Runqiu Ye
Stanford CS299
Problem Set #3
07/29/2024

# Problem Set #3: Deep Learning & Unsupervised Learning

**Problem 1 A simple neural network**
Let $X = \{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ be dataset of $m$ examples with 2 features. That is, $x^{(i)} \in \mathbb{R}^2$. Samples are classifed into 2 categorie with labels $y \in \{0, 1\}$, as shown in Figure 1. Want to perform binary classification using a simple neural networks with the architecture shown in Figure 2.
Two features $x_1$ and $x_2$, the three neurons in the hidden layer $h_1$, $h_2$, $h_3$, and the output neuron as $o$. Weight from $x_i$ to $h_j$ be $w_{i,j}^{[1]}$ for $i = 1, 2$ and $j = 1, 2, 3$, and weight from $h_j$ to $o$ be $w_j^{[2]}$. Finally, denote intercept weight for $h_j$ as $w_{0,j}^{[1]}$ and the intercept weight for $o$ as $w_0^{[2]}$. Use average squared loss instead of the usual negative log-likelihood:

$$l = \frac{1}{m}\sum_{i=1}^{m}(o^{(i)} - y^{(i)})^2.$$

(a) Suppose we use sigmoid function as activation function for $h_1$, $h_2$, $h_3$, and $o$. We have

$$h_1 = g(w_1^{[1]}x), \quad h_2 = g(w_2^{[1]}x), \quad h_3 = g(w_3^{[1]}x), \quad o = g(w^{[2]}h).$$

Hence,

$$\frac{\partial l}{\partial w_{1,2}^{[1]}} = \frac{1}{m}\sum_{i=1}^{m} 2(o^{(i)} - y^{(i)})o^{(i)}(1 - o^{(i)})w_2^{[2]}h_2^{(i)}(1 - h_2^{(i)})x_1^{(i)},$$

where $h_2^{(i)} = g(w_{0,2}^{[1]} + w_{1,2}^{[1]}x_1^{(i)} + w_{2,2}^{[1]}x_2^{(i)})$ and $g$ is the sigmoid function. Therefore, the gradient descent update to $w_{1,2}^{[1]}$, assuming learning rate $\alpha$ is

$$w_{1,2}^{[1]} := w_{1,2}^{[1]} - \frac{2\alpha}{m}\sum_{i=1}^{m}(o^{(i)} - y^{(i)})o^{(i)}(1 - o^{(i)})w_2^{[2]}h_2^{(i)}(1 - h_2^{(i)})x_1^{(i)}$$

where $h_2^{(i)} = g(w_{0,2}^{[1]} + w_{1,2}^{[1]}x_1^{(i)} + w_{2,2}^{[1]}x_2^{(i)})$.

(b) Now, suppose the activation function for $h_1$, $h_2$, $h_3$, and $o$ is the step function $f(x)$, defined as

$$f(x) = \begin{cases} 1, & (x \geq 0), \\ 0, & (x < 0). \end{cases}$$

Is it possible to have a set of weights that allow the neural network to classify this dataset with 100% accuracy? If so, provide a set of weights by completing `optimal_step_weights` wihin `src/p01_nn.py` and explain your reasoning for those weights. If not, please explain the reasoning.

There is a set of weights that allow the neural network to classify this dataset with 100% accuracy. For the step function activation, we have

$$\begin{aligned}
h_1 &= f(w_1^{[1]}x) = f(w_{0,1}^{[1]} + w_{1,1}^{[1]}x_1 + w_{2,1}^{[1]}x_2) \\
h_2 &= f(w_2^{[1]}x) = f(w_{0,2}^{[1]} + w_{1,2}^{[1]}x_1 + w_{2,3}^{[1]}x_2) \\
h_3 &= f(w_3^{[1]}x) = f(w_{0,3}^{[1]} + w_{1,3}^{[1]}x_1 + w_{2,3}^{[1]}x_2) \\
o &= f(w^{[2]}h) = f(w_0^{[2]} + w_1^{[2]}h_1 + w_2^{[2]}h_2 + w_3^{[2]}h_3).
\end{aligned}$$

Notice from Figure 1 that the label $y^{(i)} = 0$ if and only if $x^{(i)}$ satisfies

$$\begin{cases}
x_2^{(i)} > 0.5, \\
x_1^{(i)} > 0.5, \\
x_1^{(i)} + x_2^{(i)} < 4.
\end{cases}$$

Now, let

$$w_1^{[1]} = \begin{bmatrix} 0.5 \\ 0 \\ -1 \end{bmatrix}, \quad w_2^{[1]} = \begin{bmatrix} 0.5 \\ -1 \\ 0 \end{bmatrix}, \quad w_3^{[1]} = \begin{bmatrix} -4 \\ 1 \\ 1 \end{bmatrix}, \quad w_1^{[1]} = \begin{bmatrix} -0.5 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

This set of weights will capture all the conditions and allow the nerual network to classify this dataset with 100% accuracy.

■