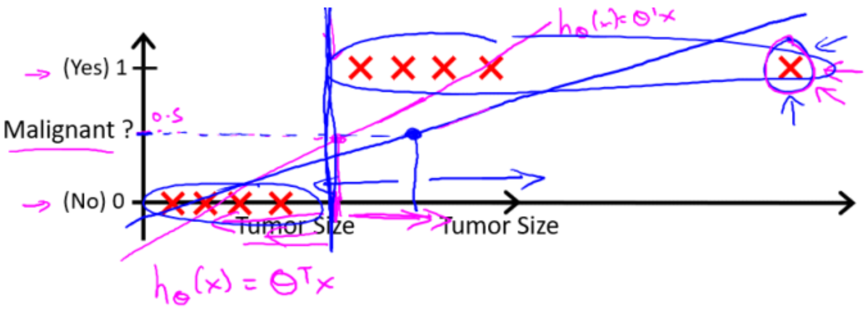
6 逻辑回归

2022年10月24日　15:54

6.1分类

一些分类的例子：判断是否为垃圾邮件；判断网上交易是否存在欺诈；判断良性和恶行肿瘤。

可以用0表示负类，用1表示正类。

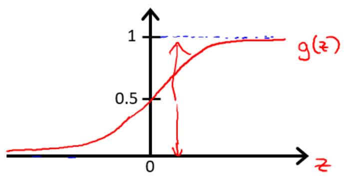


可以用线性回归模型来处理肿瘤的分类问题，将阈值设为0.5即可。但显然，额外的例子对模型有非常大的影响。因而将线性回归模型应用于分类问题不是一个好主意。

6.2假设表示

sigmoid函数/logistic函数：

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



假设函数：

$$h_\theta(x)=g(\theta^T x)$$

Example: If $x=\begin{bmatrix} x_0 \\ x_1 \end{bmatrix}=\begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$

$h_\theta(x)=0.7$　　y=1

Tell patient that 70% chance of tumor being malignant

以肿瘤分类的例子，对于特征量为x的病人，其肿瘤为恶性，也即y=1的概率为70%。y=1的概率和y=0的概率之和为1。

6.3决策边界(Dicision Boundary)

$$h_\theta(x) = g(\theta^T x)$$
$$g(z) = \frac{1}{1+e^{-z}}$$

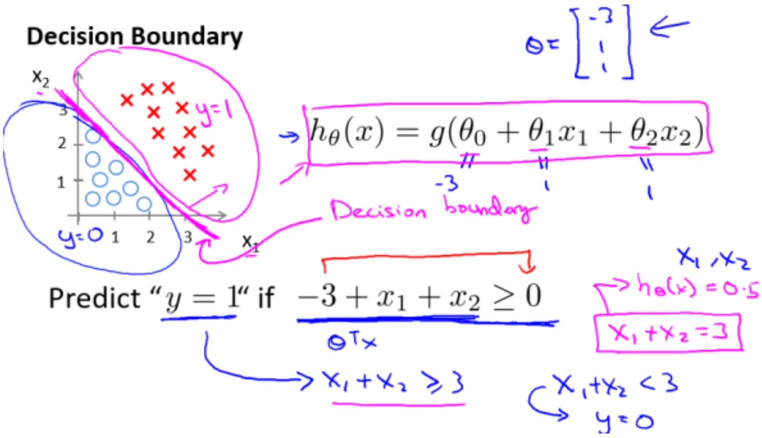
Suppose predict "$y = 1$" if $h_\theta(x) \geq 0.5$

$\theta^T x \geq 0$

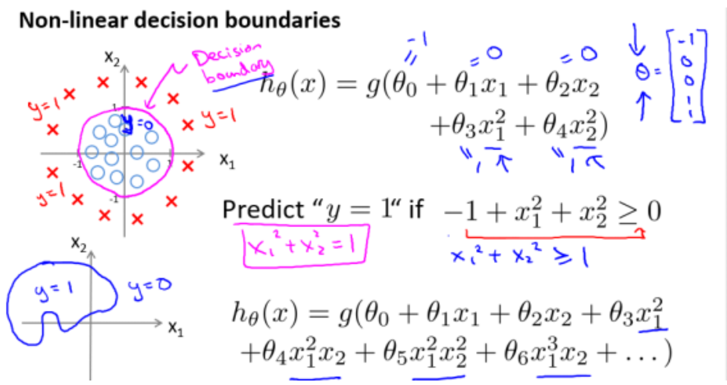
predict "$y = 0$" if $h_\theta(x) < 0.5$

$\theta^T x < 0$

判定假设函数是否大于0.5就是判断$\theta^T x$是否大于0。



以二特征为例，可以根据参数构造出分类边界。



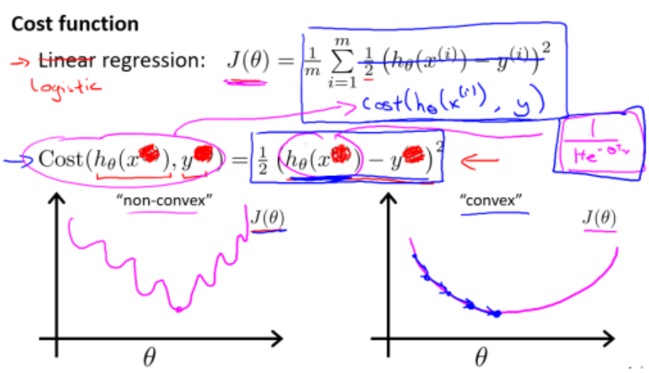
对于非线性的边界亦同理。

决策边界的属性并不是训练集的属性，而是假设本身和其参数的属性。

更加复杂的多项式也会相应地构造出更加复杂的决策边界。

6.4代价函数

重点在于前述的参数θ怎么确定。



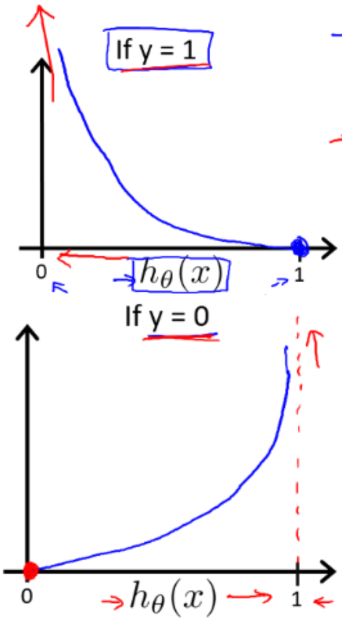
类似于线性回归的代价函数，若用相同的式子对其进行表示，由于逻辑假设函数的复杂性，对该代价函数进行绘制会发现可能会有很多的局部最优值，梯度下降法对其不

适用。我们希望构造出一个凸函数的代价函数从而有利于用梯度下降法解决问题。

逻辑回归代价函数定义如下：

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

对其进行绘制后如下：



6.5简化的代价函数与梯度下降

前面一节的分段代价函数可以用一个式子来表达：

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)\right]$$

它是一个凸函数。

依旧使用梯度下降法得到最小的 θ 取值。

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

(simultaneously update all θ_j)

可以看到它和线性回归的式子是一模一样的，但是由于假设函数不一样，所以算法并不是相同的。

特征缩放同样适用于逻辑回归的梯度下降，使其下降收敛更快。

6.6高级优化

高级优化算法能使算法的计算速率优于梯度下降，也使其更适于解决大量数据的问题。

- 共轭梯度算法
- BFGS
- L-BFGS

优点：

- 不需要手动选择学习率
- 收敛速度远远快于梯度下降

缺点：

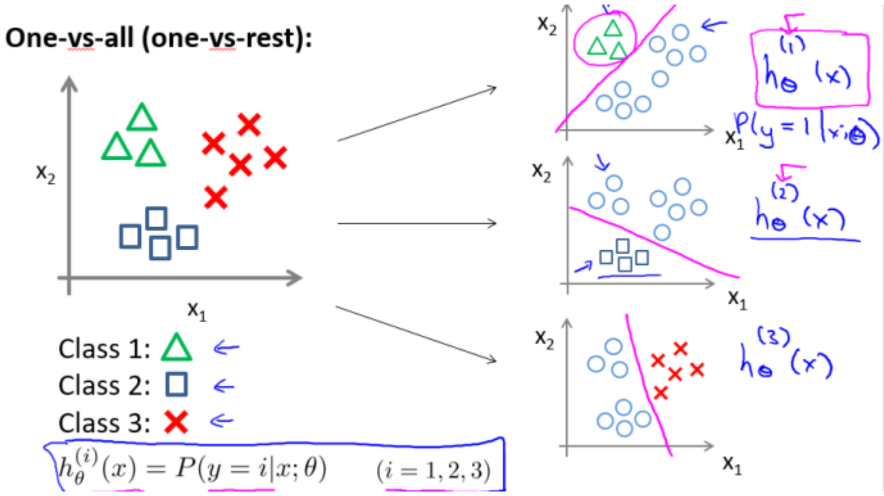
比梯度下降算法更复杂

6.7多元分类：一对多

多类别的分类问题(Multi-class classification)：
举例而言，将邮件分成工作、朋友、家庭、爱好等；对病人病情的诊断，没有生病、感冒、流感等；将天气分为晴朗、多云、有雨、有雪等。

前述已经讨论过二元分类的具体方法，比如通过构造假设函数直线进行分类。那么对于多类别的分类该如何处理？

一对余算法：将多分类转变为多个独立的二元分类问题。如下面这个三元分类为例：



构造一个“伪训练集”，对于一个目标类别为“正样本”，将剩余的类别作为一个类别为“负样本”。从而拟合出三个分类器。那么对于一个新的输入，为了预测它的类别，将其通过三个分类器计算后选择计算值最大，可信度最高、效果最好的。