线性模型

本文是观看斯坦福大学机器学习公开课2~4讲视频及阅读配套讲义notes1后所做笔记。本文主要介绍线性回归、逻辑回归和一般线性模型的逐步引入和推导过程。

by Jimmy

2016年2月29日

1 线性回归

在回归问题中，取样本的维度（即每个样本中特征的数目）为n、样本数目为m，则线性回归模型假设可以表示为

$$h_\theta(x) = \sum_{j=0}^{n} \theta_j x_j = \theta^T x$$

在这里取$x_0 = 1$（即增加截距项θ_0），为了衡量模型与目标变量（即样本对应的真实值）的偏差程度，取最小二乘方程作为代价函数（cost function）

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

当代价函数$J(\theta)$取值最小时，模型显然对已有样本的描述最为准确，我们需要找到使代价函数取最小值的参数θ。

1.1 θ的解法

1.1.1 梯度下降算法

利用梯度下降算法求θ的过程为

1. 初始化θ（任意初值）
2. 计算$J(\theta)$对θ的偏导（一个向量，即全局梯度∇_θ），按照梯度相反的方向更新θ，更新法则为$\theta := \theta - \alpha\nabla_\theta$，其中$\alpha$称为学习率，表示更新的步长。如果学习率太小，那么收敛太慢；如果学习率太大，则不能保证收敛
3. 重复步骤2直至收敛（即前后两次θ相差小于预设阈值）或者迭代次数上限，输出θ

对一个样本可以求得偏导（式1）

$$
\begin{aligned}
\frac{\partial}{\partial\theta_j}J(\theta) &= \frac{\partial}{\partial\theta_j}\frac{1}{2}(h_\theta(x)-y)^2 \\
&= \frac{1}{2}\cdot 2(h_\theta(x)-y)\frac{\partial}{\partial\theta_j}(h_\theta(x)-y) \\
&= (h_\theta(x)-y)\frac{\partial}{\partial\theta_j}(\sum_{i=0}^{n}\theta_i x_i - y) \\
&= (h_\theta(x)-y)x_j
\end{aligned}
$$

因此参数的更新法则为

$$\theta_j := \theta_j - \alpha\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_j^{(i)} \qquad \text{(for every j)}$$

算法每一步更新都需要遍历所有的样本，虽然最后可以得到一个全局的最优解，但是计算耗时、收敛缓慢。

1.1.2 随机梯度下降算法

利用随机梯度下降算法求θ的过程为

1. 初始化θ，将当前样本标志flag置为1
2. 利用（式1）计算当前样本的偏导（即当前样本的梯度），然后按照更新法则 $\theta_j := \theta_j - \alpha(h_\theta(x^{(i)})-y^{(i)})x_j^{(i)}$更新$\theta$
3. 当前样本标志flag+1，如果flag=n则将flag重置为1。重复步骤2直至收敛，输出θ

随机梯度下降算法算法的每一次更新只用到了一个样本，当样本数目n很大时，甚至可能出现没有遍历完所有样本就收敛的情况，因此速度快很多。需要注意的是，这里的每一次更新并非朝着全局最优方向，而是局部最优方向（意味着θ中有些位置收敛很快，有些很慢）；而且由于噪音的存在，甚至有可能朝着错误的方向更新。随机梯度下降算法最终通常能够使θ收敛在其全局最优解附近。

值得一提的是，得益于随机梯度下降算法这种增量更新的方式，它也经常用于在线学习。

1.1.3 θ的解析解

事实上，我们也可以从矩阵运算的角度出发求得θ的解析解：$\theta = (X^TX)^{-1}X^T\vec{y}$。其中，$X$是所有样本构成的矩阵（design matrix），$\vec{y}$是目标变量构成的向量（target vector）。

1.2 最小二乘的概率解释

在线性回归中我们选择最小二乘方程作为代价函数，下面给出这样选择在概率意义上的解释。假设输入和目标变量存在以下关系

$$y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)}$$

其中，$\epsilon^{(i)}$表示模型假设与目标变量之间的误差。假设误差相互独立同分布，且$\epsilon \in N(0,\sigma)$（

σ为一个常数）。那么$\epsilon^{(i)}$的概率密度函数为

$$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{(\epsilon^{(i)})^2}{2\sigma^2})$$

对所有误差的似然估计$L(\theta)$取对数可得

$$\begin{aligned} l(\theta) &= \log L(\theta) \\ &= m\log\frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2}\cdot\frac{1}{2}\sum_{i=1}^{m}(y^{(i)} - \theta^T x^{(i)})^2 \end{aligned}$$

可以看到最大化$l(\theta)$等价于最小化最小二乘方程，因此从误差的角度出发，最小二乘方程是作为代价函数是合适（符合最大似然原理，即最符合表述当前的误差）。

1.3 局部加权回归

局部加权回归是一个非参数学习算法，这意味着模型的参数会随着样本的增多而增多。在线性回归中，代价函数中每一项的权重都相等且取1，然而在局部加权回归中，会对误差项取不同权重。局部加权回归的基本假设为

1. Fit θ to minimize $\sum_i \omega^{(i)}(y^{(i)} - \theta^T x^{(i)})^2$
2. Output $\theta^T x$

$\omega^{(i)}$表示一个非零的权值，一般有

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

其中，x称为查询点，τ称为波长（bandwidth）。可以看到，靠近查询点的误差项权重接近于1，远离查询点的误差项权重接近于0，这便是局部加权的意义所在。在局部加权回归中，每次都需根据查询点重新建立新的模型，这点类似于K近邻算法。

2 逻辑回归

回归模型一般无法直接用于分类，因为模型的输出是一个连续值。为了将回归模型用于分类，我们需要在线性模型中增加一个映射层。在二分类问题中，令目标变量$y \in \{0, 1\}$（0表示负例，1表示正例），那么模型假设可以表示为

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

其中，

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

称为sigmoid函数，此函数将$\theta^T x$映射到区间(0,1)，可以描述样本取正例的概率大小。需要注意的是，由于映射层$g(z)$的存在，这里的模型假设并不是关于$\theta^T x$的线性函数，也就是说这是

一个非线性模型。

2.1 θ的解法

这里我们利用最大似然原理来解θ。样本取正例或负例的概率为

$$p(y \mid x;\theta) = (h_\theta(x))^y(1-h_\theta(x))^{1-y}$$

对所有样本有似然估计

$$\begin{aligned} L(\theta) &= p(\vec{y} \mid X;\theta) \\ &= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)};\theta) \\ &= \prod_{i=1}^{m} (h_\theta(x^{(i)}))^{y^{(i)}}(1-h_\theta(x^{(i)}))^{1-y^{(i)}} \end{aligned}$$

其中，X表示design matrix，$\vec{y}$表示target vector。取对数后得

$$\begin{aligned} l(\theta) &= \log L(\theta) \\ &= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log(1-h(x^{(i)})) \end{aligned}$$

类似于线性模型，我们利用梯度上升算法来逼近似然估计的极大值。（事实上，$l(\theta)$相当于模型对所有样本的熵的相反数，从熵的角度看也应该使$l(\theta)$取最大值，因为这时混乱程度最低，即分类效果最好。）对一个样本有

$$\frac{\partial}{\partial\theta_j} l(\theta) = (y - h_\theta(x))x_j$$

那么参数的更新法则为

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

可以看到，这个法则和线性回归中的法则是一致的，that's amazing！

2.2 牛顿法解最大似然估计

考虑另一种方法解最大似然估计。给定函数$f:\mathbb{R} \mapsto \mathbb{R}$，而我们的目标是找到实数$\theta$使$f(\theta)=0$，那么牛顿法可以表示为

$$\theta := \theta - \frac{f(\theta)}{f'(\theta)}$$

将牛顿法移植到解最大似然估计时，我们可以得到新的更新法则

$$\theta := \theta - \frac{l'(\theta)}{l''(\theta)}$$

在实际的应用中θ是一个向量，所以继续修改更新法则

$$\theta := \theta - H^{-1}\nabla_\theta l(\theta)$$

其中H表示n-by-n（如果加上截距项，就是n+1-by-n+1）的Hessisan矩阵，其中

$$H_{ij} = \frac{\partial^2 l(\theta)}{\partial\theta_i \partial\theta_j}$$

牛顿法收敛很快，但计算Hessian矩阵的逆比较耗时。此外，当初始点x_0靠近极值点时算法收敛速度最快，但是当初始点x_0距离极值点较远时，每一步迭代甚至不一定是朝向极值点的方向，这和似然估计的图形有关。

3 一般线性模型

在之前的讨论中，对回归问题，我们假设$y \mid x; \theta \sim \mathcal{N}(\mu, \sigma^2)$，利用最小二乘方程建模；对二分类问题，我们假设$y \mid x; \theta \sim \text{Bernoulli}(\phi)$，利用sigmoid函数建模，事实上它们都属于一般线性模型。

3.1 指数分布族

定义指数分布族的形式为

$$p(y;\eta) = b(y)\exp(\eta^T T(y) - a(\eta))$$

其中，η称为自然参数，$T(y)$称为充分统计量。对于Bernoulli分布有

$$\begin{aligned} p(y;\phi) &= \phi^y(1-\phi)^{1-y} \\ &= \exp(y\log\phi + (1-y)\log(1-\phi)) \\ &= exp((log(\frac{\phi}{1-\phi}))y + \log(1-\phi)) \end{aligned}$$

符合指数分布族的形式，且对应有$\eta = \log(\phi/(1-\phi))$，即$\phi = 1/(1+e^{-\eta})$（式2）。对于高斯分布（假设$\sigma^2 = 1$）有

$$\begin{aligned} p(y;\mu) &= \frac{1}{\sqrt{2\pi}}\exp(-\frac{1}{2}(y-\mu)^2) \\ &= \frac{1}{\sqrt{2\pi}}\exp(-\frac{1}{2}y^2)\cdot\exp(\mu y - \frac{1}{2}\mu^2) \end{aligned}$$

同样符合指数分布族的形式，且对应有$\eta = \mu$（式3）。此外泊松分布、beta分布等也属于指数分布族。

3.2 构造一般线性模型

为了构造一般线性模型解决分类和回归问题，我们需要作出以下三个假设

1. $y \mid x; \theta \sim \text{ExpFamily}(\eta)$

2. 给定x，我们的目标是得到$T(y)$在给定x下的期望。在一般情况下，$T(y) = y$，也就是说，我们的模型假设$h(x)$需满足$h(x) = E[y \mid x]$
3. $\eta = \theta^T x$（当η是向量时有$\eta_i = \theta_i^T x$），可以理解为一种设计策略

3.2.1 逻辑回归

如果样本集服从参数为ϕ伯努利分布，那么根据（式2）可构造一般线性模型如下

$$h_\theta(x) = E[y \mid x; \theta] = \phi = 1/(1 + e^{-\eta}) = 1/(1 + e^{-\theta^T x})$$

得到了逻辑回归的模型假设。

3.2.2 线性回归

如果样本集服从期望为μ，方差为σ^2 的高斯分布，那么（式3）可构造一般线性模型如下

$$h_\theta(x) = e[y \mid x; \theta] = \mu = \eta = \theta^T x$$

得到了线性回归的模型假设。

3.2.3 Softmax回归

鉴于推导过程比较复杂，这里就简单介绍一下Softmax回归。Softmax回归是逻辑回归的一个扩展版，用于解决多分类问题。假设目标变量$y \in \{1, 2, \ldots, k\}$，且$p(y = i \mid x; \theta) = \phi_i$，构造一般线性模型可得模型假设$h_\theta(x)$（一个向量），其中

$$h_\theta(x)_i = \frac{\exp(\theta_i^T x)}{\sum_{j=1}^k \exp(\theta_j^T x)}$$

表示的是目标变量取i的概率大小。由此可构造似然估计

$$\begin{aligned} l(\theta) &= \sum_{i=1}^m \log p(y^{(i)} \mid x^{(i)}; \theta) \\ &= \sum_{i=1}^m \log \prod_{l=1}^k \left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{i=1}^k e^{\theta_j^T x^{(i)}}}\right)^{1\{y^{(i)} = l\}} \end{aligned}$$

接下来就是找到使似然估计取最大值的θ（一个矩阵），可以用之前提到的梯度下降算法或牛顿法。