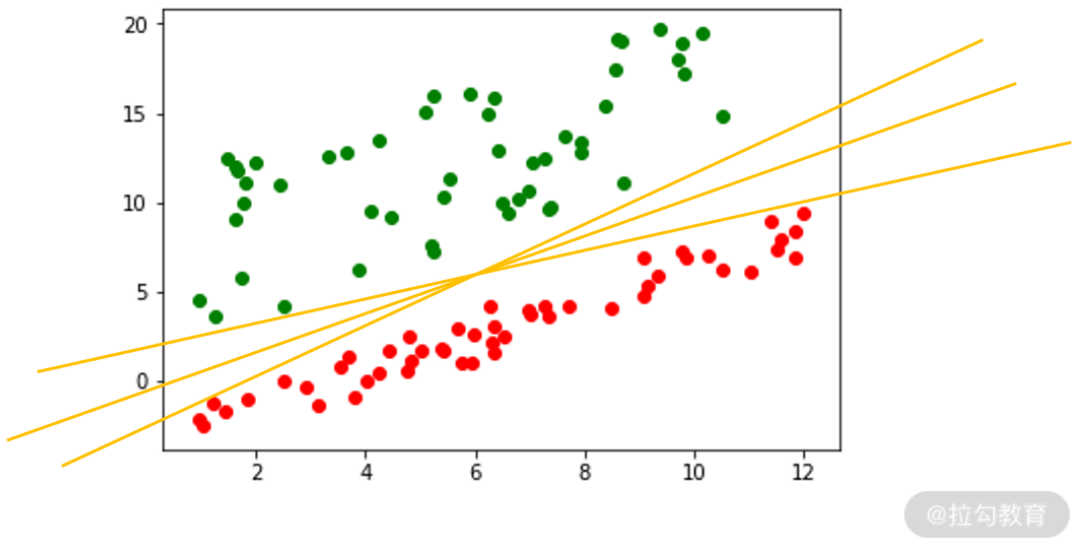
今天要介绍的算法叫作支持向量机（Support Vector Machine，SVM）算法。这个算法在 1995 年就已经被发表出来了，由于在文本分类任务上面表现优异，SVM 算法很快就如日中天，成为机器学习的主流算法。在后面很长一段时间里，都有大量的学者对它进行了深入研究和改进，甚至写了很多相关的书籍。下面我们从一个例子出发，去看看这个算法是基于什么样的思路产生的。

一个例子



秋天来了，丰收的季节到了。我们把收获的红豆和绿豆从豆荚里摘出来，然后铺在地上晾晒。大伯拉着红豆和绿豆来到一条公路附近，把绿豆铺在公路的左边，红豆铺在公路的右边。

后来，陆陆续续还有人来晒豆子，为了划分好红豆和绿豆，防止混合到一起，这个时候我们在红豆和绿豆之间画一根线，这样就可以比较明显地分出红豆和绿豆的界限。

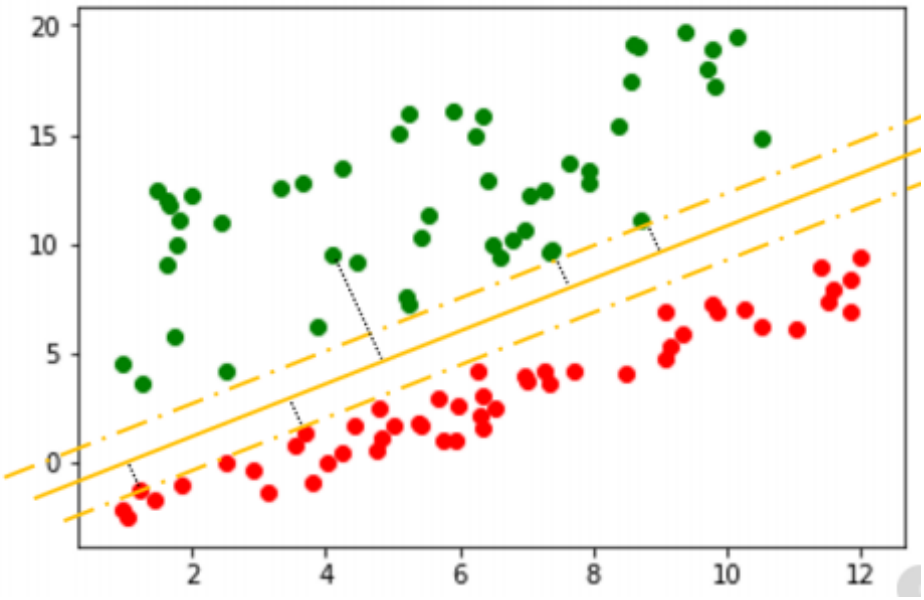
在目前的情况下，红豆和绿豆的中间有一条宽阔的马路，你可以在马路上任意地画出一条线，只要不超出这个界限就可以分割红豆和绿豆了，但是，如果是斜着划线，那么只要铺的豆子足够长，这条线早晚会超出公路的范围，而把一些红豆划为绿豆，所以要划的线肯定要跟笔直公路平行，这样至少不管画多远都不会超出公路。而沿着公路，仍然有很多条线可以画，但是想到有些豆子可能不太听话，有时候会跑到公路上面一两颗，这样的话，当然是选马路中轴线作为我们的划分直线最好了。

算法原理

上面就是 SVM 支持向量机的思考来源，SVM 要解决的就是怎么**找到那条中轴线。**虽然这个想法十分简单，然而具体做起来有很大的难度。实际上，在我们的数据中并没有一条公路，而是只有两堆豆子，SVM 要解决的问题就是首先要找到一些线，这些线都可以分割红豆和绿豆；然后再找到正确的方向 / 斜率的那条线；最后确认马路的宽度，当然是越宽越好，这时候得到最优解——马路的中轴线。

比起其他分割线，这个马路的中轴线对于未知的数据宽容度最大，使用中轴线来作为分界线，当新数据发生了一些波动，超过了马路的边界，仍然不会对结果产生影响，因此它被认为是最优解，所以 SVM 就是基于优美的数学推导不断探索最优解的一个算法。

到这里，我们已经知道了 SVM 的思考路径，然而 SVM 所涉及的内容远没有介绍完，下面我们再来看几个 SVM 所涉及的名词。



什么是超平面?

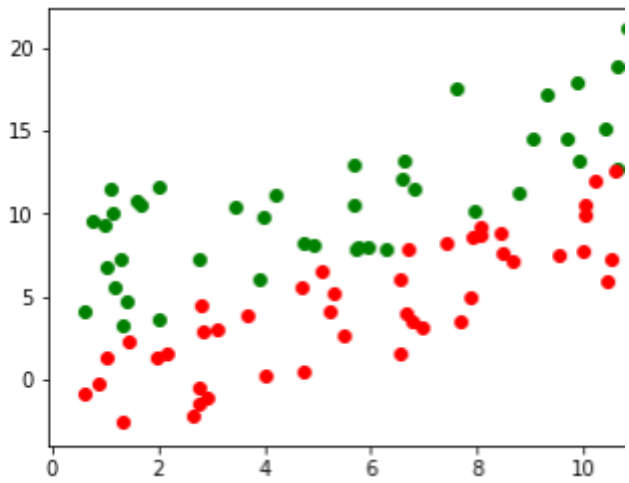
我们首先想一想什么是平面。根据定义，在三维空间中，平面就是到两个点距离相同的点的轨迹。一个平面没有厚度，而且可以把空间分割成两部分。而超平面就是在这个基础上进行的延伸，在维度大于三维的时候仍然满足上面的条件，而且它的自由度比空间维度小 1。对于这样的一个数学概念，就称为超平面。通俗地讲，在二维中就是直线，在三维中是平面，在三维以上的维度中就是超平面。

什么是支持向量?

假设我们已经找到了一条线（不一定是最优的那条）可以分割红豆和绿豆，红豆和绿豆中距离这条线最近的几个样本点就被称为**支持向量（Support Vector）**，这些点到这条线的距离称为间隔，SVM 的思路就是要找到有最大间隔的那条线（超平面）。**在决定最佳超平面时只有支持向量起作用，而其他数据点并不起作用**，如果移动非支持向量，甚至删除非支持向量都不会对最优超平面产生任何影响。即支持向量对模型起着决定性的作用，这也是 “支持向量机” 名称的由来。

如何处理不清晰的边界?

在我们晒豆子的时候，不小心把一些豆子踢到了马路上，最后红豆和绿豆产生了一些交叉，甚至有一两个绿豆完全进入了红豆的区域，这大概也比较符合我们平时遇到的数据的情况。这时就有一个新的概念：**软间隔**，也就是说在这个间隔区域里允许出现一定数量的样本，我们就称为这个间隔为软间隔；像最开始那种划分非常清晰，在间隔中间没有任何红豆和绿豆的理想状态下，间隔就称为**硬间隔**。



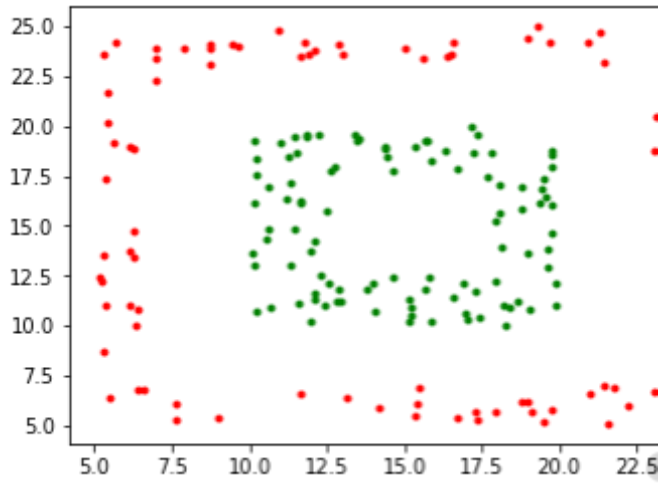
像我上面画的这幅图，红豆和绿豆的分布间隔并不是那么明显，甚至有些红豆和绿豆都已经分布在同样的位置，这个时候使用硬间隔，不允许间隔中出现任何豆子明显是不现实的，我们永远没办法找到一条线来把这两种豆子划分清楚，更没办法找到一个硬间隔区间。在这种情况下，我们允许间隔区域有一定的豆子出现，从而作为软间隔区间来进行划分。

如何处理非线性可分?

上面我们画出的绿豆和红豆是**线性可分**的，可是在数据中，还有很多是**线性不可分**的情况。如下面这张图，我们在一个方形的广场上晾晒豆子，把豆子摆成方形，内圈是绿豆，外圈是红豆。这就没有办法画一条线把红豆和绿豆切分出来了。

在 SVM 中采取的办法是**把不可划分的样本映射到高维空间中**，让样本在高维空间中可以线性可分。当然了，在高维空间，比如在三维空间中，线性可分就是可以画一个平面把红豆和绿豆切分成两个部分。

试想本来落在地上的红豆和绿豆本来没办法划分，这个时候在广场的底部有一个巨大的风机，这个风机可以提供稳定的风，从豆子下方往上方吹。由于红豆和绿豆的密度和质量不同，当风机开动之后，红豆和绿豆飞起来；等待稳定之后，红豆和绿豆就会悬停在不同的高度上，这个时候我们就可以放一个巨大的平面进去，把红豆和绿豆划分开来。



当然了，在 SVM 中没有吹风机，在 SVM 中借助 “**核函数**” 来实现映射到高维的操作。常见的核函数有线性核函数、多项式核函数、高斯核函数等。使用核函数主要是模拟特征转化到高维空间的内积计算结果，在实现了映射到高维的同时，降低了计算量，同时也能够节省计算用的内存。

关于 SVM 算法的主体部分到这里讲得差不多了，当然，SVM 里面的高深的数学原理我们还远未涉及，如果大家对公式的推导很感兴趣，可以找一些资料来进行学习。接下来我们看一下，对于实际的工作，我们该注意一下 SVM 有什么优缺点。

算法的优缺点

优点

- **有严格的数学理论支持，可解释性强。** SVM 所获得的结果是全局最优解而不是局部最优解，很多算法为了降低复杂性只给出了一个局部最优解，比如我们前面提到的 “决策树算法”，而 SVM 的最优化求解所获得的一定是全局最优解。
- **算法的鲁棒性很好。** 由于计算主要依赖于关键的支持向量，所以只要支持向量没有变化，样本发生一些变化对算法没有什么影响。

缺点

- **训练所需要的资源很大。** 由于运算量与存储量都很高，SVM 训练的开销也是巨大的，因此支持向量机只适合比较小的样本量，比如几千条数据，当样本量太大时训练资源开销过大。

- **只能处理二分类问题。** 经典的 SVM 算法十分简洁，正如上面的例子一样，画一条线分割两个类别，如果需要处理多类别的分类问题，需要使用一些组合手段。
- **模型预测时，预测时间与支持向量的个数成正比。** 当支持向量的数量较大时，预测计算复杂度较高。因此支持向量机目前只适合小批量样本的任务，无法适应百万甚至上亿样本的任务。

尝试动手

与之前的课时相同，下面我们亲自来练练手，用代码实现使用 SVM 算法。在本节的代码中，在数据获取与数据处理阶段仍然沿用了之前的方法，没有任何改动，主要的区别是我们引入的算法包为 SVM 包，在进行分类时使用的是 SVM 分类器。

```
from sklearn import datasets


from sklearn import svm


import numpy as np


np.random.seed(0)


iris=datasets.load_iris()


iris_x=iris.data


iris_y=iris.target


indices = np.random.permutation(len(iris_x))


iris_x_train = iris_x[indices[:-10]]


iris_y_train = iris_y[indices[:-10]]


iris_x_test  = iris_x[indices[-10:]]


iris_y_test  = iris_y[indices[-10:]]


clf = svm.SVC(kernel = 'linear')


clf.fit(iris_x_train,iris_y_train)


iris_y_predict = clf.predict(iris_x_test)


score=clf.score(iris_x_test,iris_y_test,sample_weight=None)
```

```
print('iris_y_predict = ')

print(iris_y_predict)

print('iris_y_test = ')

print(iris_y_test)

print('Accuracy:',score)
```

使用 SVM 算法得到的结果如下，可见第二个位置的预测仍然是错误的，这个位置的数据看来十分难分对。

```
iris_y_predict =

[1 2 1 0 0 0 2 1 2 0]

iris_y_test =

[1 1 1 0 0 0 2 1 2 0]

Accuracy: 0.9
```

如果你有时间，尽量把代码亲手敲一遍以加深自己的印象，同时可以去 sklearn 网站上查看一下该算法的各种使用方法和接受参数，尝试进行一些改变，看看是否会有一些不同的结果。

总结

完成了我们的代码部分，我们这一课时的讲解就告一段落了。本课时我们介绍支持向量机 SVM 算法的基本内容，我觉得 SVM 是众多算法中最复杂的一个算法，当然这节课我屏蔽了那些艰涩的部分，期望能够以一种简单易懂的方式向你介绍 SVM 的原理。关于 SVM 的细节，尤其是数学上的推导我们没有涉及，如果你有兴趣可以进行更深入的学习。

>