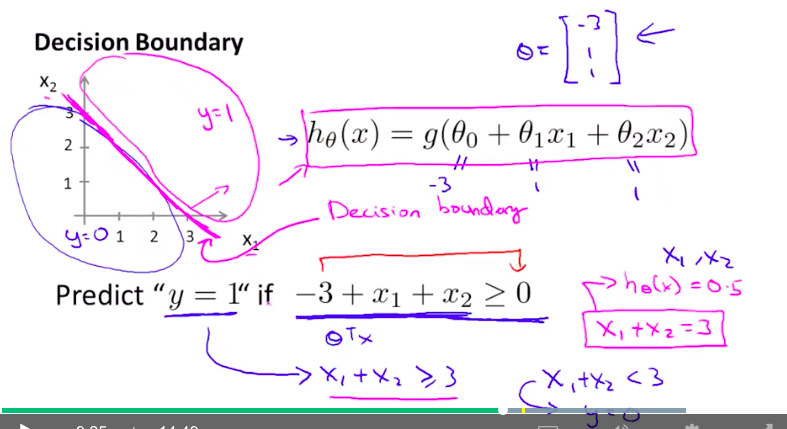
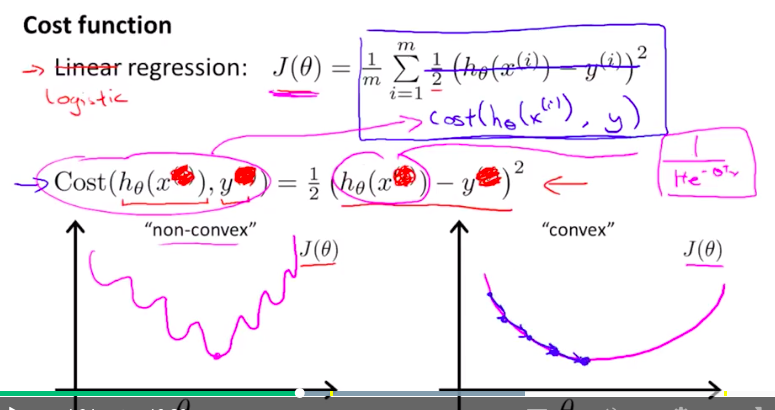
* LR拟合的过程和原理，包括cost function，h(θ)
* 除了Gradient Descent外，还介绍了别的高级拟合算法，如BFGS
* 防止Overfitting，underfitting的方法
* Regularization

**LR拟合的过程和原理，包括cost function，h(θ)**

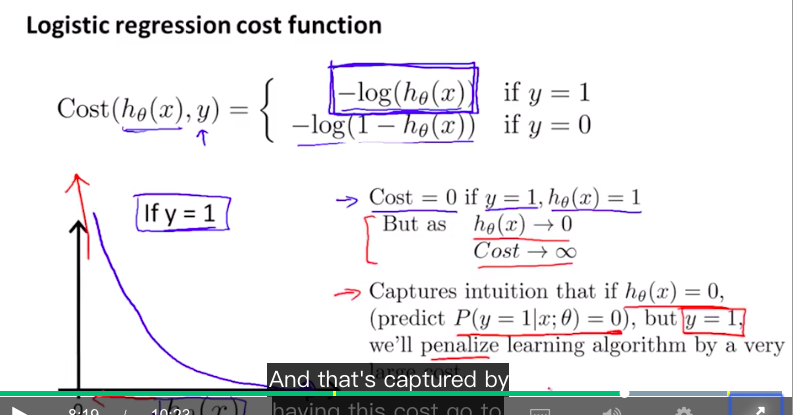
h(θ）>=0.5，既Decision Boundary 决策边界>=0



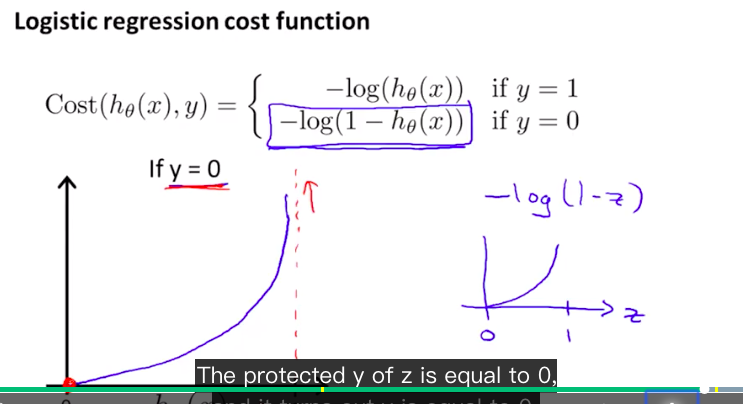
Non-convex非凸函数，找不到global minimum



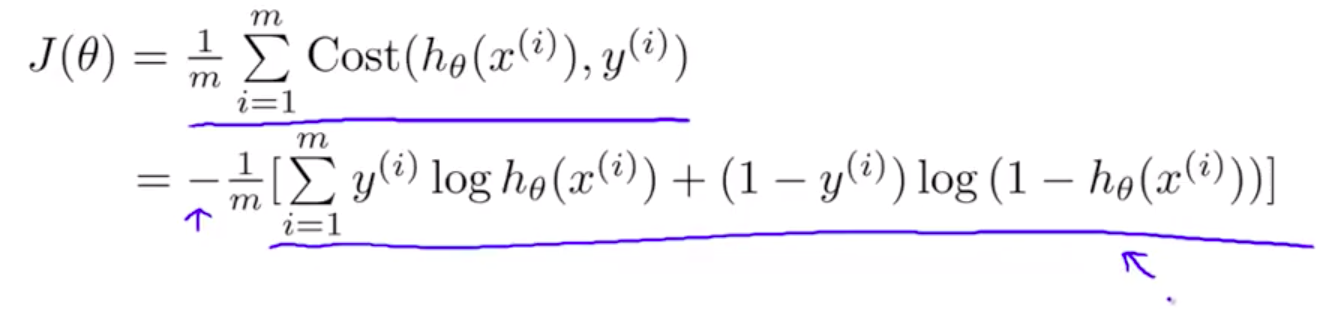
以下这种cost function保证逻辑回归的成本函数是凸函数，既可导，且找到全局最优。

当h(θ)=1, 实际y=1，那cost为0。如果h(θ)=0，但y=1，既预测上y=1的概率为0，那我们会对学习算法进行很大的成本惩罚，既cost趋向于正无穷。

**反之亦然**

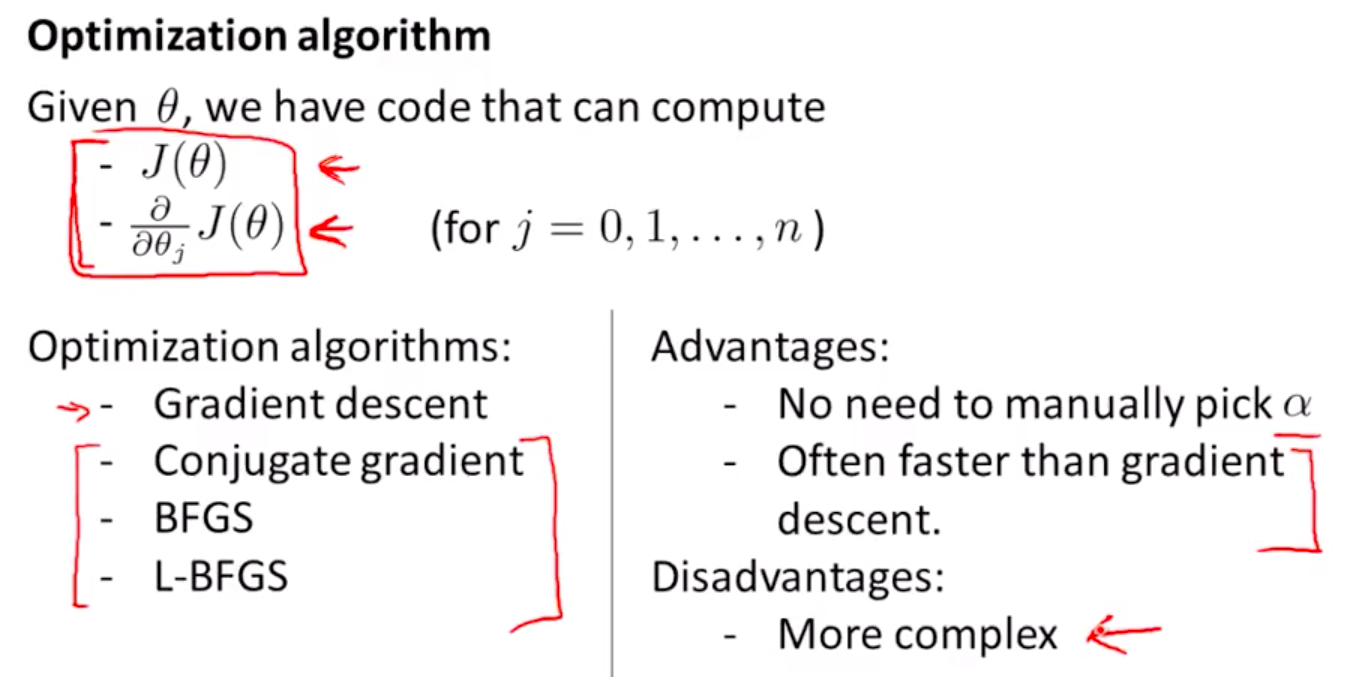


把分段函数整合成一个

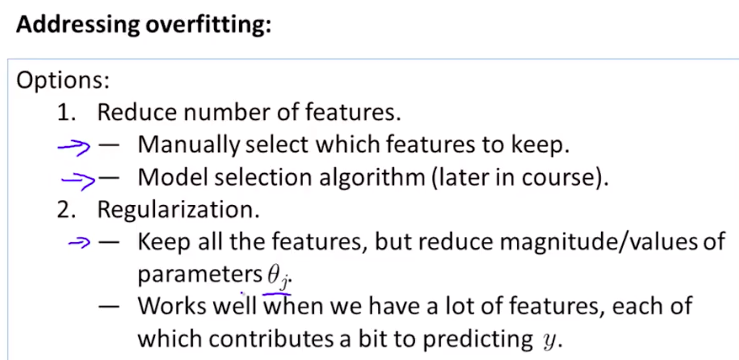


除了gradient descent外的优化算法，一般比gradient descent运算更快，更加适合大的数据集

**除了Gradient Descent外，还介绍了别的高级拟合算法，如BFGS**

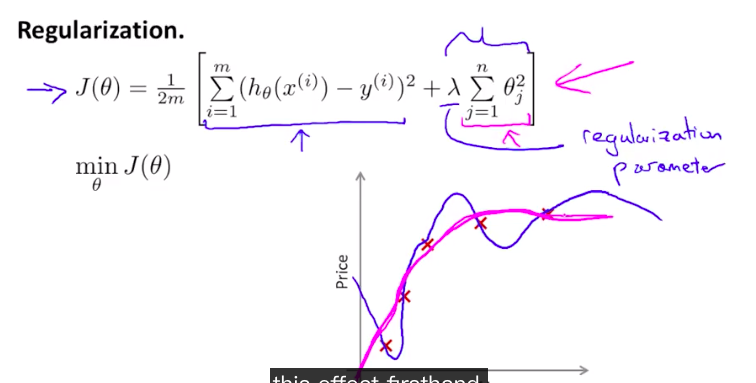


**防止Overfitting，underfitting的方法**

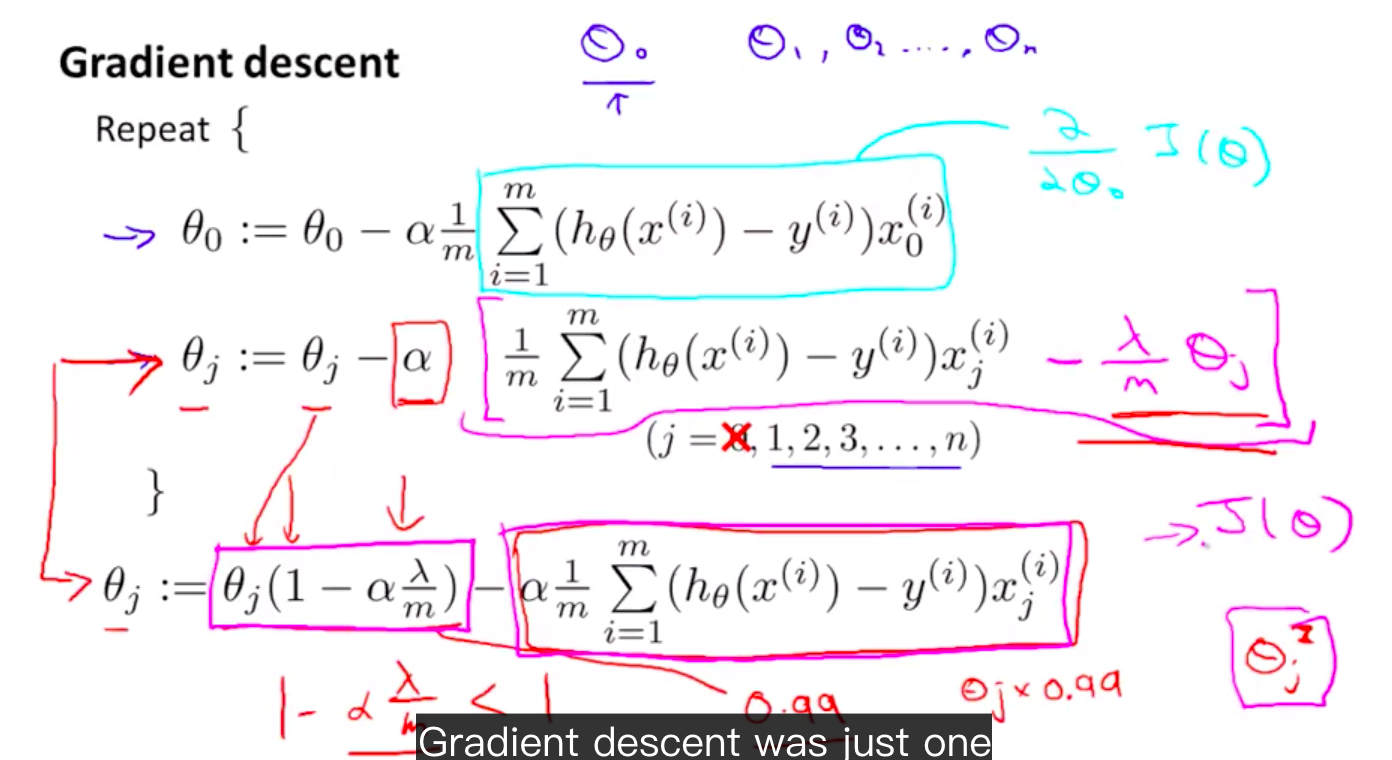


**Regularization**

实际上regularization是在原来loss function or cost function的基础上加上惩罚函数，实现两者的trade off，从中抑制了theta的大小。尤其随着λ越大，θ则会越小，试想如果λ=10^6，θ则几乎为0。



加上regularization后的cost function的gradient descent



Regularization还可解决不可逆的问题

Recall that if m < n, then [](https://www.codecogs.com/eqnedit.php?latex=XX%5ET%0)is non-invertible. However, when we add the term λ, then

[](https://www.codecogs.com/eqnedit.php?latex=XX%5ET%0)*+* λ⋅L becomes invertible.

