Neural Network (Classification). $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$.
$L =$ total number of layers.
$s_l =$ number of neuron (unit), not counting bias unit, in layer l.

Binary Classification. $y = 0$ or 1. In last layer, only have 1 output unit, i.e., $h_\Theta(x) \in \mathbb{R}$, or $s_L = K = 1$.

Multi-class classification (K classes). $y \in \mathbb{R}^K$. K output units, $h_\Theta(x) \in \mathbb{R}^K$, $s_L = K$, $K \geq 3$.

Cost Function. Regularized Logistic regression:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} log h_\theta(x^{(i)}) + (1 - y^{(i)}) log(1 - h_\theta(x^i))\right] + \frac{\lambda}{2m}\sum_{j=1^n} \theta_j^2$$

Neural network: $h_\Theta(x) = \mathbb{R}^K, (h_\Theta(x))_i = i$th output.

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1} y_k^{(i)} log(h_\Theta(x^{(i)}))_k + (1 - y_k^{(i)}) log(1 - (h_\Theta(x^{(i)}))_k)\right] - \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_t}\sum_{j=1}^{s_l+1}(\Theta_{ji}^{(l)})^2$$

Gradient Computation. We want to minimize $J(\Theta)$. Need code to compute $J(\Theta)$, $\frac{\partial}{\partial\Theta_{ij}^{(l)}} J(\Theta)$. Imagine there is only one training example (x, y). Forward propagation:

$$\begin{aligned}
a^{(1)} &= x \\
z^{(2)} &= \Theta^{(1)} a^{(1)} \\
a^{(2)} &= g(z^2) \text{add } a_0^{(2)} \\
z^{(3)} &= \Theta^{(2)} a^{(2)} \\
a^{(3)} &= g(z^3) \text{add } a_0^{(3)} \\
z^{(4)} &= \Theta^{(3)} a^{(3)} \\
a^{(4)} &= h_\Theta(x) = g(z^{(4)})
\end{aligned}$$

Gradient Computation: Backpropagation algorithm