

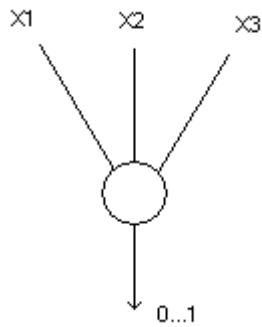
Neural Networks

Inductive Learning Problem

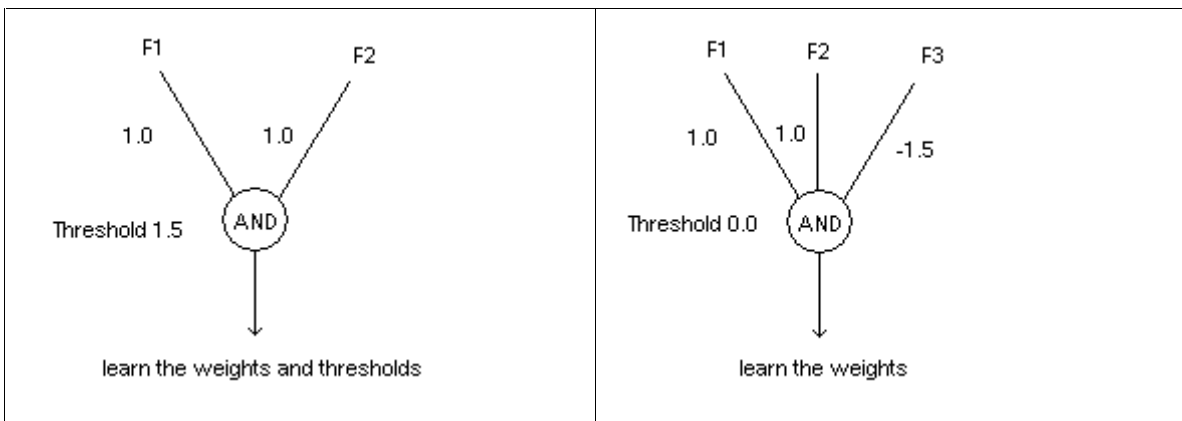
Features

Features					Class
F1	F2	F3	F4	...	
T	T	F	T	...	T
T	F	F	F	...	T
F	T	F	T	...	T
F	F	T	F	...	T
F	T	T	T	...	???

Perceptron



Thresholds as weights



Perceptron learning algorithm

- has no local minima
- will converge on a set of weights that correctly classifies the example, provided the classes are linearly separable

f_{ij} = jth feature value of training example i

$class_i$ = class of training example i

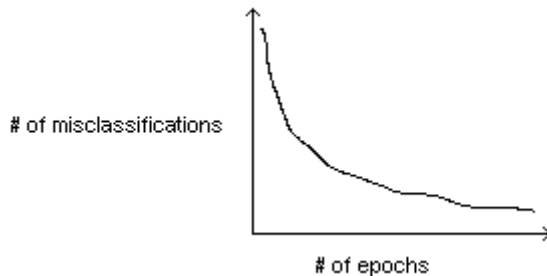
assign each weight w_j a random value

repeat

<pre>for each training example i do $o_i := g(w_1 f_{i1} + w_2 f_{i2} + \dots)$ for each weight j do $w_j := w_j + \alpha f_{ij}(class_i - o_i)$</pre>	← epoch
--	---------

alpha or α is the learning rate (small constant)

This algorithm runs until all examples are correctly predicted or a stopping criterion is reached.



Steepest (gradient) descent

Consider a perceptron without the final thresholding:

Error function

Error

$$= \frac{1}{2} \sum_i (class_i - O_i)^2$$

$$= \frac{1}{2} \sum_i (class_i - w_1 f_{i1} - w_2 f_{i2} - \dots)^2$$

gradient descent on the weights

$$w_j = w_j - \alpha \text{Error} / dw_j$$

$d\text{Error}/dw$

$$= \sum_i (class_i - O_i) d(class_i - O_i) / dw_j$$

$$= \sum_i (class_i - O_i) (-f_{ij})$$

batch update: $w_j = w_j + \alpha \sum_i f_{ij} (\text{class}_i - O_i)$

incremental update: $w_j = w_j + \alpha f_{ij} (\text{class}_i - O_i)$ *iterated over all training examples

Linearly Separable

*All the diagrams from class are in the book.

A network of perceptrons can express an XOR

Multi-layer Network of Perceptrons

