Logistic回归_吴恩达_机器学习笔记

1、分类

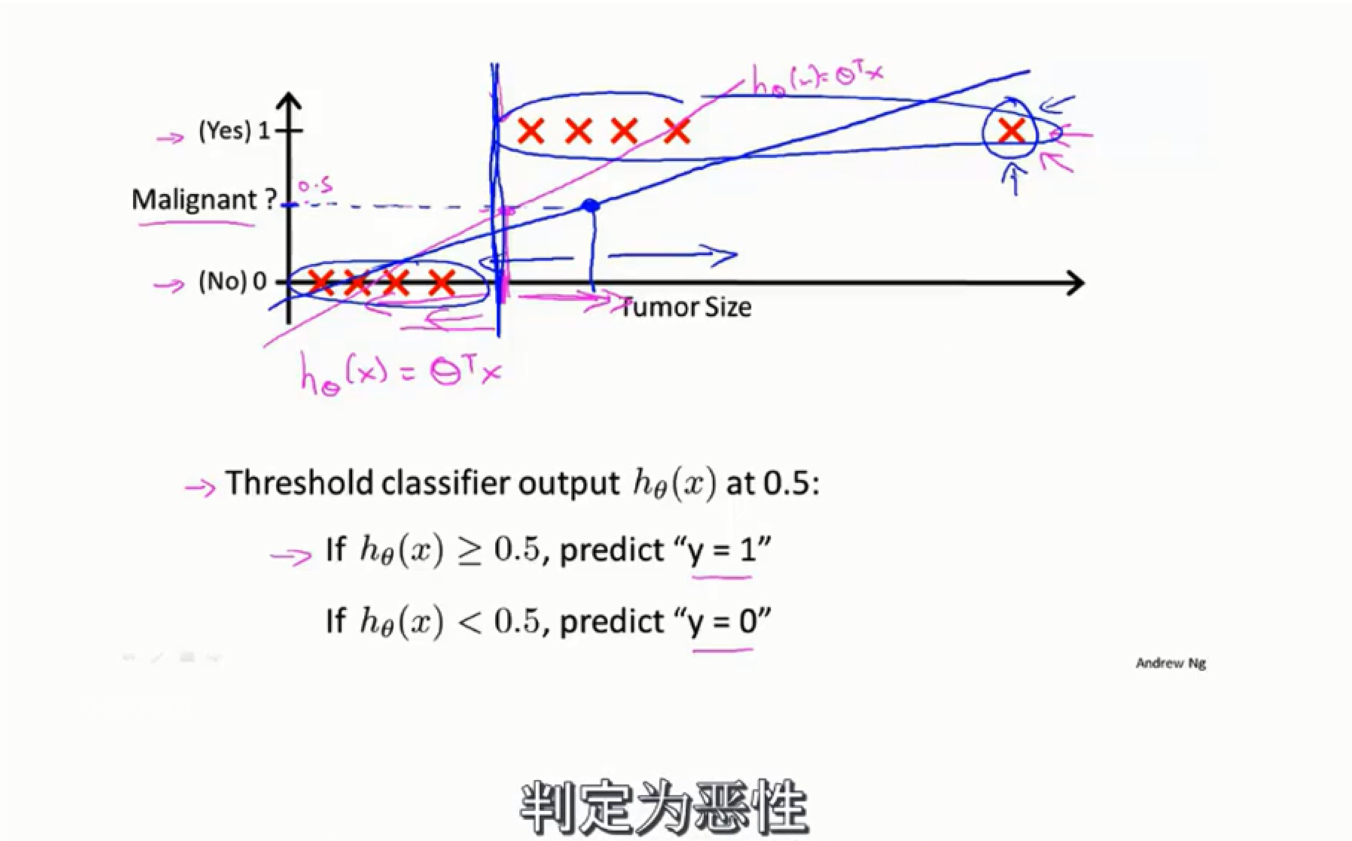
Logistic回归一般用来预测二分类问题（当然，可以通过一对多的方式进行多分类），此处我们先说二分类。一般会将label处理成（0和1）或者（-1,1）两个类别，其中1表示正类，0或者-1表示负类。

一般，我们的假设决策函数如下：

$$h_\Theta(x) >= 0.5, 1 | h_\Theta(x) < 0.5, 0$$

- **当假设函数的取值大于或等于0.5的时候，我们将label结果判定为正类。**
- **当假设函数的取值小于0.5的时候，我们将label结果判定为负类。**

假设使用线性回归来进行二分类问题判别的时候，无论采用什么样的线性函数，对结果的判定仍然会存在偏差，无法准确的判定结果。具体如下中的离群值所示：



实际上稍微偏远一点的样本，因为线性函数的判别，误判为了负类，实际上，这是一个正类。

2、假设陈述

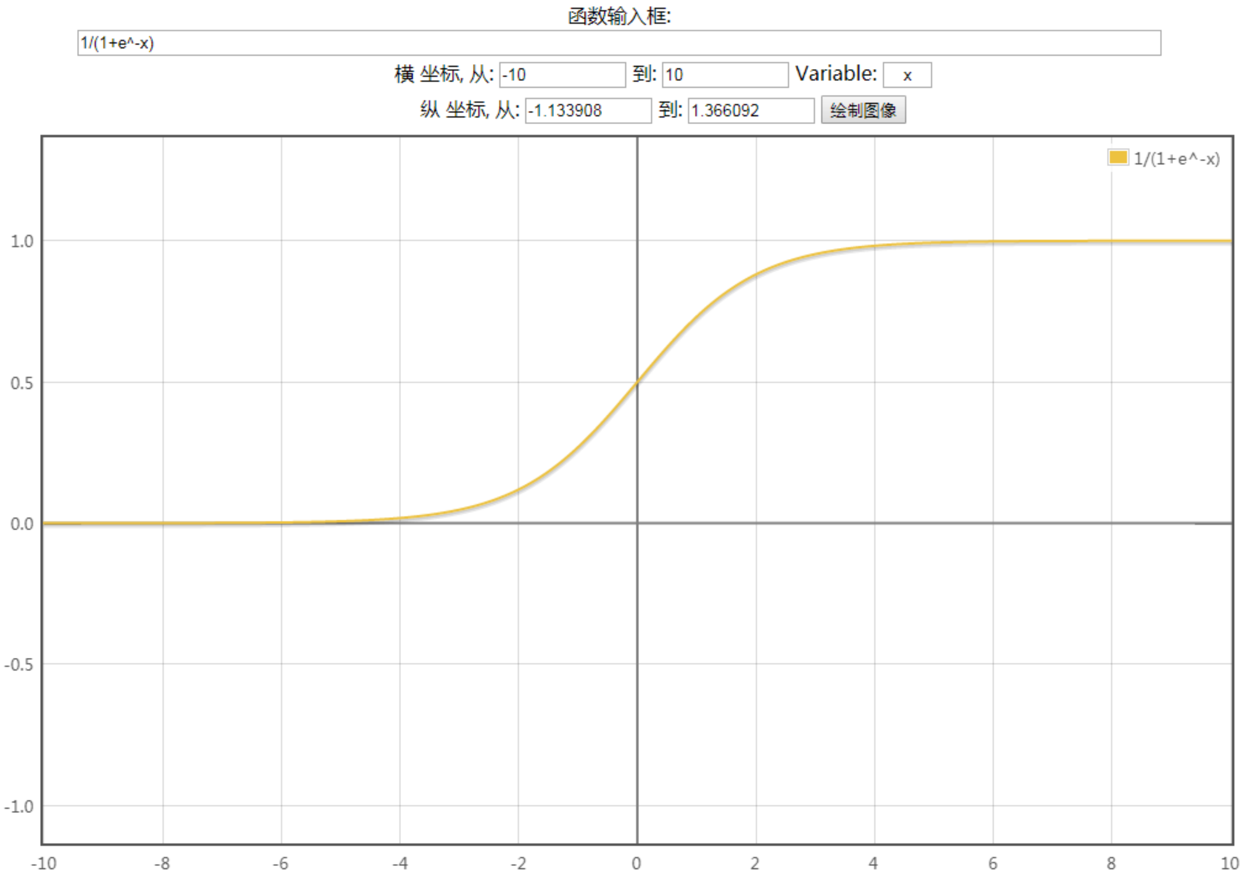
下面来说一下，如何将假设函数的输出值保证在0到1之间，这个时候经典的sigmod函数就出场了：

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

这就是sigmod函数的公式，这个公式的神奇之处在于，无论你取什么参数z值，这个函数的最终结果都会在0-1之间。所以，这个函数用来做Logistic的决策函数：

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

我们具体画一下这个函数的图像看看吧，会对函数更加深刻的理解。



从图上我们可以看到，实际上，当假设函数的取值越大时，那么他被判定为正类的概率就越大，从取值超过4开始，几乎就等于1了。

这里很重要的一点就是，上图中的参数θ，要求到合适的参数θ的值，才能拟合我们的函数，然后最终，才能够将结合输出，转换为判定函数。

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

从下面这个图，可以看出来，我们要取到的值实际上是一个条件概率表达式可以表达的内容。

首先，当y = 0的时候，那么我们x的特征所形成的条件概率，表明了例如一个人患癌症的概率是多少，当y=1的时候，那么我们x的特征所形成的条件概率，表明了 例如一个人不患癌症的概率是多少。

那么，实际上在这个模型中，只会生病或者不生病，他们加起来的概率正好就等于1。当完整的理解这个以后，将非常有助于我们待会儿去理解逻辑回归的代价函数。

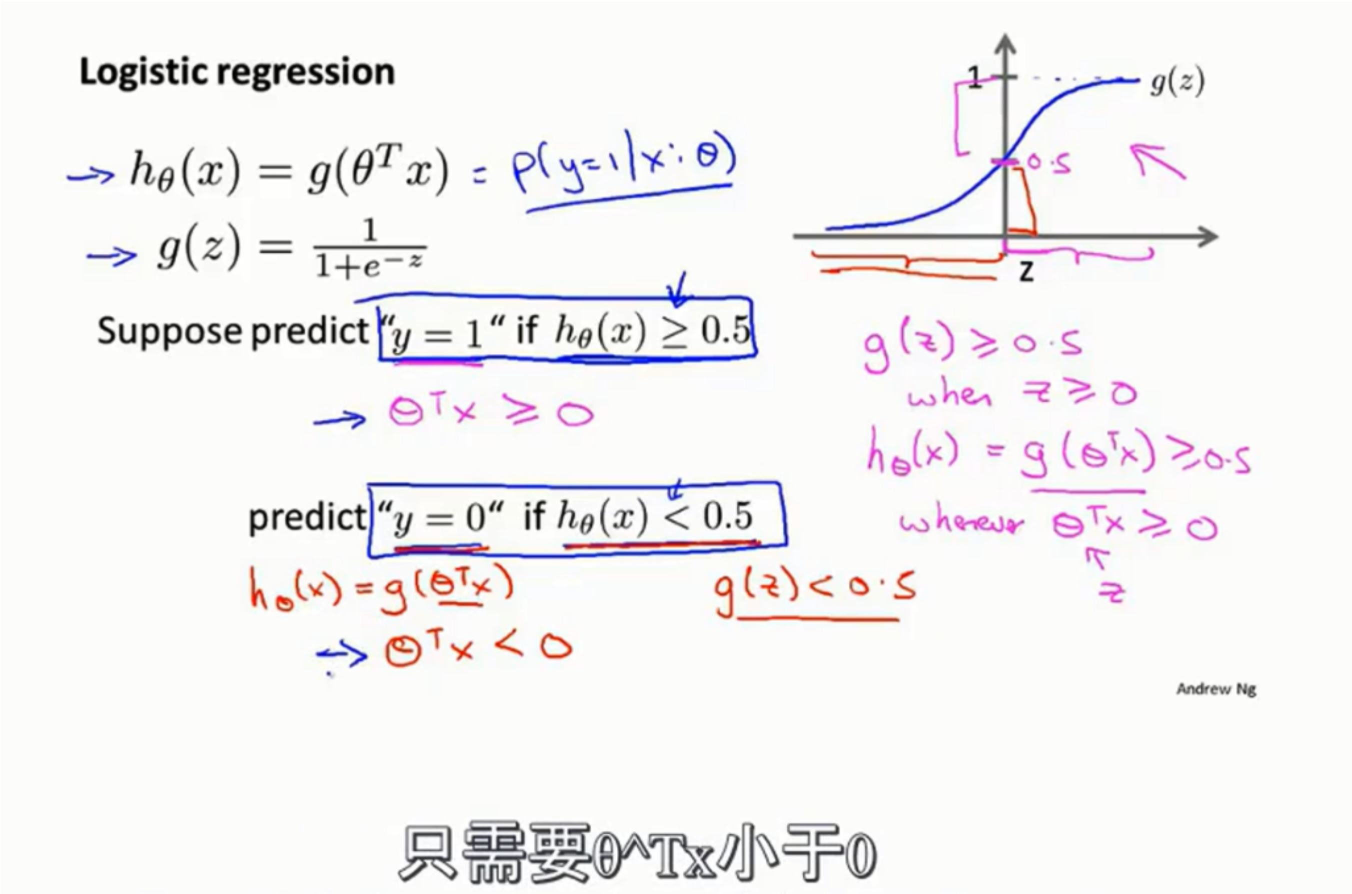
实质上，当计算出某个人判定正类的概率的时候，也将计算出判定为负类的概率，这就是第二个公式的意义。

$$\rightarrow P(y=0|x;\theta) + P(y=1|x;\theta) = 1$$
$$P(y=0|x;\theta) = 1 - P(y=1|x;\theta)$$

Andrew Ng

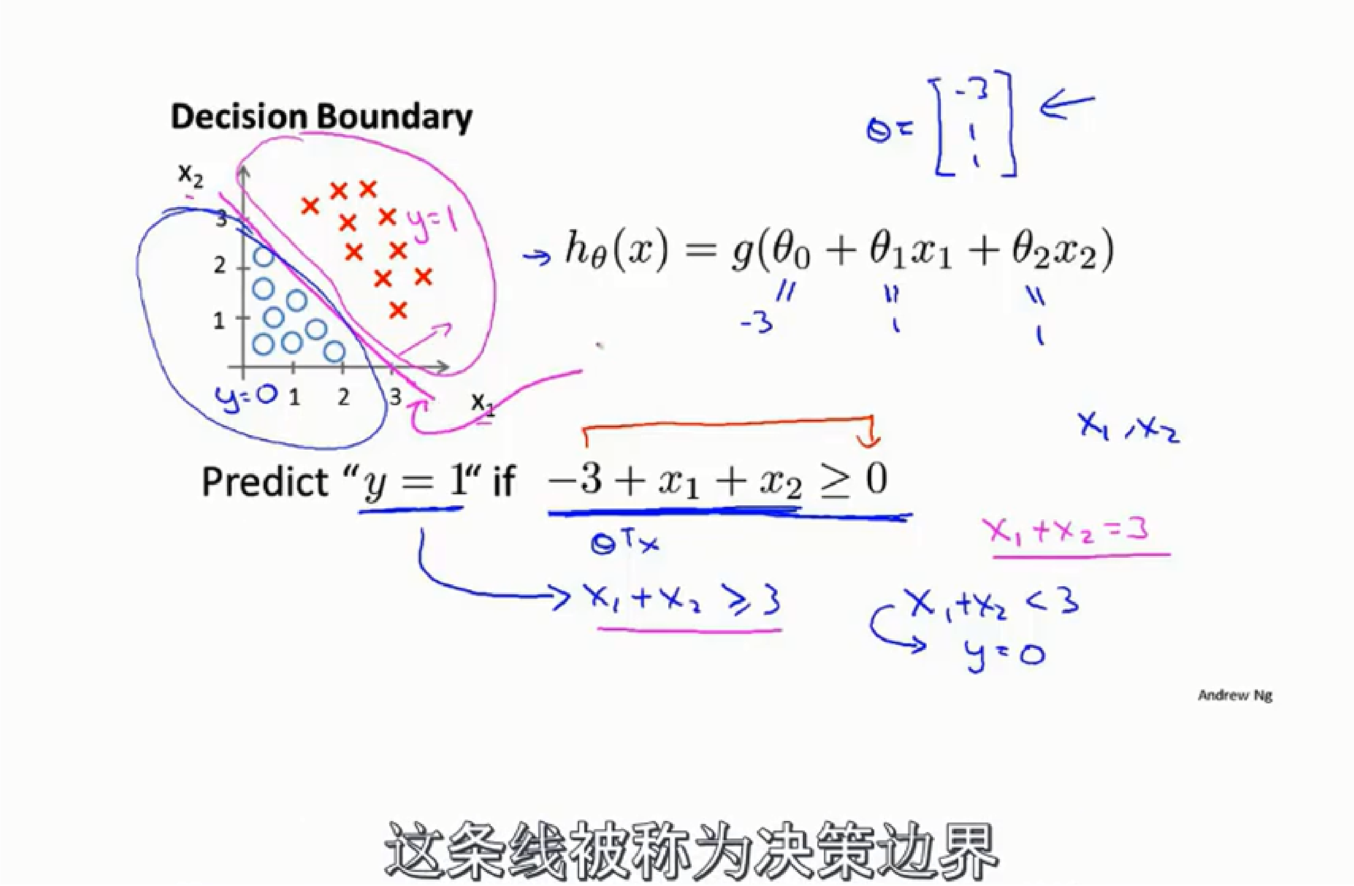
3、决策界限如何决策

要理解决策边界是如何进行分类决策的，下面这张图的信息量，可以说是非常的大。具体我们看图说明如下：



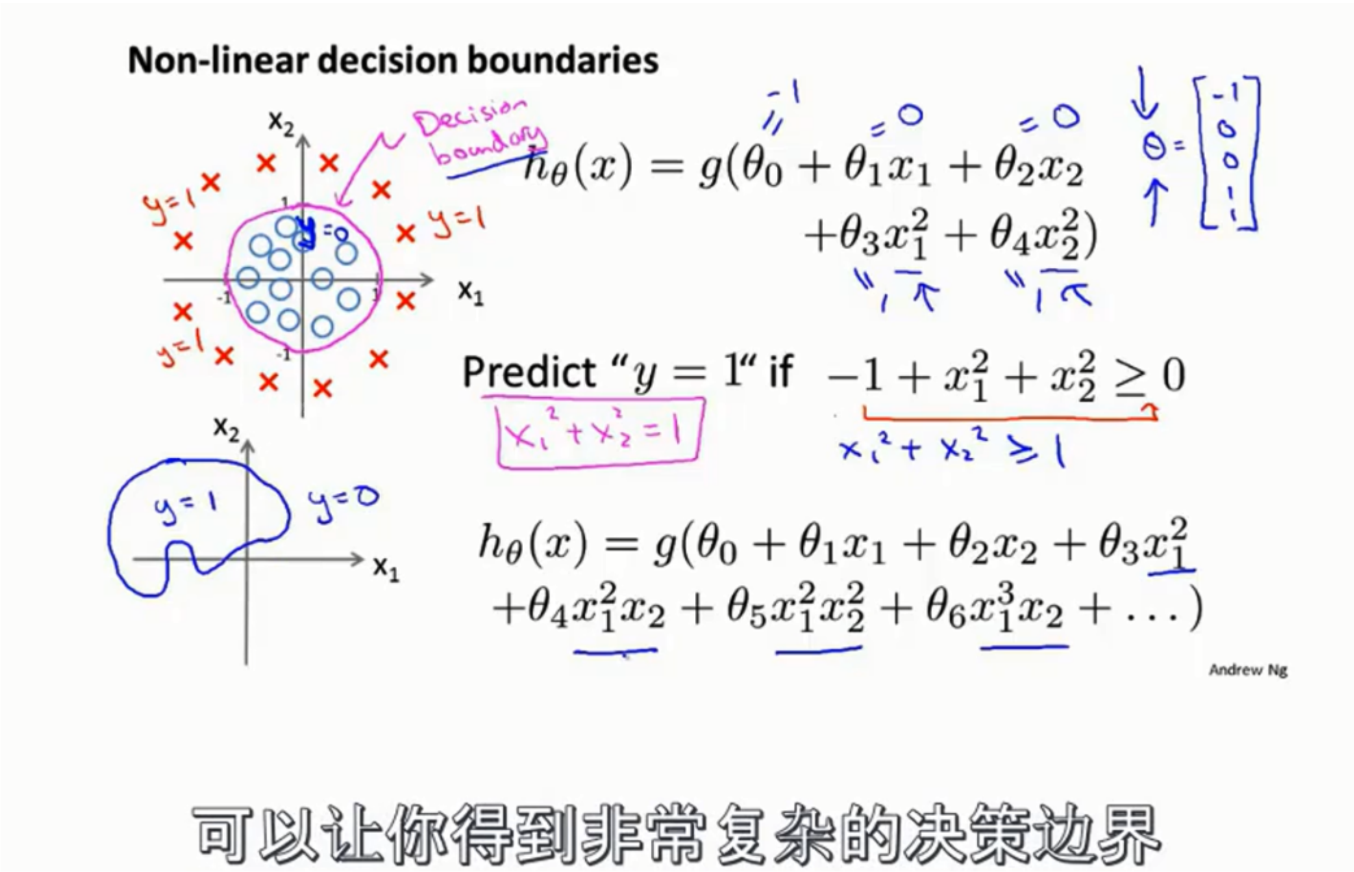
- 首先，sigmod函数中，以（0,0.5）为界限，当z>=0的时候，那么0.5<=g(z)<1,这里的实质就是θ^T*X取值大于等于0的时候，那么就会被判定为正类。
- 相反，可以得到θ^T*X取值为负数的时候，那么就会被判定为负类。

再通过一张图直观的了解决策边界在到底从事什么事情：



- 假设这个图上，已经拟合好了参数向量θ = [-3,1,1]
- 那么当前的决策边界函数就是-3+x1+x2 = 0，可以变换为x1+x2 = 3，那么可以看到x1+x2>=3的部分，就是我们的正类，x1+x2<3的部分，就是我们的负类，这个和上面标示的结果完全一致，非常完美。

最后，我们需要重点强调两个点：

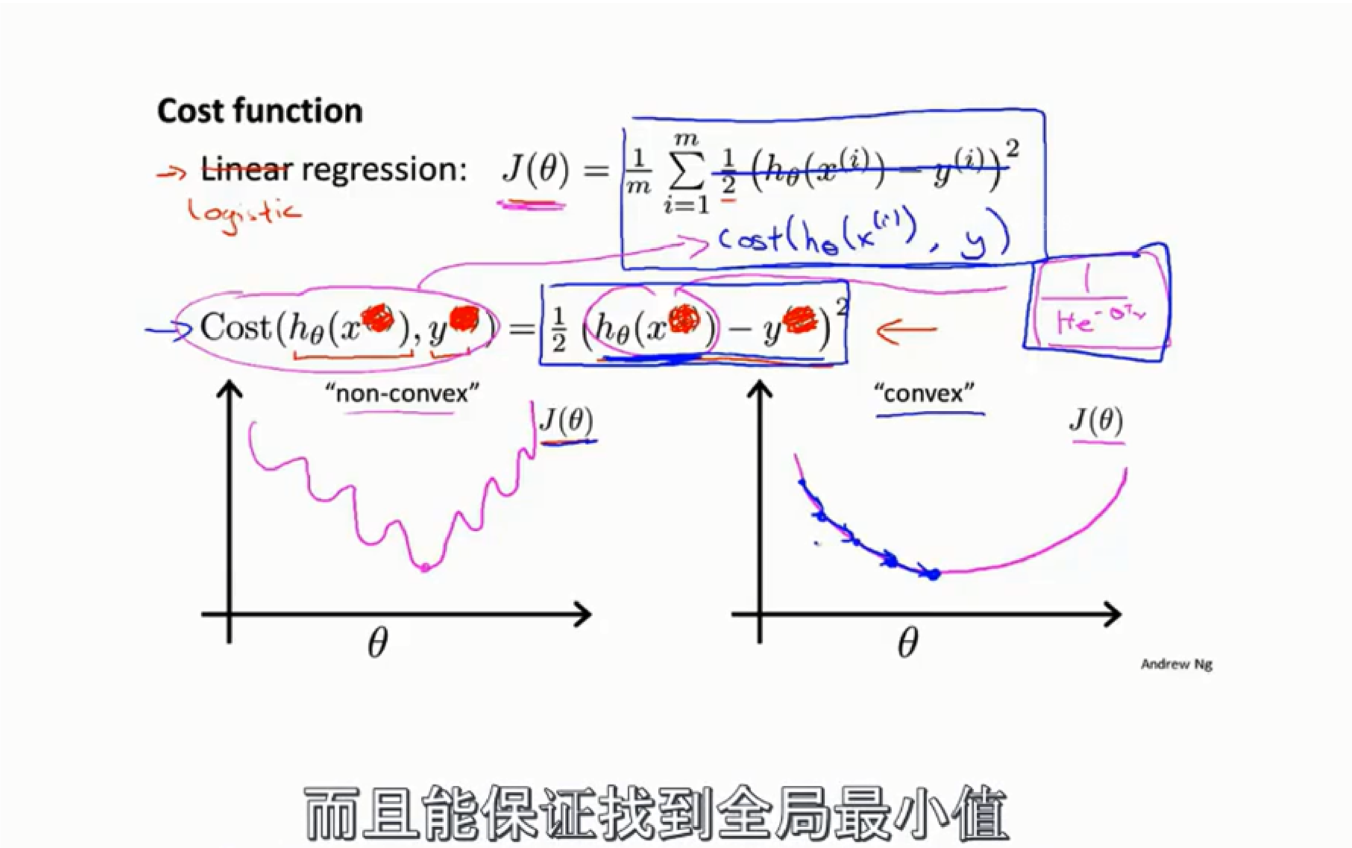


- 决策边界可以是参数向量和高阶项，这个观点至少打破了的固定思维模式，以为逻辑回归只能做线性可分的数据。 因此，在特征项中自行的添加特征的平房项或者多个特征的内积，那么也是可以实现更好的拟合状态的。
- 当高阶项很多的时候，甚至可以用决策边界来分类一些非常曲折的类型数据。具体可以参考图中，当然，这个地方的高阶项的组合方式，是非常复杂的，当数据量非常大的时候，那么高阶项和项与项之间的内积将会导致计算非常复杂。
- 到这个时候，Logistic回归将无法解决数据量过大的问题，这时候，神经网络的出现，将非常有效。
- **再次强调，决策边界不是训练集的属性，而是假设本身及其参数的属性，只要给定了参数向量θ，圆形的决策边界就确定下来了。训练集只是用来拟合参数向量θ的。但是，一旦拥有了参数向量θ的具体值，就确定了决策边界。**

4、代价函数

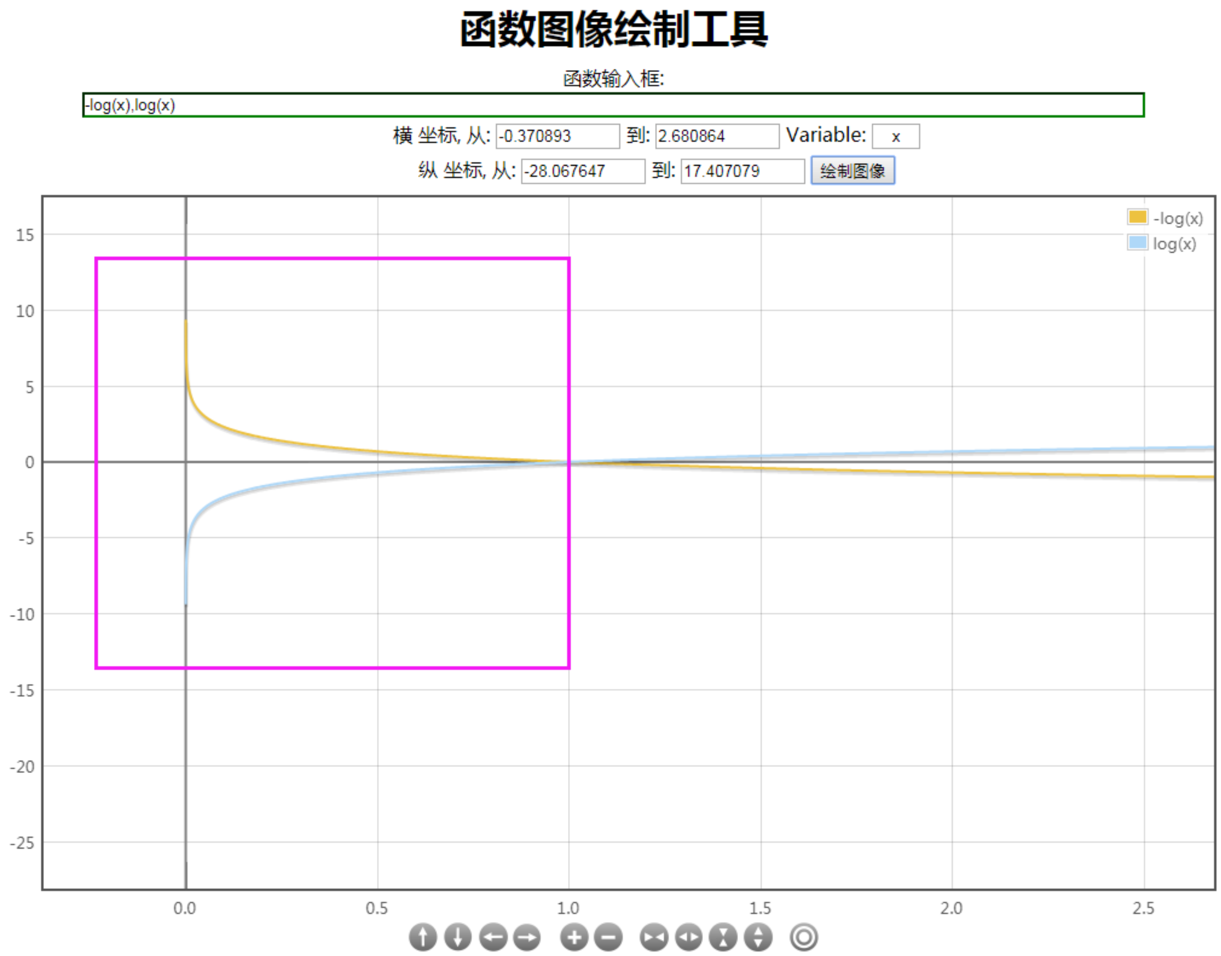
本节主要阐述如何拟合假设函数的参数θ。

首先我们来考虑一种可能，是否可以直接用线性回归的代价函数作为Logistic回归的代价函数呢？答案是不行的，原因我们结合图进行说明：



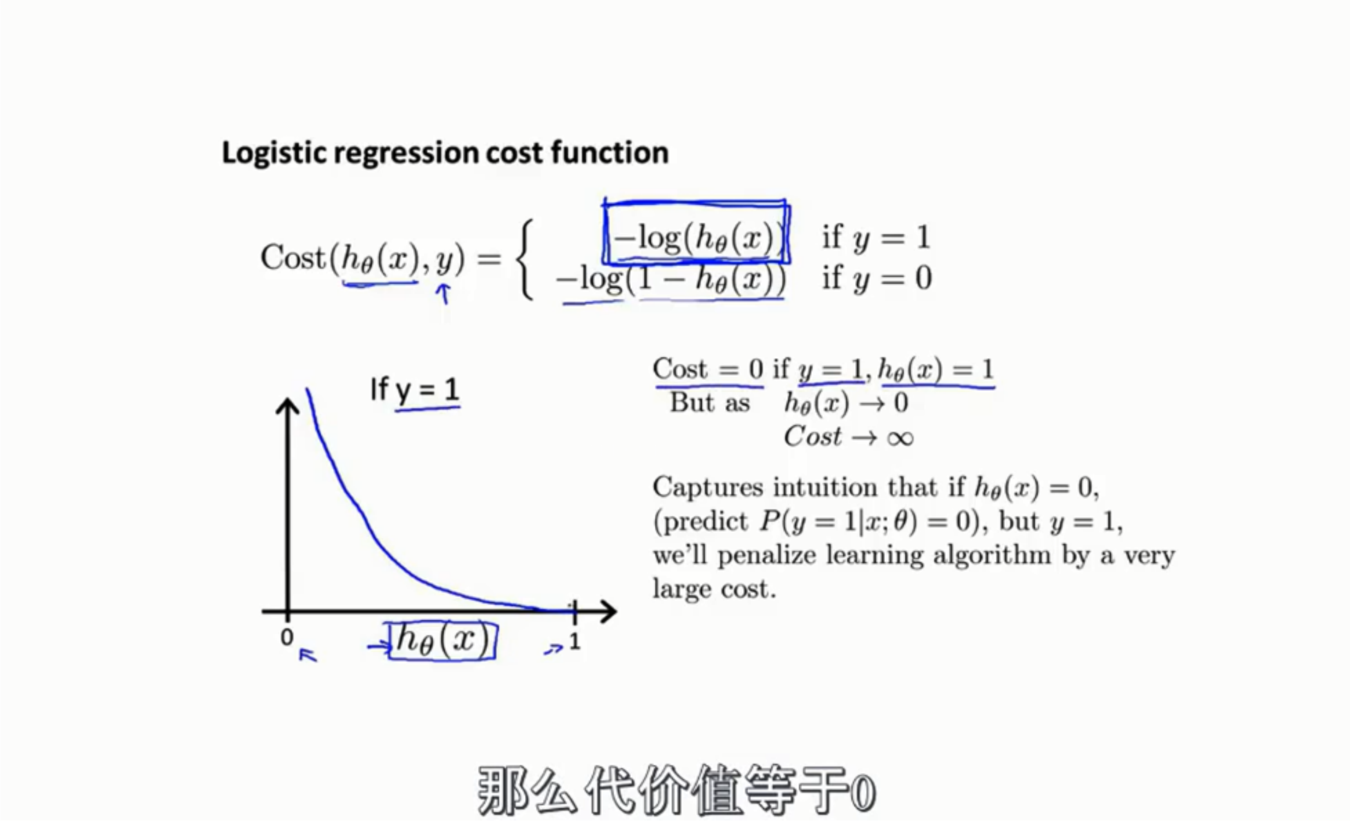
仔细看上图，我们看到的是线性回归的代价函数，先去掉上标的干扰，更加容易理解，那么我们如果直接用这个代价函数，那么其中hθ（x)项中，实际上是一个sigmod函数，这个函数如何组合到一起的话，是一个非凸函数，这个非凸函数有非常多的局部最小值，那么使用梯度下降法，是不能保证收敛到全局最小值的。因此，我们需要找到一个像上图中右图一样的凸函数。

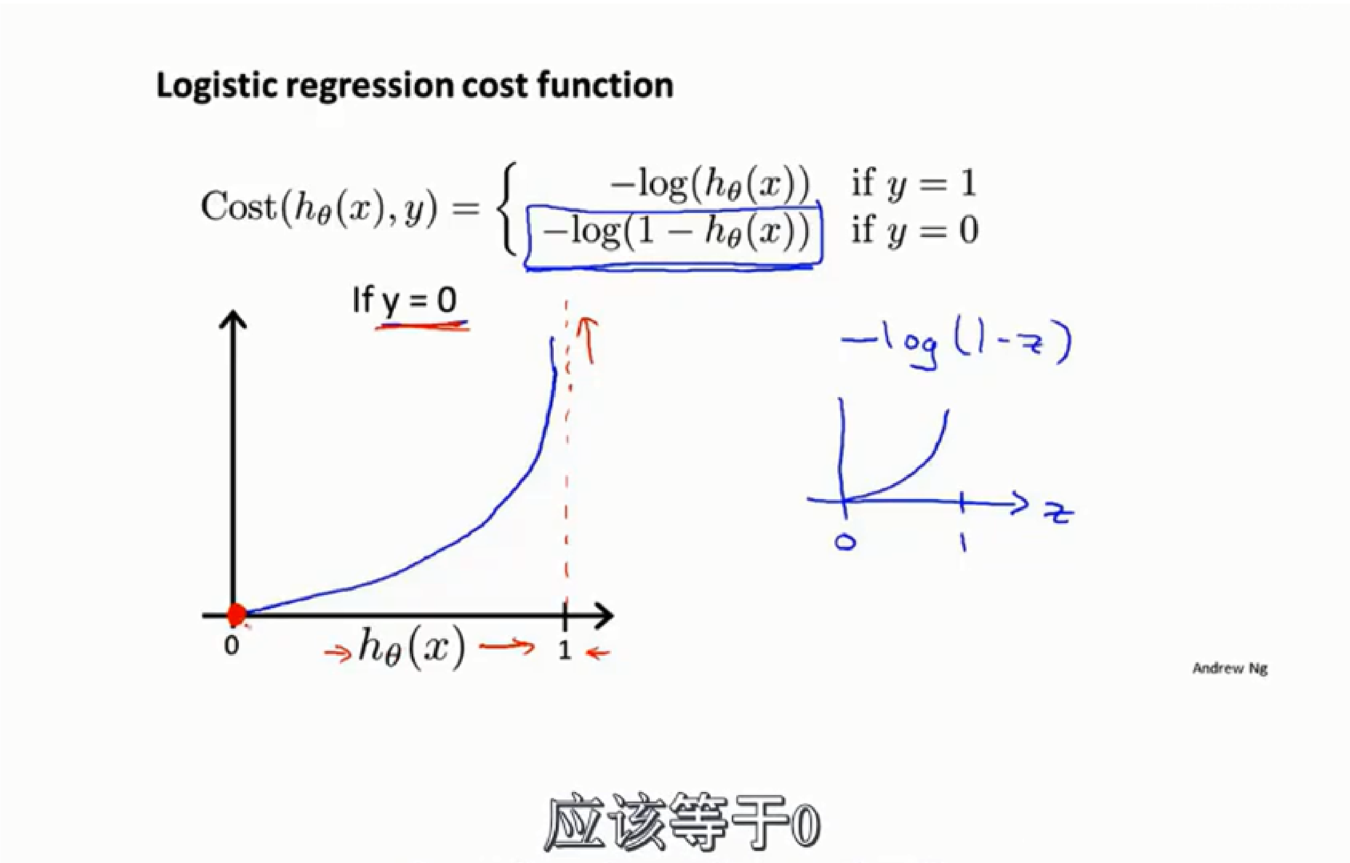
因此，我们来看看具体正式的凸函数的代价函数： 那么对数函数恰好是这样一种我们想要的函数，第一他是凸函数，第二，他经过一定的变换以后，可以在（0,1）这个区间，给到我们一个非常想要变换区间。 首先，我们来理解对数函数的样子：



仔细观察对数函数取负数以后-log(X)在（0,1）区间的图像，那么实际上，他的 图像的趋势是，在X取趋近0的时候，那么Y也是趋势正无穷的，当X取趋近正无穷的时候，Y的趋势是趋近于0的，恰巧是相反的。理解这个函数的特性将非常有好处。

为何要去关注0到1这个区间的取值呢，因为，上图的上图告诉我们，hθ（x)是一个sigmod函数，这个函数的取值，就只能在（0,1)区间，所以我们最终的代价函数，取值也重点关注这个区间。





继续看这个函数成为我们的代价函数以后的形态，非常有意思。

- 假如我们的代价函数为0，表示我们完美的拟合了数据，对吧。那么，这时候条件是hθ（x)必须为1，而且这是真实值y也是为1的。这说明预测结果和真实结果是一致的。
- 但是如果我们的hθ（x)=0的时候，但是y的真实值是0，那么就蛋疼了，代价函数的结果会区域正无穷，也就是说，代价函数值很大，拟合效果非常差。
- 相反，当预测结果为0，真实结果也为0的时候，上图同样得到的结果是-log(1),实际代价函数结果仍然为0，那么拟合效果也非常棒（上面组图的第二张图已经明确了该意义）。

5、简化代价函数与梯度下降

5.1 简化代价函数

首先来看之前定义好的代价函数：

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

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

怎么来将上述的表达式合并为一个表达式呢，答案如下：

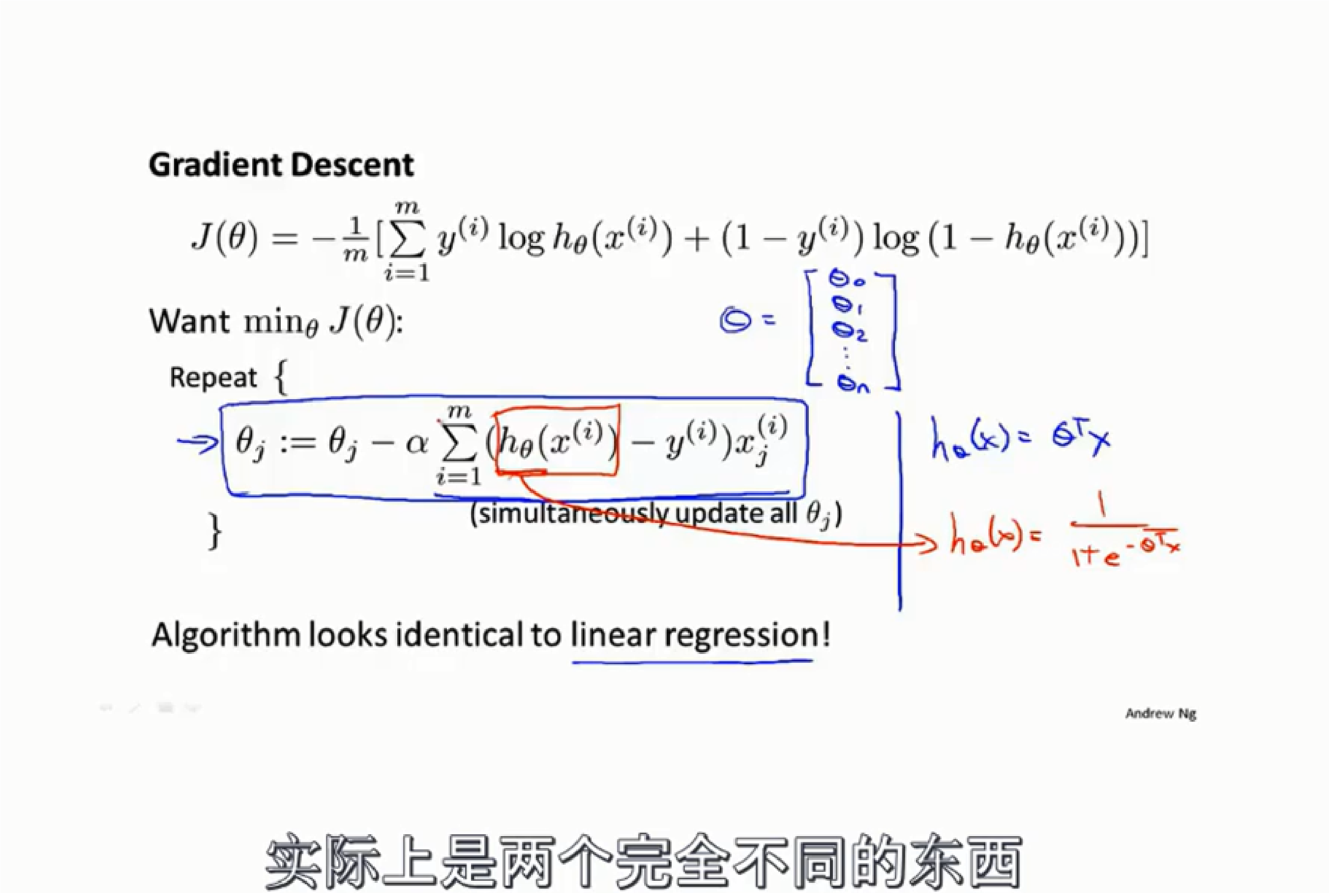
$$J(\Theta) = \frac{1}{m}[\sum_{m}^{1} -ylog(h_\Theta x) - (1-y)log(1-h_\Theta x)] \rightarrow\rightarrow\rightarrow\rightarrow$$

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

- 首先来看，1/m里面的中括号部分
- 假设当y= 1的时候，那么里面的部分实际上就是-ylog(hθ(x))
- 假设当y= 0的时候，那么里面的部分实际上就是-log(1-hθ(x))
- 这样，我们就完美的将这个代价函数的表示方法合二为一了，唉，统计学家们真是天才。
- 红色部分就是我们最终向量化的代价函数最终结果。
- 这个公式是从统计学中的极大似然法的出来的结果，他可以为不同的统计模型快速寻找参数。
- 这是一个凸函数，那么也保证了可以使用梯度下降的方法进行优化。

5.2 如何对该函数运行梯度下降

还是一张图就可以解释多个方向的东西：



看上图，我们会得到几个很重要的信息：

- 梯度下降法的算法仍然是一样的，如果有n个特征，那么就会同样的产生n个参数，组合成一个向量就是[θ1，θ2，θ3，θ4，......，θn]。
- 我们需要同步更新上面的所有的θ参数，而不是一个个更新，可以使用for循环，向量化的手段同步更新这些参数。
- 注意观察，这个梯度下降中，实际上是需要对我们上面所述的代价函数求偏导数，求出来的偏导数居然和线性回归的偏导数看起来是同一个东西。
- 但是，实际上，由于hθ(x)变化了，原来是θ^T*X，现在是sigmod函数，因此实际上，他们是有很大区别的。

最后注意一点，特征缩放对Logistic回归同样重要，那么，这么来看，任何算法，先做一个特征缩放，肯定是有必要的。 理解这里的参数值非常多的情况，将对后来神经网络的学习，更加容易。

6、高级优化

吴恩达在这里介绍了基于牛顿法和拟牛顿法的相关算法，这些算法都比较复杂，看着就很蛋疼，涉及到许多矩阵的变换，反正我还没有学会。 BGFS、LBFGS无约束优化类问题文章：
https://www.cnblogs.com/ljy2013/p/5129294.html
https://www.cnblogs.com/shixiangwan/p/7532830.html

他们的优点非常的明显，就是，这些算法都能够比梯度下降方法更快的收敛（快很多），同时，不需要设置学习率，他们自己的算法就可以自动搜索最好的学习率。

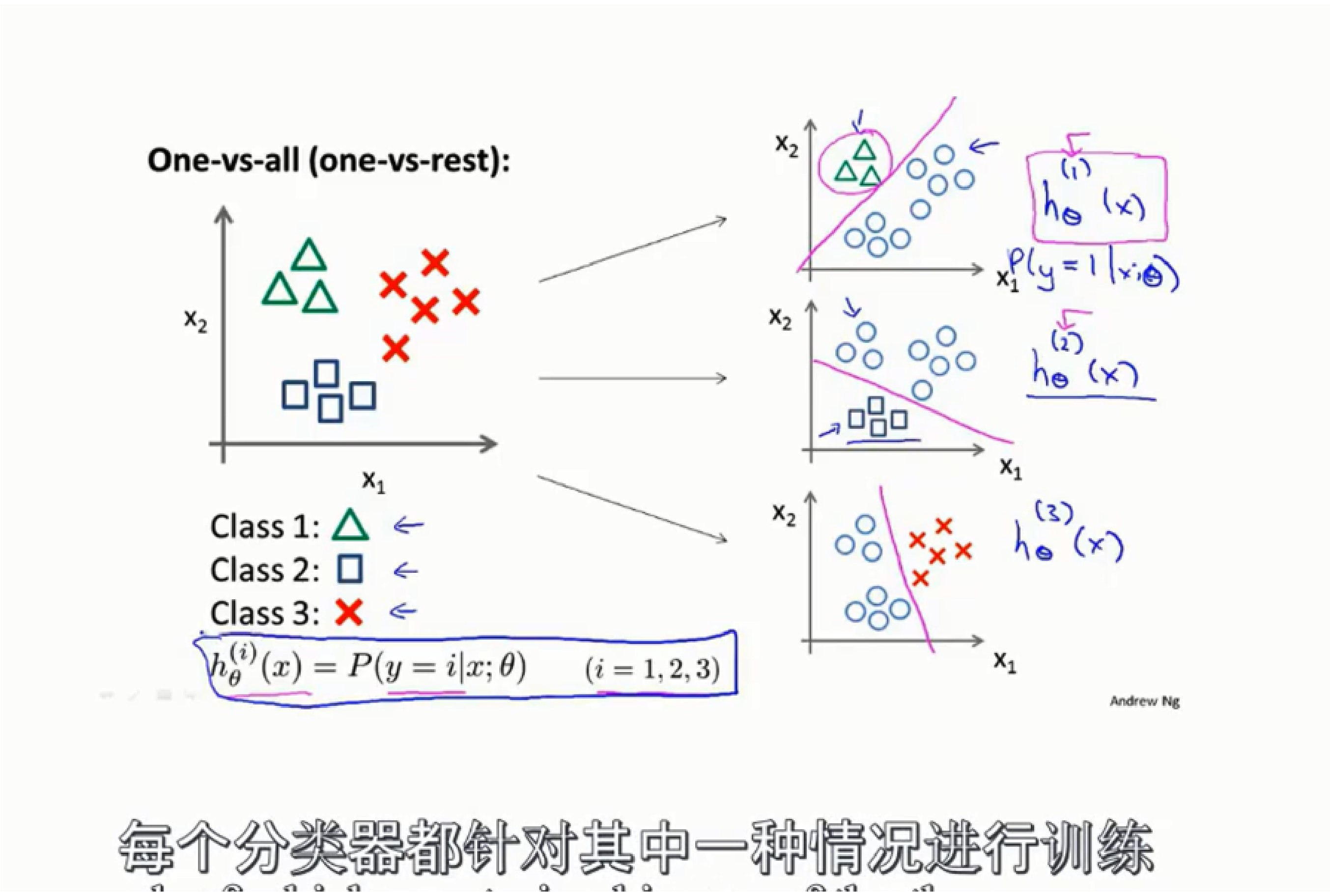
实际上，sklearn上是直接调包就可以得出想要的结果，这些原理并非需要像学习代价函数那些那么迫切，但是，梯度下降至少是必须要懂得的。

吴大佬具体演示的代码我这里就不做笔记了。

7、多元分类：一对多

使用Logistic回归处理多分类问题，原理如下：

- 假设为需要分类的类别是三个类别
- 第一步是将其分为三个独立的二分类问题
- 将其中一个类别划分为正类，另外两个类别一切归纳为负类，开始训练分类器1
- 将其中第二个类别划分为正类，另外两个类别一切归纳为负类，开始训练分类器2
-   - 将其中第三个类别划分为正类，另外两个类别一切归纳为负类，开始训练分类器3
- 得到三个分类器，每个分类器都对其中一个情况开始训练，然后会得到一个概率值
- 每一个分类器都会计算一个判定结果，就是P(y=1|Xi(θ))的概率
- 预测阶段，对任何一个样本，看他在三个分类器中，谁的预测结果概率取值更高，那么，我们就判定他属于最高概率的分类器中的正类。



这就是解决多分类问题的终极奥义。