斯坦福机器学习整理5——第五周

Neural learning

Cost Function

代价函数：

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log h_\theta(x^{(i)})_k + (1 - y_k^{(i)}) \log(1 - h_\theta(x^{(i)})_k)] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\theta_j^{(l)})^2$$

Forward Propagation(前向传播)

Back Propagation(反向传播)

用来计算代价函数的导数。 反向传播算法就是计算一个项$\delta_j^{(l)}$，代表第l层上单元j的激励误差。

假设训练集为$\{(x^{(1)}, y^{(1)}), \cdots, (x^{(m)}, y^{(m)})\}$，算法流程：

1. 令$\Delta_{ij}^{(l)} = 0 \ (for \ all \ l, i, j)$
2. For $i = 1$ to m
    1. 令 $a^{(1)} = x^{(i)}$
    2. 用前向传播计算$a^{(l)}$ for $l = 2, 3, \cdots, L$
    3. 用$y^{(i)}$，计算$\delta^{(L)} = a^{(L)} - y^{(i)}$
    4. 计算$\delta^{(L-1)}, \delta^{(L-2)}, \cdots, \delta^{(2)}$
    5. $\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)}\delta_i^{(l+1)}$
3. $D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)} + \lambda\theta_{ij}^{(l)}$ if $j \neq 0$

   $D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)}$ if $j = 0$

   $$\frac{\partial}{\partial\theta_{ij}^{(l)}}J(\theta) = D_{ij}^{(l)}$$
4. 代价函数的偏导数为

Gradient Checking(梯度检验)