

## LMS算法

在上一小节中，我们定义了代价函数 $J(\theta)$ ，当其值最小时，我们则称找寻到了拟合数据集最佳的参数 $\theta$ 的值。那么我们又该如何选择参数 $\theta$ 的值使得代价函数 $J(\theta)$ 最小化？

我们不妨考虑使用搜索算法（Search Algorithm），其先将参数 $\theta$ 初始化，然后不断给参数 $\theta$ 赋予新值，直至代价函数 $J(\theta)$ 收敛于最小值处。因此，我们可以梯度下降算法（Gradient Descent Algorithm），其更新规则为：

$$\begin{aligned} & \text{Repeat until convergence} \{ \\ & \quad \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \quad (\text{for every } j) \\ & \} \end{aligned}$$

其中， $\alpha$ 表示学习速率，其值过小会导致迭代多次才能收敛，其值过大可能导致越过最优点而发生震荡现象。

当只有一个训练实例时，偏导数的计算公式如下：

$$\begin{aligned} \frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_{\theta}(x) - y)^2 \\ &= 2 * \frac{1}{2} (h_{\theta}(x) - y) * \frac{\partial}{\partial \theta_j} (h_{\theta}(x) - y) \\ &= (h_{\theta}(x) - y) * \frac{\partial}{\partial \theta_j} \left( \sum_{j=0}^n \theta_j x_j - y \right) \\ &= (h_{\theta}(x) - y) x_j \end{aligned}$$

因此，梯度下降算法的更新规则可改写为如下：

$$\begin{aligned}
 & \textit{Repeat until convergence} \{ \\
 & \quad \theta_j := \theta_j - \alpha \sum_{i=1}^m (h_{\theta}(x^i) - y^{(i)}) x_j^{(i)} \quad (\textit{for every } j) \\
 & \}
 \end{aligned}$$

其中，对于单个训练实例其更新规则为：

$$\theta_j := \theta_j - \alpha (h_{\theta}(x^i) - y^{(i)}) x_j^{(i)}$$

这规则被称为Widrow-Hoff学习规则，其也被称最小二乘法（LMS，Least Mean Squares）。

由于该算法在计算参数 $\theta_j$ 时都要遍历训练集，因此该算法也称为批量梯度下降算法（Batch Gradient Descent）。

对于训练集较大时，使用批量梯度下降算法其计算成本较大。因此，我们推荐采用随机梯度下降算法（Stochastic Gradient Descent）（也称为增量梯度下降算法，Increment Gradient Descent）。其更新规则如下：

$$\begin{aligned}
 & \textit{Repeat until convergence} \{ \\
 & \quad \textit{for } i = 1 \textit{ to } m, \{ \\
 & \quad \quad \theta_j := \theta_j - \alpha (h_{\theta}(x^i) - y^{(i)}) x_j^{(i)} \quad (\textit{for every } j) \\
 & \quad \} \\
 & \}
 \end{aligned}$$

该算法在计算参数 $\theta_j$ 时，只需一个训练实例，其大大提高了运行效率。但该算法无法收敛至最优值处，只能无限接近该值。一般情况下，其值也足够我们最佳地拟合训练集了。