TS6. 逻辑回归

H1

6.1 逻辑函数

H2

一些回归算法也可以用于分类（反之亦然）。 Logistic 回归（也称为 Logit 回归）通常用于估计一个实例属于某个特定类别的概率（例如，这电子邮件是垃圾邮件的概率是多少？）。 如果估计的概率大于 50%，那么模型预测这个实例属于当前类（称为正类，标记为“1”），反之预测它不属于当前类（即它属于负类，标记为“0”）。 这样便成为了一个二元分类器。

就像线性回归模型一样，Logistic 回归模型计算输入特征的加权和（加上偏差项），但它不像线性回归模型那样直接输出结果，而是把结果输入 `logistic()` 函数进行二次加工后进行输出。稍有区别的是，逻辑回归模型输出的是一个概率估计。

$$\begin{aligned} \hat{y} &= f(x) \\ \hat{p} &= f(x) \end{aligned} \tag{1}$$

$$\hat{y} = \begin{cases} 0, \hat{p} < 0.5 \\ 1, \hat{p} \geq 0.5 \end{cases} \tag{2}$$

$$\hat{y} = f(x) = \theta^{\mathrm{T}} \cdot x_b \tag{3}$$

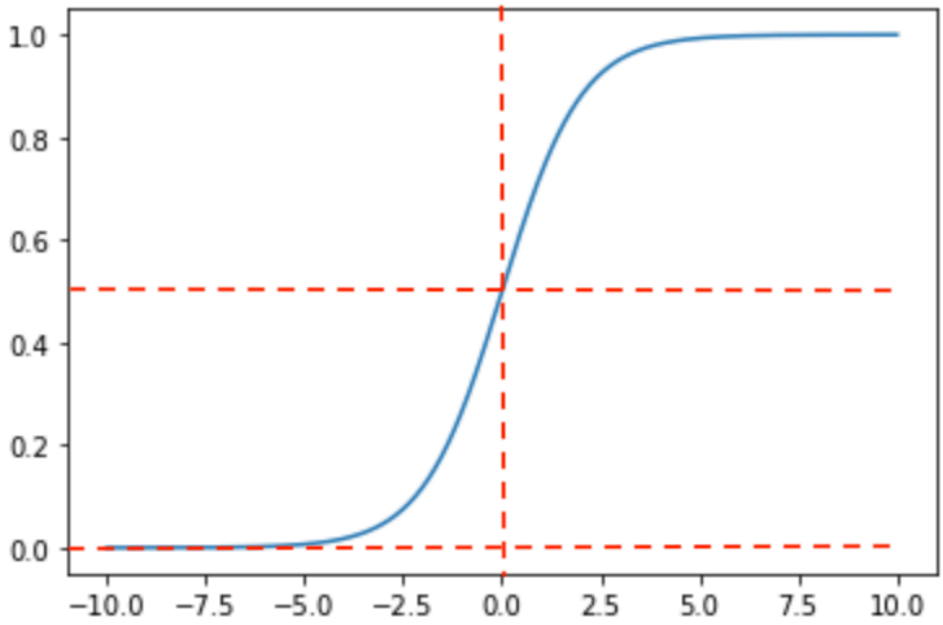
逻辑回归作为分类算法用，只能解决二分类问题，但通过改造可以解决多分类。

上面线性回归，可以获得一个值域为无限范畴的值。但输出概率最好能在$[0, 1]$的值域，于是我们考虑将线性的结果，能否通过一个σ函数转换为一个$[0, 1]$的值域。

$$\hat{p} = \sigma(\theta^{\mathrm{T}} \cdot x_b) \tag{4}$$

那这个函数是什么样子的呢？逻辑函数（Sigmoid）

$$\sigma(t) = \frac{1}{1 + e^{-t}} \tag{5}$$



此时，y的值域就是$[0,1]$的值域。一旦 Logistic 回归模型估计得到了 `x` 属于正类的概率$\hat{p}=\sigma(\theta^{\mathrm{T}}\cdot x_b)$，那它很容易得到预测结果$\hat{y}$。

$$\begin{aligned}\hat{p}=\sigma(\theta^{\mathrm{T}}\cdot x_b)&=\frac{1}{1+e^{-\theta^{\mathrm{T}}\cdot x_b}}\\ \hat{y}&=\begin{cases}0,\hat{p}<0.5\\1,\hat{p}\geq 0.5\end{cases}\end{aligned}\tag{6}$$

注意当 `t < 0` 时 `σ(t) < 0.5`，当 `t ≥ 0` 时 `σ(t) ≥ 0.5`，因此当$\theta^{\mathrm{T}}\cdot x_b$是正数的话，逻辑回归模型输出 1，如果它是负数的话，则输出 0。

6.2 损失函数

如何找到θ如何最大程度获得样本数据X对应分类输出y? 即如何在训练集上找到最优的拟合效果。

H2

单个样本损失函数我们可以写作：

$$cost=\begin{cases}\text{如果}y=1,p\text{越小，}cost\text{越大}\\\text{如果}y=0,p\text{越大，}cost\text{越大}\end{cases}\leq 0\tag{7}$$

$$cost=\begin{cases}-log(\hat{p})\ ,if\ y=0\\-log(1-\hat{p})\ ,if\ y=0\end{cases}\tag{8}$$

这个损失函数是合理的，因为当 `t` 接近 0 时， `-log(t)` 变得非常大，所以如果模型估计一个正例概率接近于 0，那么损失函数将会很大，同时如果模型估计一个负例的概率接近 1，那么损失函数同样会很大。 另一方面，当 `t` 接近于 1 时， `-log(t)` 接近 0，所以如果模型估计一个正例概率接近于 0，那么损失函数接近于 0，同时如果模型估计一个负例的概率接近 0，那么损失函数同样会接近于 0， 这正是我们想的。上述公式可以通过一个公式来表达：

$$cost = -y\log(\hat{\mathrm{p}}) - (1-y)\log(1-\hat{\mathrm{p}}) \tag{9}$$

当有m的样本时：

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(\hat{\mathrm{p}}^{(\mathrm{i})}) + (1-y^{(i)})\log(1-\hat{\mathrm{p}}^{(\mathrm{i})})] \tag{10}$$

但是这个损失函数对于求解最小化损失函数的$J(\theta)$是没有公式解的（没有等价的正规方程）。 但好消息是，这个损失函数是凸的，所以梯度下降（或任何其他优化算法）一定能够找到全局最小值（如果学习速率不是太大，并且你等待足够长的时间）。

$$\hat{p}^{(i)} = \sigma(X_b^{(i)} \cdot \theta) = \frac{1}{1+e^{-X_b^{(i)}\cdot\theta}} \tag{11}$$

可以转化为：

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(\sigma(\mathrm{X}_\mathrm{b}^{(\mathrm{i})} \cdot \theta)) + (1-y^{(i)})\log(1-\sigma(\mathrm{X}_\mathrm{b}^{(\mathrm{i})} \cdot \theta))] \tag{12}$$

$$\begin{aligned} \frac{J(\theta)}{\theta_j} &= \frac{1}{m}\sum_{i=1}^{m}(\sigma(X_b^{(i)}\theta) - y^{(i)}) \cdot X_j^{(i)} \\ &= \frac{1}{m}\sum_{i=1}^{m}(\hat{y} - y^{(i)}) \cdot X_j^{(i)} \end{aligned} \tag{13}$$

这个公式看起来非常像之前线性函数公式：首先计算每个样本的预测误差，然后误差项乘以第 `j` 项特征值，最后求出所有训练样本的平均值。 一旦你有了包含所有的偏导数的梯度向量，你便可以在梯度向量上使用批量梯度下降算法。 也就是说：你已经知道如何训练 Logistic 回归模型。 对于随机梯度下降，你当然只需要每一次使用一个实例，对于小批量梯度下降，你将每一次使用一个小型实例集。

最终，向量化过程可以表示为：

$$\nabla J(\theta) = \frac{1}{m} \cdot X_b^T \cdot (\sigma(X_b\theta) - y) \tag{14}$$

6.3 实现逻辑回归

H2