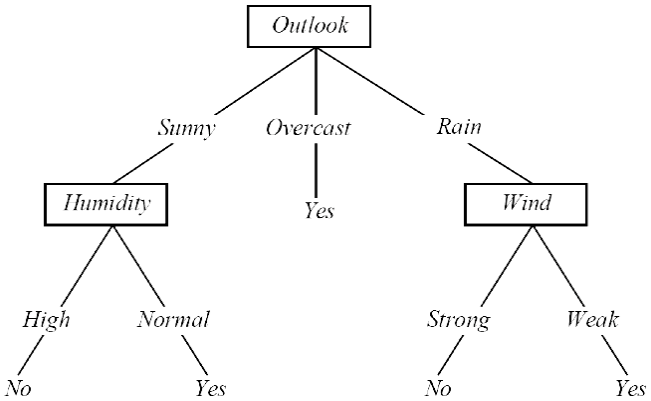
Learning

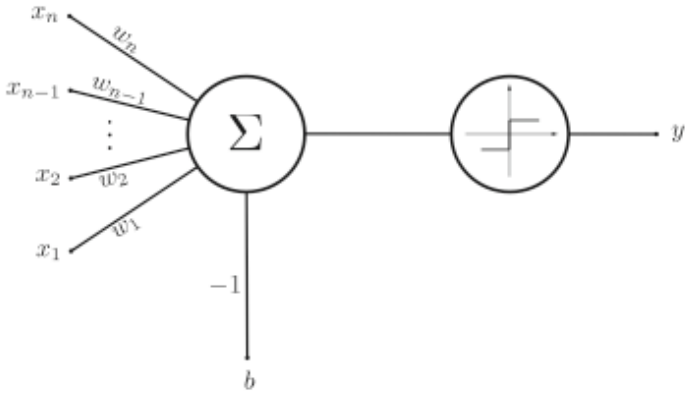
Chap 19 Alex

We have seen machine learning with different representations:

(1) Decision trees -- symbolic representation of various decision rules -- “disjunction of conjunctions”

(2) Perceptron -- learning of weights that represent alinear decision surface classifying a set of objects into two groups



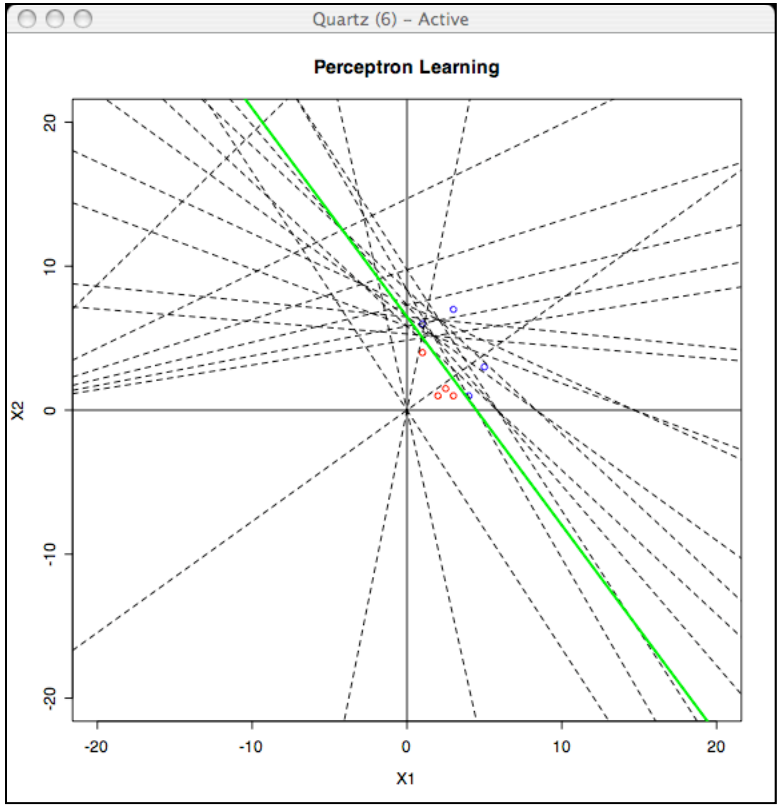


Different representations give rise to different hypothesis or model spaces.

Machine learning algorithms search these model spaces for the best fitting model.

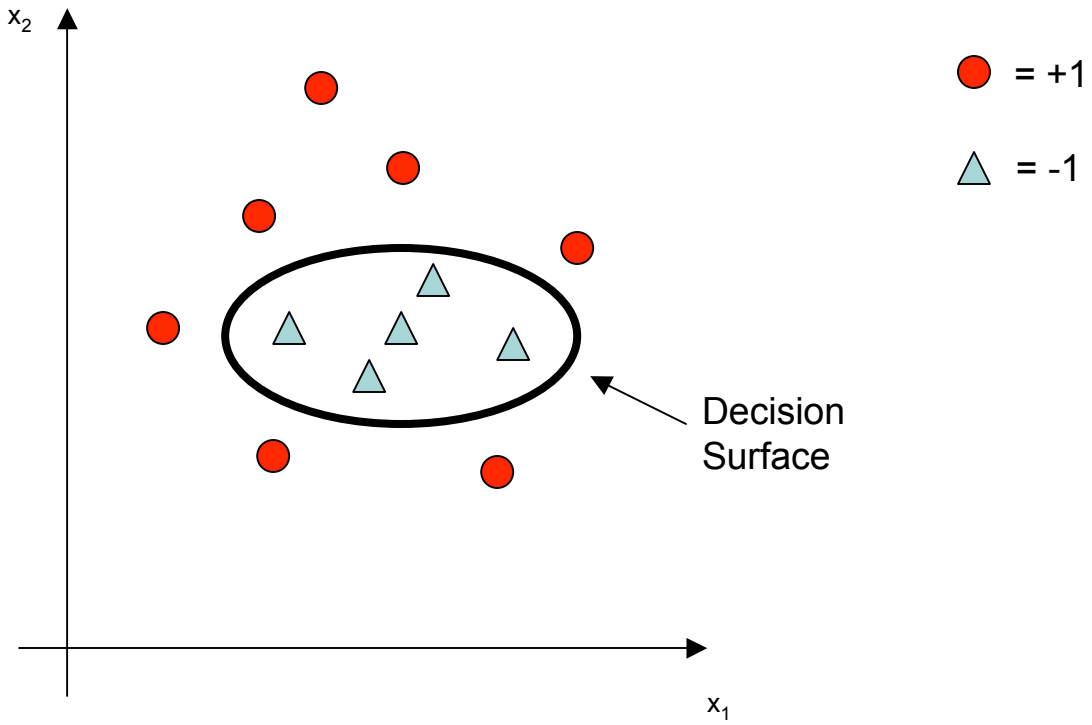
Perceptron Learning Revisited

Initialize $\overline{w}$ and b to random values.
repeat
  for each $(\overline{x}_i, y_i) \in D$ **do**
    if $\hat{f}(\overline{x}_i) \neq y_i$ **then**
      Update $\overline{w}$ and b incrementally.
    end if
  end for
until D is perfectly classified.
return $\overline{w}$ and b



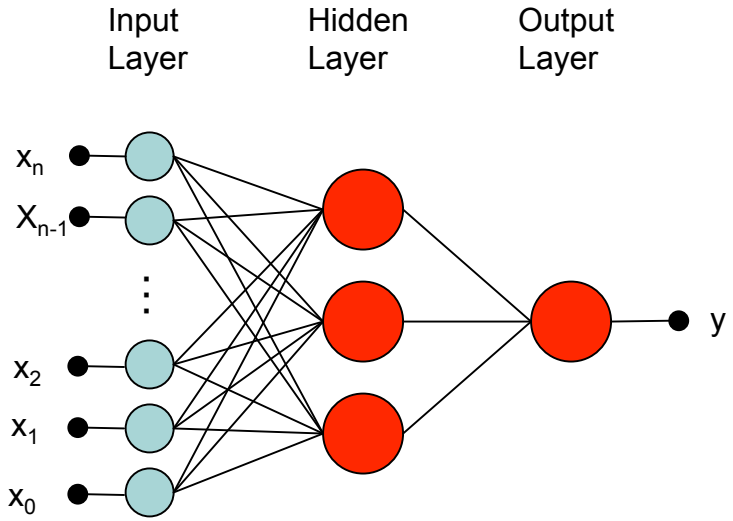
R Demo

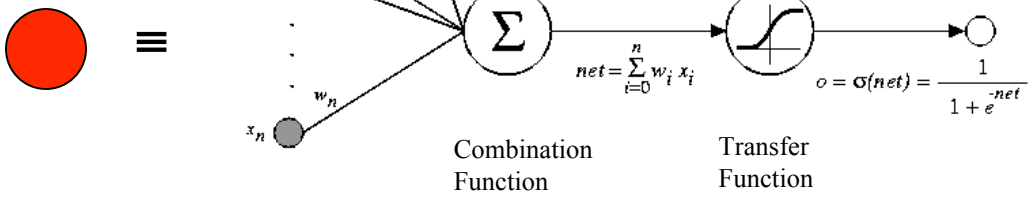
What About Non-Linearity?



Can we learn this decision surface? …Yes! Multi-Layer Perceptronsl.

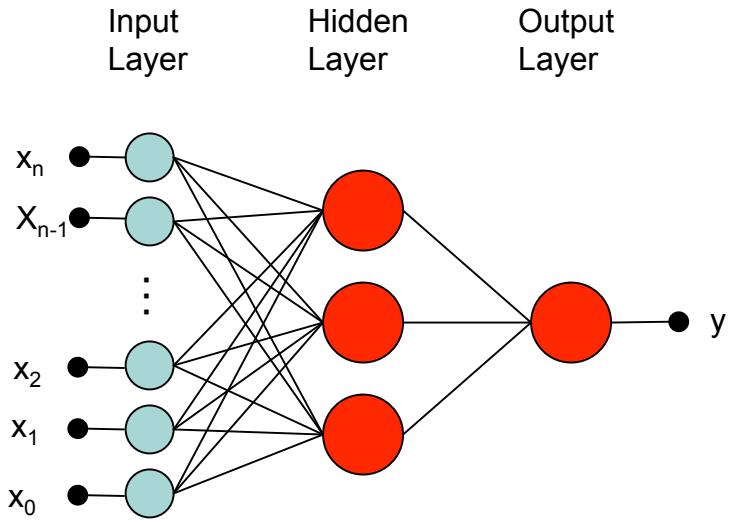
Multi-Layer Perceptrons





Artificial Neural Networks

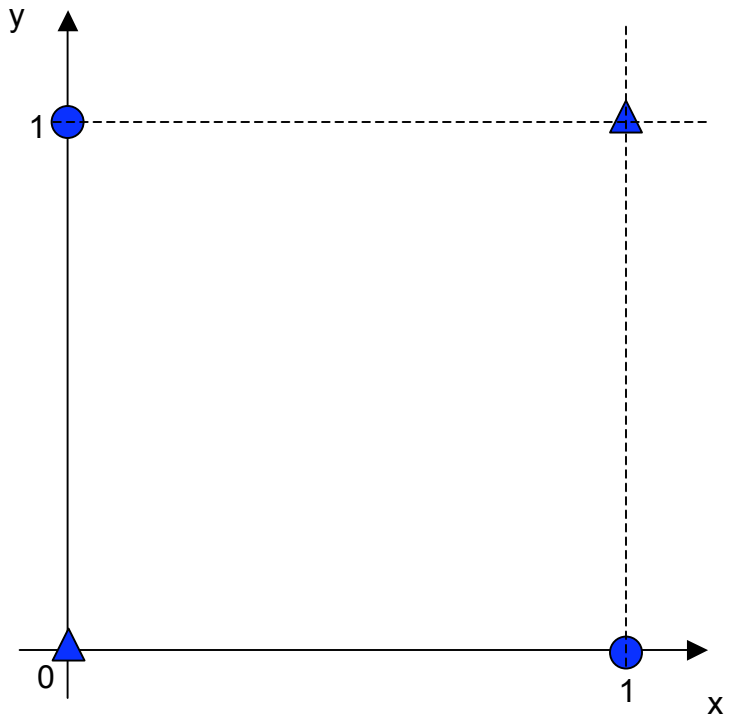
Feed-forward with Backpropagation



Signal Feed-forward

Error Backpropagation

Learning the XOR Function

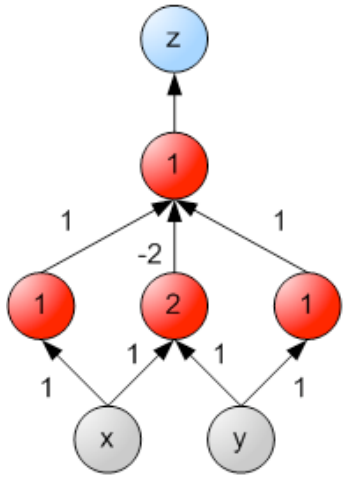


Note: no linear decision surface exists for this dataset.

= TRUE

= FALSE

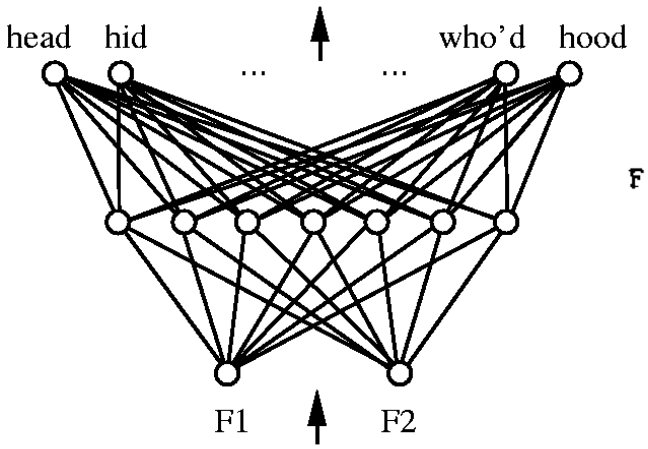
z = XOR(x, y)

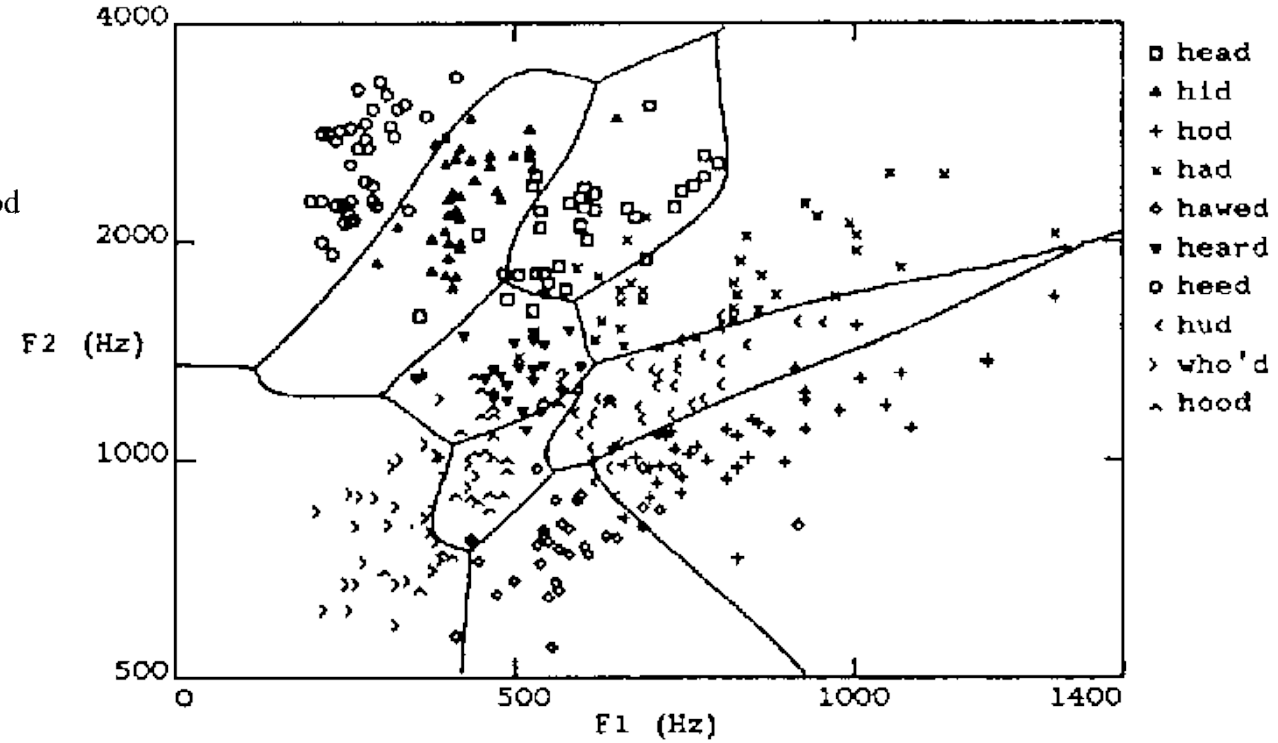


A multi-layer Perceptron capable of calculating XOR. The numbers within the perceptrons represent each perceptrons' explicit threshold. The numbers that annotate arrows represent the weight of the inputs. This net assumes that if the treshhold is not reached, zero (not -1) is output.

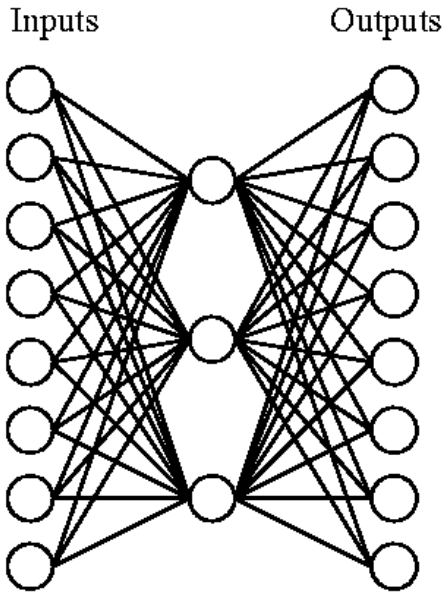
Representational Power

- Every bounded continuous function can be approximated with arbitrarily small error by a network with one hidden layer.
- Any function can be approximated to arbitrary accuracy by a network with two hidden layers.





Hidden Layer Representations

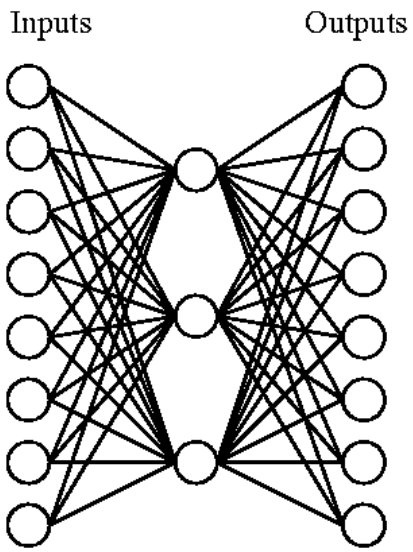


Target Function:

Input		Output
10000000	→	10000000
01000000	→	01000000
00100000	→	00100000
00010000	→	00010000
00001000	→	00001000
00000100	→	00000100
00000010	→	00000010
00000001	→	00000001

Can this be learned?

Hidden Layer Representations



Input		Hidden Values				Output
10000000	→	.89	.04	.08	→	10000000
01000000	→	.01	.11	.88	→	01000000
00100000	→	.01	.97	.27	→	00100000
00010000	→	.99	.97	.71	→	00010000
00001000	→	.03	.05	.02	→	00001000
00000100	→	.22	.99	.99	→	00000100
00000010	→	.80	.01	.98	→	00000010
00000001	→	.60	.94	.01	→	00000001

1 0 0
0 0 1
0 1 0
1 1 1
0 0 0
0 1 1
1 0 1
1 1 0

Hidden layers allow a network to invent appropriate internal representations.