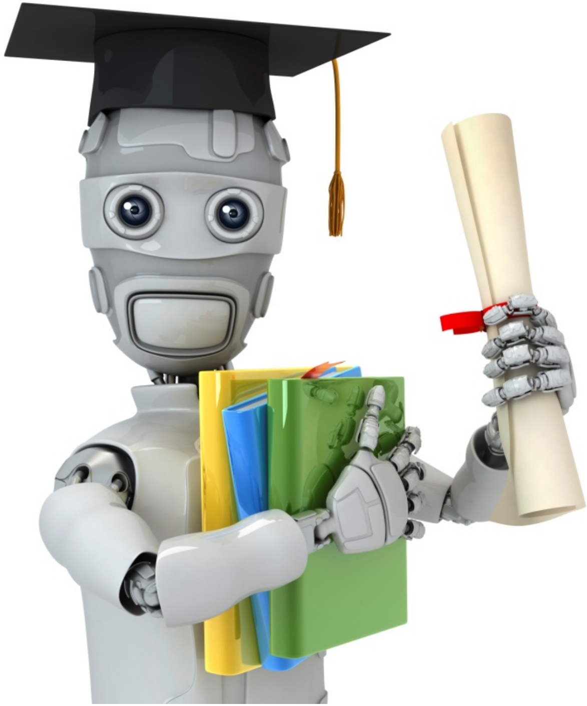


Machine Learning

Neural Networks: Learning

Random initialization

Initial value of Θ

For gradient descent and advanced optimization method, need initial value for Θ.

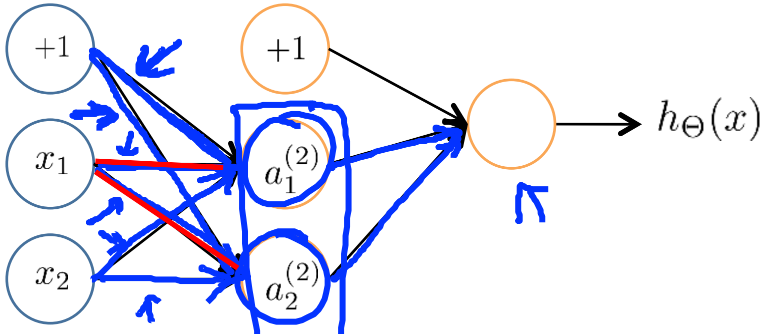
```
optTheta = fminunc(@costFunction,
           initialTheta, options)
```

Consider gradient descent

Set `initialTheta = zeros(n,1)` ?

n funcona em redes neuronais

Zero initialization



$\rightarrow \Theta_{ij}^{(l)} = 0$ for all i, j, l.

$a_1^{(2)} = a_2^{(2)}$. Also $\delta_1^{(2)} = \delta_2^{(2)}$.

$$\underline{\frac{\partial}{\partial \Theta_{01}^{(1)}} J(\Theta)} = \underline{\frac{\partial}{\partial \Theta_{02}^{(1)}} J(\Theta)}$$

$$\underline{\Theta_{01}^{(1)}} = \underline{\Theta_{02}^{(1)}}$$

After each update, parameters corresponding to inputs going into each of two hidden units are identical.

$$\underline{a_1^{(2)} = a_2^{(2)}}$$

Random initialization: Symmetry breaking

- Initialize each $\Theta_{ij}^{(l)}$ to a random value in $[-\epsilon, \epsilon]$
  (i.e. $-\epsilon \leq \Theta_{ij}^{(l)} \leq \epsilon$)

E.g.

random 10×11 matrix (betw. 0 and 1)

```
Theta1 =  rand(10,11)*(2*INIT_EPSILON)
          - INIT_EPSILON;
```

$[-\epsilon, \epsilon]$

```
Theta2 =  rand(1,11)*(2*INIT_EPSILON)
          - INIT_EPSILON;
```

Andrew Ng