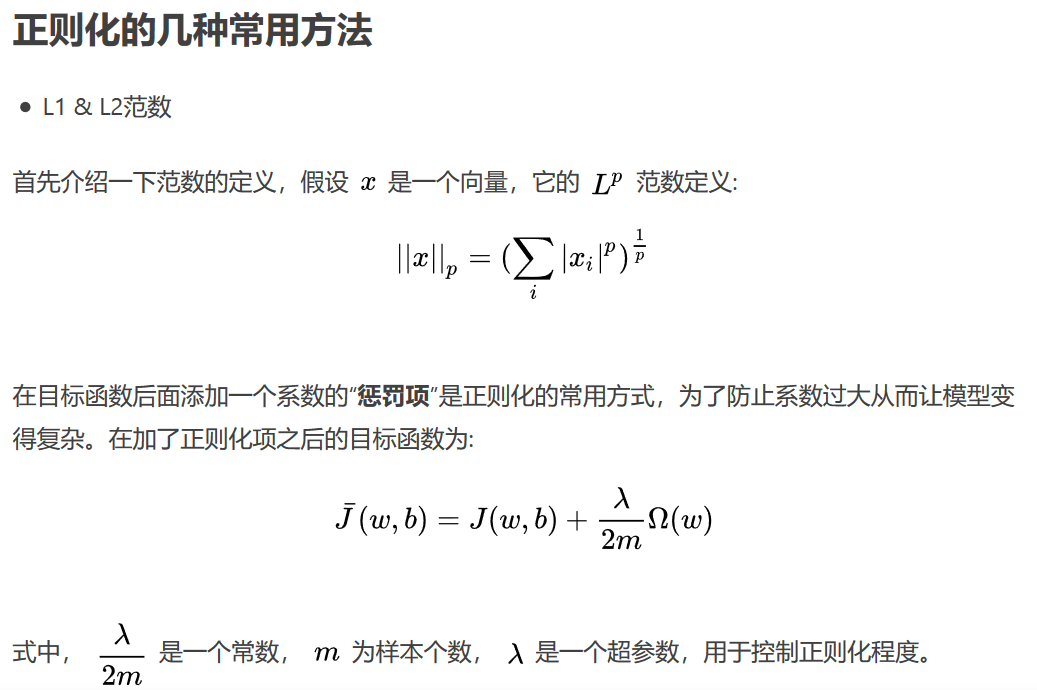
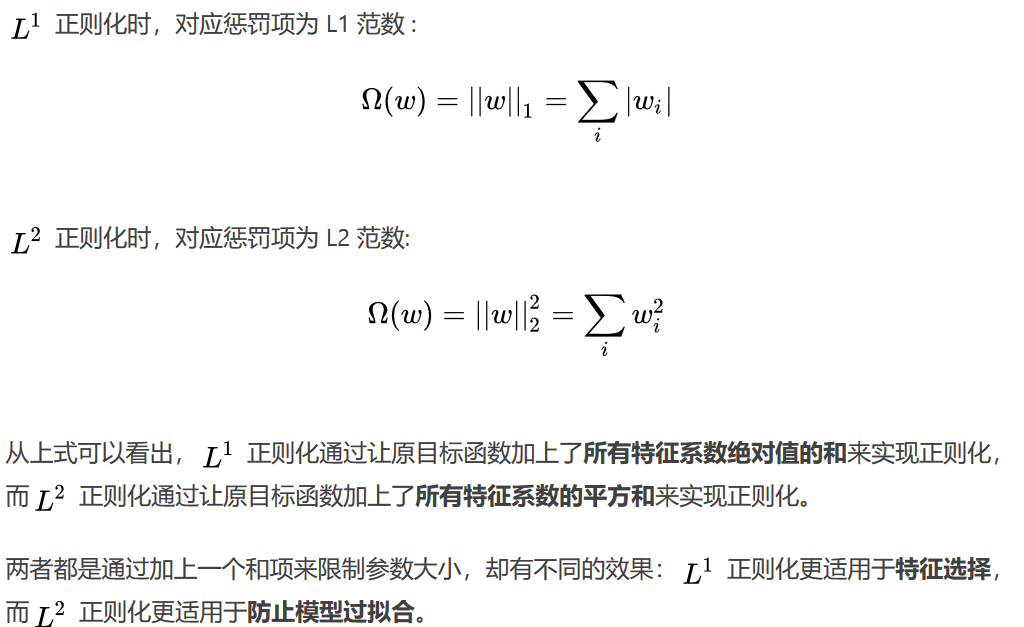
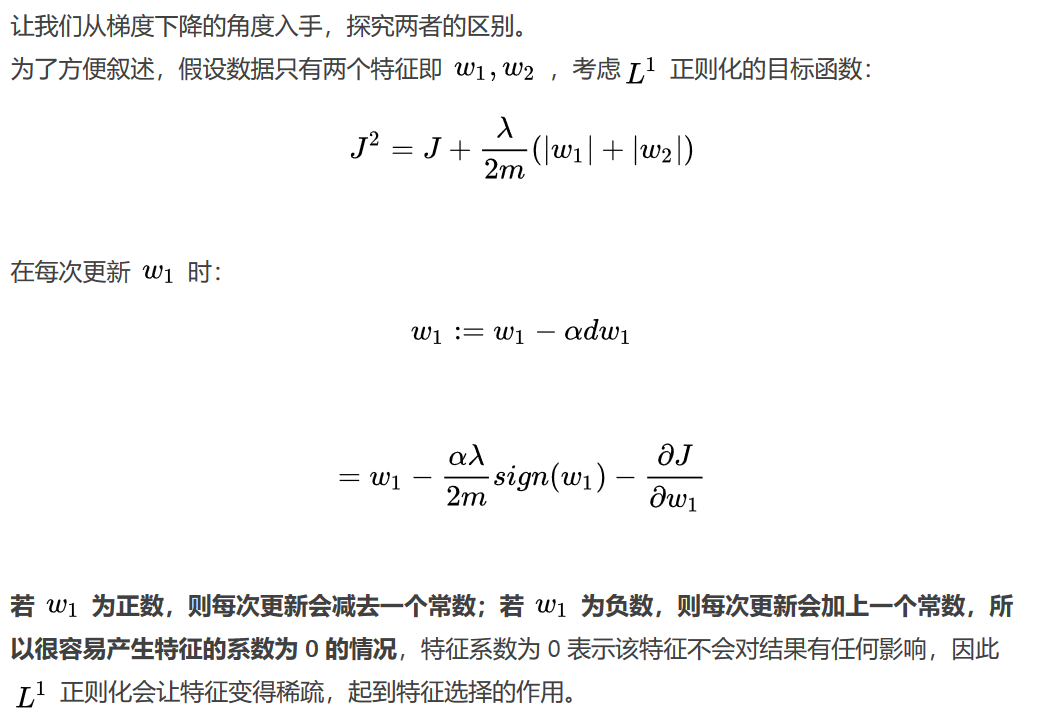
https://www.jianshu.com/p/569efedf6985

简单来说，正则化是一种为了减小**测试误差**的行为(有时候会增加**训练误差**)。我们在构造机器学习模型时，最终目的是让模型在面对新数据的时候，可以有很好的表现。当你用比较复杂的模型比如神经网络，去拟合数据时，很容易出现过拟合现象(训练集表现很好，测试集表现较差)，这会导致模型的泛化能力下降，这时候，我们就需要使用**正则化**，降低模型的复杂度。

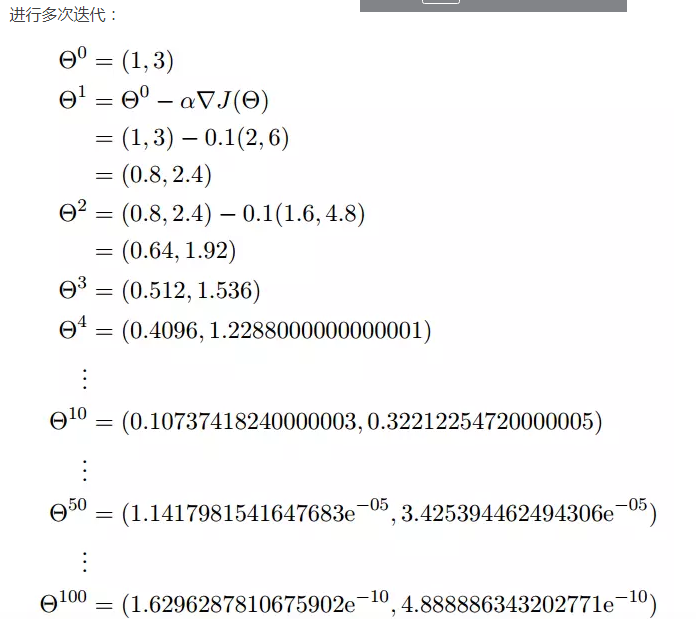




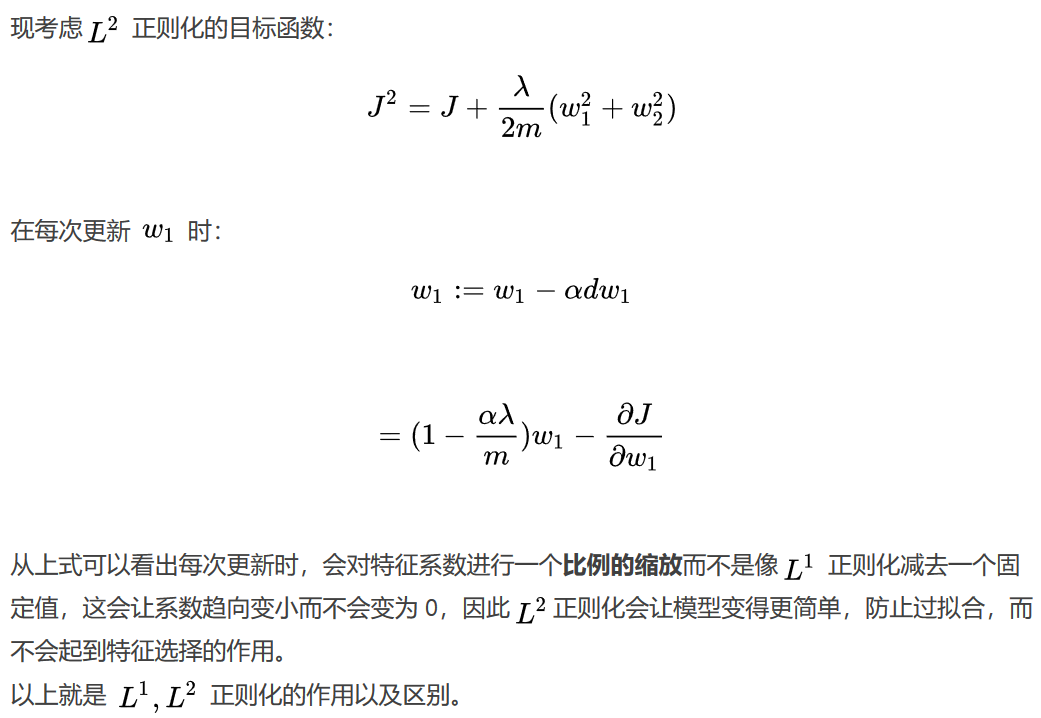


也就是说L1范数的约束之下，在梯度下降的过程中很有可能让一些参数Wi朝着0发展，最终变成0，之后Wi上梯度不再变化，那么这个特征也就失去了作用，这样算是起到了稀疏特征的作用。

其实可以这样总结，我们看常用的损失函数，均方误差，是幂函数，平方项，其梯度变化的一个示例我们看一下



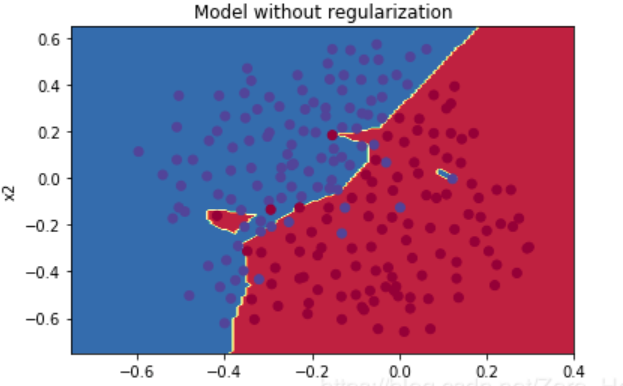
可以看出，梯度虽然在不断变小，但是是趋近与零而不是等于零，而对于一个不加正则项的目标函数，也就是损失函数，一般由于其自身特征，比如均方误差，交叉熵函数，不易使得参数变成0。但是在均方误差函数上再加一个L1正则项的话，L1正则项和我们的平均绝对值误差又很像，其特性显而易见，梯度相同，每次下降距离相同，在均方误差函数上再加一个L1正则项的话，确实使得有些特征的系数w变成0的可能性变高。



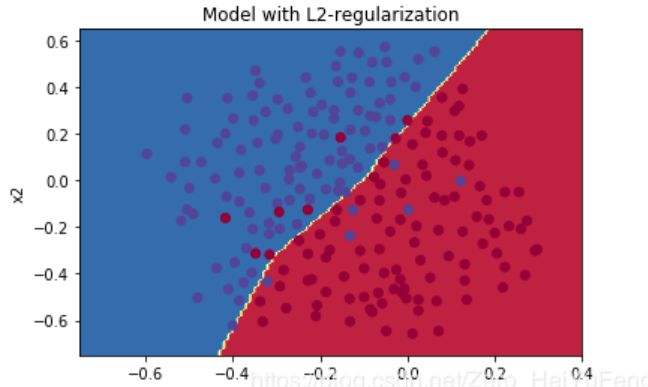
还是没有参透这种加速w变化的操作，怎么就可以防止过拟合了。

L2拟合过程中通常都倾向于让权值尽可能趋近于零，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现象。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。线性组合本身可以用加权求和的思想去看，不同的特征的系数都很小，这样限定每一个特征对最终的回归结果都影响很小，所以扰动能力弱

当不使用正则化，发生过拟合时



使用 正则化，正常拟合



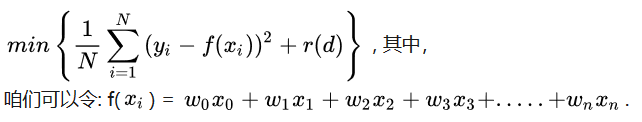
图中的边界可以看做是描述样本的函数f(x),在图中可以有比较直观的感受是，过拟合时，分类边界的起伏会更大，也就是在部分点斜率更大，而正常拟合时，分类边界更加平缓。这也是为什么在目标函数中加入“惩罚项”可以达到正则化的效果，“惩罚项”可以使每个参数趋向于零，在求导时斜率也会更小，等于变相的让模型更加简单了，更加简单的模型当然更加不容易过拟合。

对于线性回归模型，

使用L1正则化的模型建叫做Lasso回归，

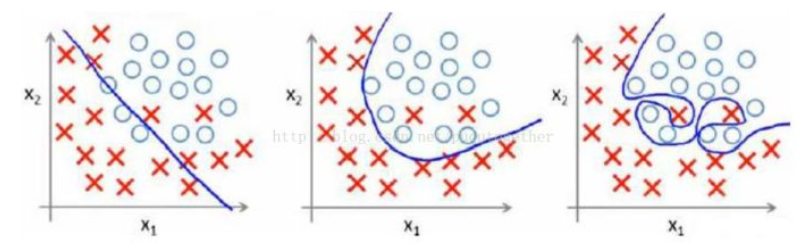
使用L2正则化的模型叫做Ridge回归（岭回归）

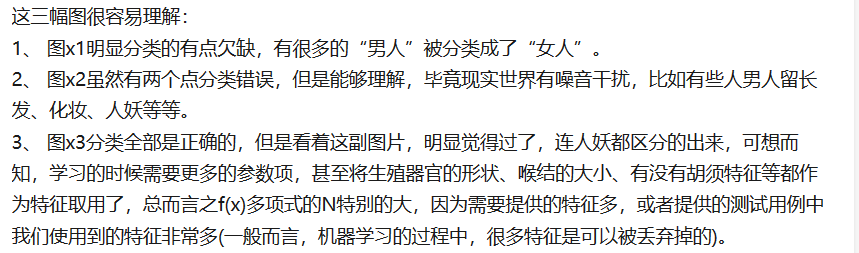
<https://www.zhihu.com/question/20924039/answer/131421690>



r(**d**)可以理解为有d的参数进行约束，或者 **D** 向量有d个维度。

首先，用一个例子来理解什么是过拟合，假设我们要根据特征分类{男人X，女人O}。  
请看下面三幅图，x1、x2、x3；

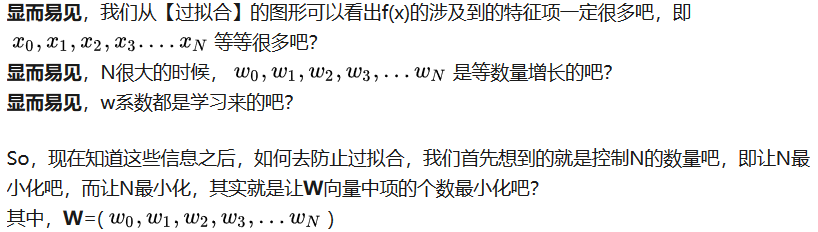




多项式越多，函数越复杂，表征能力就越强，可以逼近任何函数，对于x1只有一个特征，故只能表征一条直线，x2有两个特征可以表示一条略复杂的曲线，f(x)多项式的N越大，学习的越好，明显的f(x)1 欠拟合、f(x)3 过拟合

（上图中x1容易理解，对于决策函数f(x)，只取一个特征的话，其决策边界就是一条直线，正常来讲的话，无论有多少多项式，它始终是线性组合，表达能力不强，其实上图中x2，x3都是不是一次函数的线性组合而是高阶函数画出来的，比如

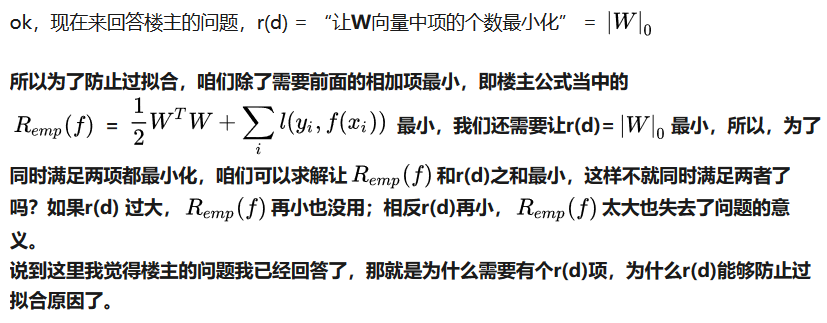
Y=w0+w1\*x+w2\*x2+w3\*x3(次方),所谓的线性是对参数w而言，而不是x，这才是为什么多项式的组合可以拟合这么复杂的曲线，参数w是通过目标函数学习得来的，回到决策函数，用目标函数确定的最优参数w，决策函数fx 变量变成了x了，w变成固定系数，这时其实我们fx对w是线性的，对于x的可能就不是线性组合函数了）

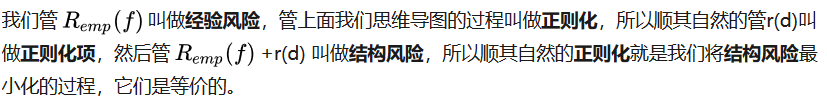


问题最终导向：如何求解“让**W**向量中项的个数最小化”这个问题

学过数学的人是不是看到这个问题有点感觉？对，没错，这就是0范数的概念！什么是范数，我在这里只是给出个0-2范数定义，不做深究，以后有时间在给大家写点文章去分析范数的有趣玩法；  
0范数，向量中非零元素的个数。  
1范数，为绝对值之和。  
2范数，就是平方的和。

向量中0元素，对应的x样本中的项我们是不需要考虑的，可以砍掉。因为0\*Xi没有啥意义，说明Xi项没有任何权重,相当于摒弃了一些特征对于模型的影响





而经验风险函数一般就是我们所说的损失函数，结构风险函数一般就是我们所说的目标函数。

0范数难求，咱们就求1范数呗，然后就研究出了下面的等式：



上面概括而言就是一句话总结：1范数和0范数可以实现稀疏，1因具有比L0更好的优化求解特性而被广泛应用，**L2范数是指向量各元素的平方和然后求平方根。我们让L2范数的正则项||W||2最小，可以使得W的每个元素都很小，都接近于0，但与L1范数不同，它不会让它等于0，而是接近于0，这里是有很大的区别的哦；所以大家比起1范数，更钟爱2范数。**