One of Pioneer of Neutral Network：Yann LeCun

Multi-class classification:

Suppose we have four class;

The output layer is the essentially four logistic regression classifiers. Each of them will capture one of the class.

Training set is the (x1,y1), (x2,y2),(x3,y3)….(xm,ym)

Y is [1,0,0,0]/[0,1,0,0]/[0,0,1,0]/[0,0,0,1] to represent the four classes.

Backpropagation algorithm:

Intuition: delta\_l\_j equals the “error” of node j in layer l.

E.g: for each output unit(layer L = 4)

delta\_4 = a\_4 – y;

delta\_3 = Theta\_3' \* delta\_4 .\* g'(z\_3)(note: g'(z\_3) equals a\_3 .\* (1 – a\_3));

delta\_2 = Theta\_2' \* delta\_3 .\* g'(z\_2)(note: g'(z\_2) equals a\_2 .\* (1 – a\_2));

Implementation of Back Propagation Algorithm:

Training set {(x1,y1), (x2, y2), … , (xm,ym)}

set Delta\_l\_ij = 0 (for all l, I, j)

For I = 1 to m

Set a\_1 = x\_i

Perform forward propagation to compute the a\_l for l = 2, 3, … , L

Using y\_i, compute delta\_L = a\_L – y\_i

Compute delta\_L-1， delta\_L-2, …, delta\_2

Delta\_l\_ij = Delta\_l\_ij + a\_l \_j \* delta\_l+1\_i

D\_l\_ij := 1/m \* Delta\_l\_ij + lambda \* Theta\_l\_ij if j != 0

D\_l\_ij := 1/m \* Delta\_l\_ij if j == 0(bias unit!)

Note: partial derivative of cost J(Theta)\_l\_ij equals to D\_l\_ij;