斯坦福 ML 公开课笔记 3

这篇笔记针对公开课的第三个视频，主要内容包括局部加权回归（LWR，Locally/Loess Weighted Regression）、最小二乘法的概率解释、逻辑斯蒂回归（logistic regression）、感知器算法。

首先，Ng 分析了过拟合与欠拟合的概念。举了三个例子，第一个例子还是房价-大小，即用大小这一个变量构建一个线性回归模型来拟合房价，第二个例子是房价-大小/大小的平方，即用两个变量构建一个线性回归模型来拟合房价，尽管第二个变量与第一个变量呈平方关系，但这样或许会拟合的更好；第三个例子是对于一个有七个样例的房价-大小关系，用六次多项式对其进行建模来拟合房价。显然，若第二个例子的方法比第一个例子要好的话，那么第一个例子就是欠拟合（underfitting），第三个例子是过拟合（overfitting）。

过拟合的形式化定义在《ML》[1]中这样给出：给定一个假设空间 H，一个假设 h 属于 H，如果存在其他的假设 h_1，使得在训练样例上 h 的错误率比 h_1 好，但在整个实例分布上 h_1 的错误率比 h 小，那么就说假设 h 是过拟合。欠拟合的定义与之相似。

然后，给出了参数学习算法（Parametric Learning Method）和非参数学习算法（Non-Parametric Learning Method）的定义。参数学习算法是一类有固定数目参数的用来进行数据拟合的算法，线性回归即是此类。非参数学习算法则是一类参数数目随数据集增大而变多（一般是线性增大）的算法，接下来要讨论的局部加权回归即是此类。

局部加权回归

局部加权回归算法是对线性回归的扩展，当目标假设是线性模型时，使用线性回归自然能拟合的很好，但如果目标假设不是线性模型，比如一个忽上忽下的的函数，这时用线性模型就拟合的很差。为了解决这个问题，当我们在预测一个点的值时，我们选择和这个点相近的点而不是全部的点做线性回归。基于这个思想，就有了局部加权回归算法，它的目标函数是加权的最小二乘：

$$J(\theta) = \sum_i w^{(i)}(y^{(i)} - \theta^T x^{(i)})^2 \tag{1}$$

[1] Tom Mitchell 著

其中，$w^{(i)}$是权值，它的作用在于根据要预测的点与数据集中的点的距离来为数据集中的点赋权值，当某点距离待预测点较远时，其权重较小，否则较大。一个较好的函数如下：

$$w^{(i)} = \exp(-\frac{(x^{(i)} - x)^2}{2\tau^2}) \tag{2}$$

该函数被称为指数衰减函数。其中，τ被称为波长参数，它控制了权值随距离下降的速率。该函数比较像但不是高斯分布（Gaussian Distribution）或正态分布（Normal Distribution）。

在学习这个算法的时候，不仅需要学习线性回归的参数，函数要学习波长参数。

这个算法的问题在于，对于每一个要查询的点，都要重新依据整个数据集计算一个线性回归模型出来，这样做使得计算代价极高。Andrew Moore[2]的 kd-tree 算法[3]可以对该问题进行优化。

最小二乘法的概率解释

接下来对线性回归的最小二乘法的合理性做了概率解释，即为什么选择平方函数作为目标函数会使得效果比较好？

假设一，对于每一个样例$(x^{(i)},y^{(i)})$，特征值 x 和目标值 y 的关系可以表示成：

$$y^{(i)} = \theta^T x^{(i)} + \varepsilon^{(i)} \tag{3}$$

其中，$\varepsilon^{(i)}$表示线性模型与目标值的误差。

假设二，$\varepsilon^{(i)}$服从正态分布：

$$\varepsilon \sim N(0, \sigma^2) \tag{4}$$

假设一只是一种表示方法，成立很合理。那么假设二为何会成立呢？这是因为影响误差的因素有很多，这些因素都是随机分布，根据中心极限定理（Central Limit Thoery），即许多独立随机变量的和趋向于正态分布，我们可以得到假设二。

有了假设二之后，我们可以得到：

$$\mathrm{P}(\varepsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{(\epsilon^{(i)})^2}{2\sigma^2}) \tag{5}$$

[2] 其个人主页：http://www.cs.cmu.edu/~awm/

[3] 应该是这篇：Very Fast EM-based Mixture Model Clustering Using Multiresolution KD-trees

这也表示，当给定参数θ和 x 时，目标值 y 也服从正态分布：

$$\mathrm{P}\left(y^{(i)}|x^{(i)};\theta\right)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{\left(\theta^{T}x^{(i)}-y^{(i)}\right)^{2}}{2\sigma^{2}}\right) \tag{6}$$

注意到$x^{(i)}$与θ间的分号，它表示的是θ是已知变量而非随机变量。

假设三：对于各个样例的误差$\varepsilon^{(i)}$，它们是 iid（独立同分布，independent and identical distribution）随机变量。

这样，我们就可以得到似然函数：

$$\begin{aligned}\ell(\theta)&=\mathrm{P}(\mathrm{Y}|\mathrm{X};\theta)\\&=\prod_{i=1}^{m}\mathrm{P}\left(y^{(i)}|x^{(i)};\theta\right)=\prod_{i=1}^{m}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{\left(\theta^{T}x^{(i)}-y^{(i)}\right)^{2}}{2\sigma^{2}}\right)\end{aligned} \tag{7}$$

其中，Y 是一个长度为样例数的向量，X 是样例数*特征数的矩阵。

似然函数的意义是什么？似然函数表示的是在参数θ下，数据集出现的概率。似然函数与概率的概念其实相似，不同之处在于似然函数把θ作为变量，找到使得数据集出现的概率最大时的参数，就称为极大似然估计。

对公式 7 的右端取对数，我们发现将似然函数最大的问题可以转化为使得平方和最小的问题。如公式 8 所示。

$$\begin{aligned}\mathrm{L}(\theta)=\log\ell(\theta)&=\log\prod_{i=1}^{m}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{\left(\theta^{T}x^{(i)}-y^{(i)}\right)^{2}}{2\sigma^{2}}\right)\\&=-mlog\sqrt{2\pi}\sigma-\frac{1}{2\sigma^{2}}\sum_{i=1}^{m}\left(\theta^{T}x^{(i)}-y^{(i)}\right)^{2}\end{aligned} \tag{8}$$

逻辑斯蒂回归

对于目标值是连续变量的问题来说，使用线性回归可能会解决的很好，即便其问题不是线性模型所能解决的，也可以用局部加权回归解决。但对于目标值是离散变量的分类问题来说，应用线性模型会有一定的困难。当然，有些这类问题也可以应用线性模型，但线性模型绝不是一种通用的解决这类问题的方法。

对于目标值是离散变量的两类分类问题，假设目标值为{0,1}，所以先改变模型使其预测值在[0,1]之间，我们选择这样一个函数：

$$h_{\theta}(x)=g(\theta^{T}x)=\frac{1}{1+e^{-\theta^{T}x}} \tag{9}$$

$$g(z) = \frac{1}{1+e^{-z}} \tag{10}$$

其中，函数 g 被称为 logistic 函数或 sigmoid 函数。至于为什么会选择这个函数，会在以后提到。

有了这个函数，对于一个样例，我们就可以得到它分类的概率值：

$$P(y=1|x;\theta) = h_\theta(x) \tag{11}$$

$$P(y=0|x;\theta) = 1 - h_\theta(x) \tag{12}$$

将公式 11 和公式 12 组合起来，我们可以得到公式如下：

$$P(y|x;\theta) = \left(h_\theta(x)\right)^y (1-h_\theta(x))^{(1-y)} \tag{13}$$

通过分情况可以证明，公式 13 与公式 11 和公式 12 的组合等价。

这样，我们就得到了函数 h 在整个数据集上的似然函数：

$$\ell(\theta) = P(Y|X;\theta) = \prod_i P\left(y^{(i)}|x^{(i)};\theta\right)$$

$$= \prod_i \left(h_\theta\left(x^{(i)}\right)\right)^{y^{(i)}} \left(1-h_\theta\left(x^{(i)}\right)\right)^{(1-y^{(i)})} \tag{14}$$

同样的，为了计算方便，对似然函数取对数：

$$L(\theta) = \log \ell(\theta)$$

$$= \sum_{i=1}^{m} [y^{(i)} log\, h_\theta\left(x^{(i)}\right) + (1-y^{(i)}) \log(1-h_\theta\left(x^{(i)}\right))] \tag{15}$$

对公式 15 应用梯度下降算法，我们得到更新规则如下：

$$\theta := \theta + \alpha \nabla_\theta L(\theta) \tag{16}$$

求公式 16 中的导数时，我们回忆一下笔记 1-2 中的为最小二乘法求梯度下降的导数时的做法，我们先假设有一个样例，这样，导数的解法如下：

$$\nabla_{\theta_j} L(\theta) = \nabla_{\theta_j} \left[y \log(1-h_\theta(x)) + (1-y)\log(1-h_\theta(x))\right]$$

$$= \frac{y}{h_\theta(x)} \nabla_{\theta_j} h_\theta(x) + (1-y) \frac{1}{1-h_\theta(x)} (-\nabla_{\theta_j} h_\theta(x))$$

$$= \frac{x_j y e^{-\theta^T x}}{1+e^{-\theta^T x}} + (y-1) \frac{1}{1-h_\theta(x)} (\nabla_{\theta_j} h_\theta(x))$$

$$= \frac{x_j y e^{-\theta^T x}}{1+e^{-\theta^T x}} + \frac{(y-1)x_j}{1+e^{-\theta^T x}} = x_j y - \frac{x_j}{1+e^{-\theta^T x}}$$

$$= x_j(y - h_\theta(x)) \tag{17}$$

考虑到多个样例，所以更新规则为：

$$\theta := \theta + \alpha \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)} \tag{18}$$

公式 18 与最小二乘的形式一样，但是实际上是不一样的，因为函数 h 不一样。但这并不是巧合，这几乎是一种通用的规则，你可以选择不同的假设，但如果使用梯度下降算法的话，更新规则都是如公式 18 的形式。

感知器算法

感知器算法强迫函数输出值为{0,1}离散值而不是概率。其假设为：

$$h_\theta(x) = g(\theta^T x) = \begin{cases} 0 & if(\theta^T x > 0) \\ 1 & otherwise \end{cases} \tag{19}$$

在这个假设的基础上，我们使用与公式 18 形式相同的规则，就得到了感知器算法。感知器算法是人工神经网络的基础，在后面的学习理论中，将把它作为分析的起点。