逻辑回归

吴思思 15320171151911 经济系西方经济学

2019 年 4 月 13 日

一、绪论

1.1 与线性回归的比较

1.2 逻辑回归的类型

二、逻辑回归问题的常规步骤

2.1 构造预测函数

2.2 决策边界

2.3 构造损失函数

2.4 最小化损失函数-梯度下降算法

一、绪论

分类和回归是机器学习可以解决两大主要问题，从预测值的类型上看，连续变量预测的定量输出称为回归；离散变量预测的定性输出称为分类。逻辑回归是一种分类算法，用于将观测值分配给一组离散的类。与输出连续数值的线性回归不同，逻辑回归使用逻辑 sigmoid 函数转换其输出并返回至概率值，然后可以将概率值映射到两个或更多个离散类。

1.1 与线性回归的比较

给出学习时间和考试成绩的数据，线性回归和逻辑回归可以预测不同的东西：

线性回归可以帮助我们以 0-100 的数值范围预测学生的测试分数，线性回归预测是连续的（范围内的数字）。

逻辑回归可以帮助预测学生是否通过考试，逻辑回归预测是离散的（仅允许特定值或类别）。我们还可以查看模型分类背后的概率分数。

方法	自变量（特征）	因变量（结果）	关系
线性回归	连续或离散	连续实数	线性
逻辑回归	连续或离散	特定值或类别	非线性

1.2 逻辑回归的类型

* 二元。举例：邮件为垃圾邮件/非垃圾邮件？肿瘤是恶性/良性？考试通过或未通过？

* 多元。举例：出行方式选择公交/地铁/私家车？

* 序数。举例：对药物剂量的反应是无/轻微/适度/剧烈？

二、逻辑回归问题的常规步骤

2.1 构造预测函数

逻辑回归的假设输出介于 0 与 1 之间，即：

$0 \leq h_\theta(x) \leq 1$

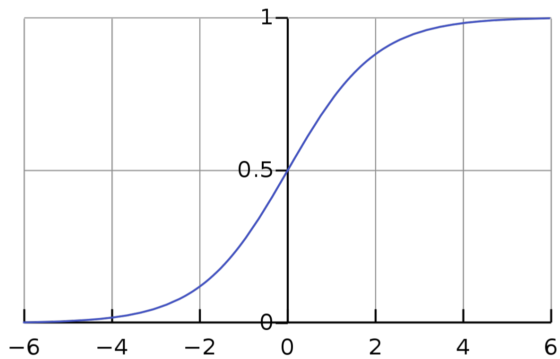
函数形式为：

$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$，其中 θ 为参数

这里 g 称为 Sigmoid 函数或者逻辑函数, 具体表达式为：

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

sigmoid 函数的形状如下图所示：



预测函数输出的直观解释为：$h_\theta(x)$ = 对于给定的输入 x，y=1 时估计的概率。

用数学表示如下：

$h_\theta(x) = P(y = 1 \mid x ; \theta)$

$P(y = 0 \mid x ; \theta) = 1 - h_\theta(x)$

2.2 决策边界

我们当前的预测函数输出的是 0 到 1 之间的概率分数。为了将其映射到离散类，我们选择一个阈值或临界点。假设给定的阈值是 0.5，

当 h_θ(x) $\geq$0.5 时，y = 1；

当 h_θ(x) <0.5 时，y = 0 。

例如，如果我们的阈值是 0.5 并且我们的预测函数输出为 0.7，我们将此观察分类为正。如果我们的预测是 0.2，我们会将观察分类为负。对于具有多个类的逻辑回归，我们可以选择具有最高预测概率的类。

2.3 构造损失函数

线性回归的损失函数选择的是 L2 函数，在逻辑回归中我们选择对数似然函数作为损失函数。

当 y = 1 时，Cost(h_θ(x),y) = - log(h_θ(x))

当 y = 0 时，Cost(h_θ(x),y) = - log(1-(h_θ(x))

$\mathrm{J}(\theta) = \frac{1}{m}\sum_{i=1}^{m}\mathrm{y}^{(i)}\mathrm{log}h_\theta(\mathrm{x}^{(i)})+(1\text{-}\mathrm{y}^{(i)})\mathrm{log}(1\text{-}h_\theta(\mathrm{x}^{(i)}))$

实际上 J(θ) 是通过极大似然估计推导得到的：我们将预测函数和似然概率分布写成更紧凑的形式：

$\mathrm{P}（\mathrm{y} \mid \mathrm{x}\,;\theta) = h_\theta(\mathrm{x})^{y}(1\text{-}h_\theta(\mathrm{x}))^{1-y}$

由最大似然估计原理，我们可以通过 m 个独立生成的训练样本值，来估计参数值：

$\mathrm{L}（\theta） = \mathrm{P}（\overrightarrow{y} \mid \mathrm{X}\,;\theta)$

$=\prod_{i=1}^{m}\mathrm{P}（\mathrm{y}^{(i)} \mid \mathrm{x}^{(i)}\,;\theta)$

$=\prod_{i=1}^{m}(h_\theta(\mathrm{x}^{(i)}))^{y^{(i)}}(1\text{-}h_\theta(\mathrm{x}^{(i)}))^{1-y^{(i)}}$

求 log:

$\ell(\theta)=\mathrm{logL}（\theta）$

$=\sum_{i=1}^{m}\mathrm{y}^{(i)}\mathrm{log}h_\theta(\mathrm{x}^{(i)})+(1\text{-}\mathrm{y}^{(i)})\mathrm{log}(1\text{-}h_\theta(\mathrm{x}^{(i)}))$

根据“最大似然估计”，求 $\ell(\theta)$ 取最大值时的 θ，定义损失函数 J(θ) 为：

$\mathrm{J}(\theta)=-\ \frac{1}{m}\ell(\theta)$

所以最后目标变成取 J(θ) 最小值时的 θ 为最佳参数。

2.4 最小化损失函数-梯度下降算法

梯度下降是迭代法的一种, 可以用于求解最小二乘问题 (线性和非线性都可以)。在求解机器学习算法的模型参数，即无约束优化问题时，梯度下降（Gradient Descent）是最常采用的方法之一，另一种常用的方法是最小二乘法。在求解损失函数的最小值时，可以通过梯度下降法来一步步的迭代求解，得到最小化的损失函数和模型参数值。梯度下降法的计算过程就是沿梯度下降的方向求解极小值。举一个非常简单的例子，如求函数 $f(x) = x^2$ 的最小值。利用梯度下降的方法解题步骤如下：

1. 求梯度，∇=2x

2. 向梯度相反的方向移动 x，如下：

$x \leftarrow x-\gamma \cdot \nabla$, 其中,$\gamma$ 为步长。如果步长足够小，则可以保证每一次迭代都在减小，但可能导致收敛太慢，如果步长太大，则不能保证每一次迭代都减少，也不能保证收敛。

3. 循环迭代步骤 2，直到 x 的值变化到使得 f(x) 在两次迭代之间的差值足够小，比如 0.00000001，也就是说，直到两次迭代计算出来的 f(x) 基本没有变化，则说明此时 f(x) 已经达到局部最小值了。

4. 此时, 输出 x，这个 x 就是使得函数 f(x) 最小时的 x 的取值。

现在，我们使用梯度下降算法求使损失函数最小化的参数值 θ。

θ 更新过程：

$\theta_j := \theta_j - \alpha \frac{\delta}{\delta_{\theta_j}} J(\theta)$

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

$= -\frac{1}{m}\sum_{i=1}^{m}(y^{(i)}\frac{1}{g(\theta^T x^{(i)})} - (1-y^{(i)})\frac{1}{1-g(\theta^T x^{(i)})})\frac{\delta}{\delta_{\theta_j}}\ g(\theta^T x^{(i)})$

$= -\frac{1}{m}\sum_{i=1}^{m}(y^{(i)}\frac{1}{g(\theta^T x^{(i)})} - (1-y^{(i)})\frac{1}{1-g(\theta^T x^{(i)})})g(\theta^T x^{(i)})(1-g(\theta^T x^{(i)}))\frac{\delta}{\delta_{\theta_j}}\theta^T x^{(i)}$

$= -\frac{1}{m}\sum_{i=1}^{m}(y^{(i)}(1-g(\theta^T x^{(i)})) - (1-y^{(i)})g(\theta^T x^{(i)}))x_j^{(i)}$

$= -\frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - g(\theta^T x^{(i)}))x_j^{(i)}$

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

θ 更新过程可以写成：

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

参考文献

[1]Machine Learning Cheatsheet, Logistic Regression

[2]http://cs229.stanford.edu/notes/cs229-notes1.pdf

[3]http://cs229.stanford.edu/extra-notes/loss-functions.pdf

[4]https://en.wikipedia.org/wiki/Sigmoid_function

[5]Coursera 公开课笔记: 斯坦福大学机器学习第六课“逻辑回归 (Logistic Regression)”

[6] 百度百科-梯度下降