Logistic回归
关于逻辑回归的读书笔记

梁军
郑州大学
2014 年 5 月 5 日

摘要

Logistic回归——我们经常用一条直线对一些点进行拟合（该线成为最佳拟合直线），这个拟合过程就称作回归。[2]利用Logistic回归进行分类的主要思想是：根据现有数据对分类边界线建立回归公式，以此进行分类。Logistic回归的一般过程为：

1. 收集数据：采用任意方法收集数据。
2. 准备数据：由于需要进行距离计算，因此要求数据类型采用数值型。另外，结构话数据格式则最佳。
3. 分析数据：采用任意方法对数据进行训练。
4. 训练算法：大部分时间将用于训练，训练的目的是为了找到最佳的分类回归函数。
5. 测试算法：一旦训练步骤完成，分类将会很快。

Keywords: 机器学习, 逻辑回归, 最优化算法

1 介绍

好了，以上摘自机器学习实战这本书。写的有点抽象……看了这些，maybe一些童鞋对逻辑回归有那么一丢丢的理解，但一定还是不知其所以然，那么下面我们从回归、线性回归和逻辑方程来逐步深入逻辑回归:)

1.1 神马是回归

回归 按照我的理解就是有一堆数据，而且我们又事先知道这些数据之间的关系，举个例子说吧：比如我们知道弹簧的弹力与弹簧的形变量之间存在线性关系：F=kx，但是却不知道系数k的值，那么我们可以通过大量的弹力和与之对应的形变量来计算出弹性系数，这个过程就叫做回归。也就是

说，在进行回归的时候我们首先要知道变量与因变量之间的关系，而仅仅是去计算参数集而已。可是现实中，我们往往对于数据之间的关系不得而知，这就需要去观察数据然后预测数据之间可能的关系，然后再去回归参数集。

当然，根据数据间的关系，我们可以将回归分为线性回归（一元一次方程、二元一次方程……）和非线性回归（N元M次方程、Log 方程、指数方程……）

1.2 神马是线性回归

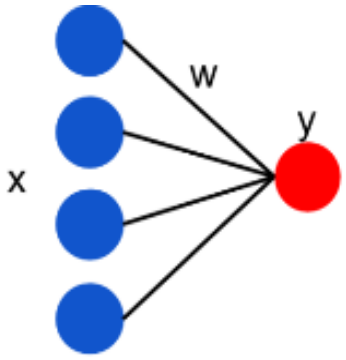

图 1: 线性回归

线性回归 我们举个最简单的例子，还是上面弹簧弹力与形变量之间的线性关系问题，它们之间是一元一次方程的关系，根据我们初中OR小学的知识我们可以知道：解这个方程仅仅需要一个F 值和一个与之对应的x值就OK了，可是现实中由于这些数据都是测量出来的，不可避免的会出现误差甚至的错误，那么我们就需要大量数据来预测一个较为准确的弹性系数。通常采用的方式是，定义一个代价函数，通过最小化代价函数来估计参数集；或者定义代价函数为最大似然函数，通过最大化似然函数来估计参数集。

1.3 神马是Logistic方程

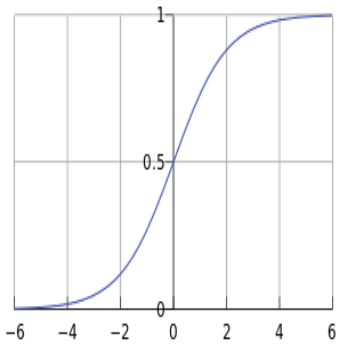

图 2: Logistic方程

Logistic方程 ——直觉上，一个线性模型的输出值y 越大，这个事件$P(Y = 1|x)$ 发生的概率就越大。另一方面，我们可以用事件的几率（odds）来表示事件发生与不发生的比值，假设发生的概率是p ，那么发生的几率（odds）是$p/(1-p)$，odds 的值域是0 到正无穷，几率越大，发生的可能性越大。将我们的直觉与几率联系起来的就是下面这个（log odds）或者是Logistic函数(有点牵强- -!)：

$$Log\frac{p}{1-p} = w^T \cdot x$$

其中，w和x就为行向量。进而可以求出p关于$w \cdot x$的表示：

$$P(Y = 1|x) = \frac{e^{w^T \cdot x}}{1 + e^{w^T \cdot x}}$$

看到这里，恭喜你！终于见到传说中的Logistic函数了，其实就是人为的将线性回归的结果与对数概率结合在了一起！以$w \cdot x$为自变量的函数图2所示。

我们可以看得出，当横坐标拉伸时，该函数图像就像是一个阶跃函数一样，因此可以用它来做分类!

2 参数估计方法

上面也简单提到了一下，对于回归我大致了解到的有两种参数估计方法：

1. 最大似然估计法
2. 平方损失函数估计法

2.1 最大似然法

逻辑回归输出的是分到每一类的概率，参数估计的方法自然可以用最大似然估计(MLE) 咯。对于训练样本来说，假设每个样本是独立的，输出（标签）为$y = 0, 1$，样本的似然函数就是将所有训练样本label 对应的输出节点上的概率相乘, 令$p = P(Y = 1|x)$,如果$y = 1$, 概率就是p，如果$y = 0$, 概率就是$1 - p$，（好吧，我是个罗嗦的家伙), 将这两种情况合二为一，得到似然函数：

$$L = \prod_{i=1}^{N} [P(Y = 1|x_i)]^{y_i} [P(Y = 0|x_i)]^{1-y_i}$$

有连乘，这个好办，取对数就OK：

$$\begin{aligned} L(w) &= \sum_{i=1}^{N} y^i log(P(Y = 1|x_i)) + (1 - y^i) log(P(Y = 0|x_i)) \\ &= \sum_{i=1}^{N} y^i log(P(Y = 1|x_i)) + log(P(Y = 0|x_i)) - y^i log(P(Y = 0|x_i)) \\ &= \sum_{i=1}^{N} y^i log \frac{P(Y = 1|x_i)}{P(Y = 0|x_i)} + log(P(Y = 0|x_i)) \end{aligned}$$

根据上面的对数概率公式，我们可以得到：

$$L(w) = \sum_{i=1}^{N} y^i (\vec{w}^T \cdot \vec{x}) - log(1 + e^{w^T \cdot x})$$

对w求导可以得到：

$$\frac{\partial L(w)}{\partial w} = \sum_{i=1}^{N} y_i x - \sum_{i=1}^{N} \frac{e^{w^T x}}{1 + e^{w^T x}} x = \sum_{i=1}^{N} (y_i - P(Y = 1|x)) x$$

2.2 平方损失函数法

平方损失函数 通常在定义了一个模型($f(x; w_0, w_1) = w_0 + w_1 x$)之后，衡量一个特定模型与数据点接近程度的普遍方法是真实值与模型预测值之间的平方差[1]。平方差可以定义为：

$$(t_n - f(x_n; w_0, w_1))^2$$

这个表达式就称之为**平方损失函数**。

如果$f(x_n; w_0, w_1)$仅是一个线性模型，那么该方法就是用于线性回归的参数拟合；但是在$f(\cdot)$外层加上激活函数Logistic函数，那么该方法就是用于逻辑回归的参数拟合。由此，我们可以看出，从线性模型到逻辑模型仅

仅是在线性函数外层嵌套一个Logistic函数而已，那么我们不禁要问什么时候使用线性模型，什么时候使用逻辑模型呢？根据我的理解，这个需要依据已有数据的结果集来判断：

1. 当结果集中属于二分类问题时，一般选用逻辑模型。
2. 当结果集中为连续型预测值时，一般选用线性模型。

对于整个数据集，考虑其平均损失，即：

$$\pounds_n(t_n, f(x_n; w_0, w_1)) = (t_n - f(x_n; w_0, w_1))^2$$

$$\pounds = \frac{1}{N}\sum_{n=1}^{N} \pounds_n(t_n, f(x_n; w_0, w_1))$$

£就是我们要定义的损失函数，为了得到更符合实际的参数集，我们当然希望损失函数越小越好。因此，我们需要根据数据集调整w_0和w_1来产生一个模型，词模型得到平均损失的最低值。使用数学表达式可以表示为：

$$\arg\min_{w_0,w_1} \frac{1}{N}\sum_{n=1}^{N} \pounds_n(t_n, f(x_n; w_0, w_1))$$

argmin项是数学上“找到最小化参数”的缩写，即“argument of the minimum”，这个曾经也是困惑了笔者好久，o(╯□╰)o

$$\begin{aligned}
\pounds &= \frac{1}{N}\sum_{n=1}^{N} \pounds_n(t_n, f(x_n; w_0, w_1)) \\
&= \frac{1}{N}\sum_{n=1}^{N} (t_n - f(x_n; w_0, w_1))^2 \\
&= \frac{1}{N}\sum_{n=1}^{N} (t_n - (w_0 + w_1 x_n))^2 \\
&= \frac{1}{N}\sum_{n=1}^{N} (w_1^2 x_n^1 + 2w_1 x_n(w_0 - t_n) + w_0^2 - 2w_0 t_n + t_n^2)
\end{aligned}$$

然后对于损失函数£分别关于w_0和w_1求导，令得到的偏导数等于0，求出w_0和w_1的表达式：

$$\widehat{w_0} = \bar{t} - w_1\bar{x}$$

$$\widehat{w_1} = \frac{\bar{x}t - \bar{x}\bar{t}}{\bar{x^2} - (\bar{x})^2}$$

对于该例中仅有两个参数w_0和w_1，该方法还可以使用，但如果有成百上千个参数呢？如此做法，一定会令你痛不欲生:(

事实上，我们一般采用矩阵的形式来表示，首先我们做如下约定：使用黑体表示矩阵或者向量，矩阵用大写字母表示，向量用小写字母表示，并且所有向量均为行向量，列向量使用行向量的转置表示。那么，我们定义$\boldsymbol{w}$表示参数向量，$\boldsymbol{x_i}$表示第i个数据集的特征向量，$\boldsymbol{X}$表示整个数据集的特征向量，$\boldsymbol{t}$表示结果向量集。那么，代价函数可以表示为：

$$\begin{aligned}\pounds &= \frac{1}{N}(\boldsymbol{t}-\boldsymbol{X}\boldsymbol{w}^T)^T(\boldsymbol{t}-\boldsymbol{X}\boldsymbol{w}^{\boldsymbol{T}}) \\ &= \frac{1}{N}(\boldsymbol{t}^T\boldsymbol{t}-\boldsymbol{t}^T\boldsymbol{X}\boldsymbol{w}^T-\boldsymbol{w}\boldsymbol{X}^T\boldsymbol{t}+\boldsymbol{w}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}^T) \\ &= \frac{1}{N}(\boldsymbol{t}^T\boldsymbol{t}-2\boldsymbol{w}\boldsymbol{X}^T\boldsymbol{t}+\boldsymbol{w}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}^T)\end{aligned}$$

根据向量微分的求导规则：

$f(w)$	$\frac{\partial f}{\partial w}$
$\boldsymbol{w}^T\boldsymbol{x}$	$\boldsymbol{x}$
$\boldsymbol{x}^T\boldsymbol{w}$	$\boldsymbol{x}$
$\boldsymbol{w}^T\boldsymbol{w}$	$2\boldsymbol{w}$
$\boldsymbol{x}^T\boldsymbol{C}\boldsymbol{w}$	$2\boldsymbol{C}\boldsymbol{x}$

得到：

$$\frac{\partial \pounds}{\partial w} = \frac{2}{N}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}-\frac{2}{N}\boldsymbol{X}^T\boldsymbol{t}$$

3 最优化算法

我们可以采用梯度下降算法。(未完待续……)

参考文献

[1] Simon Rogers and Mark Girolami. *A first course in machine learning*. CRC Press, 2011.

[2] Peter Harrington. *Machine Learning in Action*. Manning Publications Co., 2012.