面试问题

、对于最优化的问题 $\min\limits_w f(w)$.

L1 正则化 $\min\limits_w L_1(w) = \min\limits_w f(w) + \frac{\lambda}{n}\sum_{i=1}^{n}|w_i|$

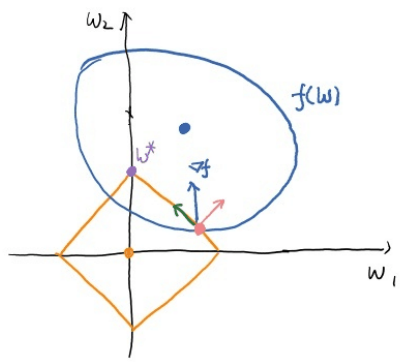
L2 正则化 $\min\limits_w L_2(w) = \min\limits_w f(w) + \frac{\lambda}{2n}\sum_{i=1}^{n} w_i^2$

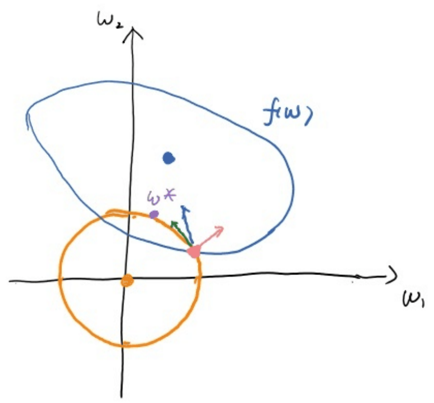
* L1 比 L2 更容易获得 sparse 的 w. L2 比 L1 更容易获得 smooth 的 w.

① 从优化问题视角来看.

$\min\limits_w L_1(w) \Leftrightarrow \min\limits_w f(w)$ s.t. $\sum_{i=1}^{n}|w_i| < C_1$

$\min\limits_w L_2(w) \Leftrightarrow \min\limits_w f(w)$ s.t. $\sum_{i=1}^{n} w_i^2 < C_2$





* 如图情况, w^* 一定在 ◇ 边缘上.
* 在 • 时, ↑ 表示 f(w) 的梯度方向. ↑ 表示脱离 ◇ 的方向. 由于不可以脱离 ◇, 因此 ↑ 只会产生 ↑ 的分量, 将 w 往靠近 ◇ 顶点方向引导.
* L1 使 w^* 更容易落在顶点上, 那就是 w 为 0. 因此获得更 sparse 的 w

* 如图情况, w^* 一定在 ○ 边缘上
* 在 • 时, 同左, ↑ 只会产生 ↑ 的分量, 将 w 在 ○ 上移动.
* 当移动到 • 时获得 w^*. 这样的 w 的每个值都比较小, 但却不向 L1 那样 sparse. 因此获得更 smooth 的 w

② 从梯度视角来看

$$\frac{\partial L_1(w)}{\partial w_i} = \frac{\partial f(w)}{\partial w_i} + \frac{\lambda}{n}\,\mathrm{sign}(w_i)$$

$$w_i' = w_i - \eta\frac{\partial L_1(w)}{\partial w_i}$$

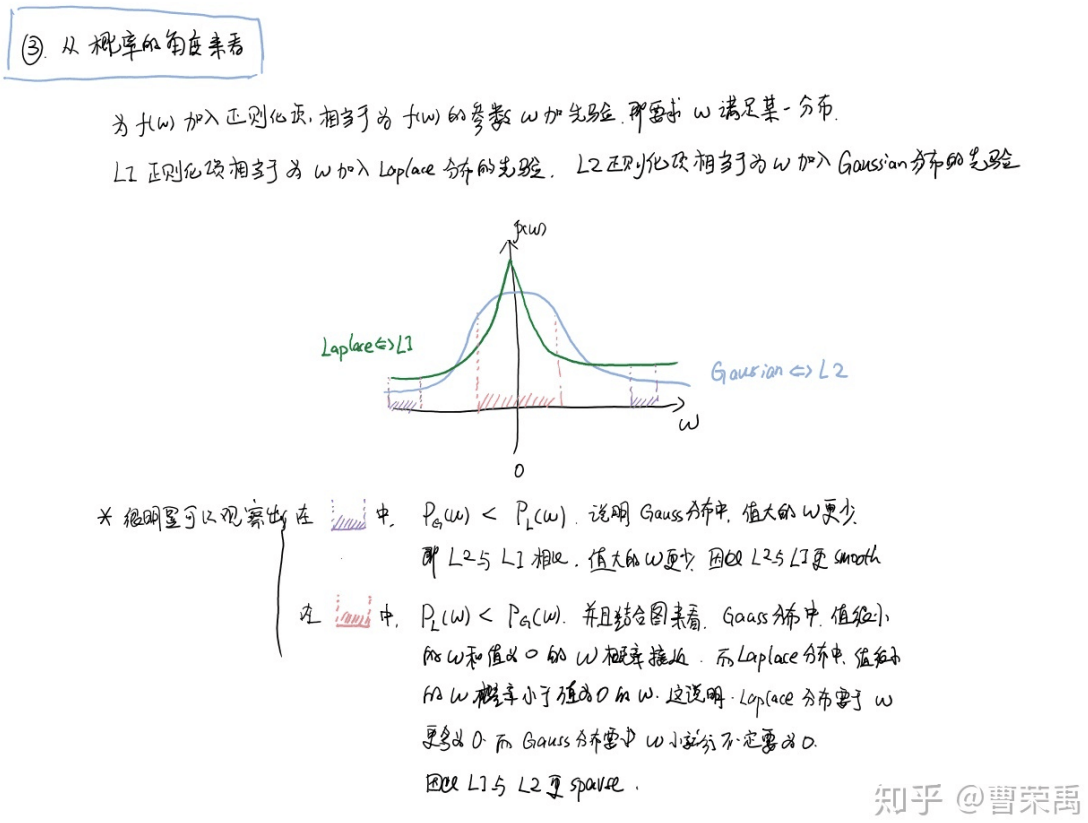
$$w_i' = w_i - \eta\frac{f(w)}{\partial w_i} - \boxed{\eta\frac{\lambda}{n}\,\mathrm{sign}(w_i)}$$

$$\frac{\partial L_2(w)}{\partial w_i} = \frac{\partial f(w)}{\partial w_i} + \frac{\lambda}{n} w_i$$

$$w_i' = w_i - \eta\frac{\partial L_2(w)}{\partial w_i}$$

$$w_i' = w_i - \eta\frac{f(w)}{\partial w_i} - \boxed{\eta\frac{\lambda}{n} w_i}$$

* 从 L1 和 L2 的 ▭ 来看. L1 与 L2 不一样的地方在于 L1 会减 $\mathrm{sign}(w_i)$ 倍的 $\eta\frac{\lambda}{n}$ 而 L2 会减 w_i 倍的 $\eta\frac{\lambda}{n}$
当 w_i 在 $[1,+\infty)$ 时. L2 获得比 L1 更快的减小速率. 当 w_i 在 $(0,1)$ 时, L1 比 L2 获得更快的减小速率.
并且当 w 越小时. L1 更容易减小接近于 0. 而 L2 更不容易变化. 因此 L1 会获得更多的接近于 0 的 w.
即 L1 比 L2 更容易获得 sparse 的 w.



1.L1和L2区别

L1范数（L1　norm）是指向量中各个元素绝对值之和，也有个美称叫“稀疏规则算子”（Lasso　regularization）。
比如　向量A=[1，-1，3]，　那么A的L1范数为　|1|+|-1|+|3|.

简单总结一下就是：
L1范数:　为x向量各个元素绝对值之和。
L2范数:　为x向量各个元素平方和的1/2次方，L2范数又称Euclidean范数或者Frobenius范数
Lp范数:　为x向量各个元素绝对值p次方和的1/p次方.

在支持向量机学习过程中，L1范数实际是一种对于成本函数求解最优的过程，因此，L1范数正则化通过向成本函数中添加L1范数，使得学习得到的结果满足稀疏化，从而方便人类提取特征，**即L1范数可以使权值稀疏，方便特征提取。**
L2范数可以防止过拟合，提升模型的泛化能力。

L1和L2的差别，为什么一个让绝对值最小，一个让平方最小，会有那么大的差别呢？看导数一个是1一个是w便知，　在靠进零附近，　L1以匀速下降到零，　而L2则完全停下来了.　这说明L1是将不重要的特征(或者说，　重要性不在一个数量级上)尽快剔除，　L2则是把特征贡献尽量压缩最小但不至于为零.　两者一起作用，　就是把重要性在一个数量级(重要性最高的)的那些特征一起平等共事(简言之，　不养闲人也不要超人)。

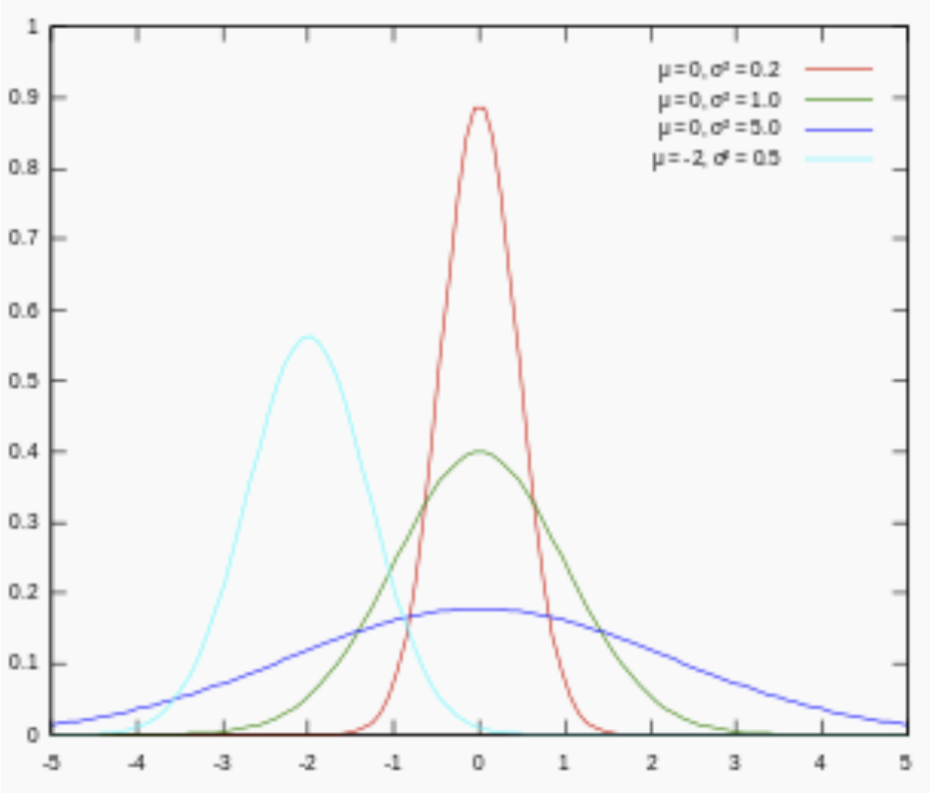
2.L1和L2正则先验分别服从什么分布?

L1是拉普拉斯分布，L2是高斯分布。

先验就是优化的起跑线，有先验的好处就是可以在较小的数据集中有良好的泛化性能，当然这是在先验分布是接近真实分布的情况下得到的。从信息论角度看，向系统加入了正确先验这个信息，肯定会提高系统的性能。

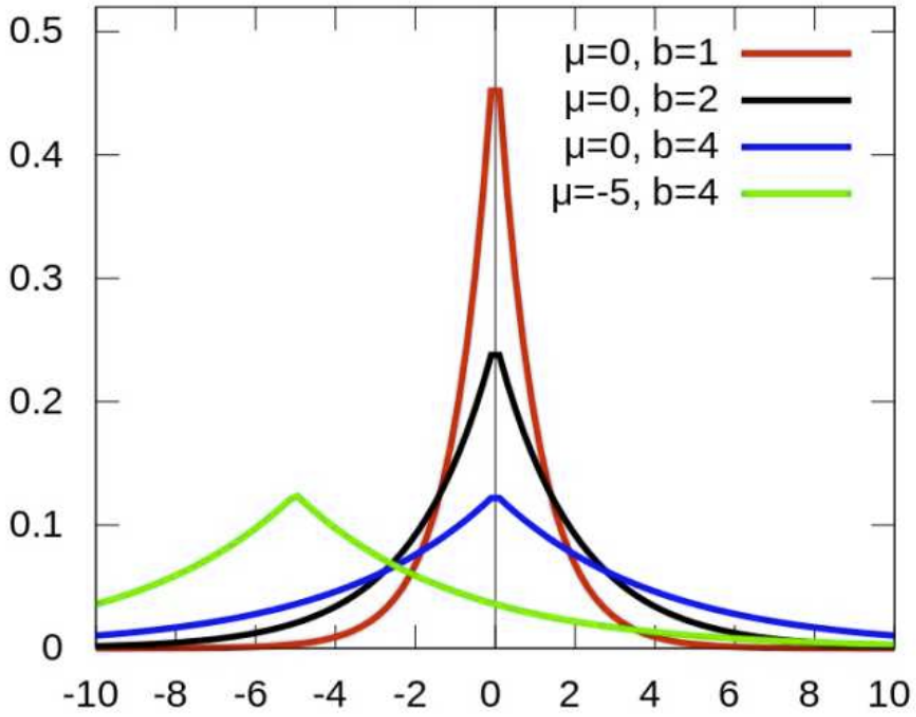
对参数引入高斯正太先验分布相当于12正则化:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



对参数引入拉普拉斯先验等价于11正则化，如下图:

$$f(x \mid \mu, b) = \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right)$$



从上面两图可以看出，12先验趋向零周围，11先验趋向零本身。

一般的目标函数都包含下面两项：

$$Obj(\Theta) = L(\Theta) + \Omega(\Theta)$$

误差函数： 我们的模型有多拟合数据。

正则化项：惩罚复杂模型

其中，误差/损失函数鼓励我们的模型尽量去拟合训练数据，使得最后的模型会有比较小的bias，正则化项鼓励更加简单的模型。因为当模型简单之后，有限数据拟合出来结果的随机性比较小，不容易过拟合，使得最后模型的预测更加稳定。

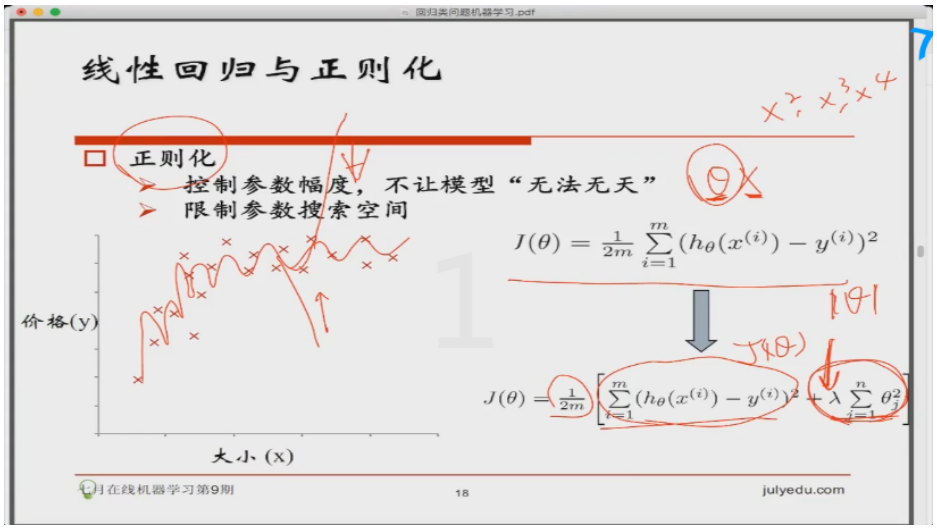
1.过拟合

过拟合表现在训练数据上的误差非常小，而在测试数据上误差反而增大。其原因一般是模型过于复杂，过分得去拟合数据的噪声和outliers。

为什么正则化可以减少过拟合？

几点解释：（意思都差不多）

- 正则化则是对模型参数添加先验，使得模型复杂度较小，对于噪声以及outliers的输入扰动相对较小。
- 过拟合的时候，拟合函数的系数往往非常大，而正则化是通过约束参数的范数使其不要太大，所以可以在一定程度上减少过拟合情况。
- 最简单的解释就是加了先验。在数据少的时候，先验知识可以防止过拟合。比如最小二乘回归问题：加2范数正则等价于加了高斯分布的先验，加1范数正则相当于加拉普拉斯分布先验。（贝叶斯学派观点）
- 正则化削减了（容易过拟合的那部分）假设空间，从而降低过拟合风险



2.L1正则化和L2正则化

L1正则化和L2正则化可以看做是损失函数的惩罚项，所谓“惩罚”是指对损失函数中的某些参数做一些限制。

对于线性回归模型，使用L1正则化的模型叫做Lasso回归，使用L2正则化的模型叫做Ridge回归（岭回归）。

Lasso回归的损失函数：

$$\min_w \frac{1}{2n_{samples}}||Xw - y||_2^2 + \alpha||w||_1$$

岭回归的损失函数：

$$\min_w ||Xw - y||_2{}^2 + \alpha||w||_2{}^2$$

一般回归分析中w表示特征的系数，从上式可以看到正则化项是对系数做了处理（限制）。L1正则化和L2正则化的说明如下：

- L1正则化是指权值向量w中各个元素的绝对值之和，通常表示为||w||1
- L2正则化是指权值向量w中各个元素的平方和然后再求平方根，通常表示为||w||2

L1和L2正则化的作用：

- L1正则化可以产生稀疏权值矩阵，即产生一个稀疏模型，可以用于特征选择
- L2正则化可以防止模型过拟合，一定程度上，L1也可以防止过拟合

2.1 稀疏模型与特征选择

为什么要生成一个稀疏矩阵？

稀疏矩阵指的是很多元素为0，只有少数元素是非零值的矩阵，即得到的线性回归模型的大部分系数都是0。通常机器学习中特征数量很多，难以选择，但是如果代入这些特征得到的是一个稀疏模型，表示只有少数特征对这个模型有贡献，绝大部分特征是没有的或者贡献微小

（因为它们前面的系数是0或者是很小的值，即使去掉对模型也没有什么影响），此时我们就可以只关注系数是非零值的特征。这就是稀疏模型与特征选择的关系。

2.2 L1和L2正则化的直观理解

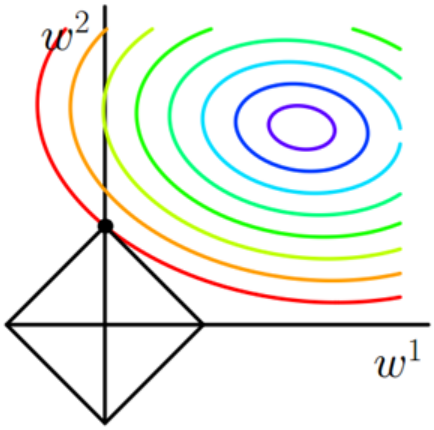
为什么L1正则化可以产生稀疏模型（L1是怎么让系数等于0的），为什么L2正则化可以防止过拟合？

2.2.1 L1正则化和特征选择

假如有如下带L1正则化的损失函数

$$J = J_0 + \alpha \sum_w |w| \tag{1}$$

其中J0是原始的损失函数，阿尔法是正则化系数。J是带有绝对值符号的函数，因此J是不完全可微的。机器学习的任务就是要通过一些方法（比如梯度下降）求出损失函数的最小值。当我们在J0后添加L1正则化项时，相当于对J0做了一个约束。令L= α ∑w|w|，则J=J0+L，此时我们的任务变成**在L约束下求出J0取最小值的解**。考虑二维的情况，即只有两个权值w1和w2，此时L=|w1|+|w2|，对于梯度下降法，求解J0的过程可以画出等值线，同时L1正则化的函数L也可以在w1w2的二维平面上画出来。一个圈代表一个目标函数值，圆心就是样本观测值（假设一个样本），半径就是误差值。



重点解释该图：

图中等值线是J_0的等值线，黑色方形是L函数的图形。在图中，当J_0等值线与L图形首次相交的地方就是最优解。上图中J_0与L在L的一个顶点处相交，这个顶点就是最优解。注意到这个顶点的值是$(w^1, w^2) = (0, w)$。可以直观想象，因为L函数有很多『突出的角』（二维情况下四个，多维情况下更多），J_0与这些角接触的机率会远大于与L其它部位接触的机率，而在这些角上，会有很多权值等于0，这就是为什么L1正则化可以产生稀疏模型，进而可以用于特征选择。

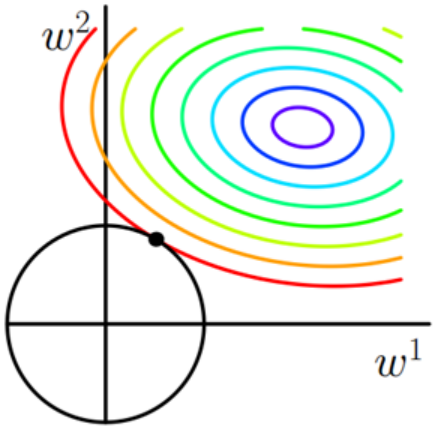
也就是，最优参数只可能在坐标轴上，所以就会出现0权重参数，使得模型稀疏。

α 越大，对原始模型中特征系数的惩罚作用（降低过拟合作用）越明显，黑色方框越小，此时才会使得最优解中的非零值越小，也就是起到了很好的惩罚作用；相反 α 越小，惩罚作用越不明显，黑色方框越大，最优解越大。其他方面具体大小根据具体需要分析。

类似，假设有如下带L2正则化的损失函数：

$$J = J_0 + \alpha \sum_w w^2 \tag{2}$$

同样可以画出他们在二维平面上的图像，如下：



二维平面下L2正则化的函数图形是个圆，与方形相比，被磨去了棱角。因此J0与J相交时使得w1或w2等于零的几率小了很多，所以L2正则化不具有稀疏性。

2.2.2 L2正则化和过拟合

拟合过程中通常都倾向于让权值尽可能小，最后构造一个所有参数都比较小的模型。因为一般认为参数值小的模型比较简单，能适应不同的数据集，也在一定程度上避免了过拟合。可以设想一下对于一个线性回归方程，若参数很大，那么只要数据偏移一点点，就会对结果造成很大的影响；但如果参数足够小，数据偏移得多一点也不会对结果造成什么影响，专业一点的说法是『抗扰动能力强』。

为什么L2正则化可以获得值很小的参数？？

以线性回归中的梯度下降法为例，假设要求的参数为θ，$h_\theta(x)$是我们的假设函数，那么线性回归的代价函数如下：

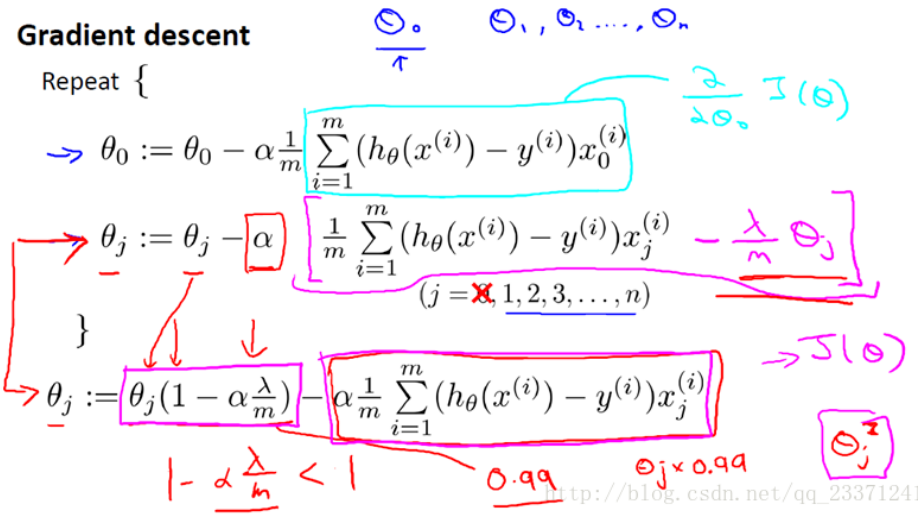
$$J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})^2 \qquad (3)$$

那么在梯度下降法中，最终用于迭代计算参数θ的迭代式为：

$$\theta_j:=\theta_j-\alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_j^{(i)} \qquad (4)$$

阿尔法是学习率。如果加入12正则化项，损失函数和迭代公式变成如下：

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



其中λ就是正则化参数。从上式可以看到，与未添加L2正则化的迭代公式相比，每一次迭代，θj都要先乘以一个小于1的因子，从而使得θj不断减小，（权重衰减：weigth decay的由来），当然考虑到后面的导数项，w最终的值可能增大也可能减小。

到目前为止，我们只是解释了L2正则化项有让w“变小”的效果，但是还没解释为什么w“变小”可以防止overfitting？一个所谓“显而易见”的解释就是：更小的权值w，从某种意义上说，表示网络的复杂度更低，对数据的拟合刚刚好（这个法则也叫做奥卡姆剃刀），而在实际应用中，也验证了这一点，L2正则化的效果往往好于未经正则化的效果。

2.3 正则化参数的选择

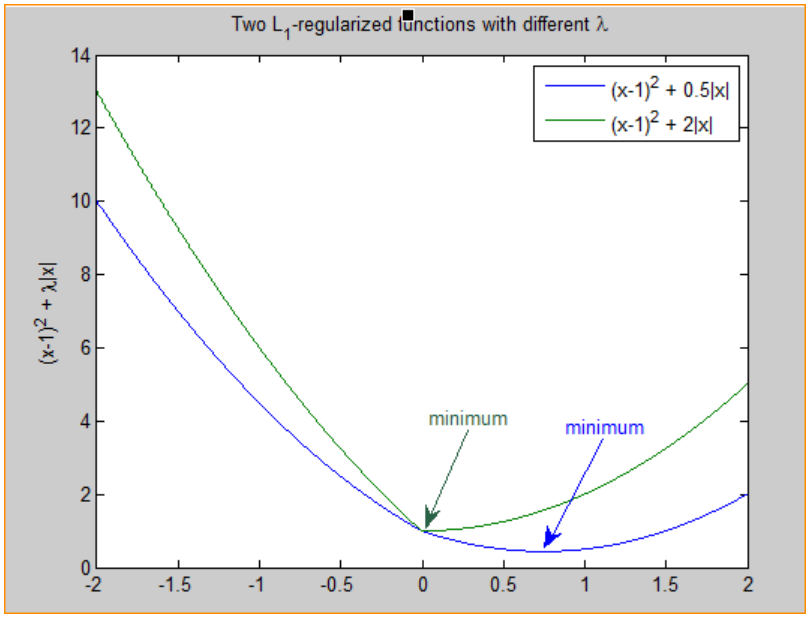
2.3.1 L1正则化参数

通常越大的λ λ可以让代价函数在参数为0时取到最小值。下面是一个简单的例子，这个例子来自Quora上的问答。为了方便叙述，一些符号跟这篇帖子的符号保持一致。

假设有如下带L1正则化项的代价函数：

$$F(x)=f(x)+\lambda ||x||1$$

其中x是要估计的参数，相当于上文中提到的w以及θ. 注意到L1正则化在某些位置是不可导的，当λ λ足够大时可以使得F(x)在x=0时取到最小值。如下图：



2.3.2 L2正则化参数

从公式5可以看到， λ 越大， θ j衰减得越快。另一个理解可以参考图2， λ 越大，L2圆的半径越小，最后求得代价函数最值时各参数也会变得很小。