假设函数表示

对于一个二分类，我们希望分类器输出值在0到1之间，但如果使用线性回归，函数的输出值可能不在这个范围，逻辑回归算法可以让输出值在0到1之间。

虽然逻辑回归算法中出现回归，但它其实是分类算法，与回归无关。

我们引入一个新的模型，逻辑回归，该模型的输出变量范围始终在0和1之间。 逻辑回归模型的假设是： $h_\theta\left(x\right)=g\left(\theta^{T}X\right)$

X 代表特征向量 g 代表逻辑函数（**logistic function**)是一个常用的逻辑函数为**S**形函数（**Sigmoid function**），公式为： $g\left(z\right)=\frac{1}{1+e^{-z}}$。

$h_\theta\left(x\right)$的作用是，对于给定的输入变量X，根据选择的参数计算y=1的可能性（**estimated probablity**）即$h_\theta\left(x\right)=P\left(y=1|x;\theta\right)$

> *注：y=1表示正向类*

判定边界 （Decision Boundary）

在逻辑回归中，我们预测：

当$h_\theta\left(x\right)>=0.5$时，预测 $y=1$。

当$h_\theta\left(x\right)<0.5$时，预测 $y=0$ 。

根据上面绘制出的 **S** 形函数图像，我们知道当

$z=0$ 时 $g(z)=0.5$

$z>0$ 时 $g(z)>0.5$

$z<0$ 时 $g(z)<0.5$

又 $z=\theta^{T}x$ ，即： $\theta^{T}x>=0$ 时，预测 $y=1$ $\theta^{T}x<0$ 时，预测 $y=0$

让我们来看另一个例子：

假设我们已经拟合好了一条直线，并且参数θ是向量[-3 1 1]。

则当$-3+x_1+x_2\geq 0$时，模型将预测y=1。我们可以绘制直线$x_1+x_2=3$，这条线便是我们模型的分界线，将预测为1的区域和预测为0的区域分隔开。

代价函数

这节将介绍如何拟合逻辑回归模型的参数θ。

对于线性回归模型，我们定义的代价函数是所有模型误差的平方和。理论上来说，我们也可以对逻辑回归模型沿用这个定义，但是问题在于，当我们将$h_\theta\left(x\right)=\frac{1}{1+e^{-\theta^Tx}}$带入到这样定义了的代价函数中时，我们得到的代价函数将是一个非凸函数（**non-convexfunction**）。

这种代价函数有许多局部最小值，这将影响梯度下降算法寻找全局最小值。所以我们需要找另外一种比较合适的代价函数。

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

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

> *注：y只能是0或1*

这个图像有个好处：当y=1时，即病人确实有肿瘤，h(x)=1的代价为0，而h(x)=0代价为无穷大，也就是说病人有病的情况下未检测出来的代价无穷大。当y=0时，h(x)=0的代价为0，当h(x)=1，代价无穷大。

将构建的 $Cost\left(h_\theta\left(x\right),y\right)$简化如下：
$Cost\left(h_\theta\left(x\right),y\right)=-y\times log\left(h_\theta\left(x\right)\right)-\left(1-y\right)\times log\left(1-h_\theta\left(x\right)\right)$ 带入代价函数得到：
$J\left(\theta\right)=\frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}\log\left(h_\theta\left(x^{(i)}\right)\right)-\left(1-y^{(i)}\right)\log\left(1-h_\theta\left(x^{(i)}\right)\right)]$ 即：
$J\left(\theta\right)=-\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log\left(h_\theta\left(x^{(i)}\right)\right)+\left(1-y^{(i)}\right)\log\left(1-h_\theta\left(x^{(i)}\right)\right)]$

梯度下降

在得到这样一个代价函数以后，我们便可以用梯度下降算法来求得能使代价函数最小的参数了。算法为：

Repeat { $\theta_j:=\theta_j-\alpha\frac{\partial}{\partial\theta_j}J(\theta)$ (**simultaneously update all**) }

求导后得到：

Repeat { $\theta_j:=\theta_j-\alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right)-y^{(i)}\right)x_j^{(i)}$ **(simultaneously update all**) }

这样就能找到$min(J(\theta))$对应的θ向量。

线性回归和逻辑回归的梯度下降算法看起来完全一致，但因为他们的假设函数不同，所以逻辑函数和线性回归的梯度下降是完全不同的对象。

除了梯度下降算法以外，还有一些常被用来令代价函数最小的算法，这些算法更加复杂和优越，而且通常不需要人工选择学习率，通常比梯度下降算法要更加快速。这些算法有：

- **共轭梯度（Conjugate Gradient）**
- **局部优化法(Broyden fletcher goldfarb shann,BFGS)**
- **有限内存局部优化法(LBFGS)**

多分类：一对多

二分类和多分类的数据集差别如下：

多分类的思想是：将多分类分成若干个二分类。比如将A, B, C 分类。可以分成：

- A or BC
- B or AC
- C or AB

为了能实现这样的转变，我们将多个类中的一个类标记为正向类（$y=1$），然后将其他所有类都标记为负向类，这个模型记作$h_\theta^{(1)}(x)$。接着，类似地第我们选择另一个类标记为正向类（$y=2$），再将其它类都标记为负向类，将这个模型记作 $h_\theta^{(2)}(x)$,依此类推。

最后我们得到一系列的模型简记为： $h_\theta^{(i)}(x)=p(y=i|x;\theta)$其中：$i=(1,2,3....k)$

在我们需要做预测时，我们将所有的分类机都运行一遍，然后对每一个输入变量，都选择最高可能性的输出变量，即$\max\limits_i h_\theta^{(i)}(x)$。比如一个数据x经过三个分类器运算后得到的结果为$0.1, 0.2, 0.7$。很显然第三个结果概率更大，我们就选择这一个。

正则化

过拟合问题(over-fitting problem)

下面第一张图是欠拟合，第三张是过拟合，过于强调拟合原始数据，而丢失了算法的本质。我们可以看出，若给出一个新的值使之预测，它将表现的很差，是过拟合，虽然能非常好地**适应我们的训练集**但在新输入变量进行**预测时可能会效果不好**。

如何处理过拟合：

1. 丢弃一些不能帮助我们正确预测的特征。可以是手工选择保留哪些特征，或者使用一些模型选择的算法来帮忙（例如**PCA**）
2. 正则化。 保留所有的特征，但是减少参数的大小（**magnitude**）。

代价函数

在上面的例子中我们发现，过拟合是高次项导致的，所以我们要做的就是在一定程度上减小这些参数θ 的值，这就是正则化的基本方法。我们决定要减少θ_3和θ_4的大小，我们要做的便是修改代价函数，在其中θ_3和θ_4 设置一点惩罚。

修改后的代价函数如下：$\min_\theta \ \frac{1}{2m}[\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right)-y^{(i)}\right)^2+1000\theta_3^2+10000\theta_4^2]$

> *注：如果我们想要代价函数尽量小，那必须让θ_3,θ_4尽量小。*

。假如我们有非常多的特征，我们并不知道其中哪些特征我们要惩罚，我们将对所有的特征进行惩罚，并且让代价函数最优化的软件来选择这些惩罚的程度。这样的结果是得到了一个较为简单的能防止过拟合问题的假设：$J\left(\theta\right)=\frac{1}{2m}[\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})^2+\lambda\sum_{j=1}^{n}\theta_j^2]$

其中λ又称为正则化参数（**Regularization Parameter**）。 注：根据惯例，我们不对θ_0 进行惩罚。经过正则化处理的模型与原模型的可能对比如下图所示：

如果选择的正则化参数λ 过大，则会把所有的参数都最小化了，导致模型变成 $h_\theta\left(x\right)=\theta_0$，也就是上图中红色直线所示的情况，造成欠拟合。

线性回归的正则化

正则化线性回归的代价函数为：

$J\left(\theta\right)=\frac{1}{2m}\sum_{i=1}^{m}[((h_\theta(x^{(i)})-y^{(i)})^2+\lambda\sum_{j=1}^{n}\theta_j^2)]$

$Repeat\ until\ convergence\{$

$\theta_0:=\theta_0-a\frac{1}{m}\sum_{i=1}^{m}((h_\theta(x^{(i)})-y^{(i)})x_0^{(i)})$

$\theta_j:=\theta_j-a[\frac{1}{m}\sum_{i=1}^{m}((h_\theta(x^{(i)})-y^{(i)})x_j^{(i)}+\frac{\lambda}{m}\theta_j]$

$for\ j=1,2,\dots n$

}

对上面的算法中$j=1,2,\ldots,n$ 时的更新式子进行调整可得（把θ_j）：

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

用正规方程求解正则化线性回归模型

没有正则化的正规方程解法：

$$\theta = (X^TX)^{-1}X^Ty$$

正则化的正规方程解法：

$$\theta = (X^TX+\lambda\begin{pmatrix}0 & 0 & \ldots & 0\\0 & 1 & \ldots & 0\\ & \cdots & & \\0 & 0 & \ldots & 1\end{pmatrix})^{-1}X^Ty$$

图中的矩阵尺寸为 $(n+1)*(n+1)$。

正则化的逻辑回归模型

$J\left(\theta\right)=\frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log\left(h_\theta\left(x^{(i)}\right)\right)-\left(1-y^{(i)}\right)\log\left(1-h_\theta\left(x^{(i)}\right)\right)\right]+\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$

Repeat until convergence{

$\theta_0 := \theta_0 - a\frac{1}{m}\sum_{i=1}^{m}((h_\theta(x^{(i)})-y^{(i)})x_0^{(i)})$

$\theta_j := \theta_j - a[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_j^{(i)}+\frac{\lambda}{m}\theta_j]$

$for\ j=1,2,\ldots n$

}

注：看上去同线性回归一样，但是知道 $h_\theta(x)=g\left(\theta^T X\right)$，所以与线性回归不同。