**线性回归 linear regression**

  通过现实生活中的例子，可以帮助理解和体会线性回归。比如某日，某屌丝同事说买了房子，那一般大家关心的就是房子在哪，哪个小区，多少钱一平方这些信息，因为我们知道，这些信息是"关键信息”（机器学习里的黑话叫“feature”）。那假设现在要你来评估一套二手房的价格(或者更直接点，你就是一个卖房子的黑中介，嘿嘿)，如果你对房价一无所知（比如说房子是在非洲），那你肯定估算不准，最好就能提供同小区其他房子的报价；没有的话，旁边小区也行；再没有的话，所在区的房子均价也行；还是没有的话，所在城市房子均价也行（在北京有套房和在余杭有套房能一样么），因为你知道，这些信息是有“参考价值”的。其次，估算的时候我们肯定希望提供的信息能尽量详细，因为我们知道房子的朝向，装修好坏，位置（靠近马路还是小区中心）是会影响房子价格的。

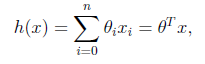
   其实我们人脑在估算的过程，就类似一个“机器学习”的过程。

a)首先我们需要“训练数据”，也就是相关的房价数据，当然，数据太少肯定不行，要尽量丰富。有了这些数据，人脑可以“学习”出房价的一个大体情况。因为我们知道同一小区的同一户型，一般价格是差不多的（特征相近，目标值－房价也是相近的，不然就没法预测了）；房价我们一般按平方算，平方数和房价有“近似”线性的关系。

b)而“训练数据”里面要有啥信息？只给你房子照片肯定不行，肯定是要小区地点，房子大小等等这些关键“特征”

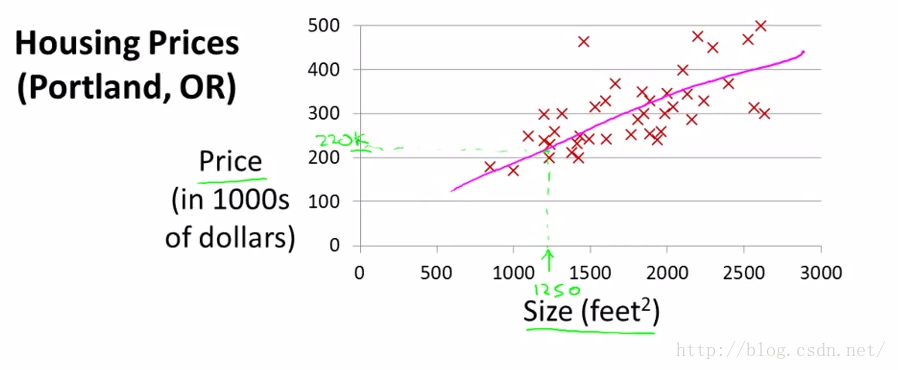
c)一般我们人肉估算的时候，比较随意，也就估个大概，不会算到小数点后几位；而估算的时候，我们会参照现有数据，不会让估算跟“训练数据”差得离谱（也就是下面要讲的让损失函数尽量小），不然还要“训练数据”干嘛。 计算机擅长处理数值计算，把房价估算问题完全可以用数学方法来做。把这里的“人肉估算”数学形式化，也就是“线性回归”。

1.我们定义线性回归函数（linear regression）为:

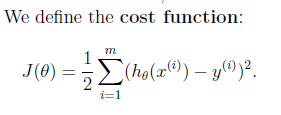


然后用h(x) 来预测y

最简单的例子，一个特征size，y是Price，把训练数据画在图上，如下图。（举最简单的例子只是帮助理解，当特征只有一维的时候，画出来是一条直线，多维的时候就是超平面了）



2.目标就是画一条直线尽量靠近这些点，用数学语言来描述就是cost function尽量小。



3 然后就是怎么求解了。如果h(x)=y那就是初中时候的多元一次方程组了，现在不是，所以要用高端一点的方法。

notes 1里面介绍3种方法

1.gradient descent （梯度下降）

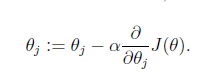
a.batch gradient descent

b.stochastic gradient descent （上面的变形）

2.the normal equations

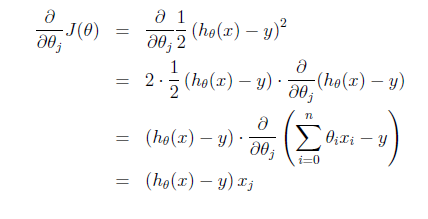
3.Newton method(Fisher scoring)

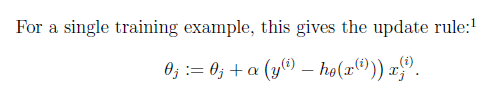
**1.gradient descent algorithm**



α is called the learning rate.

只有一个训练数据的例子：



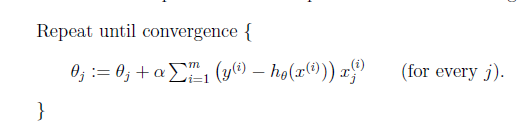


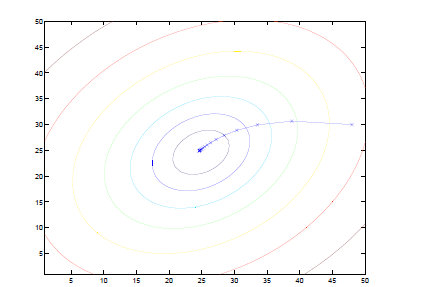
这个式子直观上也很好理解，假设xj是正数，y比预测值h(x)大的话，我们要加大θ，所以α前面是+号（当xj是负数同理）

上面的式子在具体更新的时候有小的不同

方法 a.batch gradient descent

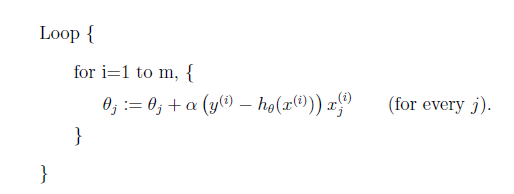
注意，要同时进行更新。因为更新θ(j+1)的时候要用hθ(x)，这里的hθ(x)用的还是老的θ1 到 θj。





直观上看，等高线代表cost function的值，横纵坐标是θ1 θ2两个参数，梯度下降就是每次一小步沿着垂直等高线的方向往等高线低（图的中心）的地方走。显然步子不能太大，不然容易扯着蛋（跨一大步之后反而到了更高的点）

方法 b.stochastic gradient descent (also incremental gradient descent)



这两种方法看公式可能不好理解，看后面的代码实现就容易区分。

**2.the normal equations。**

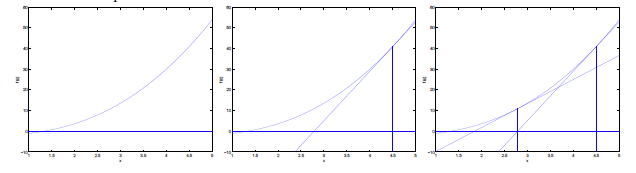
初学者可以先跳过推导过程（不是鼓励不看。），直接先记住结论http://img.blog.csdn.net/20160506213825368。

**3.牛顿法**

Another algorithm for maximizing ℓ(θ)

Returning to logistic regression with g(z) being the sigmoid function, lets now talk about a different algorithm for minimizing -ℓ(θ)。

牛顿法求函数0点，即 f (Θ) = 0



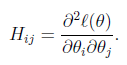
http://img.blog.csdn.net/20160506213937491这样迭代就行，f′(θ)是斜率，从图上看，就是“用三角形去拟合曲线，找0点”

因为我们是要求导数等于0，把上面的式子替换一下f(θ) = ℓ′(θ)

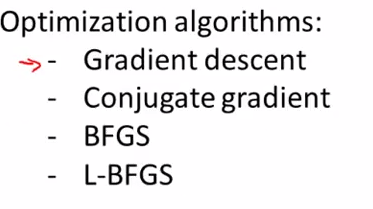
http://img.blog.csdn.net/20160506214013480

θ是多维的，有

http://img.blog.csdn.net/20160506214029319

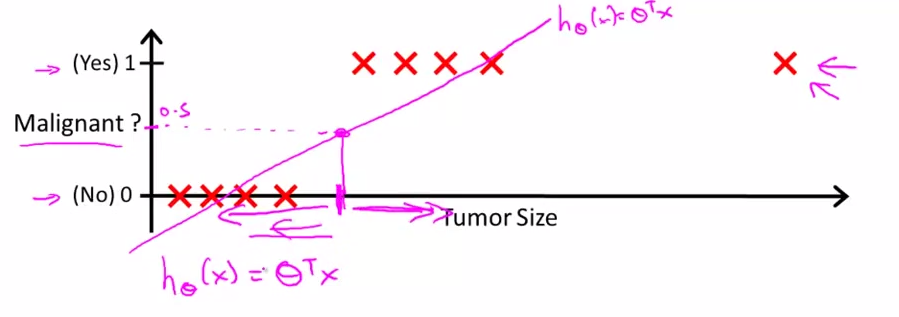
where 

When Newton’s method is applied to maximize the logistic regression log likelihood function ℓ(θ), the resulting method is also called **Fisher scoring.**

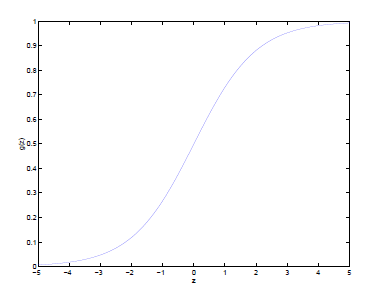


**逻辑回归logistic regression**

现在如果有一个0和1的2分类问题的，套进去线性回归去解，如下图



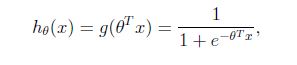
离群点会对结果影响很大，比如上图（我们以h(x)>0.5时预测y=1，一个离群点让直线大旋转，一下子把不少点误分类了）



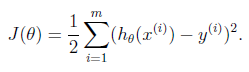
换成这个曲线就好多了，这个函数是sigmoid function

http://img.blog.csdn.net/20160506214233045g(z) 值域 (0,1)

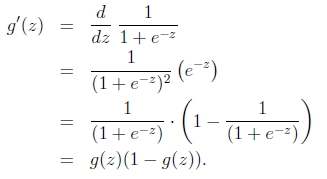
把线性回归套进去g（z）就是

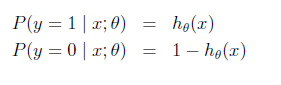


**建模好，还缺一个cost function，是不是跟linear regression一样求平方差就行？**

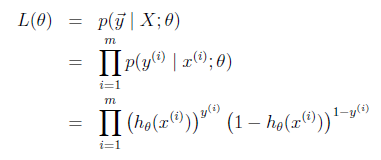
呵呵，不是，coursera课程给出的原因是套入sigmoid后，这个函数不是凸函数，不好求解了。但其实h(x) -y 算出来是一个概率，多个训练数据的概率相加是没意义的，得相乘。p(x,y) = p(x)\* p(y)。

**先讲一个useful property**

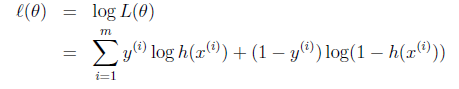


=> http://img.blog.csdn.net/20160506214549149

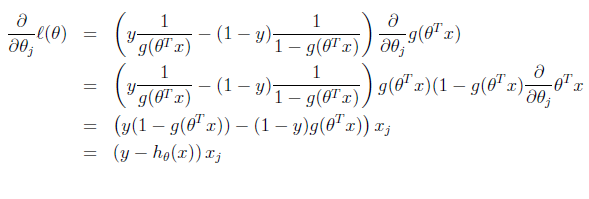
likelihood of the parameters:



log likelihood:



一个训练数据的时候（代入前面的结论http://img.blog.csdn.net/20160506214614409)



http://img.blog.csdn.net/20160506214650197

http://img.blog.csdn.net/20160506214659556