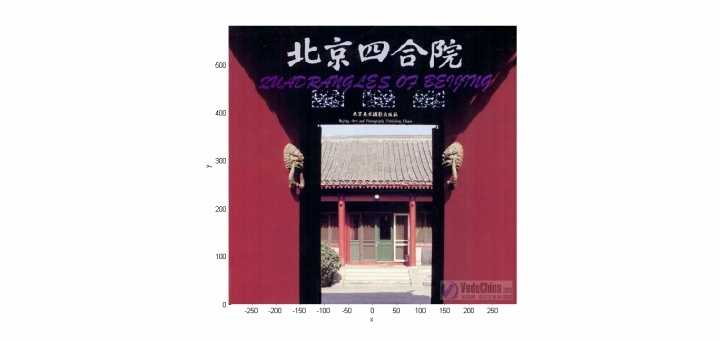
图灵提出了一种测试机器是不是具备人类智能的方法。

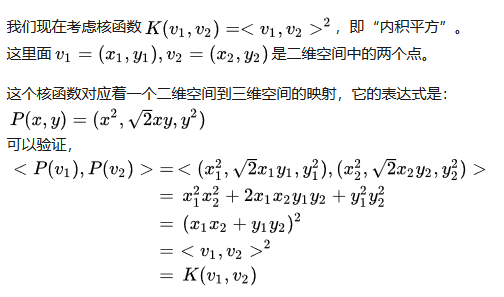
图灵测试是由[现代计算机](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=818571&ss_c=ssc.citiao.link" \t "_blank)科学之父英国人[阿兰·图灵](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=367872&ss_c=ssc.citiao.link)1950年提出的。图灵测试会在测试人在与被测试者(一个人和一台机器)隔开的情况下，通过一些装置（如键盘）向被测试者随意提问。问过一些问题后，如果被测试者超过30%的答复不能使测试人确认出哪个是人、哪个是机器的回答，那么这台机器就通过了测试，并被认为具有人类智能。

# 1一个核函数例子

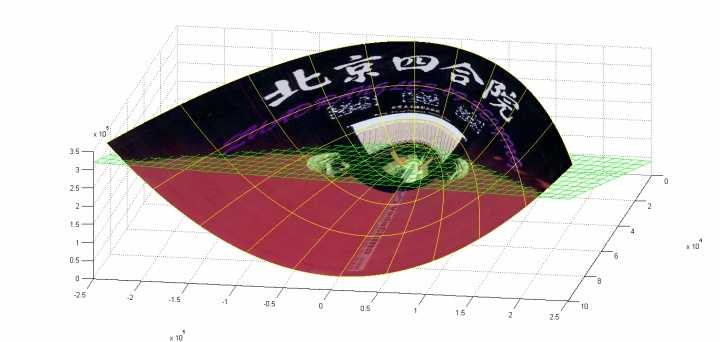
我来举一个核函数把低维空间映射到高维空间的例子。

下面这张图位于第一、二象限内。我们关注红色的门，以及“北京四合院”这几个字下面的紫色的字母。我们把红色的门上的点看成是“+”数据，紫色字母上的点看成是“-”数据，它们的横、纵坐标是两个特征。显然，在这个二维空间内，“+”“-”两类数据不是线性可分的。





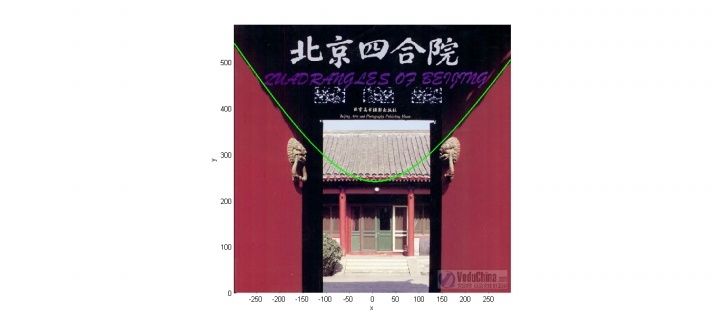
在P这个映射下，原来二维空间中的图在三维空间中的像是这个样子：



（前后轴为x轴，左右轴为y轴，上下轴为z轴）

注意到绿色的平面可以完美地分割红色和紫色，也就是说，两类数据在三维空间中变成线性可分的了。

而三维中的这个判决边界，再映射回二维空间中是这样的：



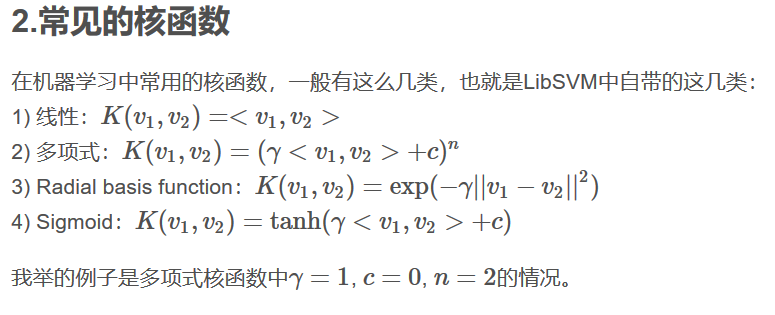
这是一条双曲线，它不是线性的。

================================================

如上面的例子所说，核函数的作用就是隐含着一个从低维空间到高维空间的映射，而这个映射可以把低维空间中线性不可分的两类点变成线性可分的。

当然，我举的这个具体例子强烈地依赖于数据在原始空间中的位置。  
事实中使用的核函数往往比这个例子复杂得多。它们对应的映射并不一定能够显式地表达出来；它们映射到的高维空间的维数也比我举的例子（三维）高得多，甚至是无穷维的。这样，就可以期待原来并不线性可分的两类点变成线性可分的了。

================================================



在实用中，很多使用者都是盲目地试验各种核函数，并扫描其中的参数，选择效果最好的。至于什么样的核函数适用于什么样的问题，大多数人都不懂。很不幸，我也属于这大多数人，所以如果有人对这个问题有理论性的理解，还请指教。

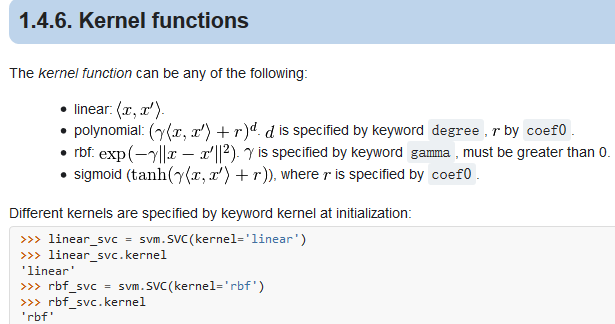
================================================

核函数要满足的条件称为[Mercer's condition](https://link.zhihu.com/?target=http%3A//www.svms.org/mercer/" \t "_blank)。使用SVM的很多人甚至都不知道这个条件，也不关心它；有些不满足该条件的函数也被拿来当核函数用。

由于我以应用SVM为主，对它的理论并不很了解，就不阐述什么了。

**核函数的有效性检验方法见后文。**

# 2常见的核函数



# 3核函数说明、推导、意义

核函数（Kernels）

考虑我们最初在“线性回归”中提出的问题，特征是房子的面积x，这里的x是实数，结果y是房子的价格。假设我们从样本点的分布中看到x和y符合3次曲线，那么我们希望使用x的三次多项式来逼近这些样本点。那么首先需要将特征x扩展到三维[clip_image002[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034289942.png)，然后寻找特征和结果之间的模型。我们将这种特征变换称作特征映射（feature mapping）。映射函数称作[clip_image004[10]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034282417.png)，在这个例子中

[clip_image006[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034285731.png)

我们希望将得到的特征映射后的特征应用于SVM分类，而不是最初的特征。这样，我们需要将前面[clip_image008[4]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034298207.png)公式中的内积从[clip_image010[16]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034294270.png)，映射到[clip_image012[42]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/20110318203429333.png)。

至于为什么需要映射后的特征而不是最初的特征来参与计算，上面提到的（为了更好地拟合）是其中一个原因，另外的一个重要原因是样例可能存在线性不可分的情况，而将特征映射到高维空间后，往往就可分了。（在《数据挖掘导论》Pang-Ning Tan等人著的《支持向量机》那一章有个很好的例子说明）

将核函数形式化定义，如果原始特征内积是[clip_image014[4]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034291172.png)，映射后为[clip_image016[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034309711.png)，那么定义核函数（Kernel）为

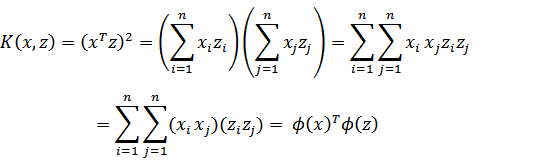
[clip_image018[8]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034302186.png)

到这里，我们可以得出结论，如果要实现该节开头的效果，只需先计算[clip_image020[10]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034315500.png)，然后计算[clip_image022[10]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034315991.png)即可，然而这种计算方式是非常低效的。比如最初的特征是n维的，我们将其映射到[clip_image024[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034326481.png)维，然后再计算，这样需要[clip_image026[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034321398.png)的时间。那么我们能不能想办法减少计算时间呢？

先看一个例子，假设x和z都是n维的，

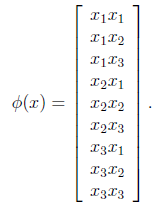
[clip_image028[4]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034331889.png)

展开后，得

[](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034337396.png)

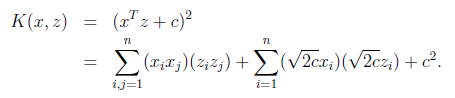
这个时候发现我们可以只计算原始特征x和z内积的平方（时间复杂度是O(n)），就等价与计算映射后特征的内积。也就是说我们不需要花[clip_image026[7]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034341266.png)时间了。

现在看一下映射函数（n=3时），根据上面的公式，得到

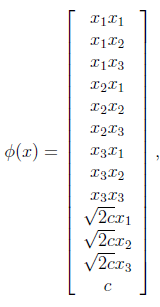
[](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034345660.png)

也就是说核函数[clip_image033[4]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034359215.png)只能在选择这样的[clip_image004[11]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034356118.png)作为映射函数时才能够等价于映射后特征的内积。

再看一个核函数

[](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034362464.png)

对应的映射函数（n=3时）是

[](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034367414.png)

更一般地，核函数[clip_image037[4]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034372921.png)对应的映射后特征维度为[clip_image039[4]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034378984.png)。（求解方法参见[http://zhidao.baidu.com/question/16706714.html](http://zhidao.baidu.com/question/16706714.html" \t "_blank)）。

由于计算的是内积，我们可以想到IR中的余弦相似度，如果x和z向量夹角越小，那么核函数值越大，反之，越小。因此，核函数值是[clip_image020[11]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034377838.png)和[clip_image041[4]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/20110318203438804.png)的相似度。

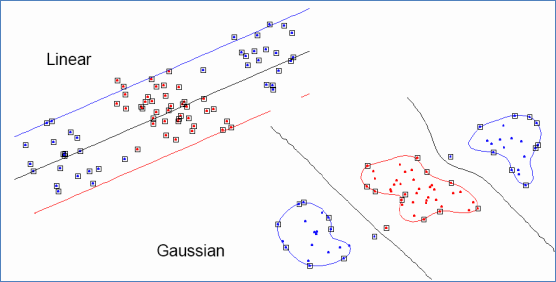
再看另外一个核函数

[clip_image042[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034396834.png)

这时，如果x和z很相近（[clip_image044[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034396245.png)），那么核函数值为1，如果x和z相差很大（[clip_image046[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034398720.png)），那么核函数值约等于0。由于这个函数类似于高斯分布，因此称为高斯核函数，也叫做径向基函数(Radial Basis Function 简称RBF)。它能够把原始特征映射到无穷维。

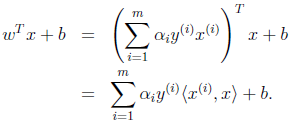
既然高斯核函数能够比较x和z的相似度，并映射到0到1，回想logistic回归，sigmoid函数可以，因此还有sigmoid核函数等等。

下面有张图说明在低维线性不可分时，映射到高维后就可分了，使用高斯核函数。

[](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/20110318203440606.png)

来自Eric Xing的slides

注意，使用核函数后，怎么分类新来的样本呢？线性的时候我们使用SVM学习出w和b，新来样本x的话，我们使用[clip_image050[8]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034414967.png)来判断，如果值大于等于1，那么是正类，小于等于是负类。在两者之间，认为无法确定。如果使用了核函数后，[clip_image050[9]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034419394.png)就变成了[clip_image052[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034419045.png)，是否先要找到[clip_image054[8]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034424312.png)，然后再预测？答案肯定不是了，找[clip_image054[9]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034436721.png)很麻烦，回想我们之前说过的

[](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034435783.png)

只需将[clip_image057[4]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034447701.png)替换成[clip_image059[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034443765.png)，然后值的判断同上。

# 4核函数有效性判定

**问题：给定一个函数K，我们能否使用K来替代计算[clip_image022[11]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034446240.png)，也就说，是否能够找出一个[clip_image061[12]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034459554.png)，使得对于所有的x和z，都有[clip_image018[9]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034452553.png)？**

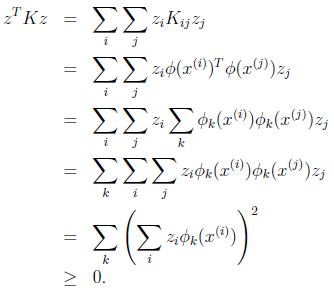
比如给出了[clip_image063[8]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034451124.png)，是否能够认为K是一个有效的核函数。

下面来解决这个问题，给定m个训练样本[clip_image065[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034464472.png)，每一个[clip_image067[8]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034463882.png)对应一个特征向量。那么，我们可以将任意两个[clip_image067[9]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034472388.png)和[clip_image069[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/20110318203448337.png)带入K中，计算得到[clip_image071[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034489191.png)。i可以从1到m，j可以从1到m，这样可以计算出m\*m的核函数矩阵（Kernel Matrix）。为了方便，我们将核函数矩阵和[clip_image073[10]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034491110.png)都使用K来表示。

如果假设K是有效地核函数，那么根据核函数定义

[clip_image075[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034496617.png)

可见，矩阵K应该是个对称阵。让我们得出一个更强的结论，首先使用符号[clip_image077[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034492124.png)来表示映射函数[clip_image020[12]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/20110318203450629.png)的第k维属性值。那么对于任意向量z，得

[](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034519417.png)

最后一步和前面计算[clip_image063[9]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/20110318203452954.png)时类似。从这个公式我们可以看出，如果K是个有效的核函数（即[clip_image073[11]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034529285.png)和[clip_image080[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034533679.png)等价），那么，在训练集上得到的核函数矩阵K应该是半正定的（[clip_image082[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034546121.png)）

这样我们得到一个核函数的必要条件：

K是有效的核函数 ==> 核函数矩阵K是对称半正定的。

可幸的是，这个条件也是充分的，由Mercer定理来表达。

|  |
| --- |
| Mercer定理：  如果函数K是[clip_image084[26]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034544104.png)上的映射（也就是从两个n维向量映射到实数域）。那么如果K是一个有效核函数（也称为Mercer核函数），那么当且仅当对于训练样例[clip_image065[7]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034545707.png)，其相应的核函数矩阵是对称半正定的。 |

Mercer定理表明为了证明K是有效的核函数，那么我们不用去寻找[clip_image061[13]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034555674.png)，而只需要在训练集上求出各个[clip_image086[6]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034565575.png)，然后判断矩阵K是否是半正定（使用左上角主子式大于等于零等方法）即可。

许多其他的教科书在Mercer定理证明过程中使用了[clip_image088[16]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034569969.png)范数和再生希尔伯特空间等概念，但在特征是n维的情况下，这里给出的证明是等价的。

核函数不仅仅用在SVM上，但凡在一个模型后算法中出现了[clip_image090[4]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034574330.png)，我们都可以常使用[clip_image073[12]](http://images.cnblogs.com/cnblogs_com/jerrylead/201103/201103182034584264.png)去替换，这可能能够很好地改善我们的算法。