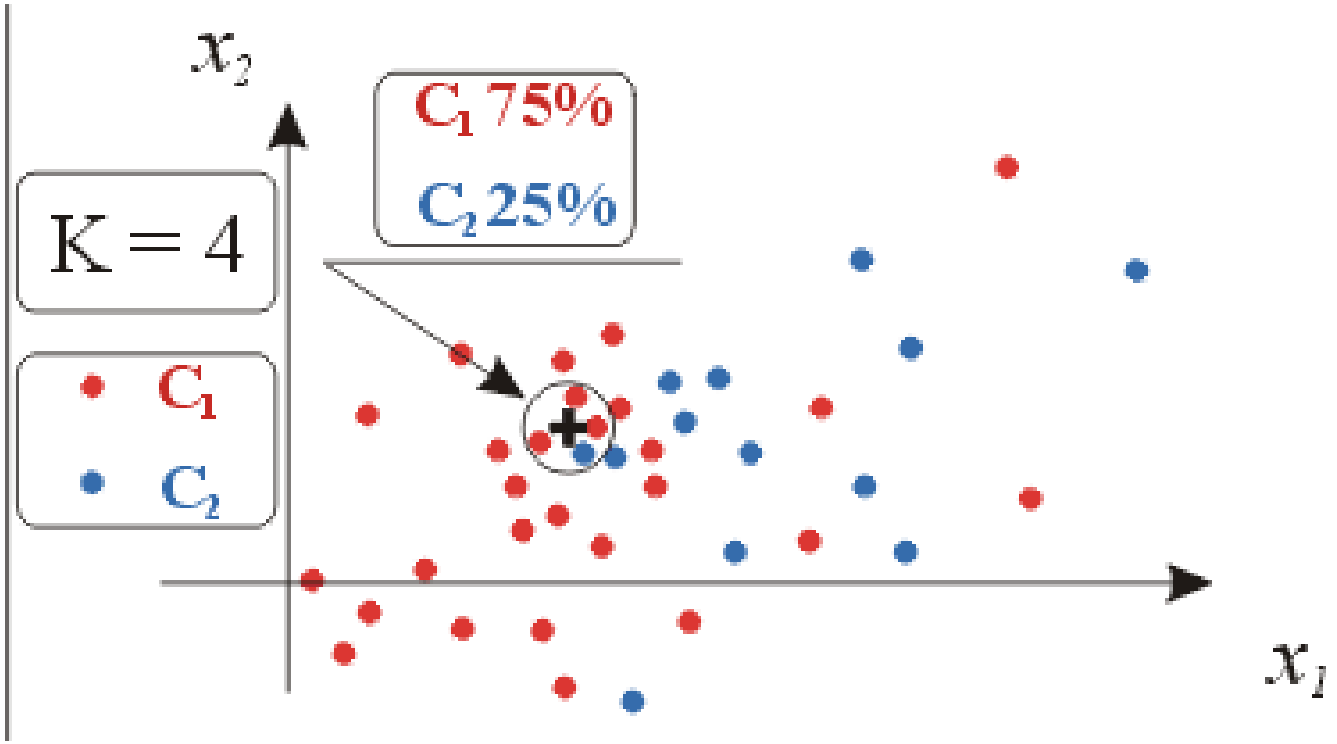
x continuous (e.g., images): Euclidean distance

$d(x_1,x_2) = || x_1-x_2 ||$ = square root of sum of squared differences between corresponding elements of data vectors

Example

Input Data: 2-D real-valued points (x_1, x_2)

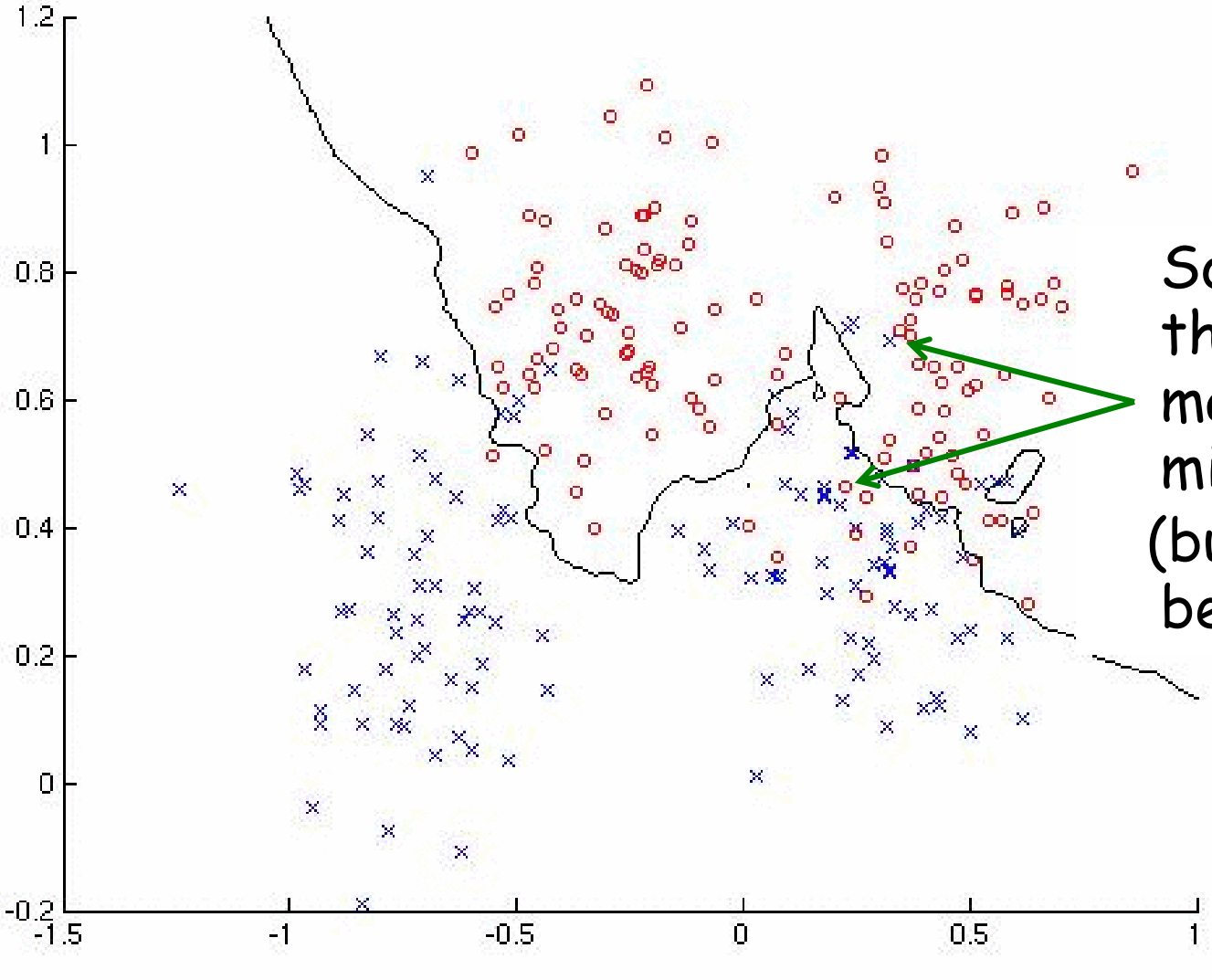
Two classes: C_1 and C_2. New Data Point +



K = 4: Look at 4 nearest neighbors.

3 are in C_1, so classify + as C_1

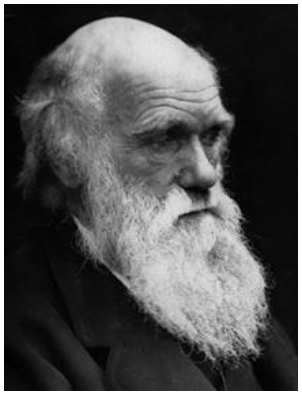
K-NN produces a Nonlinear Decision Boundary

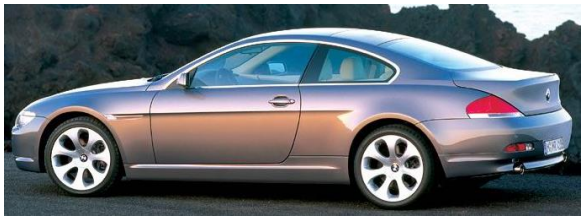


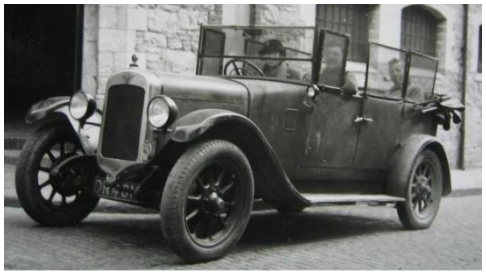
Some points near the boundary may be misclassified (but perhaps okay because of noise)

Recall: Face Detection Problem

How do we build a classifier to distinguish between faces and other objects?



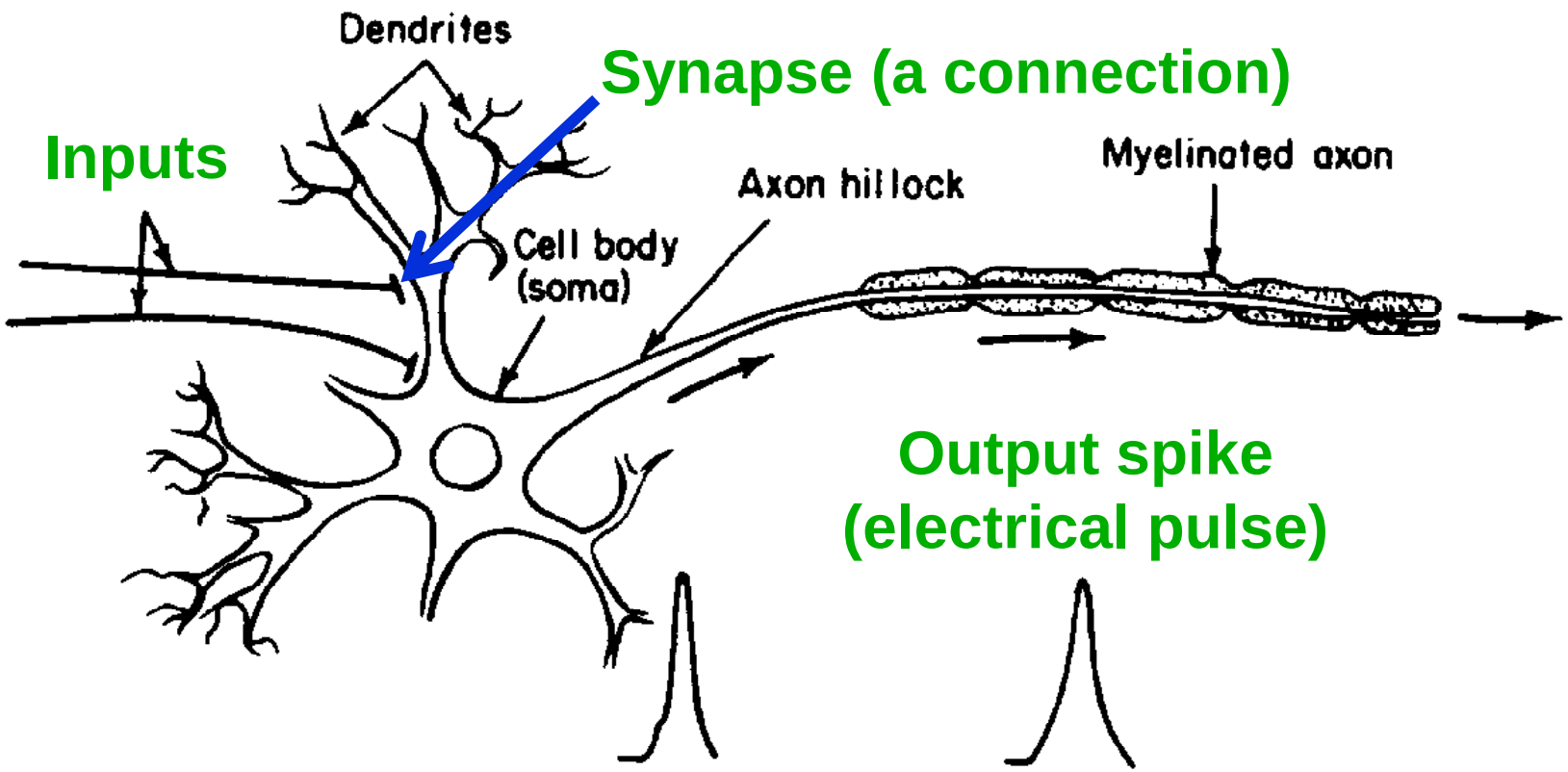




The human brain is extremely good at classifying images

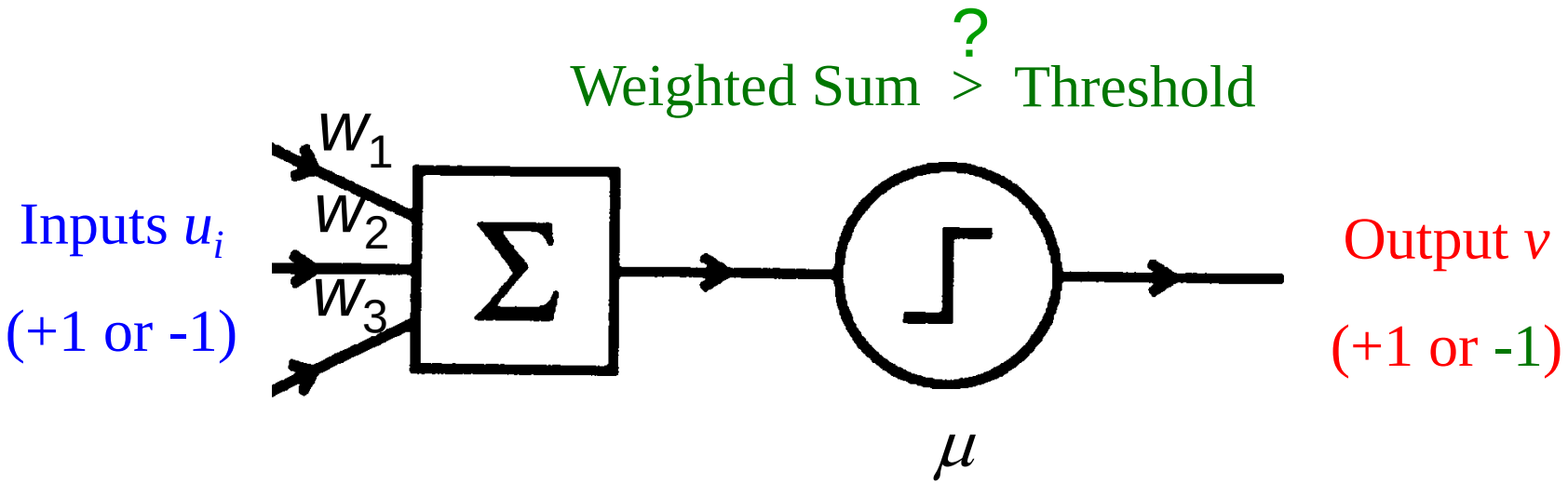
Can we develop classification methods by emulating the brain?

Neurons (Brain Cells)



Output spike roughly dependent on whether weighted sum of inputs reaches a threshold

The "Perceptron"



$$v = \Theta(\sum_i w_i u_i - \mu)$$

$\Theta(x)$ = +1 if $x > 0$ and -1 if $x \leq 0$

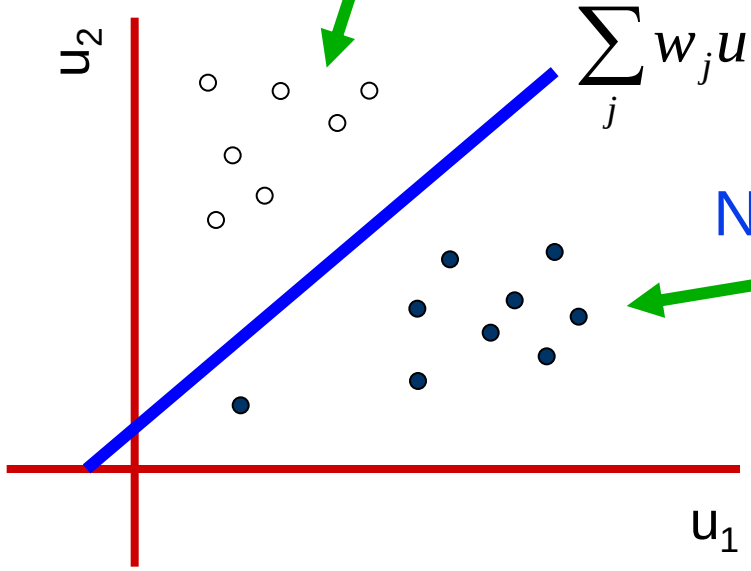
[Introduced by Rosenblatt (1958) building on McCulloch and Pitts (1943)]

Perceptrons are Classifiers!

A perceptron "neuron" defines a *hyperplane* $\sum_j w_j u_j - \mu = 0$

Spike = +1 output (class C_1)

$\sum_j w_j u_j > \mu$



$\sum_j w_j u_j - \mu = 0$

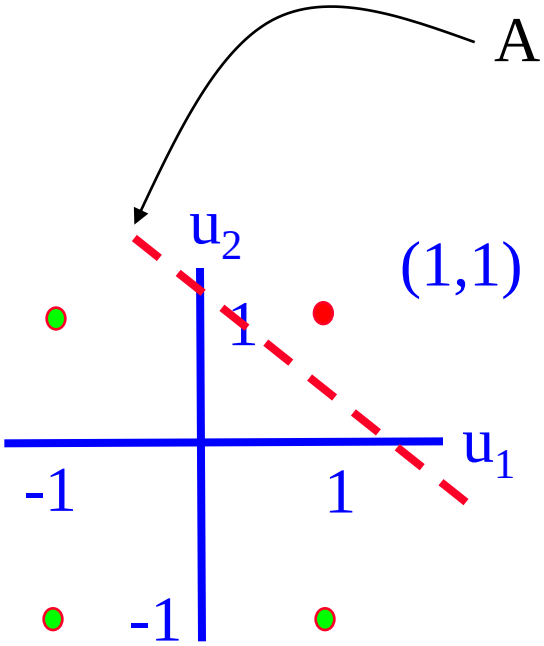
No spike = -1 output (class C_2)

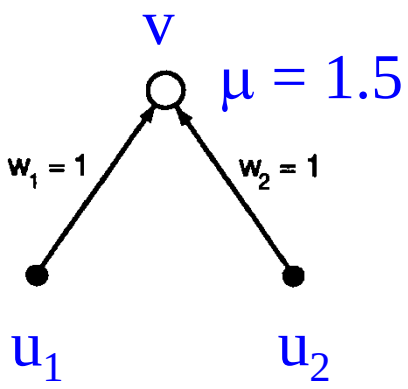
$\sum_j w_j u_j \leq \mu$

Perceptrons can compute functions

Example: AND function

u_1	u_2	AND
-1	-1	-1
1	-1	-1
-1	1	-1
1	1	1





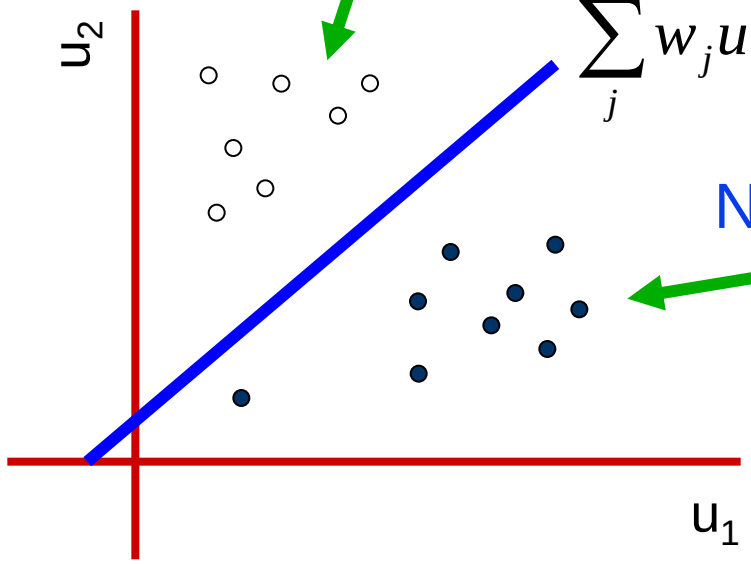
$v = 1$ iff $u_1 + u_2 - 1.5 > 0$

Similarly for OR and NOT

Perceptron Learning

Spike = +1 output (class C_1)

$$\sum_j w_j u_j > \mu$$



$$\sum_j w_j u_j - \mu = 0$$

No spike = -1 output (class C_2)

$$\sum_j w_j u_j \leq \mu$$

How do we learn the weights and threshold?

Perceptron Learning Rule

Given input u, output $v = \Theta(\sum_i w_i u_i - \mu)$, and desired output v^d

Adjust w_i and μ according to output error ($v^d - v$):

$$w_i \leftarrow w_i + \varepsilon(v^d - v)u_i$$

For positive input (u_i = +1):

Increases weight if error is positive
Decreases weight if error is negative

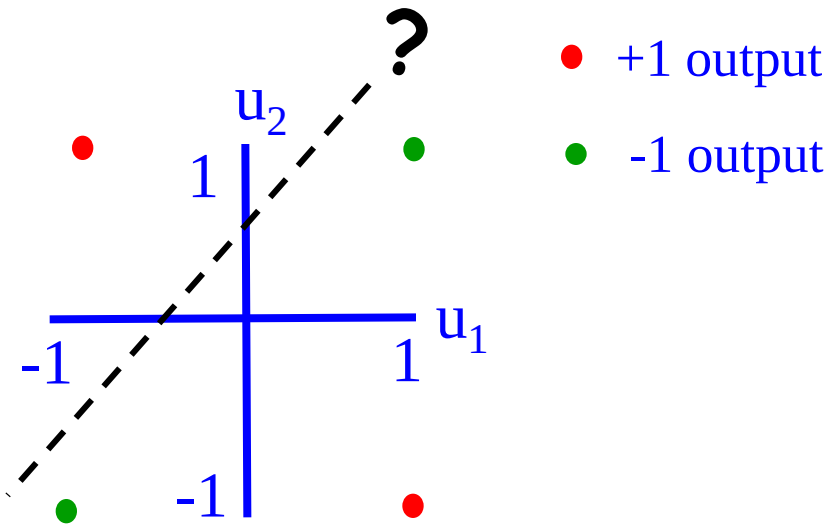
(opposite for u_i = -1)
ε is a small positive "learning rate"

$$\mu \leftarrow \mu - \varepsilon(v^d - v)$$

Decreases threshold if error is positive
Increases threshold if error is negative

Can Perceptrons learn any function?

u_1	u_2	XOR
-1	-1	-1
1	-1	+1
-1	1	+1
1	1	-1



Perceptrons can only classify linearly separable data

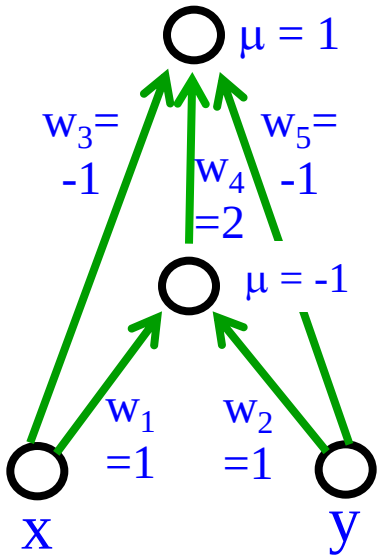
How do we handle linear inseparability?

Multilayer Perceptrons

Can classify linearly inseparable data

- Can solve XOR

An example of a two-layer perceptron that computes XOR



(Inputs and outputs are +1 or -1)

What if you want *continuous* outputs rather than +1/-1 outputs (i.e., regression)?

E.g., Teaching a network to drive

Image Source: Wikimedia Commons

Next Time

- Neural Networks for Regression
- Ensemble learning
- To Do:
  - Project 4 due this Wednesday midnight!
  - Read Chapter 18