logistic回归与softmax回归

毛润泽

April 23, 2017

1 Logistic Regression

对数几率回归最大化对数似然:

$$\boldsymbol{\beta} = \underset{\boldsymbol{\beta}}{argmax} \sum_{i=1}^{m} \ln p(\hat{y}_i = y_i | x_i; \boldsymbol{\beta})$$

使用假设函数$h(x_i)$:

$$h(x_i) = \frac{e^{\boldsymbol{\beta}^T x_i}}{1 + e^{\boldsymbol{\beta}^T x_i}}$$

则有:

$$p(\hat{y}_i = 1 | x_i; \boldsymbol{\beta}) = h(x_i) \quad , \quad p(\hat{y}_i = 0 | x_i; \boldsymbol{\beta}) = 1 - h(x_i)$$

而:

$$p(\hat{y}_i = y_i | x_i; \boldsymbol{\beta}) = y_i p(\hat{y}_i = 1 | x_i; \boldsymbol{\beta}) + (1 - y_i) p(\hat{y}_i = 0 | x_i; \boldsymbol{\beta})$$

由此可推导目标函数:

$$\begin{aligned} J(\boldsymbol{\beta}) &= -\sum_{i=1}^{m} \ln \left(y_i \frac{e^{\boldsymbol{\beta}^T x_i}}{1 + e^{\boldsymbol{\beta}^T x_i}} + (1 - y_i) \frac{1}{1 + e^{\boldsymbol{\beta}^T x_i}} \right) \\ &= -\sum_{i=1}^{m} \left(\ln(y_i e^{\boldsymbol{\beta}^T x_i} + 1 - y_i) - \ln(1 + e^{\boldsymbol{\beta}^T x_i}) \right) \end{aligned} \tag{1}$$

其中:

$$\ln(y_i e^{\boldsymbol{\beta}^T x_i} + 1 - y_i) = \begin{cases} 0 & (y_i = 0) \\ \boldsymbol{\beta}^T x_i & (y_i = 1) \end{cases}$$

因此，可以得到:

$$\ln(y_i e^{\boldsymbol{\beta}^T x_i} + 1 - y_i) = y_i \boldsymbol{\beta}^T x_i$$

因此可以继续推导(1)式:

$$J(\boldsymbol{\beta}) = \sum_{i=1}^{m} \Big(- y_i \boldsymbol{\beta}^T x_i + \ln(1 + e^{\boldsymbol{\beta}^T x_i}) \Big) \tag{2}$$

使用梯度下降法，先求梯度:

$$\begin{aligned}
\nabla J &= \frac{\partial J(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \\
&= \sum_{i=1}^{m} \Big(- y_i x_i + \frac{x_i e^{\boldsymbol{\beta}^T x_i}}{1 + e^{\boldsymbol{\beta}^T x_i}} \Big) \\
&= \sum_{i=1}^{m} \Big(p(\hat{y}_i = 1 | x_i; \boldsymbol{\beta}) - y_i \Big) x_i
\end{aligned}$$

于是得到:

$$\begin{aligned}
\boldsymbol{\beta}^{(t+1)} &= \boldsymbol{\beta}^{(t)} - \eta \nabla J \\
&= \boldsymbol{\beta}^{(t)} - \eta \sum_{i=1}^{m} \Big(p(\hat{y}_i = 1 | x_i; \boldsymbol{\beta}^{(t)}) - y_i \Big) x_i
\end{aligned}$$

2 Softmax Regression

softmax回归可以看成logistic回归在多分类上的应用。
(2)式可改写为:

$$\begin{aligned}
J(\boldsymbol{\beta}) &= - \sum_{i=1}^{m} \Big(y_i \ln e^{\boldsymbol{\beta}^T x_i} + \ln \frac{1}{1 + e^{\boldsymbol{\beta}^T x_i}} \Big) \\
&= - \sum_{i=1}^{m} \Big(y_i \ln \frac{e^{\boldsymbol{\beta}^T x_i}}{1 + e^{\boldsymbol{\beta}^T x_i}} - y_i \ln \frac{1}{1 + e^{\boldsymbol{\beta}^T x_i}} + \ln \frac{1}{1 + e^{\boldsymbol{\beta}^T x_i}} \Big) \\
&= - \sum_{i=1}^{m} \Big(y_i \ln p(\hat{y}_i = 1 | x_i; \boldsymbol{\beta}) + (1 - y_i) \ln p(\hat{y}_i = 0 | x_i; \boldsymbol{\beta}) \Big) \\
&= - \sum_{i=1}^{m} \sum_{j=0}^{1} \mathbb{I}(y_i = j) \ln p(\hat{y}_i = j | x_i; \boldsymbol{\beta}) \qquad (3)
\end{aligned}$$

其中，$\mathbb{I}(z)$为指示函数，当z为true时取值为1，z为false时取值为0。

对于多分类任务，假设y可取k个不同的值，即$y_i \in \{1, 2, \ldots, k\}$。对于每个类标签$j$，都要训练出对应的$\boldsymbol{\beta}_j$。样本$x_i$的标签为$j$的可能性为：

$$p(\hat{y}_i = j | x_i; \boldsymbol{\beta}_j) = \frac{e^{\boldsymbol{\beta}_j^T x_i}}{\sum_{l=1}^k e^{\boldsymbol{\beta}_l^T x_i}} \tag{4}$$

令：

$$\mathbf{B} = \begin{bmatrix} \boldsymbol{\beta}_1^T \\ \boldsymbol{\beta}_2^T \\ \vdots \\ \boldsymbol{\beta}_k^T \end{bmatrix}$$

则假设函数为：

$$h(x_i) = \begin{bmatrix} p(\hat{y}_i = 1 | x_i; \mathbf{B}) \\ p(\hat{y}_i = 2 | x_i; \mathbf{B}) \\ \vdots \\ p(\hat{y}_i = k | x_i; \mathbf{B}) \end{bmatrix} = \frac{e^{\mathbf{B} x_i}}{\sum_{j=1}^k e^{\boldsymbol{\beta}_j^T x_i}}$$

可将(3)式扩展为多分类的情形，得到softmax的目标函数：

$$J(\mathbf{B}) = -\sum_{i=1}^m \sum_{j=1}^k \mathbb{I}(y_i = j) \ln p(\hat{y}_i = j | x_i; \mathbf{B})$$

仍然使用梯度下降法求解，目标函数对于分量$\boldsymbol{\beta}_s$求梯度可得：

$$\nabla_{\boldsymbol{\beta}_s} J = -\sum_{i=1}^m \sum_{j=1}^k \mathbb{I}(y_i = j) \frac{\partial}{\partial \boldsymbol{\beta}_s} (\boldsymbol{\beta}_j^T x_i - \ln \sum_{l=1}^k e^{\boldsymbol{\beta}_l^T x_i})$$

其中：

$$\frac{\partial}{\partial \boldsymbol{\beta}_s} (\boldsymbol{\beta}_j^T x_i) = \begin{cases} x_i & (s = j) \\ 0 & (s \neq j) \end{cases}$$

$$\frac{\partial}{\partial \boldsymbol{\beta}_s} (\ln \sum_{l=1}^k e^{\boldsymbol{\beta}_l^T x_i}) = \frac{x_i e^{\boldsymbol{\beta}_s^T x_i}}{\sum_{l=1}^k e^{\boldsymbol{\beta}_l^T x_i}}$$

于是得到：

$$\begin{aligned}\nabla_{\boldsymbol{\beta}_s} J &= \sum_{i=1}^{m}\sum_{j=1}^{k}\mathbb{I}(y_i=j)\Big(p(\hat{y}_i=s|x_i;\boldsymbol{\beta}_s)-\mathbb{I}(s=j)\Big)x_i \\ &= \sum_{i=1}^{m}\Big(p(\hat{y}_i=s|x_i;\boldsymbol{\beta}_s)-\mathbb{I}(y_i=s)\Big)x_i\end{aligned}$$

所以，可以得到B的分量$\boldsymbol{\beta}_j$的更新公式：

$$\begin{aligned}\boldsymbol{\beta}_j^{(t+1)} &= \boldsymbol{\beta}_j^{(t)} - \eta\nabla_{\boldsymbol{\beta}_j} J \\ &= \boldsymbol{\beta}_j^{(t)} - \eta\sum_{i=1}^{m}\Big(p(\hat{y}_i=j|x_i;\boldsymbol{\beta}_j^{(t)})-\mathbb{I}(y_i=j)\Big)x_i\end{aligned}$$

3 Softmax的改进

(4)式其实是有问题的，从每个$\boldsymbol{\beta}_j$减去一个向量ψ，所得结果不变：

$$\begin{aligned}\frac{e^{(\boldsymbol{\beta}_j-\psi)^T x_i}}{\sum_{l=1}^{k} e^{(\boldsymbol{\beta}_l-\psi)^T x_i}} &= \frac{e^{\boldsymbol{\beta}_j^T x_i}e^{-\psi^T x_i}}{\sum_{l=1}^{k} e^{\boldsymbol{\beta}_l^T x_i}e^{-\psi^T x_i}} \\ &= \frac{e^{\boldsymbol{\beta}_j^T x_i}}{\sum_{l=1}^{k} e^{\boldsymbol{\beta}_l^T x_i}} \\ &= p(\hat{y}_i=j|x_i;\boldsymbol{\beta}_j)\end{aligned}$$

也就是说，模型参数过多，导致有无数多种解。

这很容易理解，因为$\sum_{j=1}^{k} p(\hat{y}_i=j|x_i;\boldsymbol{\beta}_j)=1$，所以$\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\ldots,\boldsymbol{\beta}_k$并不是无关的。

解决方法如下。

⑴ 可以固定地令$\boldsymbol{\beta}_1=\vec{0}$，相当于从所有$\boldsymbol{\beta}_j$中减去$\boldsymbol{\beta}_1$。当$k=2$时，这就相当于logistic回归。

⑵ 实际应用中，往往不会用上面的方法，而是使用L_2正则化（权重衰减）。这个正则化项会惩罚过大的参数值。

$$J(\mathbf{B}) = -\sum_{i=1}^{m}\sum_{j=1}^{k}\mathbb{I}(y_i=j)\ln p(\hat{y}_i=j|x_i;\mathbf{B}) + \frac{\lambda}{2}\sum_{j=1}^{k}\|\boldsymbol{\beta}_j\|_2$$

目标函数对B的分量$\boldsymbol{\beta}_j$的梯度为：

$$\nabla_{\boldsymbol{\beta}_j} J = \sum_{i=1}^{m} \Big(p(\hat{y}_i = j | x_i; \boldsymbol{\beta}_j) - \mathbb{I}(y_i = j) \Big) x_i + \lambda \boldsymbol{\beta}_j$$

迭代更新公式变为：

$$\boldsymbol{\beta}_j^{(t+1)} = (1-\lambda)\boldsymbol{\beta}_j^{(t)} - \eta \sum_{i=1}^{m} \Big(p(\hat{y}_i = j | x_i; \boldsymbol{\beta}_j^{(t)}) - \mathbb{I}(y_i = j) \Big) x_i$$

也可以这样理解，在空间中有多个点（或者说向量）满足最优解，而L_2正则化其实是选择了最接近原点的那个解。

4 两者区别——引用自网络

如果你在开发一个音乐分类的应用，需要对k种类型的音乐进行识别，那么是选择使用 softmax 分类器呢，还是使用 logistic 回归算法建立 k 个独立的二元分类器呢？

这一选择取决于你的类别之间是否互斥，例如，如果你有四个类别的音乐，分别为：古典音乐、乡村音乐、摇滚乐和爵士乐，那么你可以假设每个训练样本只会被打上一个标签（即：一首歌只能属于这四种音乐类型的其中一种），此时你应该使用类别数 k = 4 的softmax回归。（如果在你的数据集中，有的歌曲不属于以上四类的其中任何一类，那么你可以添加一个“其他类”，并将类别数 k 设为5。）

如果你的四个类别如下：人声音乐、舞曲、影视原声、流行歌曲，那么这些类别之间并不是互斥的。例如：一首歌曲可以来源于影视原声，同时也包含人声 。这种情况下，使用4个二分类的 logistic 回归分类器更为合适。这样，对于每个新的音乐作品 ，我们的算法可以分别判断它是否属于各个类别。

现在我们来看一个计算视觉领域的例子，你的任务是将图像分到三个不同类别中。(i) 假设这三个类别分别是：室内场景、户外城区场景、户外荒野场景。你会使用sofmax回归还是 3个logistic 回归分类器呢？ (ii) 现在假设这三个类别分别是室内场景、黑白图片、包含人物的图片，你又会选择 softmax 回归还是多个 logistic 回归分类器呢？ 在第一个例子中，三个类别是互斥的，因此更适于选择softmax回归分类器 。而在第二个例子中，建立三个独立的 logistic回归分类器更加合适。