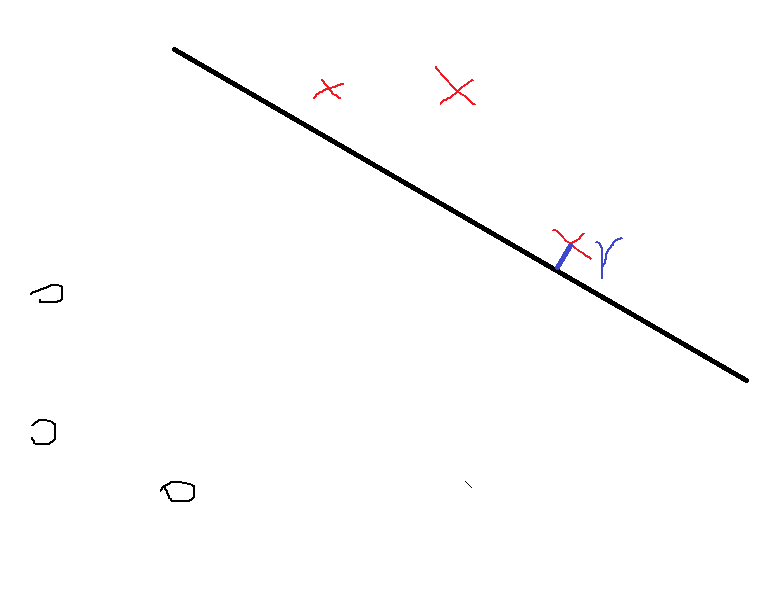
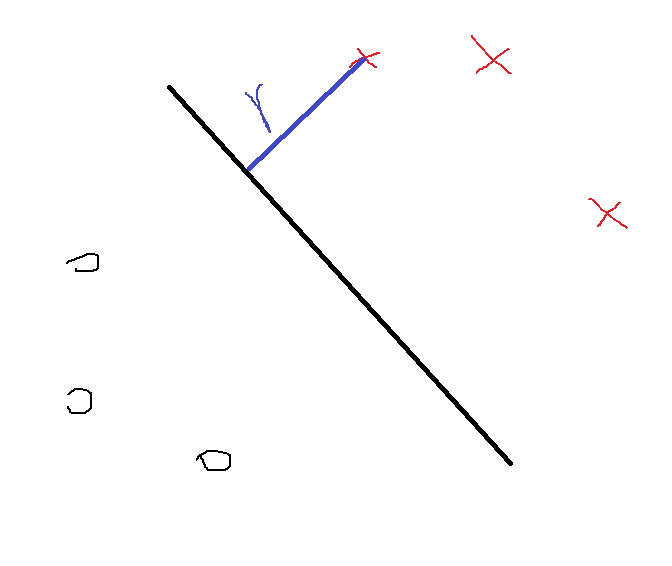
1. 线性可分支持向量机：

顾名思义，线性可分就是存在至少一个超平面，可以将样本中的正负例准确隔开，找到

这样一个超平面可以用感知机实现，但是一般这样的超平面是不唯一的，线性可分支持向量机就是在这些超平面集合中选择最优的一个。

那么最优超平面的是怎么定义的呢？在感知机相关知识中，我们知道点到超平面的距离为（https://blog.csdn.net/wzx479/article/details/83143280），因为对所有的实例点来说都是一样的，所以就省略了，对于所有正确分类的点可以用这种形式取代绝对值号：。对于一个确定的超平面π，定义这个超平面关于所有实例点的函数间隔为，最优的超平面就是使得最大的那一个，这个比较容易理解：超平面与最近的点之间的距离越大，分类时的不确定性越小，如下两个超平面，都满足感知机的条件，但是直观上来看，第一个要比第二个好一些（为什么呢？我的看法是第一个超平面划分比较靠近中间，留给两侧的“缓冲区”是差不多的，类似于均值不等式会在相等的时候获得最值，有点中庸的意思）。



所以优化的目标就是，为了排除和b成比例变化带来的影响（例如同时变为两倍，这时超平面实际没变，但r变为两倍），定义了几何间隔：，因为我们关注的是，所以把设为1，这时优化目标就变成，等价于，因为定义了=>，也就是说是所有实例点到超平面的距离最小的，所以约束条件是，综上，可以得到以下原始优化问题O1：

 O1

利用Lagrange对偶性（**https://www.cnblogs.com/breezezz/p/11303722.html**），可以将原始问题转化为最大化最小值问题：



其中，，有点像高数多元函数的有约束极值问题，首先要求，优化变量为和b，令L偏导数为0：



分别令其等于零，可以得到：



这个时候解出的和b一定使L在α固定的条件下取得极值，至于是极大还是极小，可以通过求二阶偏导数判断，这里偷个懒直接按照书上的，即取的是极小值，将带入（1）式，并利用条件，整理可得：



接下来的任务就是最大化，即：

，

优化变量是α，所以与α相关的约束条件要考虑在内：，（应该是可以小于等于0的，不过这样解出来的是-，为了避免正负号转换的麻烦，取），整理一下，最原始的优化问题经过一步步变换，转换为以下问题：



这样就把原来的三变量优化问题变为单变量优化问题（但是是个向量，所以相当于还是多变量，呵呵），求解出就可以根据和，计算b时，只需要带入一对实例点即可，因为对于所有的实例点来说计算出的b都是相等的（我猜的）

李航的书里边给了一个例子，可以帮助理解一下上面的过程。正例点x1(3,3)，x2(4,3)，

负例点x3(1,1)，求解和b，把这些点带到优化目标里：

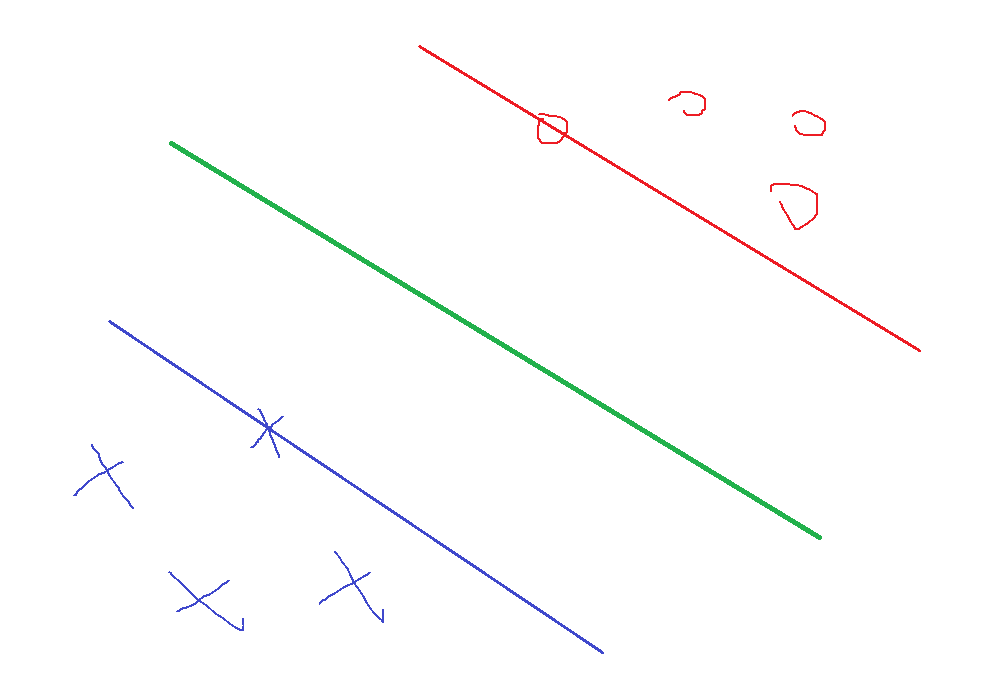


可以得到：



因为这个时候知道具体的表达式而且比较简单，可以直接求偏导数，令其等于0，而不用数值方法，求解出来，但是这个解不满足，所以正确的解在边界上，分别令，然后每次求解出其余两个α，把这三组解带到L里，看看哪一个使得L最小，最终解为。在实际的算法中，绝大多数情况肯定是没法用解析的方法求解的。

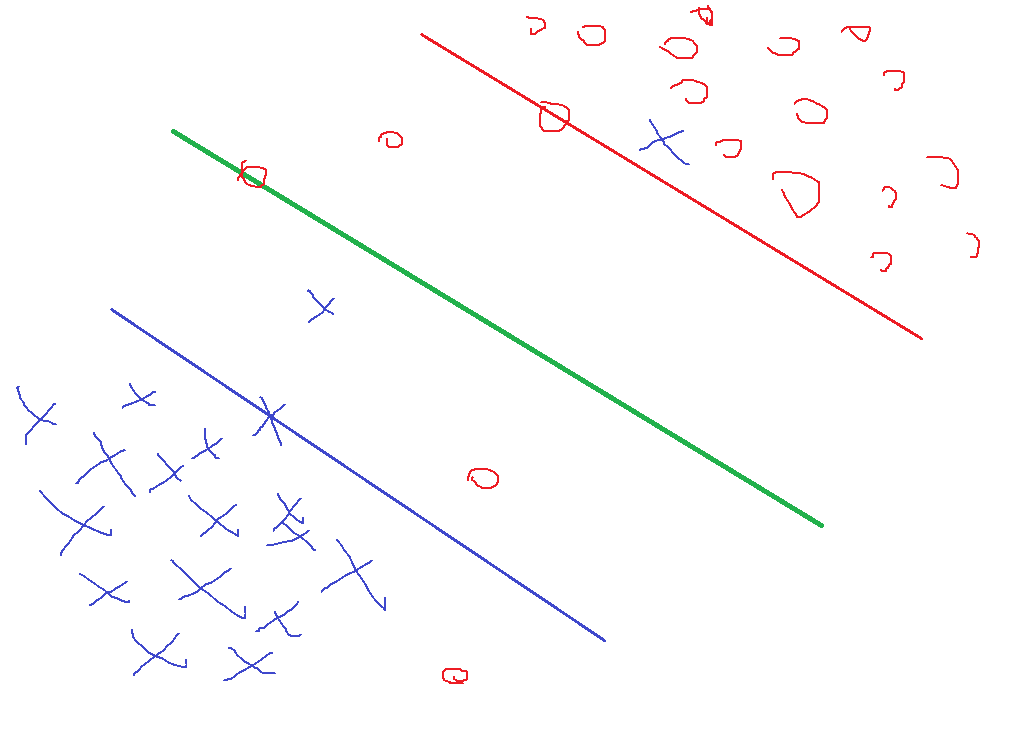
在线性可分的情况下，求解的结果应该如下图所示，其中绿线表示求出的超平面，红线和蓝线表示间隔平面。

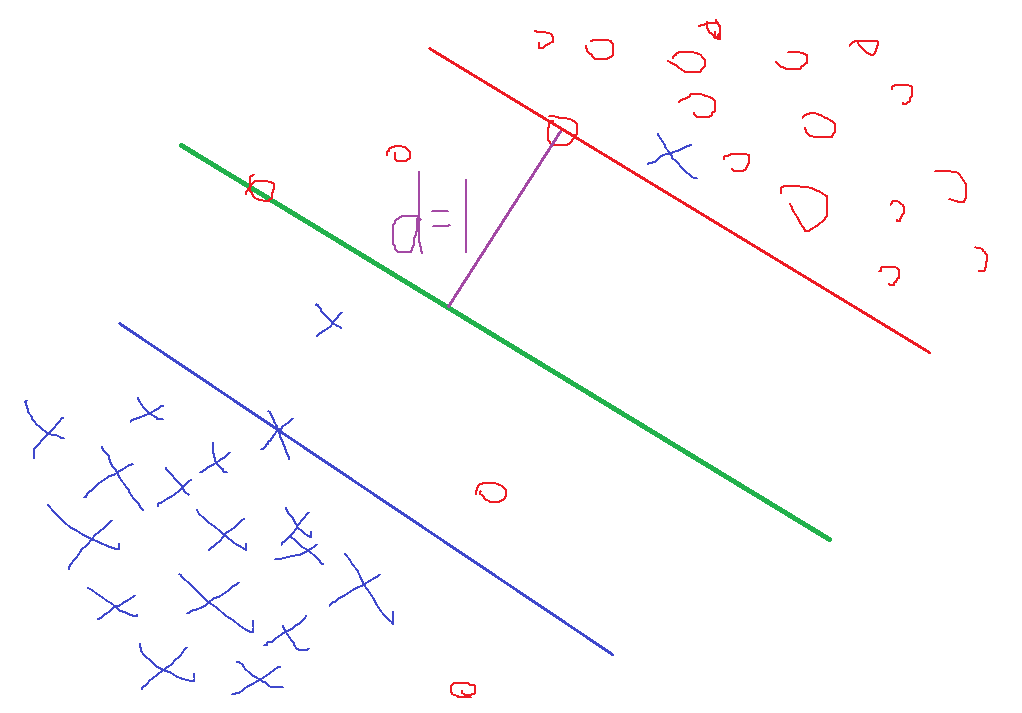


1. 软间隔最大化

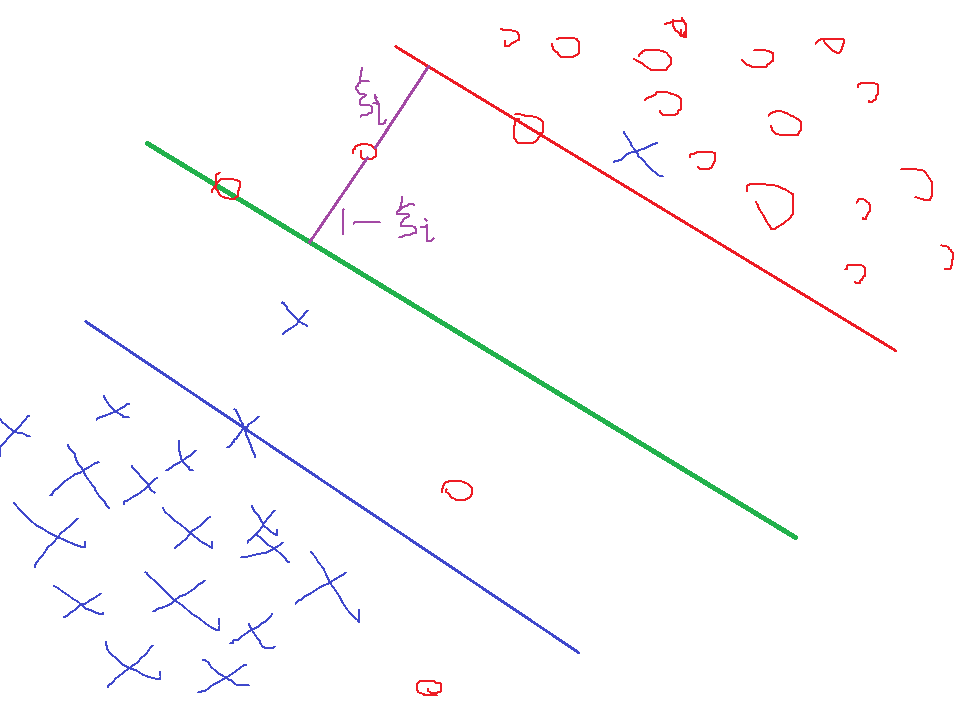
在前一节的所有论述中，一个重要前提是超平面存在，即训练空间是线性可分的，实际

应用中大部分情况肯定不是线性可分的，如果按照线性可分的要求去找超平面，在下图这种情况显然是没有解的，但是，可以看到，按照绿线的划分方式，训练出来的模型应该也有不错的分类效果。



在原先的约束条件中，这个1表示的就是下图中紫色笔画的d，当然从原始距离变为1，是经过了一些规范化的步骤的。

如果我们做出一点让步，允许某几个点（异常点）到超平面的距离小于间隔平面到超平面的距离（甚至可以为负值，负值表示点在超平面的另一侧，例如绿线以上的蓝×），这样就可以使在不是线性可分的训练空间上也可以解出绿线这种模型，并且预期的分类效果还不错。那么需要修改原先的约束条件，改为：，对于位于绿线上的红圈，只需要取=1（因为就表示实例点在超平面上），对于在绿线上侧的蓝×，只需要取>1，这时就表示蓝×落到了超平面的另一侧。



由此可以看出，加上这个变量之后，可以通过调节使所有的实例点都满足约束条件。这时优化问题变为O2：

 O2

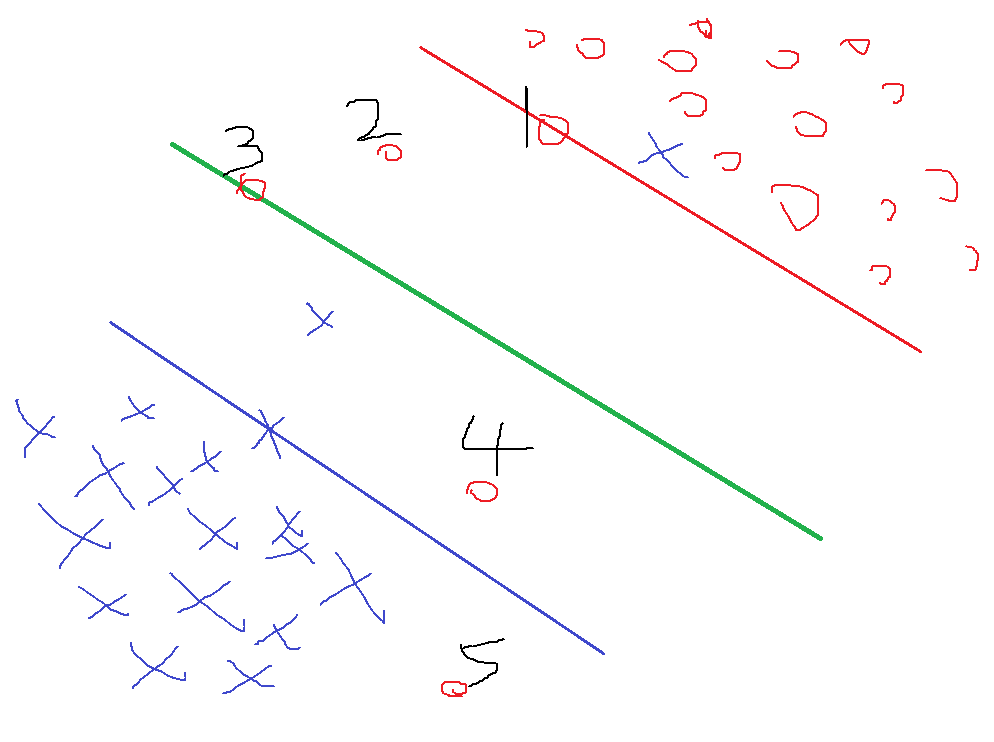
按照与求解O1类似的步骤（利用Lagrange对偶性，转换为最大化最小值问题之后偏导数，令梯度等于零，解出关于的表达式），将尚书右约束最优化问题转换为：



利用某种方法（后面会提到，不是梯度下降法）进行优化求解之后可以得到



这个解的形式与线性可分数据集的硬间隔最大化得到的解是一样的，不同的是，在软间隔最大化中，带入不同的点求解出的可能是不同的，这些都算是解，注意只能由支持向量所对应的实例点(,)求得，支持向量是指位于间隔边界上的点，即



关于上图几个点的和的取值，李航的书里是这样写的：

1点：

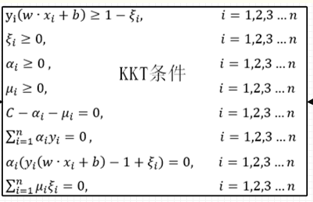
2点：

3点：

4，5点：

的取值还是比较好理解的，因为1-就表示实例点到超平面（绿线）的距离，以1点为例，它到超平面的距离为1，所以，。的取值得从KKT条件来理解：

见**https://www.cnblogs.com/houzichiguodong/p/9254898.html**



时，由于，所以，因为，所以，所以由得，所以在支持向量上；、时也可以按照KTT条件进行分析。

1. SMO算法

前面都是进行最优化问题的化简，最终都转换成关于的优化问题：



求解这个最优化问题用的是SMO算法，我还不理解原理，只知道步骤。每次选择两个进行优化，而固定其他的值。可以分为两步，一是选择两个，二是更新这两个

1. 选择和

首先说一下我对这一步的感受：太绕了，而且教材上有几个地方可以这样理解，也可以

那样理解，得多看几遍次才行。

可以用两层循环实现，第一层选，第二层选。按照上面红色文字部分的分析，要满足以下的关系：

时，；时，；时，；

就从违反这些关系的中选择，优先选择但的。如果找不到违反这些关系的，说明已经达到最优，停止搜索。

选出之后，选择的原则是使最大，其中，选出之后，按照一定方法（后面会写）更新和。如果更新之后，优化目标函数没有产生足够的下降（与预设阈值相比），就放弃当前，并且不再以最大为原则来搜索，而是选择在内的。如果再次更新和之后还是没有使优化目标函数产生足够的下降，放弃当前，并再退而求其次，从其他值中选择（注意，这里并不是说从中选一个失败之后就可以进入这个if，而是要把内的所有都遍历完，如果都失败了，才可以去或中选择）。如果再次更新之后，依然没有产生足够的下降，那么放弃当前，重新进入第一层循环以搜索。

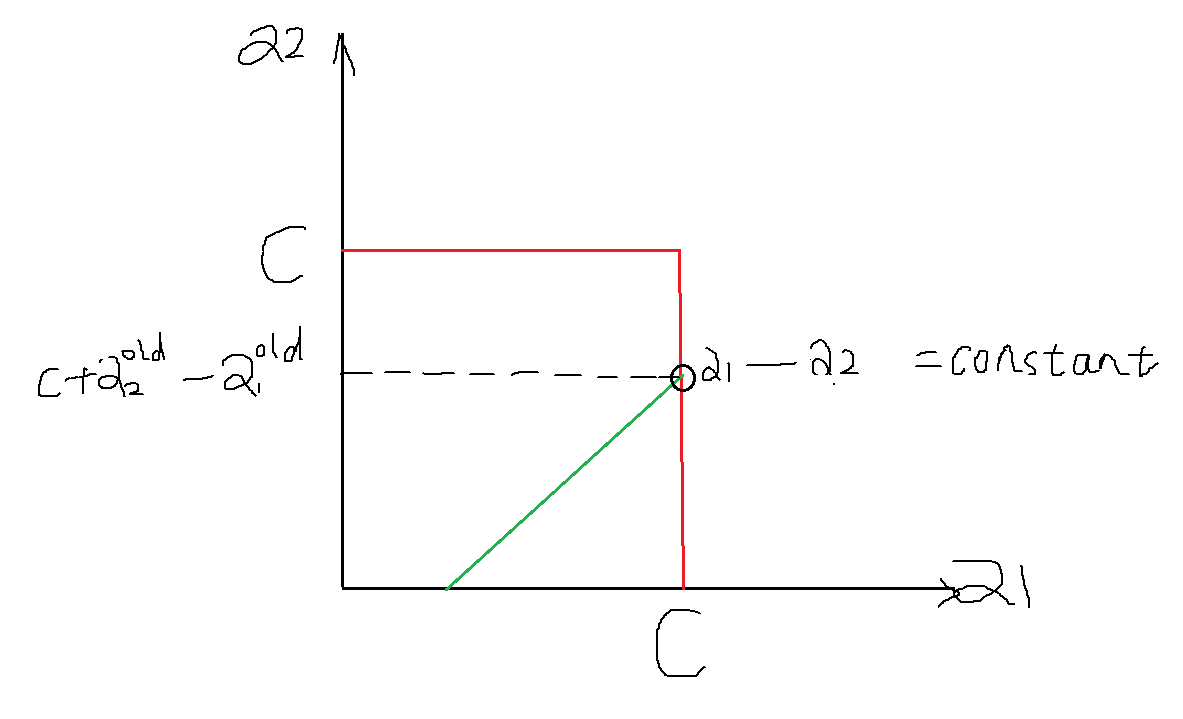
1. 更新和



因此、满足关系：，求出后根据这个更新。那是怎么求的呢？公式见下：



为什么上标要加个unc呢？因为还会受到其他约束，例如，，下面对根据受到的约束，对进行一个调整。



上图是假设且时，、的图像，可以看到这时，的上界是（这里用到了）。同理，根据、、的取值不同，可以得到的不同上下界，总结一下可以得到：

时，

时，

同一记为：，所以：



求出之后，=

其实，到这里就可以写代码了，但是为了完整，再加上核函数的概念

1. 核函数

这个地方书上有一些数学概念，我看不太懂，举个例子说一下我的理解：

假设有一个椭圆，椭圆内部都是正例点，椭圆外部都是负例点，如果不进行任何变换，虽然利用软间隔的支持向量机可以构建出模型，但是可想而知，一条直线的分类效果无论如何都好不到哪里去。如果在求解之前先进行一个变换，例如：，即，分类边界就变为：，这是个线性模型，求解出来的SVM模型准确率肯定很高。

核函数的思想就是这样，把非线性的变为线性的，然后构建SVM模型，可以达到比较好的预测效果。创建一个核函数要满足许多数学条件，因此一般用现成的，常用的有多项式核、高斯核等（见统计学习方法）。

在以上化简和求解最优化问题时，可以看到都是成对出现的，即都是以的形式出现，所以可以预先定义一个X矩阵，用于存储；对于使用了核函数的SVM，同样可以定义一个K矩阵，用于存储，并且求解步骤与一二三节完全一致，只需要把换成即可。

1. 测试

用了两个案例，上边的是线性可分的，下边是用了核函数的，用颜色表示正负例，圆

形代表训练集，三角形代表测试集。没算准确率，不过我感觉效果还是不错的。

