deeplearning.ai

One hidden layer Neural Network

Gradient descent for neural networks

Gradient descent for neural networks

Parameters: $w^{[1]}$ $(n^{[1]}, n^{[0]})$, $b^{[1]}$ $(n^{[1]}, 1)$, $w^{[2]}$ $(n^{[2]}, n^{[1]})$, $b^{[2]}$ $(n^{[2]}, 1)$

$n_x = n^{[0]}$, $n^{[1]}$, $n^{[2]} = 1$

Cost function: $J(w^{[1]}, b^{[1]}, w^{[2]}, b^{[2]}) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}, y)$

($w^{[1]}, b^{[1]}, w^{[2]}, b^{[2]}$: $\hat{y}$ $\leftarrow a^{[2]}$)

Gradient descent:

→ Repeat {

→ Compute predicts $(\hat{y}^{(i)}, i = 1 \dots m)$

$dw^{[1]} = \frac{\partial J}{\partial w^{[1]}}$, $db^{[1]} = \frac{\partial J}{\partial b^{[1]}}$, ...

$w^{[1]} := w^{[1]} - \alpha\, dw^{[1]}$

$b^{[1]} := b^{[1]} - \alpha\, db^{[1]}$

$w^{[2]} := \dots$ $b^{[2]} := \dots$

}

Formulas for computing derivatives

Forward propagation:

$z^{[1]} = W^{[1]} X + b^{[1]}$

$A^{[1]} = g^{[1]}(z^{[1]})$

$z^{[2]} = W^{[2]} A^{[1]} + b^{[2]}$

$A^{[2]} = g^{[2]}(z^{[2]}) = \sigma(z^{[2]})$

Back propagation:

$dz^{[2]} = A^{[2]} - Y$

$Y = [y^{(1)}\ y^{(2)} \dots y^{(m)}]$

$dW^{[2]} = \frac{1}{m} dz^{[2]} A^{[1]T}$

$(n^{[2]},)$

$db^{[2]} = \frac{1}{m} \text{np.sum}(dz^{[2]}, \text{axis}=1, \text{keepdims} = \text{True})$

$(n^{[2]}, 1)$

$dz^{[1]} = \underbrace{W^{[2]T} dz^{[2]}}_{(n^{[1]}, m)} * \underbrace{g^{[1]'}(z^{[1]})}_{(n^{[1]}, m)}$

$*$ ↑ element-wise product

$dW^{[1]} = \frac{1}{m} dz^{[1]} X^T$

$db^{[1]} = \frac{1}{m} \text{np.sum}(dz^{[1]}, \text{axis}=1, \text{keepdims} = \text{True})$

$(n^{[1]}, 1)$ $(n^{[1]},)$ reshape ↑

Andrew Ng