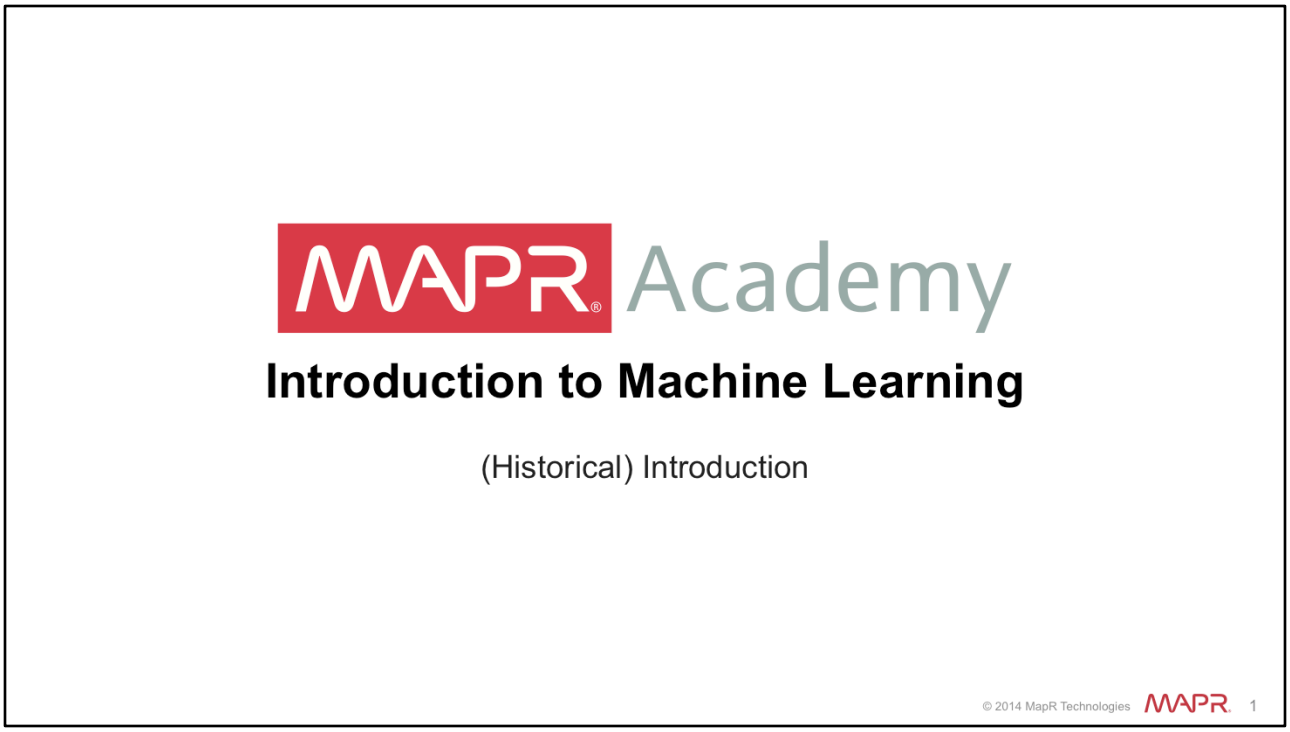
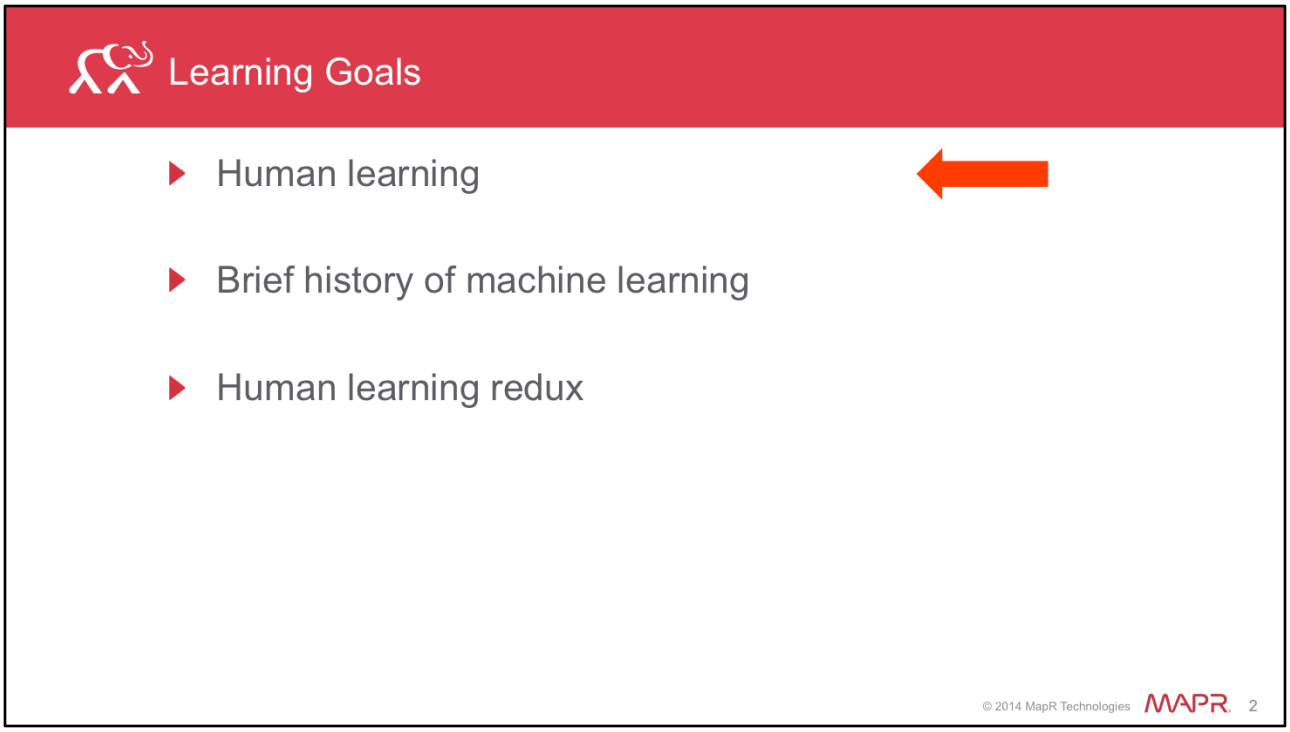
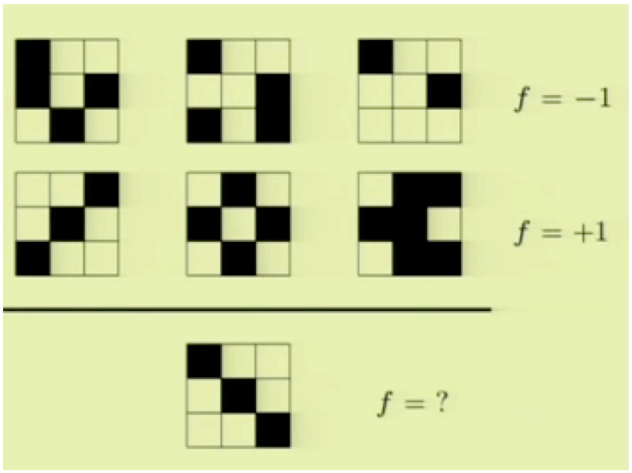


In this lesson, we will introduce the field of machine learning by examining the history of machine learning.



In this section, we motivate the concept of machine learning with human learning.

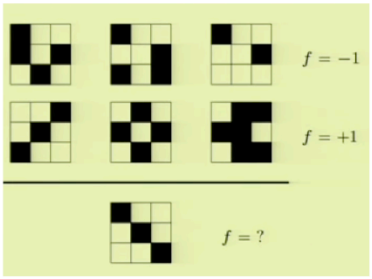
Human Learning: Example 1



The graphic above depicts 6 different images from which you may derive a pattern. The pattern that gives rise to each image is associated with either f=+1 or f=-1. The image on the bottom is a new image. The question is: which value of f do we associate with this new image based on what we "learned" from the previous 6 "labeled" examples?

What is f?

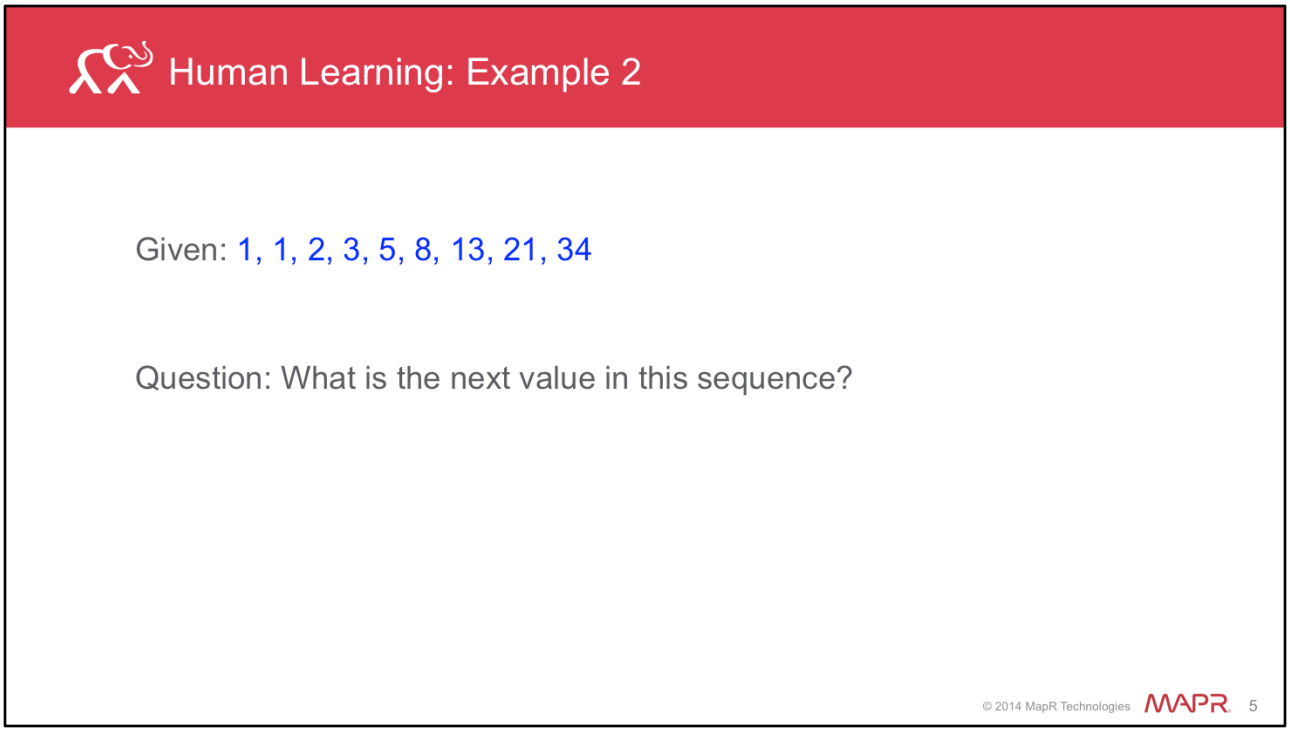
- **f=-1?**
  - Yes, if e.g. f is function of value in cell [0,0]
- **f=+1?**
  - Yes, if e.g. f is function of horizontal/vertical/diagonal symmetry
- **Which one is it?**
- **Are there more possible explanations for f?**



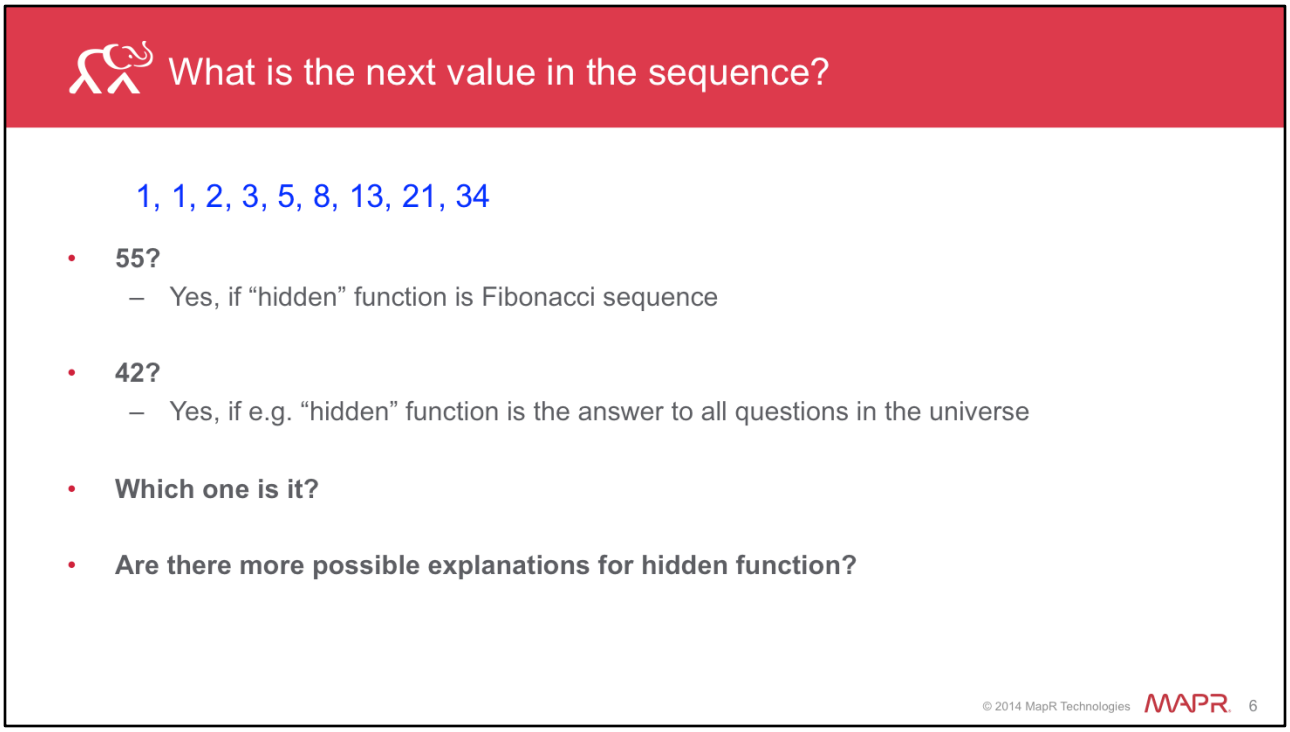
You may associate the image on the bottom with f=-1 if, for example, you only examine the top-left cell of the image. If that cell is black, then f=-1, and if that cell is white, then f=+1.

Alternatively, you may associate the image on the bottom with f=+1 if, for example, you consider symmetry of the image. If there is horizontal, diagonal, or vertical symmetry in the image, then f=+1. If there is no symmetry, then f=-1.

Which answer is correct? As it turns out, there are many ways you could interpret the relationship between the image and value of f, and none of them are necessarily "correct".



In this second human learning example, we examine a given sequence of integers. We "learn" the generating function so that we can determine the next value in the sequence.



If the generating function you learned was the Fibonacci sequence, then the next value would be 55. If the generating function you learned was that the answer to all questions is 42, then the next value would be 42.

Like our first example, there are other possible explanations for this sequence of values, and none of them are necessarily correct.

Human Learning: Example #3

Question: What is this a picture of?

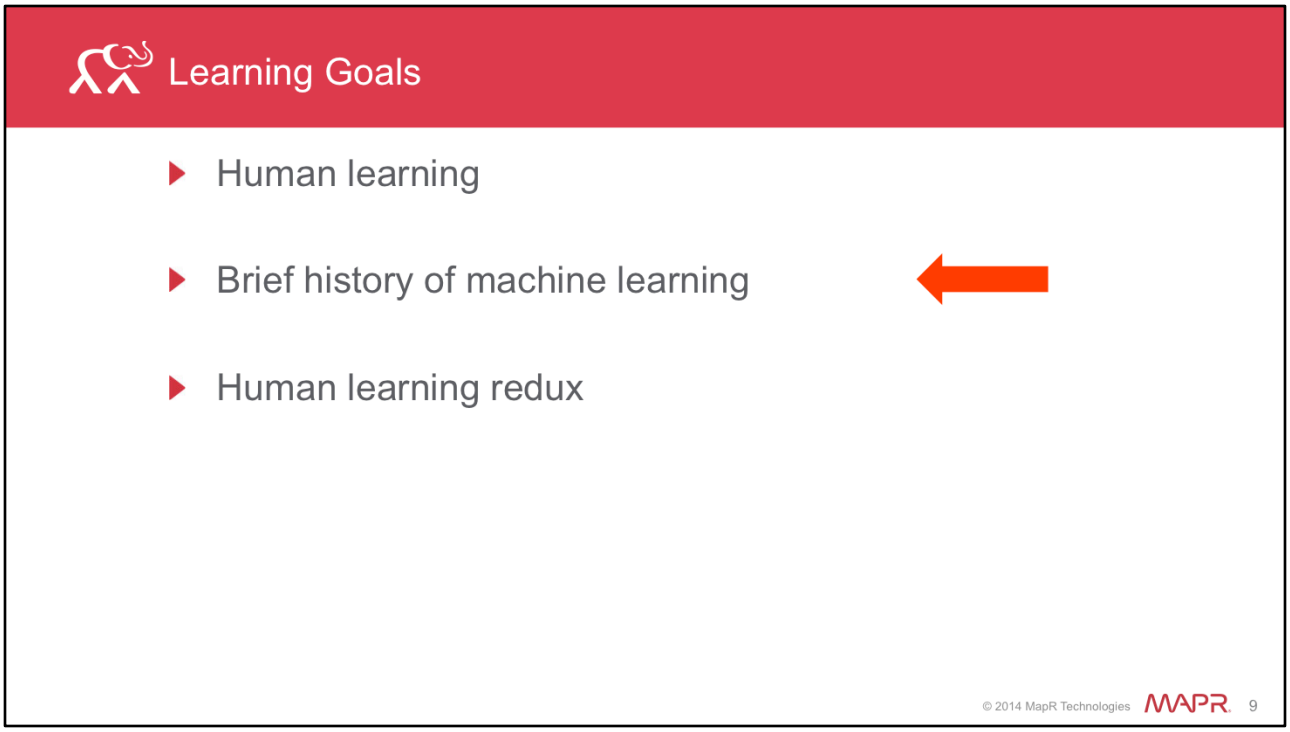
In this last human learning example, we examine this drawing and attempt to discern what it is a drawing of.

What is this a picture of?

- A pretty woman?
- A witch?
- Which one is it?
- Are there more possible interpretations?

 8

You may have guessed that it is a picture of a pretty woman, or you may have guessed that it is a picture of a witch. Like all our examples, there are more "explanations" for what this may be a picture of, and none of them are necessarily "correct".



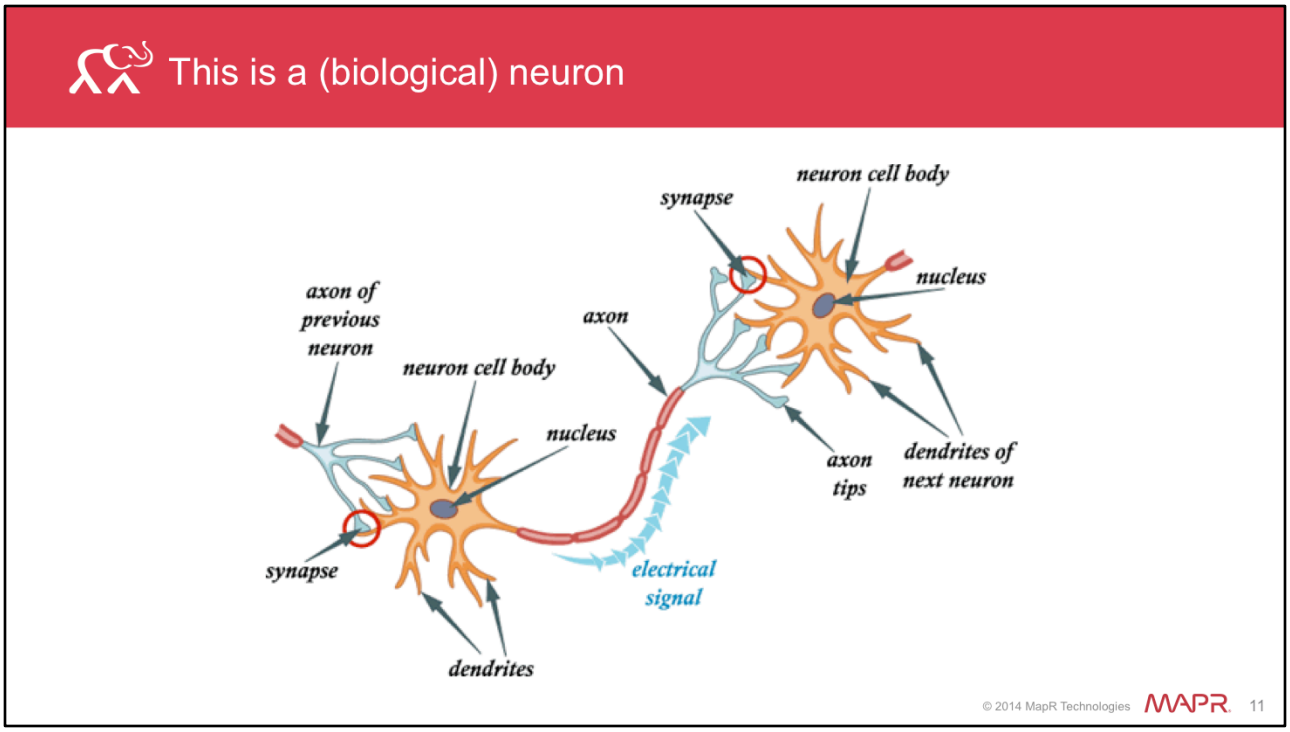
In this section, we examine a brief history of machine learning.

What is Machine Learning?

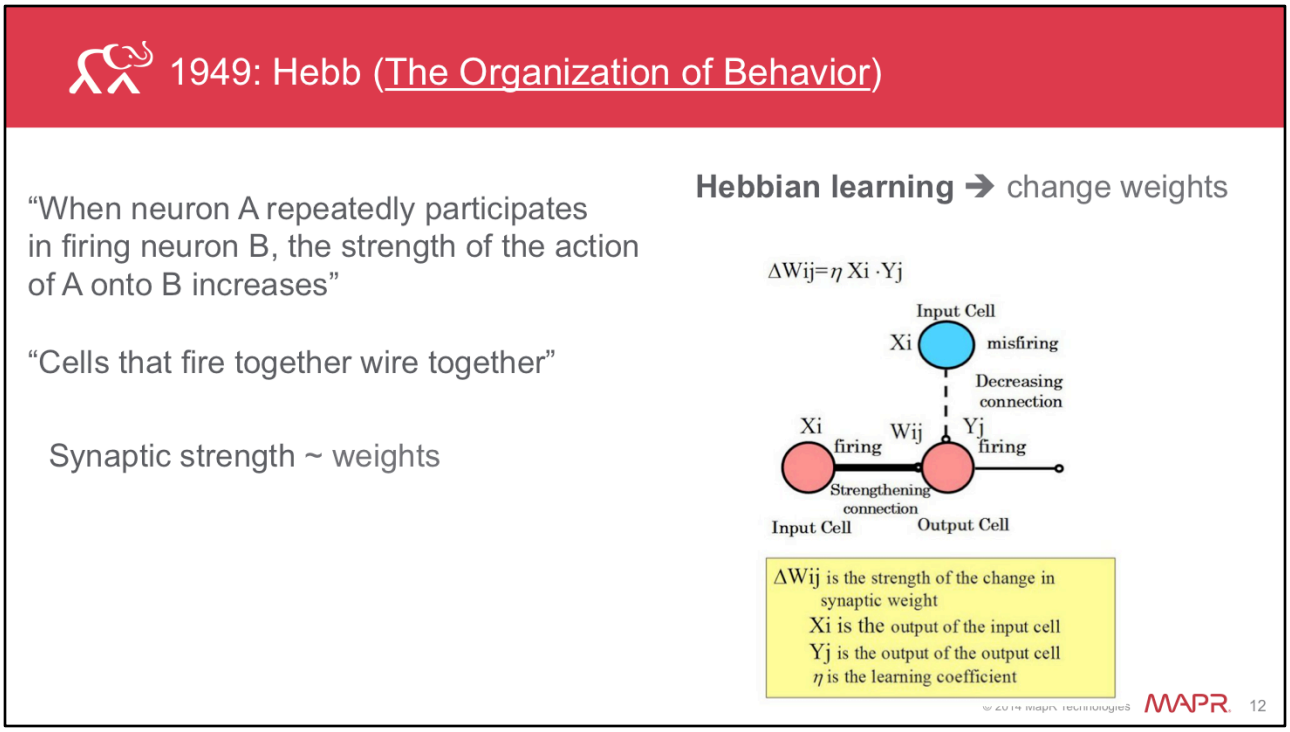
Definition from wikipedia:

Machine learning is a subfield of computer science that deals with the construction and study of systems that can learn from data, rather than follow only explicitly programmed instructions.

There are many definitions for machine learning. We consider here the definition provided by wikipedia which claims that ML is a subfield of computer science. Note there are also strong influences from statistics, mathematics, and even physics. However, ML algorithms run on computers, so we consider it part of CS. The key part of the definition is that machines may learn by examining data.

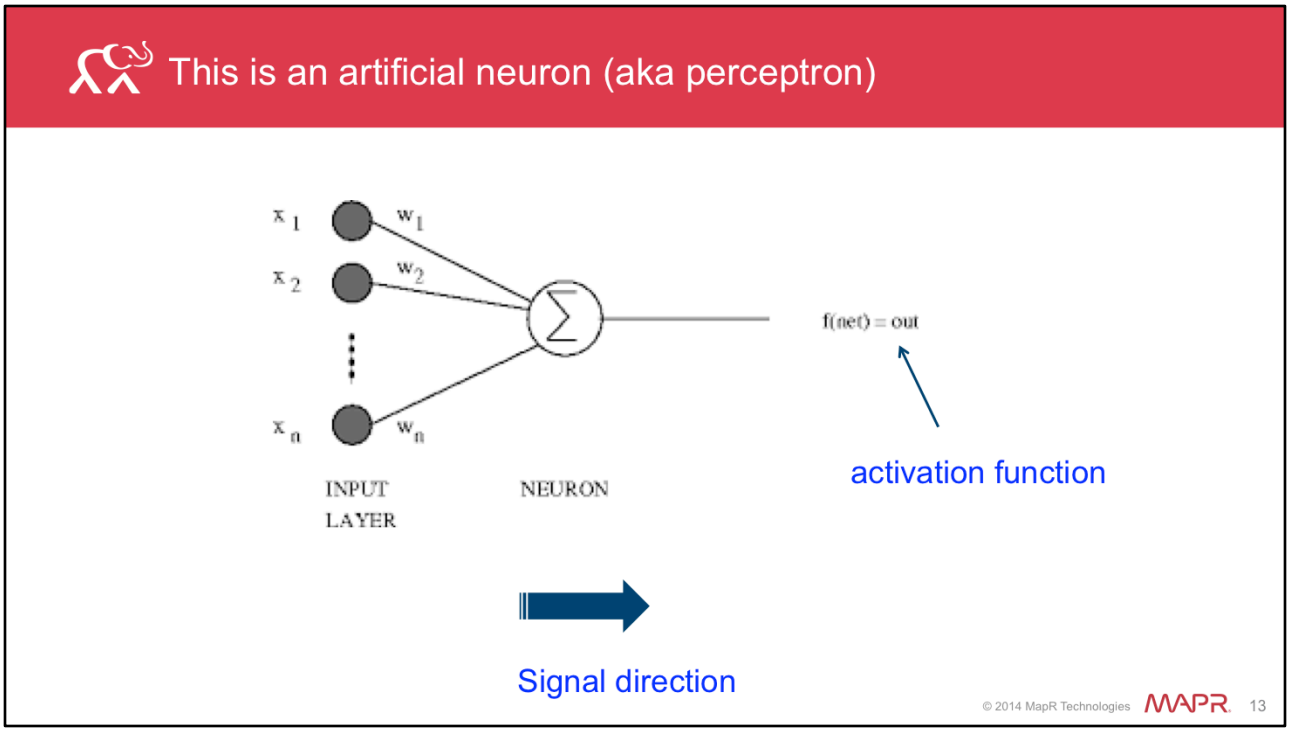


Let's examine the biological neuron – the brain cell. Shown in this graphic is a pair of neuron cell bodies that are connected. Each cell body is comprised of a nucleus, dendrites, and an axon. There are many other parts of the cell, but they aren't important for the sake of our discussion. Cells communicate with each other by sending electrical signals across the axon, out to the synapses, and across a small "cleft" separating the synapses of one cell from the dendrites of the next cell. Cells send an electrical signal when they reach a certain threshold.

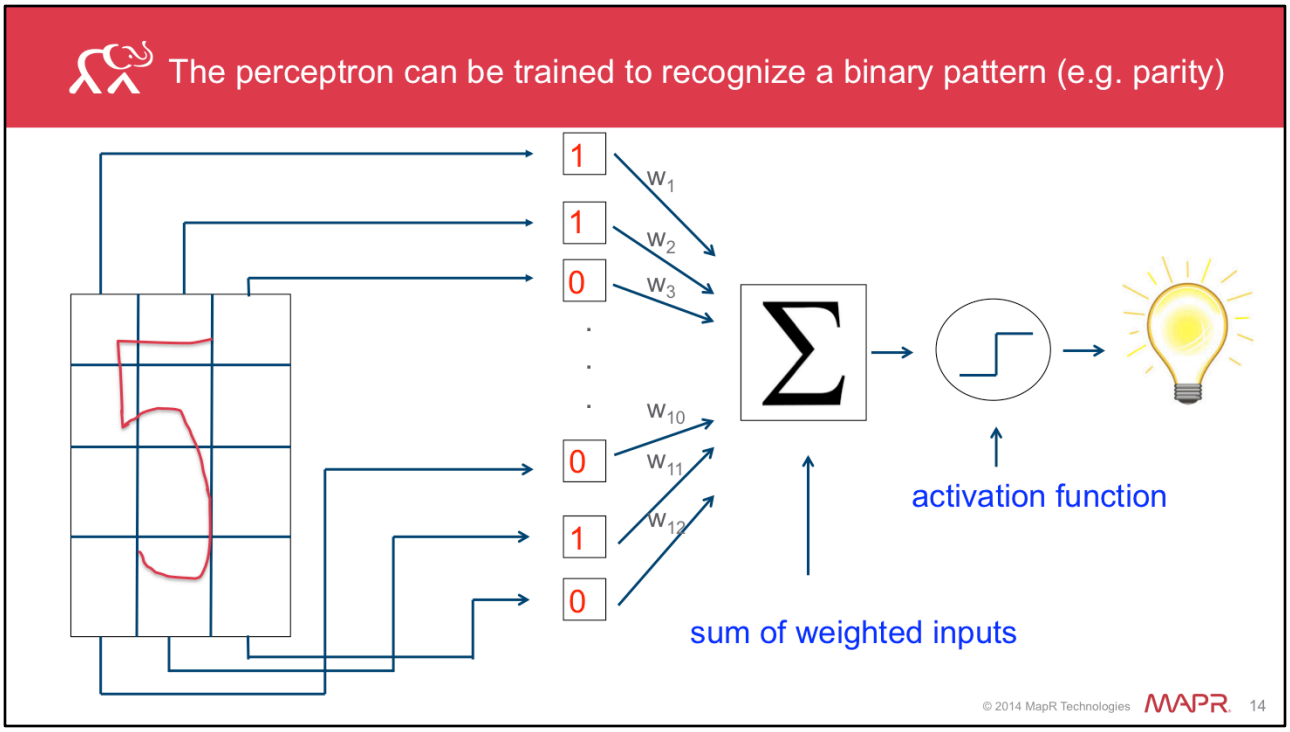


In 1949, Donald Hebb published a book called "The Organization of Behavior" in which he proposed an explanation for how neurons adapt themselves when learning. It is based on the strength (or weight) of connectivity between each neuron and all other neurons to which it is connected.

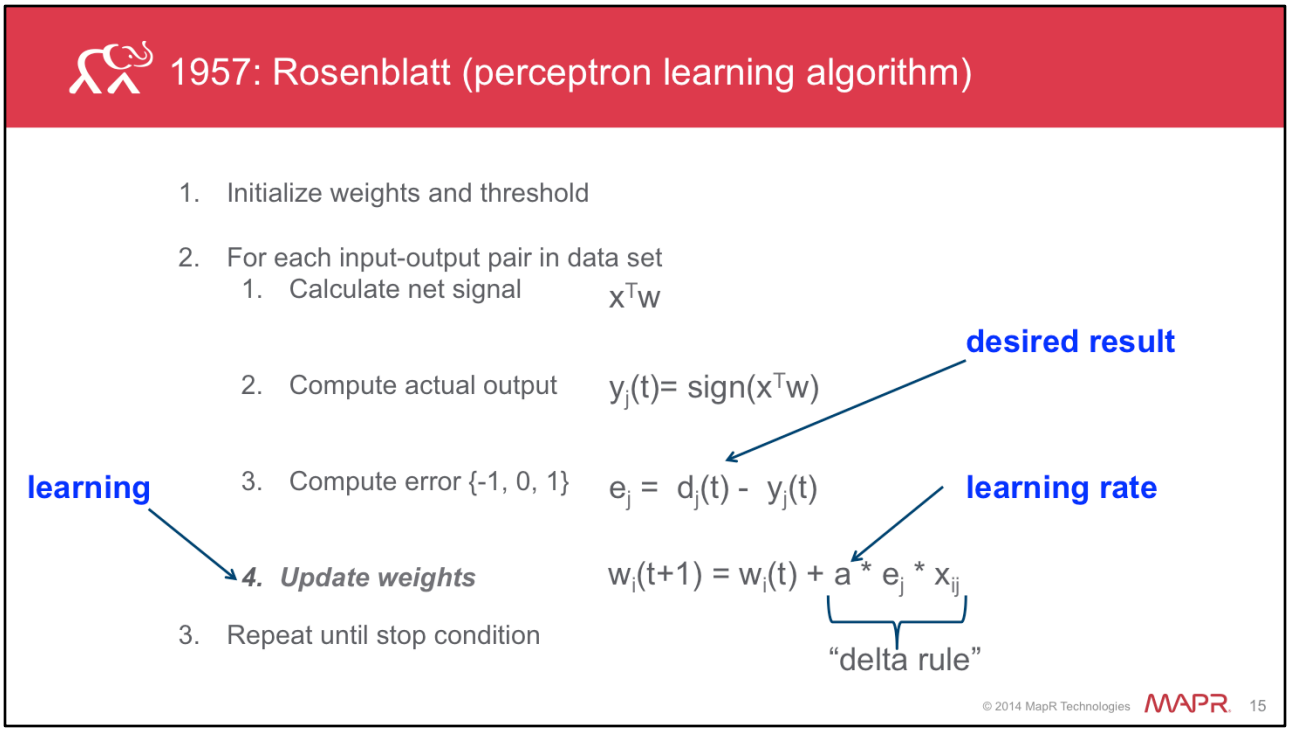
Hebb suggests that when 2 connected neurons fire simultaneously, the strengths of their connections are increased. Conversely, when 2 connected neurons fire at different times from each other, the strengths of their connections are diminished.



The graphic above depicts an artificial neuron which is motivated by the biological neuron. Like it's biological counterpart, a perceptron has a set of inputs and a set of outputs. When the inputs are collected from the input layer, they are summed and then sent to an activation function which either sends a 0 or a 1, depending on the sum of the inputs and the threshold of the function.



A simple perceptron can be learned to recognize a binary pattern (0 or 1) based on a set of inputs. In the graphic above, our perceptron reads from a grid in which a numerical digit is written. If the hand-writing appears in a cell, that cell input value is 1. If there is no hand-writing in a given cell, that cell input value is 0. The input is multiplied by a weight, and all the input-weight multiples are summed and sent to an activation function. In this case, the activation function is a step function that fires 1 if a pre-defined threshold is reached, and fires 0 if not reached. In our example, the perceptron can be trained to determine, based on the hand-written digit, whether the digit is an even or odd number.

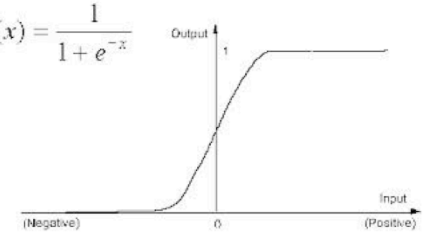


In 1957, Frank Rosenblatt devised a learning algorithm for the perceptron at the Cornell Aeronautical Laboratory. The algorithm was used for image recognition. While his prediction that an electronic computer would be to walk, talk, see, write, reproduce itself, and be conscious of its existence was a little premature, this algorithm was widely adopted.

The algorithm cycles through all the input-output pairs (in a labeled data set). In each pass, the algorithm uses the input to calculate the actual output. It then determines the error – the difference between the actual output and the desired result. That delta is used to calculate the adjustment to the weights from the inputs to the perceptron. This algorithm is repeated untila stop condition (the algorithm converges or reaches a maximum number of iterations).

1960: Widrow-Hoff (sigmoidal activation function)

$$f(x) = \frac{1}{1 + e^{-x}}$$

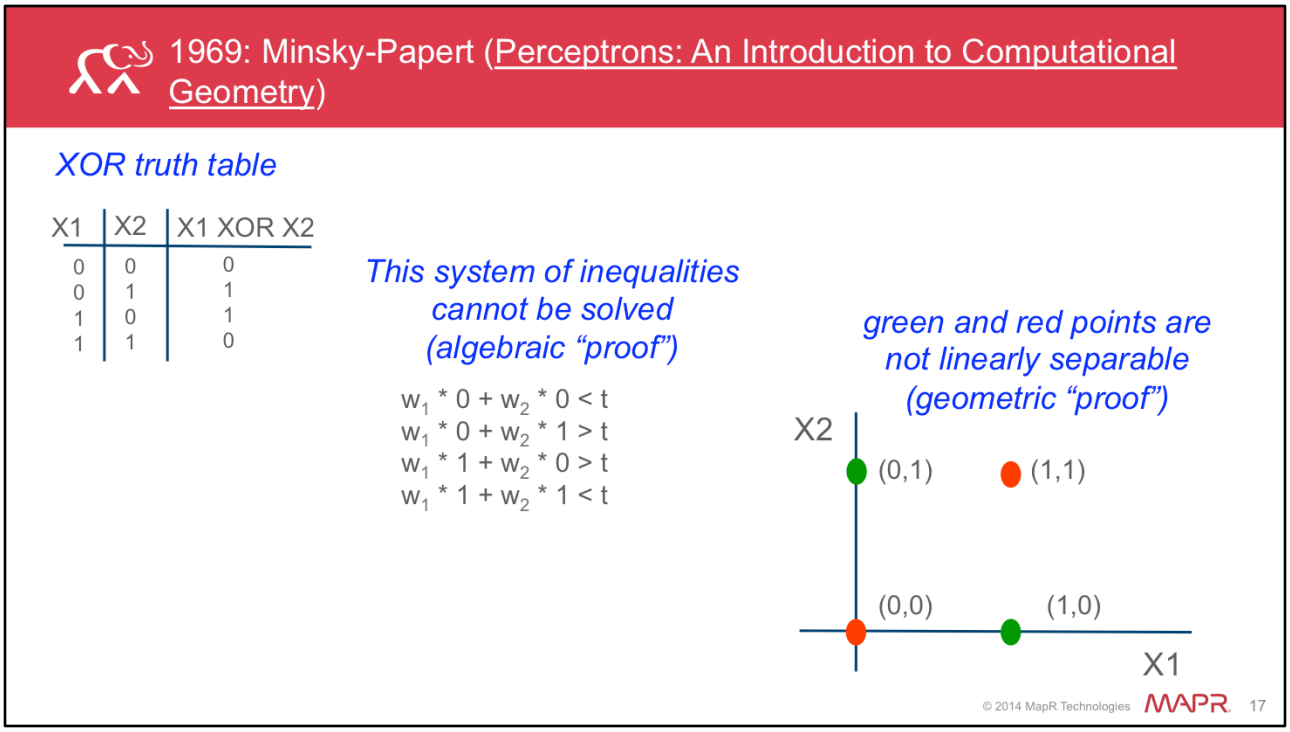


Sigmoid is a good activation function:

- Good approximation for step function
- Permits continuous output
- Solves noise saturation for large signals
- Solves noise attenuation for small signals
- Has simple 1st order derivative

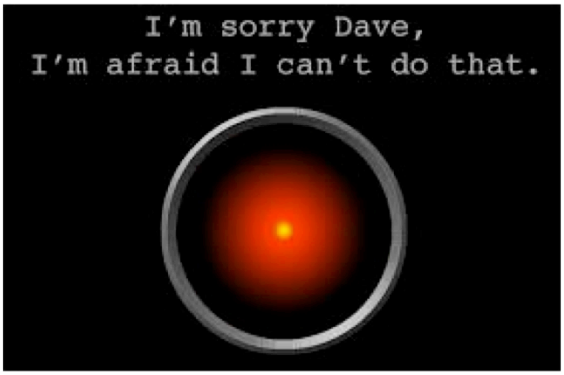
Replace discrete discrete output {0, 1} with a continuous output in [0, 1]

In 1960, Bernard Widrow and his graduate student Ted Hoff invented the least mean squares filter (LMS) adaptive algorithm. As part of that, they replaced the step activation function with a sigmoidal activation function. This yields several desirable properties, as described in the slide above.



In 1969, Marvin Minsky and Seymour Papert showed it was impossible for the perceptron to learn a non-linearly separable pattern, such as XOR. This problem is depicted in the graphic above.

Neural networking went dark for a while



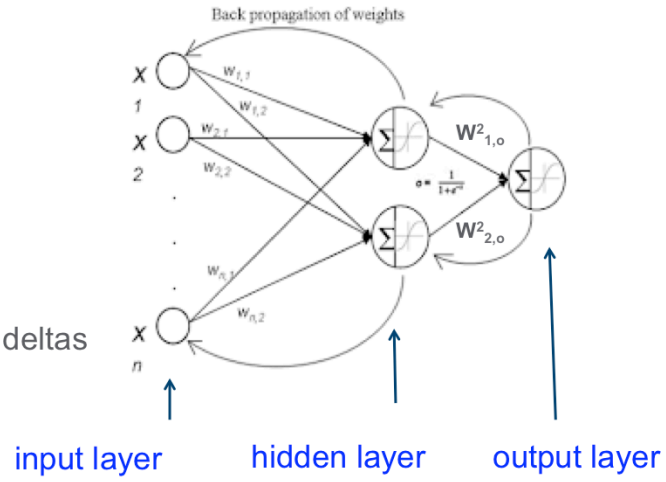
The Minsky publication caused a significant decline in interest and funding for more than 10 years.

1974: Werbos (multi-layer perceptron with back-propagation)

Repeat until convergence

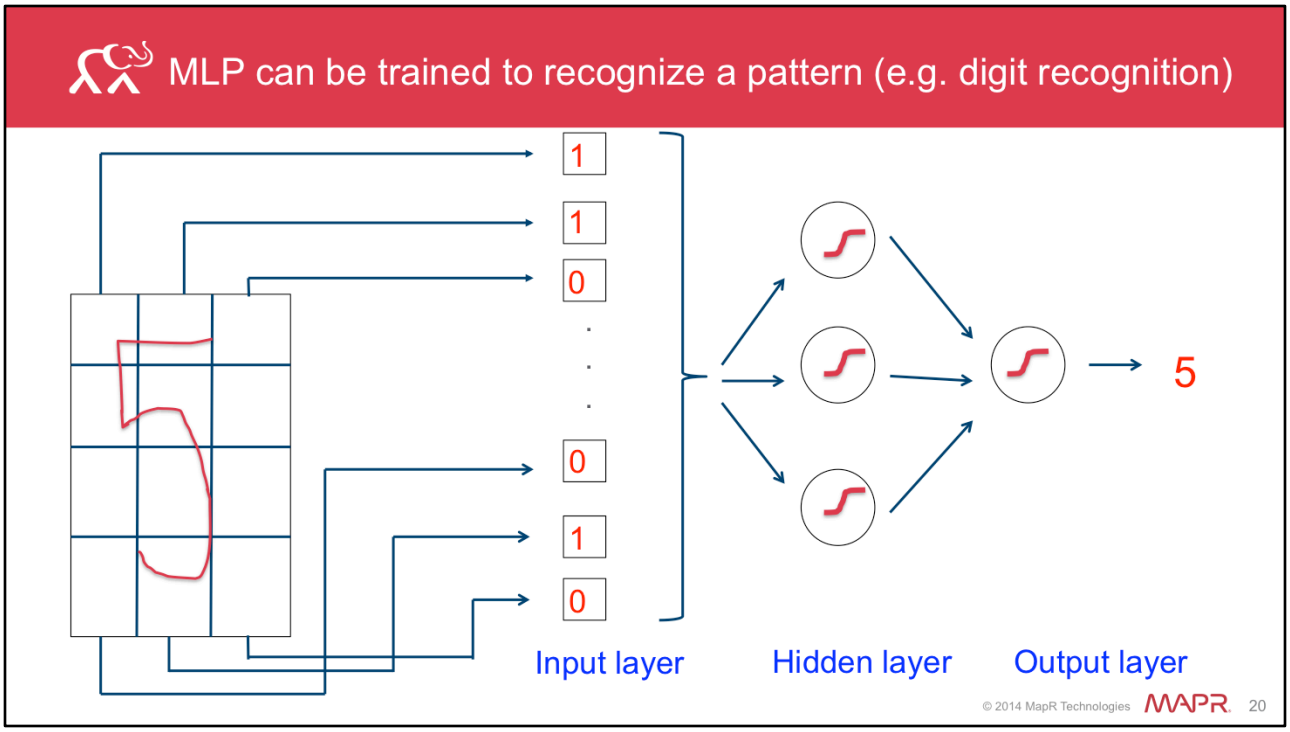
1. Forward pass:
   - Calculate output per neuron
   - Calculate deltas per neuron

2. Reverse pass
   - Calculate deltas in weights
   - Adjust weights by fraction of deltas



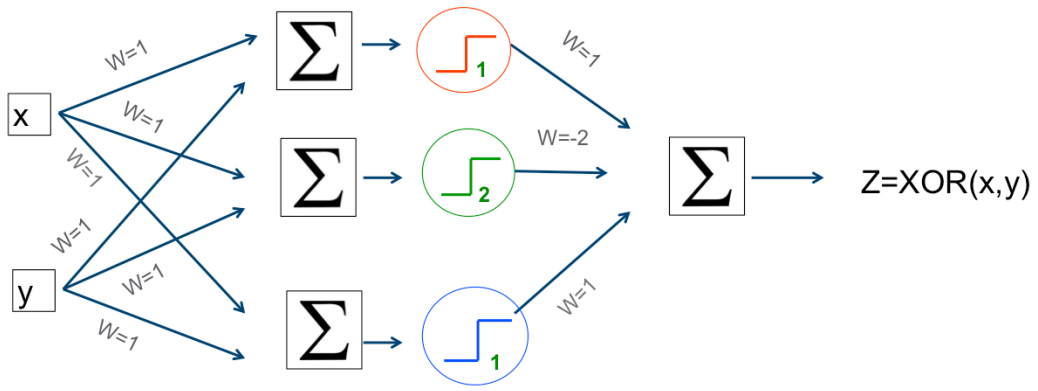
 19

Though Paul Werbos built this model in 1974, it was recognized until the early 1980's when a resurgence in the ML field occurred. Hinton, Rumelhart, Williams, and others then put the field of neural networking back on the map.



A multi-layer perceptron can be training to recognize patterns beyond binary outputs. It is commonly used in image recognition. In the slide above, you can use a MLP to recognize a hand-written character.

MLP can be trained (or coerced) to compute XOR

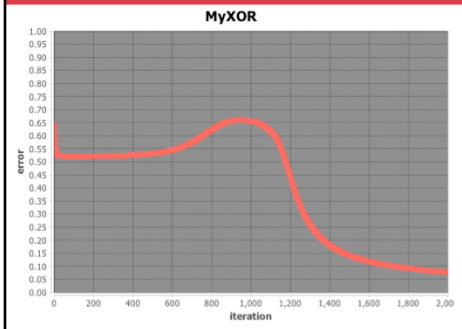


(0,0)=1*thresh{1, [0(1) + 0(1)]} + -2*thresh{2, [0(1) + 0(1)]} + 1*thresh{1, [0(1) + 0(1)]} = 0 – 0 + 0 = **0**
(0,1)=1*thresh{1, [0(1) + 1(1)]} + -2*thresh{2, [0(1) + 1(1)]} + 1*thresh{1, [0(1) + 1(1)]} = 1 – 0 + 1 = **1**
(1,0)=1*thresh{1, [1(1) + 0(1)]} + -2*thresh{2, [1(1) + 0(1)]} + 1*thresh{1, [1(1) + 0(1)]} = 1 – 0 + 1 = **1**
(1,1)=1*thresh{1, [1(1) + 1(1)]} + -2*thresh{2, [1(1) + 1(1)]} + 1*thresh{1, [1(1) + 1(1)]} = 2 – 4 + 2 = **0**

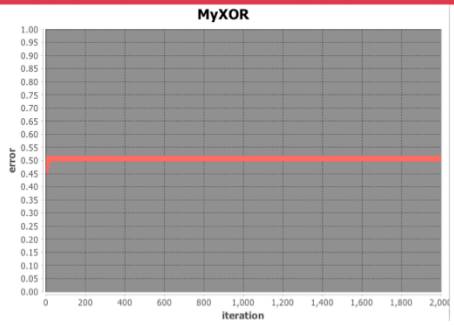
 21

And by the way, a MLP can be used to learn non-linearly separable patterns, including the infamous XOR problem.

DEMO: XOR solution using backpropagation in Java



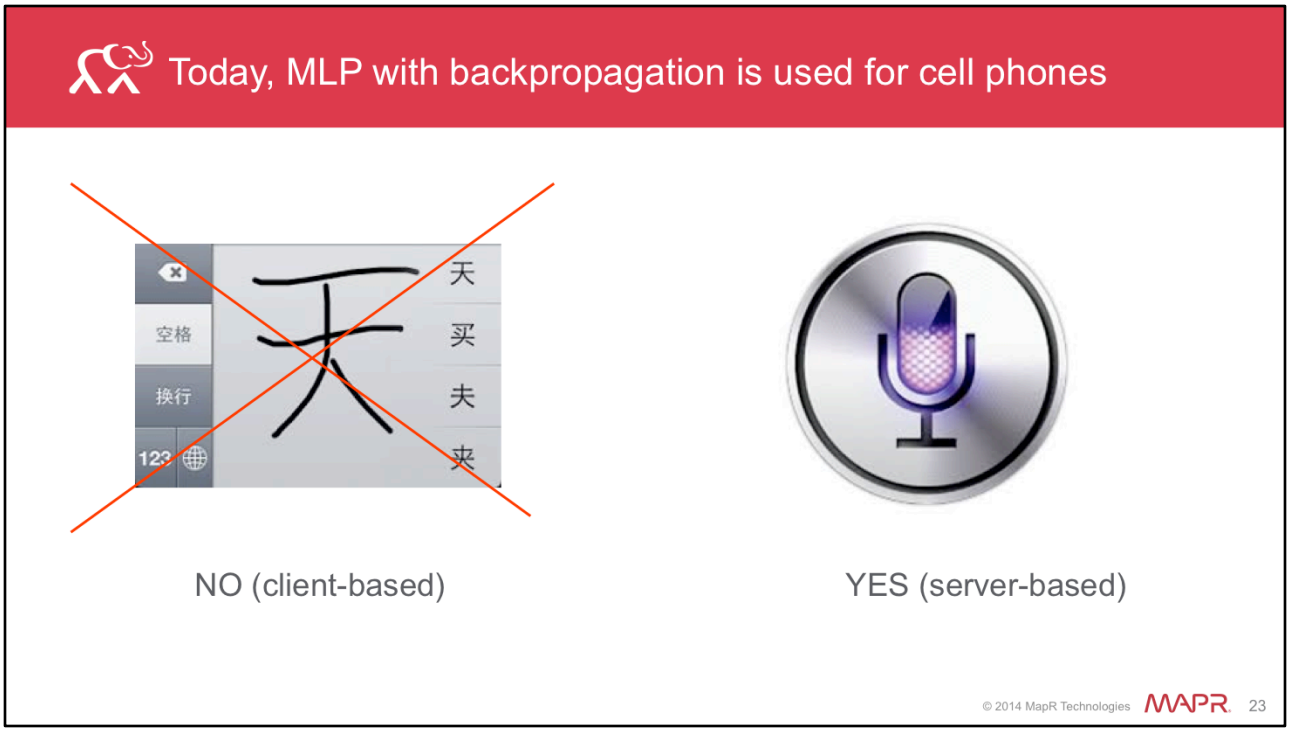

```
testing with input 1,1
actual out is 0.05695405
testing with input 1,0
actual out is 0.94305956
testing with input 0,1
actual out is 0.9227828
testing with input 0,0
actual out is 0.06565307
```



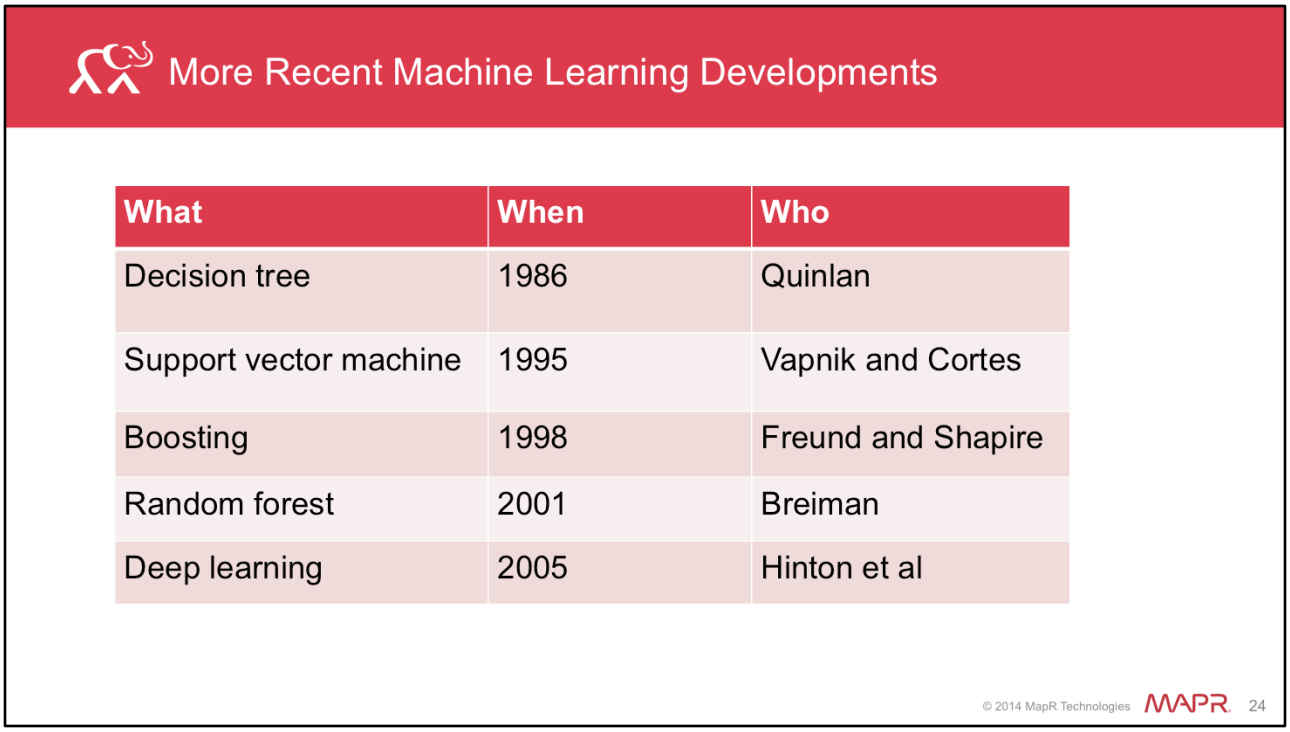

```
testing with input 1,1
actual out is 0.5122753
testing with input 1,0
actual out is 0.50953084
testing with input 0,1
actual out is 0.5074579
testing with input 0,0
actual out is 0.50321704
```

 22

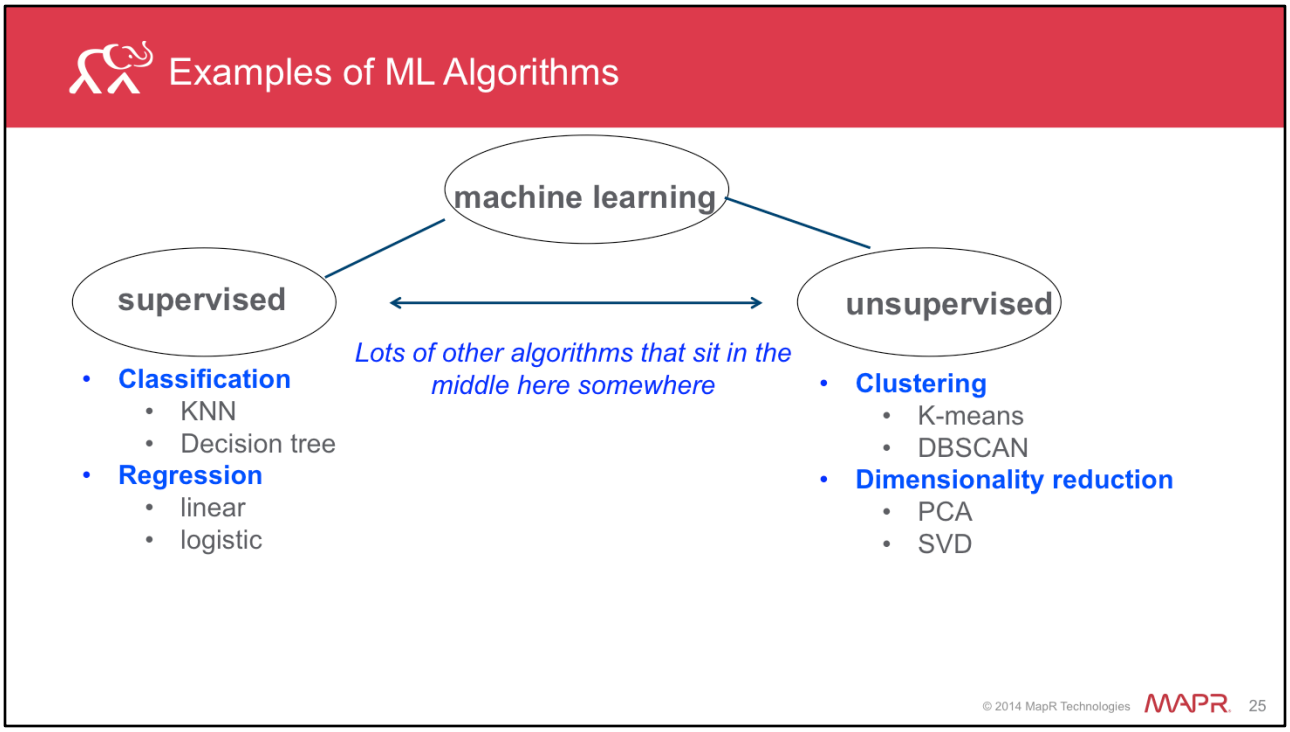
In this demo, we show that the XOR problem may be solved using a MLP. However, the algorithm we used is not guaranteed to converge. In the first case, the algorithm clearly converges after about 1,000 iterations. In the second case, the algorithm clearly doesn't converge, even after 2,000 iterations.



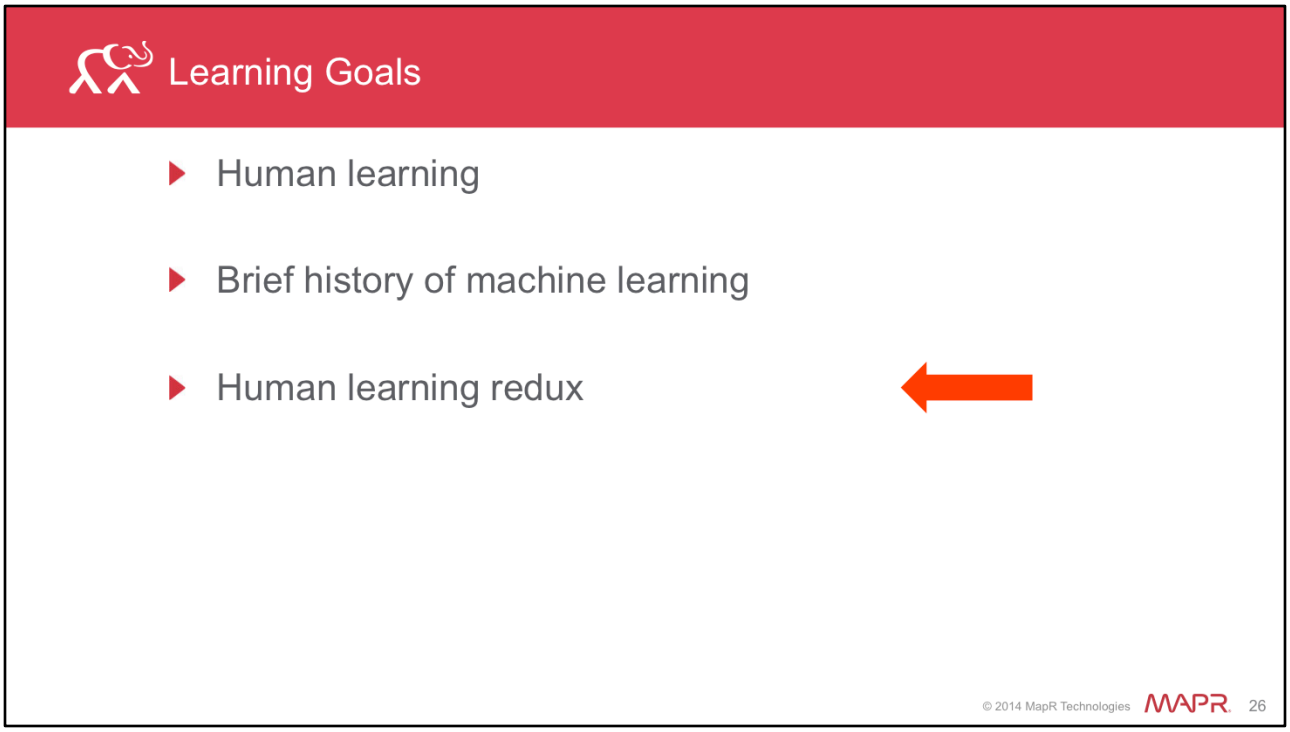
Hand-writing recognition can be achieved on a cell phone. For example, the iPhone supports interpreting hand-written Chinese characters into actual Chinese characters. These algorithms that run on the phone itself don't use MLP as the phone doesn't have sufficient resources to support the algorithm. Voice recognition programs, however, can use MLP with backpropagation. These algorithms run on an Internet server and require Internet connectivity from the mobile device.



The table above fast-forwards across history to highlight some of the more recent developments in machine learning. We will discuss all of the algorithm types shown above in this course.

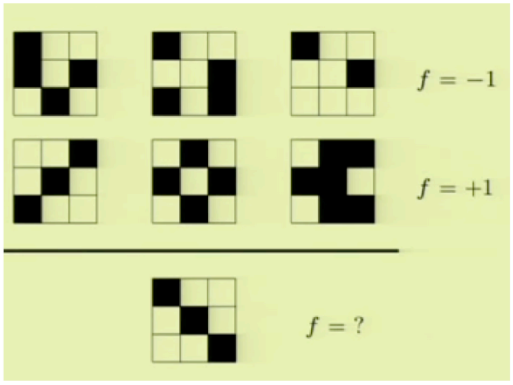


In general, machine learning may be broken down into 2 classes of algorithms: supervised and unsupervised. Supervised algorithms use so-called "labeled" data in which both the input and output are provided to the algorithm. Unsupervised algorithms do not have the outputs in advance. These algorithms are left to make sense of the data without any hints.



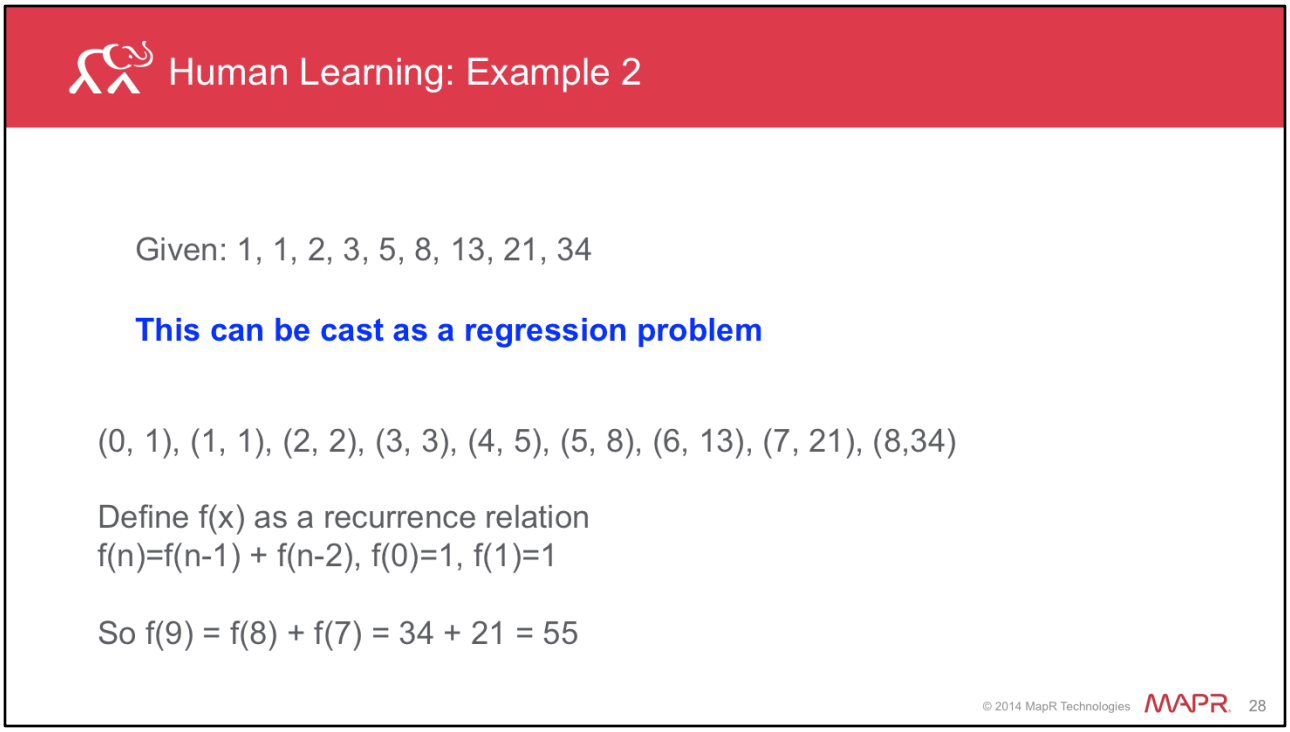
In this last section, we end where we started by reviewing the human learning examples at the beginning of this lesson.

Human Learning: Example 1



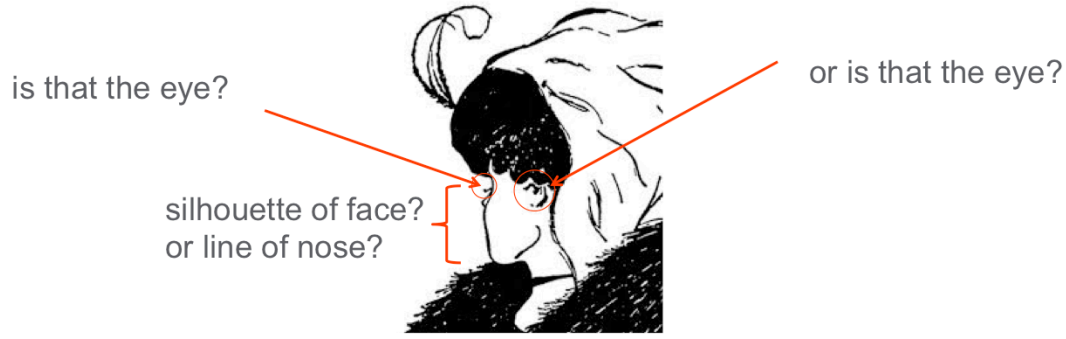
This is a binary classification problem

The problem of determining the function f can be cast as a binary classification problem. In general, classification takes an input and classifies as one of a pre-defined set of possible output values. A binary classification problem has exactly 2 possible output values. In our case, f can either be +1 or -1.



Our sequence of integers problem can be cast as a regression problem. In general, a regression algorithm attempts to predict the output value given the input value. As opposed to classification problems, regression problems do not have a finite set of possible outputs. In our case, the sequence is the Fibonacci sequence for which a recurrence relation can defined to predict the nth output value.

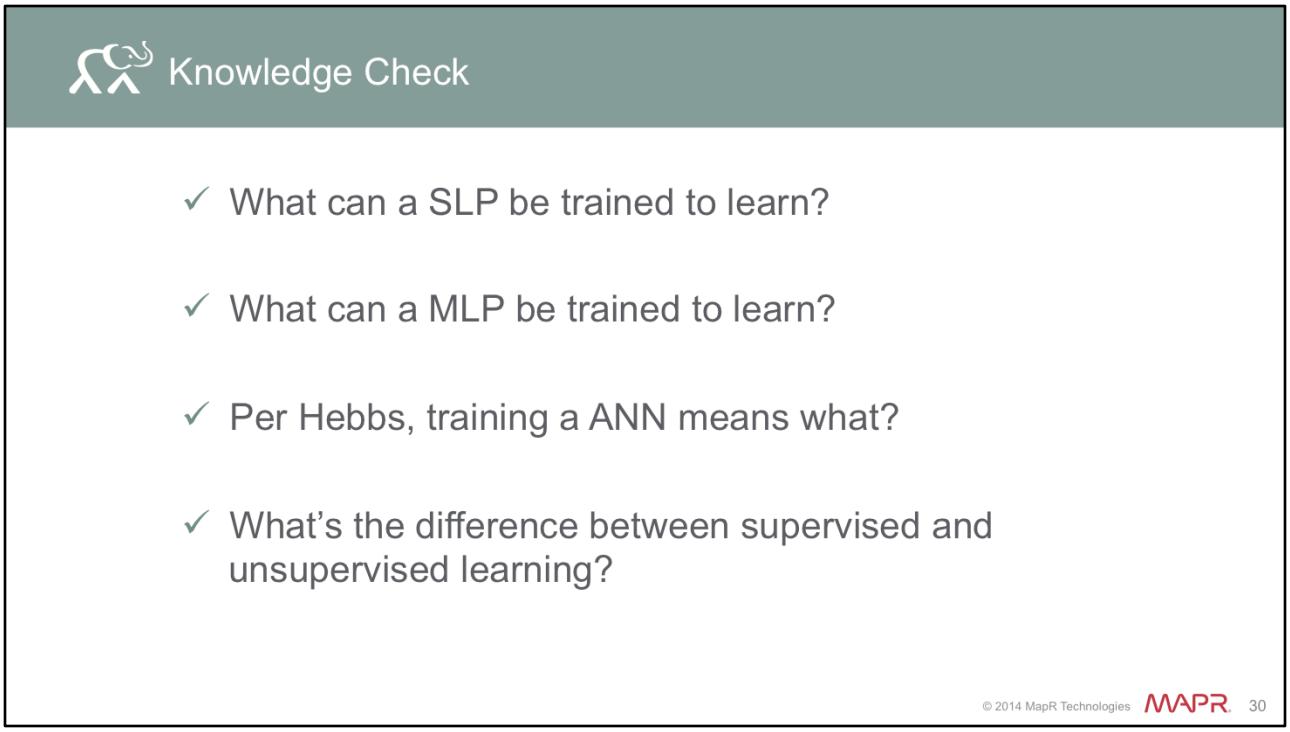
Human Learning: Example #3



Without these hints, this is an unsupervised learning problem

 29

This example in which we attempted to determine what this picture represents falls in the category of unsupervised learning problems. This is because we don't have the "answer" in advance that is associated with the input.



Answers:
SLP can be trained to learn linearly separable functions.
MLP can be trained to learn linearly and non-linearly separable functions.
Hebbian learning involves adjusting weights between neurons.
Supervised learning involves having input-output pairs during the training phase, whereas unsupervised training does not have output per se.