11 Machine Learning: Connectionist

Additional sources used in preparing the slides:
Various sites that explain how a neuron works
Robert Wilensky's slides: http://www.cs.berkeley.edu/~wilensky/cs188
Russell and Norvig's AI book (2003)

Chapter Objectives

Learn about

- **the neurons in the human brain**
- **single neuron systems (perceptrons)**
- **neural networks**

Inspiration: The human brain

- **We seem to learn facts and get better at doing things without having to run a separate "learning procedure."**
- **It is desirable to integrate learning more with doing.**

Understanding the brain (1)

“ Because we do not understand the brain very well we are constantly tempted to use the latest technology as a model for trying to understand it. In my childhood we were always assured that the brain was a telephone switchboard. (“What else could it be?”) I was amused to see that Sherrington, the great British neuroscientist, thought that the brain worked like a telegraph system. Freud often compared the brain ot hydraulic and electro-magnetic systems. Leibniz compared it to a mill, and I am told that some of the ancient Greeks thought the brain functions like a catapult. At present, obviously, the metaphor is the digital computer.”

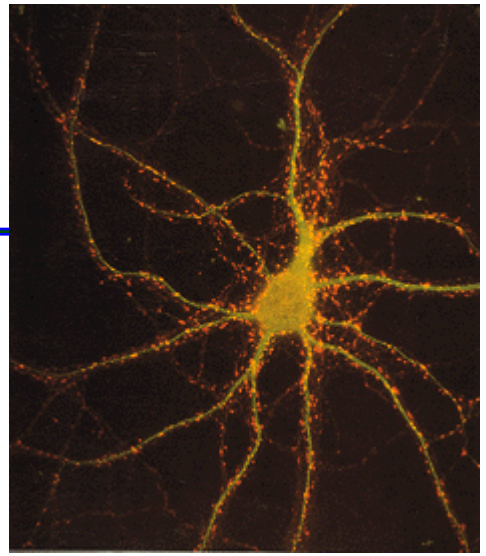
-- John R. Searle
(Prof. of Philosophy at UC, Berkeley)

Understanding the brain (2)

“ The brain is a tissue. It is a complicated, intricately woven tissue, like nothing else we know of in the universe, but it is composed of cells, as any tissue is. They are, to be sure, highly specialized cells, but they function according to the laws that govern any other cells. Their electrical and chemical signals can be detected, recorded and interpreted and their chemicals can be identified, the connections that constitute the brain’s woven feltwork can be mapped. In short, the brain can be studied, just as the kidney can.

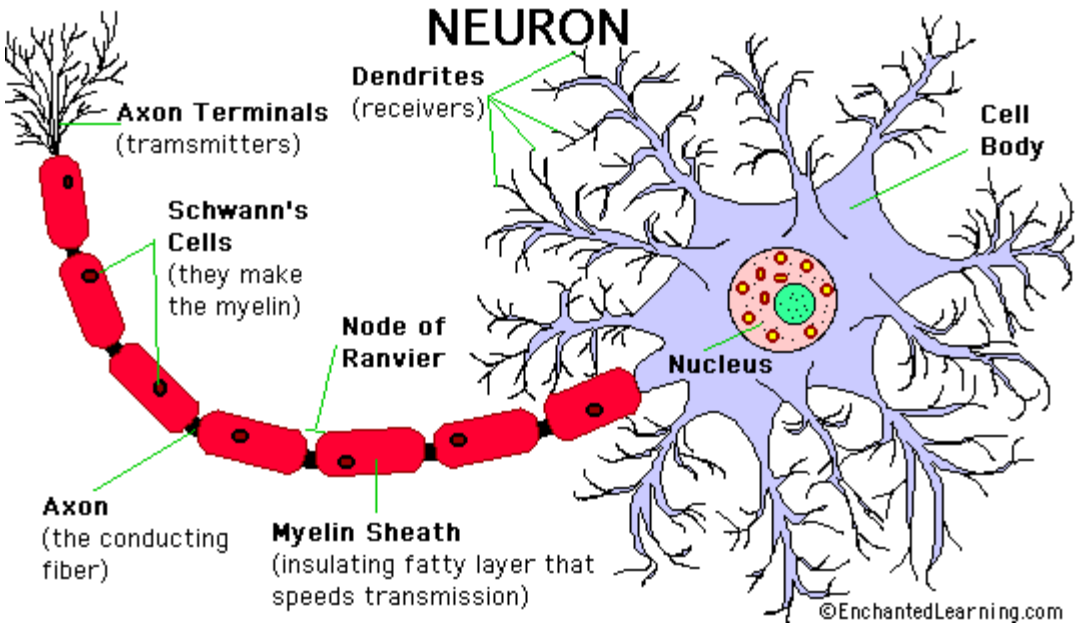
-- David H. Hubel
(1981 Nobel Prize Winner)

Biology



- **The brain doesn't seem to have a CPU.**
- **Instead, it's got lots of simple, parallel, asynchronous units, called *neurons*.**
- **Every neuron is a single cell that has a number of relatively short fibers, called *dendrites*, and one long fiber, called an *axon*.**
  - **The end of the axon branches out into more short fibers**
  - **Each fiber "connects" to the dendrites and cell bodies of other neurons**
  - **The "connection" is actually a short gap, called a *synapse***
  - **Axons are transmitters, dendrites are receivers**

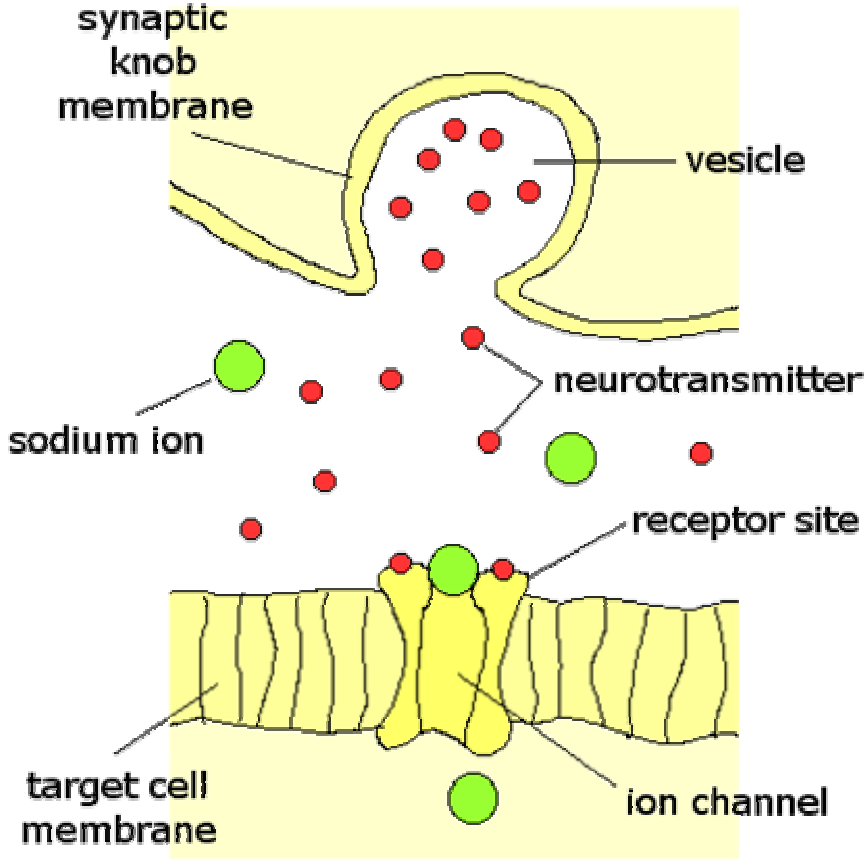
Neuron


NEURON
Dendrites
(receivers)
Axon Terminals
(tramsmitters)
Cell
Body
Schwann's
Cells
(they make
the myelin)
Node of
Ranvier
Nucleus
Axon
(the conducting
fiber)
Myelin Sheath
(insulating fatty layer that
speeds transmission)
©EnchantedLearning.com

How neurons work

- **The fibers of surrounding neurons emit chemicals (neurotransmitters) that move across the synapse and change the electrical potential of the cell body**
  - **Sometimes the action across the synapse increases the potential, and sometimes it decreases it.**
  - **If the potential reaches a certain threshold, an electrical pulse, or action potential, will travel down the axon, eventually reaching all the branches, causing them to release their neurotransmitters. And so on ...**

How neurons work (cont'd)



How neurons change

- **There are changes to neurons that are presumed to reflect or enable learning:**
  - **The synaptic connections exhibit *plasticity*. In other words, the degree to which a neuron will react to a stimulus across a particular synapse is subject to long-term change over time (*long-term potentiation*).**
  - **Neurons also will create new connections to other neurons.**
  - **Other changes in structure also seem to occur, some less well understood than others.**

Neurons as devices

- How many neurons are there in the human brain?
  - around 10^{12}
    (with, perhaps, 10^{14} or so synapses)
- Neurons are slow devices.
  - Tens of milliseconds to do something.
  - Feldman translates this into the "100 step program constraint": Most of the AI tasks we want to do take people less than a second. So any brain "program" can't be longer than 100 neural "instructions."
- No particular unit seems to be important.
  - Destroying any one brain cell has little effect on overall processing.

How do neurons do it?

- **Basically, all the billions of neurons in the brain are active at once.**
  - **So, this is truly *massive parallelism*.**

- **But, probably not the kind of parallelism that we are used to in conventional Computer Science.**
  - **Sending messages (i.e., patterns that encode information) is probably too slow to work.**
  - **So information is probably encoded some other way, e.g., by the connections themselves.**

AI / Cognitive Science Implication

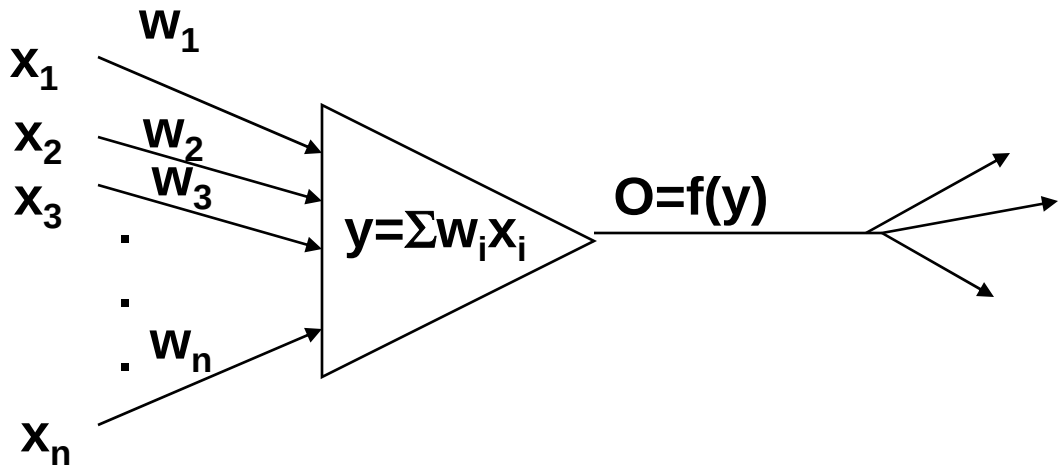
- **Explain cognition by richly connected networks transmitting simple signals.**

- **Sometimes called**
  - **connectionist computing (by Jerry Feldman)**
  - **Parallel Distributed Processing (PDP) (by Rumelhart, McClelland, and Hinton)**
  - **neural networks (NN)**
  - **artificial neural networks (ANN) (emphasizing that the relation to biology is generally rather tenuous)**

From a neuron to a perceptron

- **All connectionist models use a similar model of a neuron**
- **There is a collection of units each of which has**
  - **a number of weighted *inputs* from other units**
    - inputs represent the degree to which the other unit is firing
    - weights represent how much the units wants to listen to other units
  - **a *threshold* that the sum of the weighted inputs are compared against**
    - the threshold has to be crossed for the unit to do something (“fire”)
  - **a single *output* to another bunch of units**
    - what the unit decided to do, given all the inputs and its threshold

A unit (perceptron)



x_i are inputs w_i are weights

w_n is usually set for the threshold with x_n =-1 (bias)

y is the weighted sum of inputs including the threshold (activation level)

o is the output. The output is computed using a function that determines how far the perceptron's activation level is below or above 0

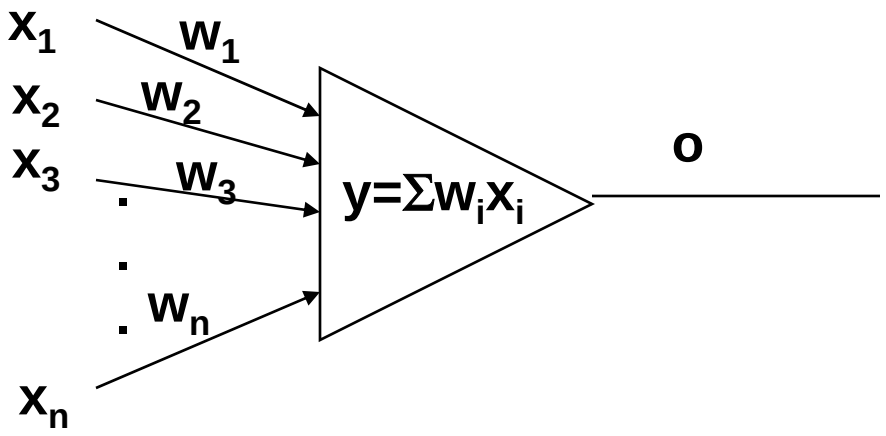
Notes

- **The perceptrons are continuously active**
  - **Actually, real neurons fire all the time; what changes is the *rate of firing*, from a few to a few hundred impulses a second**
- **The weights of the perceptrons are not fixed**
  - **Indeed, learning in a NN system is basically a matter of changing weights**

Interesting questions for NNs

- **How do we wire up a network of perceptrons?**
  - i.e., what "architecture" do we use?
- **How does the network represent knowledge?**
  - i.e., what do the nodes mean?
- **How do we set the weights?**
  - i.e., how does learning take place?

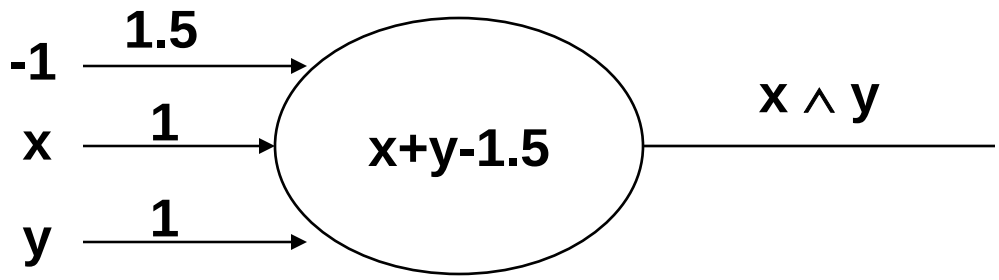
The simplest architecture: a single perceptron



A perceptron computes o = f (X . W), where
$y=X.W = w_1 * x_1 + w_2 * x_2 + \ldots + w_n * -1$, and
f(x) = 1 if y > 0 and
0 otherwise

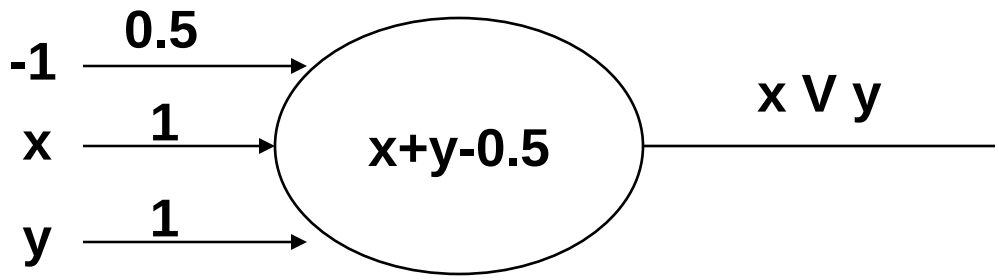
A perceptron can act as a logic gate interpreting 1 as true and -1 (or 0) as false

Logical function and



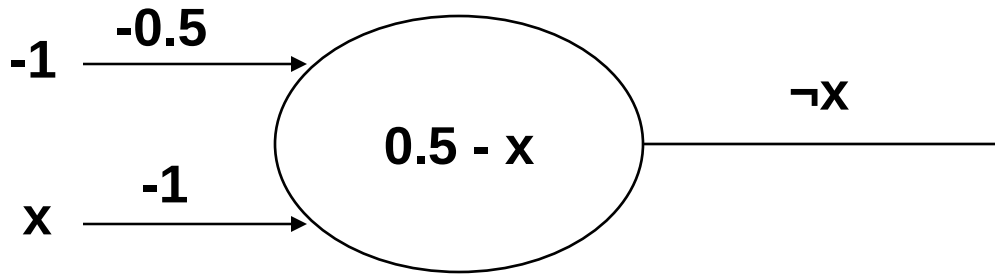
x	y	x+y-1.5	output
1	1	0.5	1
1	0	-0.5	0
0	1	-0.5	0
0	0	-1.5	0

Logical function or



x	y	x+y-0.5	output
1	1	1.5	1
1	0	0.5	1
0	1	0.5	1
0	0	-0.5	0

Logical function not



x	0.5 - x	output
1	-0.5	0
0	0.5	1

Training perceptrons

- **We can train perceptrons to compute the function of our choice**
- **The procedure**
  - **Start with a perceptron with any values for the weights (usually 0)**
  - **Feed the input, let the perceptron compute the answer**
  - **If the answer is right, do nothing**
  - **If the answer is wrong, then modify the weights by adding or subtracting the input vector (perhaps scaled down)**
  - **Iterate over all the input vectors, repeating as necessary, until the perceptron learns what we want**

Training perceptrons: the intuition

- If the unit should have gone on, but didn't, increase the influence of the inputs that are on:
  - adding the input (or fraction thereof) to the weights will do so;
- If it should have been off, but was on, decrease influence of the units that are on:
  - subtracting the input from the weights does this

Example: teaching the logical or function

Want to learn this:

Bias	x	y	output
-1	0	0	0
-1	0	1	1
-1	1	0	1
-1	1	1	1

Initially the weights are all 0, i.e., the weight vector is (0 0 0).

The next step is to cycle through the inputs and change the weights as necessary. (see the handouts)

The general procedure

- **Start with a perceptron with any values for the weights (usually 0)**
- **Feed the input, let the perceptron compute the answer**
- **If the answer is right, do nothing**
- **If the answer is wrong, then modify the weights by adding or subtracting the input vector**

$$\Delta w_i = c\,(d - f)\,x_i$$

- **Iterate over all the input vectors, repeating as necessary, until the perceptron learns what we want (i.e., the weight vector converges)**

More on $\Delta w_i = c\,(d - f)\,x_i$

c is the learning constant

d is the desired output

f is the actual output

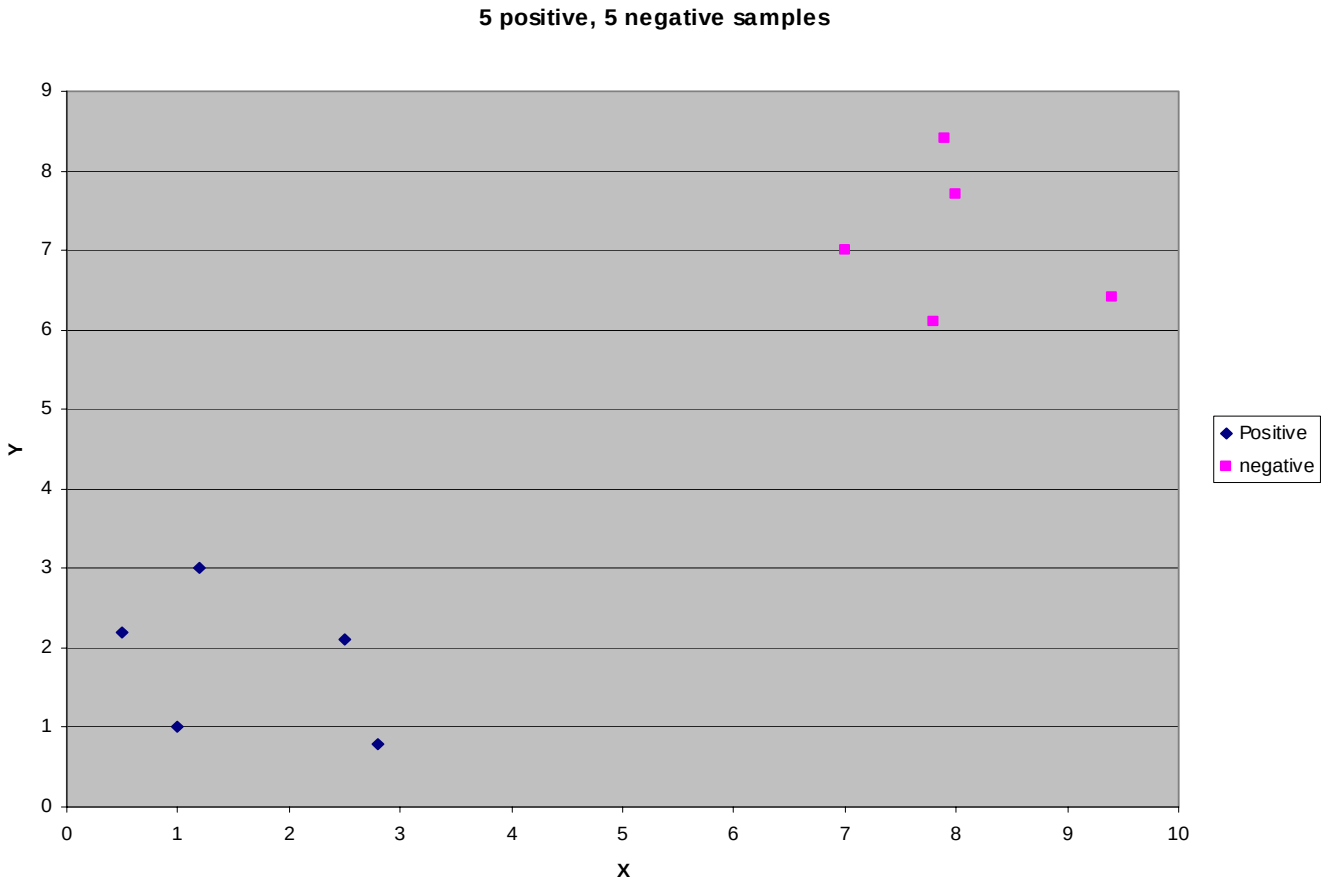
(d - f) is either 0 (correct), or (1 - 0)= 1,
or (0 - 1) = -1.

The net effect is:
When the actual output is 0 and should be 1, increment the weights on the i-th line by cx_i.
When the actual output is 1 and should be 0, decrement the weights on the i-th line by cx_i.

A data set for perceptron classification

X	Y	Output
1.0	1.0	1
9.4	6.4	0
2.5	2.1	1
8.0	7.7	0
0.5	2.2	1
7.9	8.4	0
7.0	7.0	0
2.8	0.8	1
1.2	3.0	1
7.8	6.1	0

A two-dimensional plot of the data points



The good news

- **The weight vector converges to**

(-6.0 -1.3 -0.25)

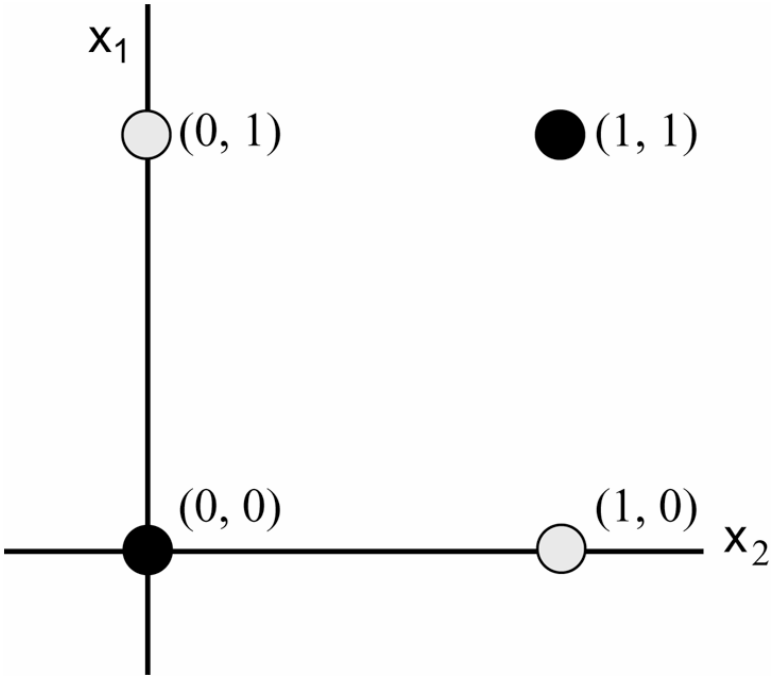
after 5 iterations.

- **The equation of the line found is**

$$-1.3 * x_1 + -0.25 * x_2 + -6.0 = 0$$

- **The Y intercept is 24.0, the X intercept is 4.6.**

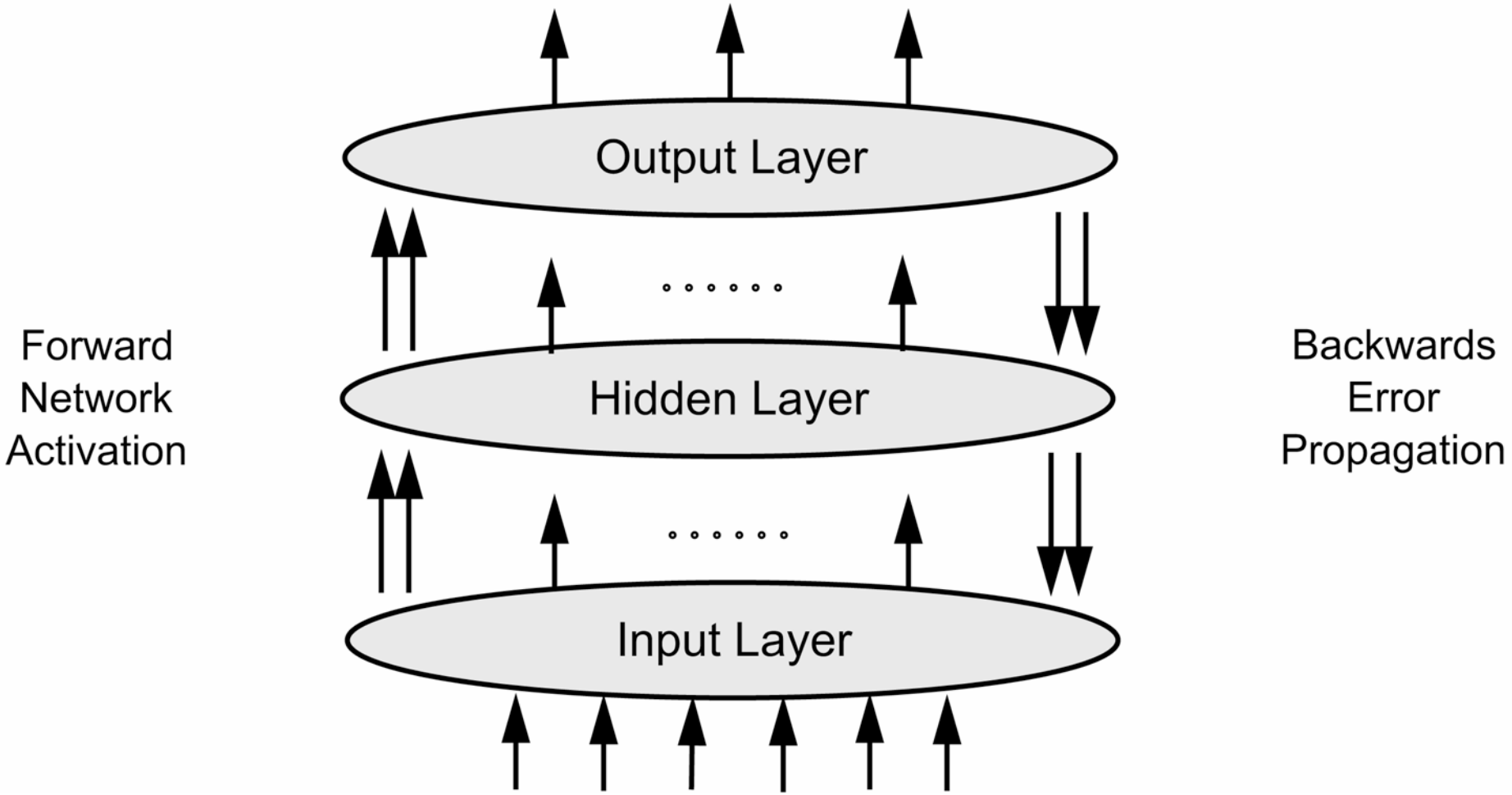
The bad news: the exclusive-or problem



No straight line in two-dimensions can separate the (0, 1) and (1, 0) data points from (0, 0) and (1, 1).

A single perceptron can only learn *linearly separable* data sets.

The solution: multi-layered NNs



The adjustment for w_{ki} depends on the total contribution of node i to the error at the output