第四章 逻辑回归

逻辑回归（Logistic Regression）是一个分类算法，常用领域:

- 医学：根据病人一系列检测指标确定是恶性肿瘤或良性肿瘤？
- 经济：根据客户的一系列特征确定接受其贷款申请或拒绝其贷款申请？
- 网络：根据Email的特征判定其是垃圾邮件或正常邮件？
- 模式识别：手写字识别，人脸识别等。

分类包括二分类问题和多分类问题，对二分类问题，通常取目标变量值为0或1，0为一类，1为另一类，可以任意指定0或1代表哪一类，但按照经验，通常用1表示“阳性”，即要寻找的那一类，比如恶性肿瘤、垃圾邮件等。

1 逻辑回归原理

对于分类问题，直观上似乎也可以用线性回归的方式来解决，比如以肿瘤大小为特征预测恶性肿瘤的概率，有一系列样本如图4-1所示。

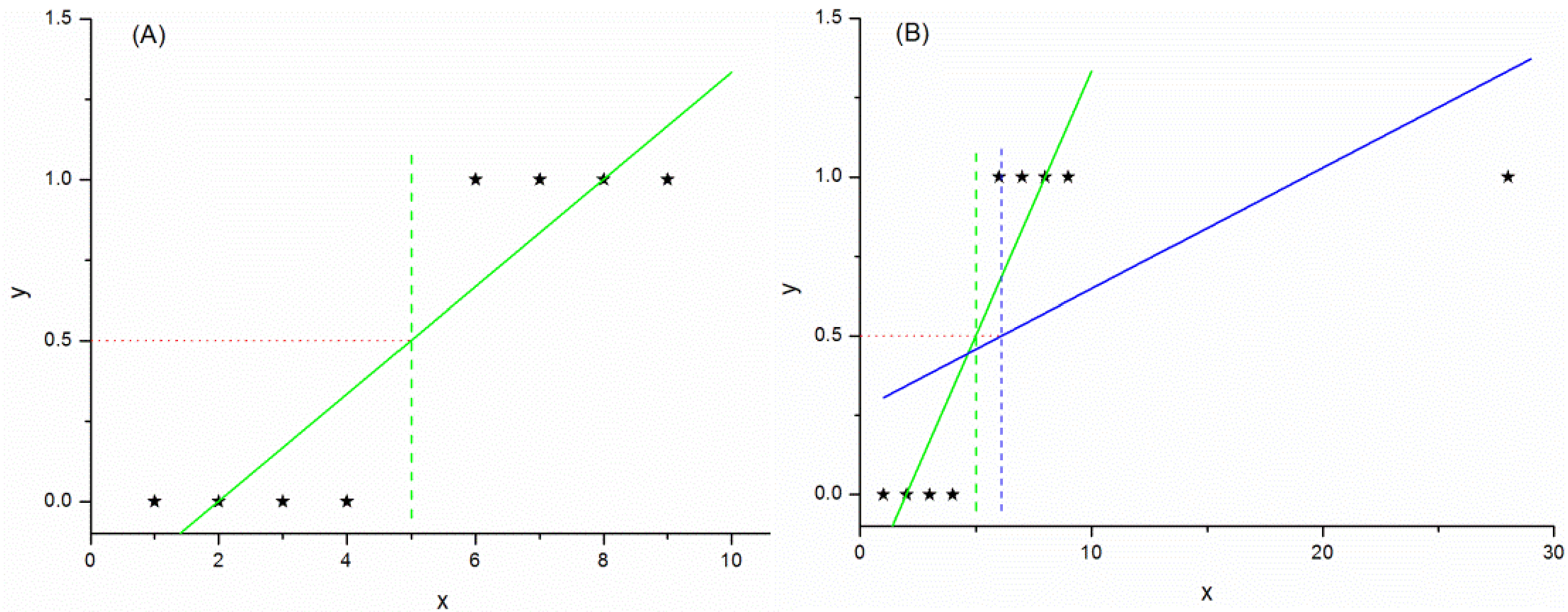

图4-1 利用线性回归处理分类问题的图示

利用线性回归处理分类问题，当有“离群点”时往往会有麻烦，所以一般不会用线性回归处理分类问题。

对于二分类问题，其输出为0或1，而线性回归模型的预测值*z*=*xW*是实数值，因此需要将实数值*z*转换为0/1值。首先可能想到单位阶跃函数：

$$\phi(z) = \begin{cases} 0, & z < 0 \\ 0.5, & z = 0 \\ 1, & \mathrm{z} > 0 \end{cases}$$

但单位阶跃函数不连续，不利于后续的数学处理。于是采用一个与单位阶跃函数“近似”的函数Sigmoid函数，也叫Logistic函数：

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

Sigmoid函数具有良好的性质，z趋于正无穷时，$g(z)$趋于1，当z趋于负无穷时，$g(z)$趋于0，这非常适合于分类模型。另外，它还有良好的导数性质：

$$g'(z) = g(z)[1 - g(z)]$$

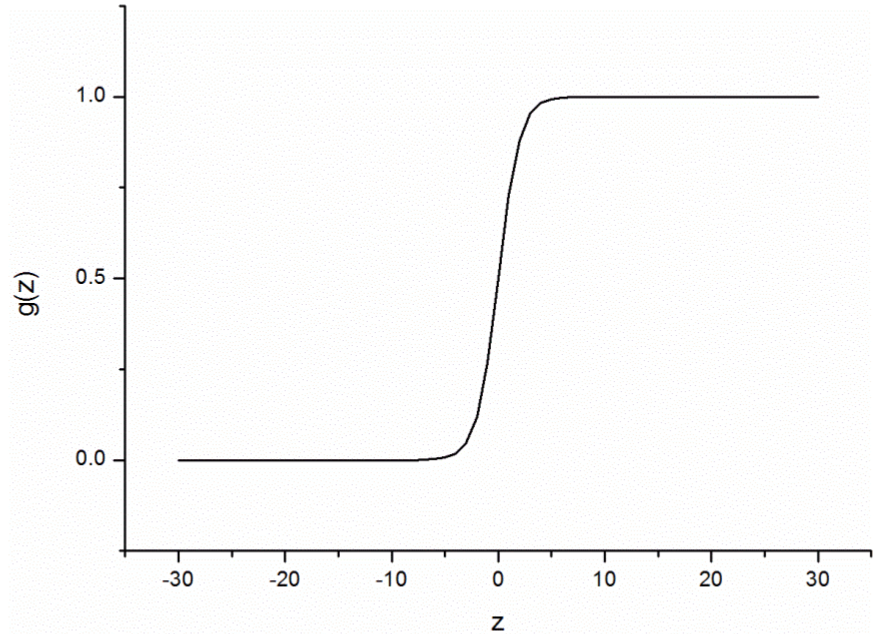

图4-2 Sigmoid函数

令 $z=xW$ 就得到逻辑回归模型的一般形式：

$$h_w(x) = \frac{1}{1 + e^{-xW}}$$

其中，$W=[w_0,w_1,\cdots,w_{m-1}]^T$为模型参数，$\boldsymbol{x}=[x_0,x_1,\cdots,x_{m-1}]$为样本输入，$h_w(\boldsymbol{x})$为模型输出，可理解为样本为阳性即y=1的概率。

可以取阀值为0.5，当$h_w(\boldsymbol{x})$≥0.5，即$\boldsymbol{x}W$≥0时y=1，当$h_w(\boldsymbol{x})$<0.5，即$\boldsymbol{x}W$<0时y=0。

对于多个样本，可以写成矩阵形式：

$$h_w(X) = \frac{1}{1 + e^{-XW}}$$

2. 逻辑回归的损失函数

前面提到，$h_w(\boldsymbol{x})$可以理解为样本$\boldsymbol{x}$为阳性即y=1的概率，即：

$$P(y = 1|x; W) = h_w(x)$$

相应地，样本x为阴性即y=0的概率为：

$$P(y = 0|x; W) = 1 - h_w(x)$$

将上面两式写为统一的形式：

$$P(y|x; W) = [h_w(x)]^y[1 - h_w(x)]^{1-y}$$

$$P(y|x;W)=[h_w(x)]^y[1-h_w(x)]^{1-y}$$

注意：式中y∈{0,1}。

对任一样本点$x^{(i)}$，它的目标值为$y^{(i)}$的概率就是：

$$P\left(y^{(i)}\middle|x^{(i)};W\right)=\left[h_w\left(x^{(i)}\right)\right]^{y^{(i)}}\left[1-h_w\left(x^{(i)}\right)\right]^{1-y^{(i)}}$$

对于训练数据集中*n*个样本点的总概率为： 类似于线性回归的差，只不过是另一种计算差的方法

$$L(W)=\prod_{i=0}^{n-1}P\left(y^{(i)}\middle|x^{(i)};W\right)=\prod_{i=0}^{n-1}\left[h_w\left(x^{(i)}\right)\right]^{y^{(i)}}\left[1-h_w\left(x^{(i)}\right)\right]^{1-y^{(i)}}$$

L(*W*)称为似然函数，它是*W*的函数。我们希望找到使得*L*(*W*)取得最大值的*W*作为最终的模型参数，这就是**极大似然估计**。为了简化计算，取负对数似然函数作为损失函数，也称**交叉熵损失函数**。

$$L(W)=\prod_{i=0}^{n-1}\left[h_w\left(x^{(i)}\right)\right]^{y^{(i)}}\left[1-h_w\left(x^{(i)}\right)\right]^{1-y^{(i)}}$$

$$J(W)=-\ln L(W)=-\sum_{i=0}^{n-1}\left\{y^{(i)}\ln\left[h_w\left(x^{(i)}\right)\right]+\left(1-y^{(i)}\right)\ln\left[1-h_w\left(x^{(i)}\right)\right]\right\}$$

取对数并不会影响函数极值点的位置，但可以将连乘符号转变为加和符号，取负号是为了将极大值问题转化为求极小值问题。

确定了损失函数就可以采用梯度下降算法求解了：

$$w_j=w_j-\eta\frac{\partial J(W)}{\partial w_j}$$

下面讨论其中偏导的求取。

$$\frac{\partial J(W)}{\partial w_j} = -\sum_{i=0}^{n-1}\left\{y^{(i)}\frac{1}{g(x^{(i)}W)} - \left(1-y^{(i)}\right)\frac{1}{1-g(x^{(i)}W)}\right\}\frac{\partial g\left(x^{(i)}W\right)}{\partial w_j}$$

只对w_j求导，对单个变量求导比较方便

$$= -\sum_{i=0}^{n-1}\left\{y^{(i)}\frac{1}{g(x^{(i)}W)} - \left(1-y^{(i)}\right)\frac{1}{1-g(x^{(i)}W)}\right\}g\left(x^{(i)}W\right)\left[1-g\left(x^{(i)}W\right)\right]\frac{\partial x^{(i)}W}{\partial w_j}$$

$$= -\sum_{i=0}^{n-1}\left\{y^{(i)}\left[1-g\left(x^{(i)}W\right)\right] - \left(1-y^{(i)}\right)g\left(x^{(i)}W\right)\right\}x_j^{(i)}$$

$$= -\sum_{i=0}^{n-1}\left[y^{(i)} - g(x^{(i)}W)\right]x_j^{(i)}$$

因此：

$$w_j = w_j + \eta\sum_{i=0}^{n-1}\left[y^{(i)} - g(x^{(i)}W)\right]x_j^{(i)}$$

所有样本的第j个属性，所以X要转置

矩阵形式：

$$W = W + \eta X^T\left[Y - g(XW)\right]$$

为什么不用最小二乘法呢？如果以误差平方和作为损失函数：

$$J(W)=\frac{1}{2n}\sum_{i=0}^{n-1}\left[h_w\left(x^{(i)}\right)-y^{(i)}\right]^2=\frac{1}{2n}\sum_{i=0}^{n-1}\left[g\left(x^{(i)}W\right)-y^{(i)}\right]^2$$

在利用梯度下降算法求解时：

$$\frac{\partial J(W)}{\partial w_j}=\frac{1}{n}\sum_{i=0}^{n-1}\left[g\left(x^{(i)}W\right)-y^{(i)}\right]g\left(x^{(i)}W\right)\left[1-g\left(x^{(i)}W\right)\right]x_j^{(i)}$$

在上式中，如果$y^{(i)}=1$：

- $g(x^{(i)}W)\rightarrow 1$时，即W接近最优点时，$\partial J(W)/\partial w_j \rightarrow 0$
- $g(x^{(i)}W)\rightarrow 0$时，即W远离最优点时，$\partial J(W)/\partial w_j \rightarrow 0$

这意味着J(W)函数存在很多极植点，不利于优化求解，容易陷入局部最优。导致这种现象的根本原因在于Sigmoid函数的导数中即包括g(z)也包括1-g(z)。

3. 逻辑回归的python实现

假设已有肿瘤预测的8个样本，数据如表4-1所示，其中“1”表示恶性肿瘤，“0”表示良性肿瘤。

表4-1 肿瘤预测样本数据

肿瘤尺寸	1	2	3	4	6	7	8	9
是否恶性肿瘤	0	0	0	0	1	1	1	1

编写逻辑回归程序，对肿瘤预测数据建立模型，打印模型输出曲线。

```
Iteration number: 4448
w =
 [[-18.27753571]
 [  3.69647296]]
```

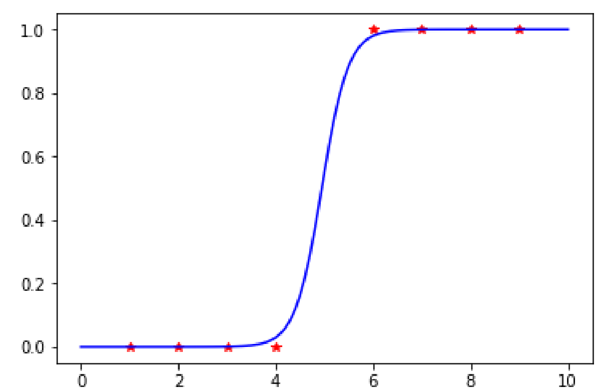

图4-3 肿瘤预测模型

4. 多分类问题

上面的逻辑回归算法只针对于二分类问题，对多分类问题，通常利用“拆解法”将多分类任务转化为二分类问题，最常用的拆分策略有：

- 一对一（One vs. One, 简称OvO）
- 一对其它（One vs. Rest, 简称OvR）
- 多对多（Many vs. Many, 简称MvM）

假设一共有K个类别，OvR只需训练K个分类器，而OvO需要训练$K(K-1)/2$个分类器，因此OvO的存储开销及计算量通常比OvR更大。但在训练时，OvR的每个分类器都使用全部训练样本，而OvO的分类器仅用到两个类别的样本，因此，在类别很多时，OvO的训练时间通常比OvR更小。而预测性能方面，取决于具体的数据分布，大多数情况下两者差不多。

5. 利用scikit-learn进行逻辑回归

在scikit-learn的linear_model模块实现了LogisticRegression类，即可用于二分类也可用于多分类逻辑回归，其用法如下：

LogisticRegression(penalty=’l2’, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver=’warn’, max_iter=100, multi_class=’warn’, verbose=0, warm_start=False, n_jobs=None)

主要属性包括：

- classes_: 所有的类标签。
- coef_: 模型参数，对于二分类问题其形状为(1, n_features)。
- intercept_: 截距项。
- n_iter_: 迭代次数。

主要方法有：

- decision_function(X): 对样本X预测结果的置信度评分。
- fit(X, y, sample_weight = None): 对给定的训练数据集X, y训练模型，可通过sample_weight指定样本权重，返回对象本身。
- predict(X): 预测样本X的类别。
- predict_proba(X): 计算样本X属于每个类别的概率，按self.classes_的次序给出。
- score(X, y, sample_weight = None): 计算对测试数据X, y的预测评分。

下面利用scikit-learn中的LogisticRegression类解决一个乳腺癌检测的问题。

在scikit-learn自带的数据集中包含一个乳腺癌检测的数据集，每个样本包含30个特征，这些特征是先从病灶影像图片中提取10个关键属性，包括radius（半径）、texture（纹理）、perimeter（周长）、area（面积）、smoothness（光滑度）、compactness（致密度）、concavity（凹度）、concave points（凹点）、symmetry（对称性）、fractal dimension（分形维度）。然后再构造出每个特征的标准差和最大值，这样每个特征又衍生出两个特征，共形成30个特征。

程序输出结果为：

```
data shape: (569, 30), num. positive: 357; num. negetive: 212
train score: 0.957286; test score: 0.953216
```