**SVM**

04

支持向量机（Support Vector Machine，SVM）最初是用于两类分类问题的线性模型，后来通过核技巧扩展到非线性模型。同Logistic回归直接从后验概率出发不同，SVM从几何角度出发，寻找特征空间上间隔最大的分类器。将SVM的思想应用到回归任务，得到支持向量回归。

本章从线性SVM出发，再引入核技巧，将线性SVM扩展到核化SVM。我们也将学习SVM的优化求解算法以及支持向量回归。最后通过应用案例介绍Scikit-Learn中SVM 的API。

## **标题2** 4.1 **SVM**的基本型

### **4.1.1** 最大间隔（**Maximize Margin**）

要理解SVM，我们先来看线性分类器和间隔的概念。

考虑一个两类的分类问题，数据点用表示，为一个维向量；类别用表示，可以取值为1或者-1，分别代表两个不同的类。线性分类器的目标是在维空间中找到一个分类超平面，将两个类分开。分类超平面的方程表示为：

|  |  |
| --- | --- |
| 。 | （4-1） |

图4-1是一个二维特征空间中的线性分类的示意图。二维平面（每个样本有两维特征）上有两种不同的点，一种为蓝色的“-”，另一种为红色的“+”。我们要在这个二维平面上找到一个超平面，在二维空间中可以是图中的黑色实线。从图中可以看出，以这条黑色线为界，将红色“+”和蓝色“-”分开，在超平面一边的数据点所对应的全是-1（蓝色“-”），而在另一边全是1（红色“+”）。

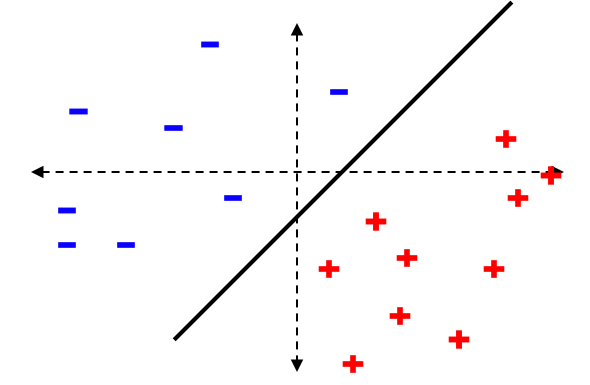


图4-1 二维特征空间中的线性分类示意图。

令分类判别函数为：

|  |  |
| --- | --- |
|  | （4-2） |

显然，如果，那么是是分类超平面上的点。我们不妨要求所有满足 的点，其对应的等于；，对应的数据点。

下面我们先假设数据是线性可分的，即假设存在这样一个超平面，超平面的每一侧都是同一类样本。一个点距离超平面的远近可以表示为分类预测的置信度或准确程度。在超平面确定的情况下，可表示点到距离超平面的距离，而的符号与类别标签的符号是否一致表示分类是否正确，所以，可以用表示分类的正确性和置信度。由此，我们引出样本到分类决策面的距离。

如图4-2所示，对于一个点，令其在分类决策超平面上的垂直投影为 ， 是垂直于超平面的一个向量，为样本到分类决策面的距离，则根据平面几何知识，有：

其中示的L2范数（模长），是单位向量。由***于***是超平面上的点，所以满足，即：

。

计算点 对应的为：

所以对于一个点到分类决策超平面的距离为：

|  |  |
| --- | --- |
|  | （4-3） |

原点到超平面的距离为：

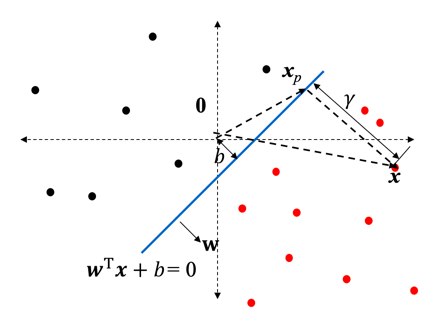
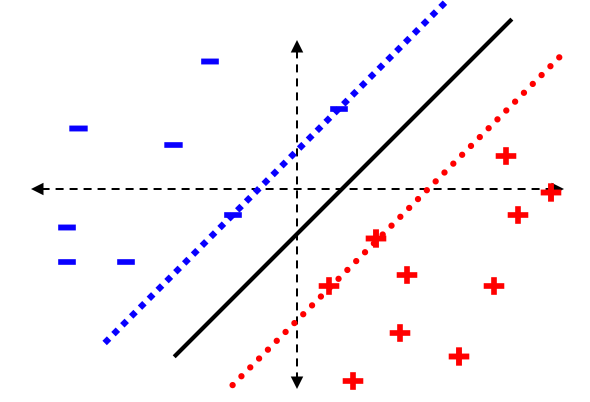
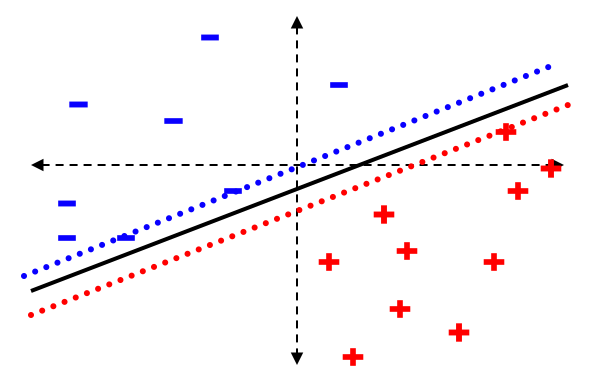


图4-2 二维特征空间中的判别函数示意图。

由上，我们已经知道，一个点到分类决策超平 的距离的绝对值（或）越大，分类的置信度越大。对于一个包含 个点的数据集，我们很自然地希望每个类别的样本到决策超平面的最小距离越大越好。如图4-3（a）和（b）中有两个不同的决策超平面，（a）中两个类到超平面的最小距离比（b）中两个类到超平面的最小距离大，我们期望（a）中的超平面的分类效果更好。因为第一个超平面能将两类分得更开，这样即使测试样本在训练样本的基础上有一些小的变化，这个超平面仍能将两类分开，泛化能力好。

  （a） （b）

图4-3 类别到超平面的距离。

我们定义间隔（margin）为个样本点到分类决策面的最小距离：

|  |  |
| --- | --- |
| ， | （4-4） |

为了使得分类的置信度高，我们希望所选择的超平面能够最大化。为了使得间隔最大，分类超平面应该位于两类的正中间（垂直平分）。由于，我们令离超平面最近点的（正类，图4-4中红色虚线上的点）或（负类，图4-4中蓝色虚线上的点），这些点被称为支持向量。关于为什么叫支持向量请参见稍后的SVM求解的KKT条件。

将 代入距离计算公式（4-3），得到：

|  |  |
| --- | --- |
|  | （4-5） |

这是正类样本到决策面的最短距离，即。考虑有正负两类样本，我们用正类样本到超平面的最短距离与负类样本到超平面的最短距离之和表示分类超平面的间隔（图4-4中红色虚线和蓝色虚线之间的距离，红色实线段长度），即：

|  |  |
| --- | --- |
|  | （4-6） |

因此我们最大间隔分类器（Maximum Margin Classifier）的目标函数为：

|  |  |
| --- | --- |
|  | （4-7） |

上述最大化优化问题等价于下面的最小化问题：

|  |  |
| --- | --- |
|  | （4-8） |

这就是SVM的基本型。

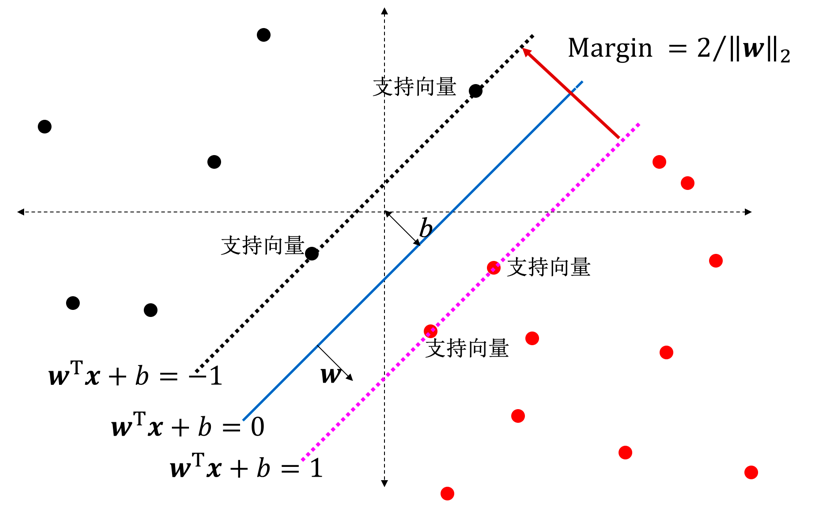


图4-4 最大间隔分类器。

SVM和Logistic回归模型均为线性分类器，判别函数为。但Logistic回归是从概率出发，而SVM从几何出发，直接根据判别函数的值进行分类，和概率没有直接联系，我们称SVM为非概率模型。

### **4.1.2 SVM**的对偶问题

SVM的目标函数（4-8）是一个凸二次规划问题，可以用现成的优化计算包求解。不过对SVM，我们有更高效的方法。

SVM的目标函数为带约束的优化问题，可采用拉格朗日乘子法变成非约束的优化问题，再进一步转换成对偶问题（dual problem）。通过求解与原问题等价的对偶问题得到原始问题的最优解，这样做的优点在于：（1）对偶问题往往更容易求解；（2）可以很自然地引入核函数，从而将线性模型推广成非线性分类模型。

SVM的原问题

对应的拉格朗日函数为：

|  |  |
| --- | --- |
|  | （4-9） |

令

根据对偶算法，目标函数变为：

|  |  |
| --- | --- |
|  | （4-10） |

其中表示问题的最优值。这个问题和我们最初的问题是等价的，称为原问题。将最小和最大的位置交换，得到对偶问题：

|  |  |
| --- | --- |
|  | （4-11） |

对偶问题的最优值，即提供了原问题最优值 的一个下界。在满足卡罗需-库恩-塔克（Karush–Kuhn–Tucker，KKT）条件的情况下，两者相等，这时我们可以通过求解对偶问题来间接求解原问题。

在对偶问题中，首先求。分别令 和 等于零：

|  |  |
| --- | --- |
|  | （4-12） |
|  |

将式（4-12）的结论代入式(4-9)，得到：

此时我们得到关于对偶变量 的优化问题：

|  |  |
| --- | --- |
|  | （4-13） |

此时的拉格朗日函数只包含了一个变量，很方便计算对偶问题中外部的最大化部分。

计算出后，便能求出 ，从而得到分类判别函数：

|  |  |
| --- | --- |
|  | （4-14） |

这个形式的有趣之处在于，对于新点的预测，只需要计算它与训练数据点的内积即可（这里 <⋅,⋅> 表示向量内积）。这一点至关重要，是稍后我们采用核方法进行非线性推广的基本前提。此外，支持向量也在这里显示出来：所有非支持向量所对应的系数，因此对于新点的内积计算实际上只要针对少量的“支持向量”而不是所有的训练数据。

为什么非支持向量对应的等于零呢？直观上理解，就是这些“后方”点对超平面是没有影响的。由于分类完全由超平面决定，所以这些无关的点并不会参与分类问题的计算。这个结论也可由SVM超平面的推导得出。回忆一下拉格朗日乘子法得到的目标函数：

对应的KKT条件（KKT条件的详细介绍请见附录B）：

|  |  |
| --- | --- |
|  | （4-15） |

训练样本可分为两类：

* 支持向量：支持向量到超平面的距离等于1，即。所以要满足式（4-15）的第3个约束，必须；
* 非支持向量：非支持向量到超平面的距离大于1，即，所以要满足式（4-15）的第3个约束，必须。从式（4-14）可以看出，由于，非支持向量对预测函数没有贡献。

## **标题2** 4.2 带松弛因子的**SVM**（**C-SVM**）

第4.1节中我们讨论的是数据是线性可分的。但实际应用可能由于数据中混入了异常点，导致不能线性可分，如图4-5（a）所示。或者数据仍然可以分开，但是会严重影响模型的泛化预测效果。如图4-5（b）所示，如果我们不考虑异常点，SVM的超平面如（b）中的黑色线所示；但是由于有一个蓝色的异常点，导致我们学习到的超平面是（c）中的黑线所示，这样会严重影响我们的分类模型预测效果（右边分类面对应的间隔小）。

缓解上述问题的办法之一是允许模型在一些样本上出错。为此，引入“软间隔”的概念。

|  |  |  |
| --- | --- | --- |
|  |  | |
| （a） | （b） | （c） |

图4-5 软间隔分类器。

我们回顾一下如式（4-8）所示的硬间隔最大化的SVM：

在硬间隔最大化中，要求所有样本都必须被正确分类，即。而软间隔允许某些样本不满足约束该约束，为此对每个样本引入了一个松弛变量 ，使函数间隔加上松弛变量大于等于1，即：

。

对比硬间隔最大化，可以看到我们对样本到超平面的函数距离的要求放松了。硬间隔要求一定要大于等于1，现在只需要加上一个大于等于0的松弛变量能大于等于1就可以了。当然，松弛变量是有成本的，每一个松弛变量，对应了一个代价，这个就得到了我们的软间隔最大化的SVM的最小化优化目标：

|  |  |
| --- | --- |
|  | （4-16） |

也就是说，我们希望尽量小，同时误分类的点尽可能的少。其中为损失的惩罚参数。显然，越大，对误分类的惩罚越大，时，迫使所有样本满足硬约束（松弛变量均为0）；越小，对误分类的惩罚越小。被误分的点的，因此为被误分点的数目的上界，可视为训练误差。

目标函数（4-15）的优化和第4.1节中线性可分SVM的优化方式类似，通过拉格朗日乘子法变成对偶问题求解。

带松弛变量的SVM的原问题（4-16）对应的拉格朗日函数为：

|  |  |
| --- | --- |
|  | （4-17） |

首先让关于最小化。令 ，和等于：

代入式（4-17）得到：

此时我们得到关于对偶变量的优化问题：

|  |  |
| --- | --- |
|  | （4-18） |

式（4-18）的目标函数与硬间隔SVM的目标函数式（4-13）相同，只是约束条件有所不同：硬间隔SVM的约束条件为，而软间隔SVM的约束条件为。这是因为软间隔SVM的约束条件为：

再加上

综合，我们可以消去，只留下，得到。

类似硬间隔的SVM，求解出后便能求出，从而得到分类判别函数：

|  |  |
| --- | --- |
|  | （4-19） |

对偶问题的目标函数（4-18）对应的KKT条件：

|  |  |
| --- | --- |
|  | （4-20） |

与硬间隔SVM类似，对训练样本，，所以：

1．：为非支持向量；

2． ：是支持向量：

（1）：则，进而根据，则恰好在最大间隔边界上；

（2）：则，此时：

若，则落在在最大间隔边界内部，，仍能被正确分类；

若，则落在在最大间隔边界外部，，样本被错误分类。

## **标题2** 4.3 合页损失函数（**Hinge Loss**）

C-SVM中，

|  |  |
| --- | --- |
|  |  |

若当时，定义；否则，即：

|  |  |
| --- | --- |
|  |  |

其中

|  |  |
| --- | --- |
|  |  |

这样C-SVM的目标函数可写为：

|  |  |
| --- | --- |
|  | （4-21） |

或者将代入，得到最小化目标函数为：

|  |  |
| --- | --- |
|  | （4-22） |

将上式与Logistic 回归的目标函数

对比，可以发现二者的形式类似，只是第一项损失函数不同。Logistic回归取得是负log似然损失

|  |
| --- |
|  |

其中模型预测的标签为1的概率。

而SVM取得是合页损失（Hinge Loss）

|  |  |
| --- | --- |
|  | （4-23） |

分类中常用的损失函数如图4-6所示。SVM中的合页损失函数取名为“合页”是因为其形状如合页。事实上，负log似然损失和合页损失都可以看成是0/1损失

|  |  |
| --- | --- |
|  | （4-24） |

的近似，被称为代理损失函数（Surrogate Loss Function），因为0/1损失函数非凸、非连续，优化计算不方便。

SVM正是因为采用合页损失函数保持了支持向量机解的稀疏性，因为合页损失函数的零区域对应的正是非支持向量的普通样本，从而所有的普通样本都不参与最终超平面的决定。

采用合页损失的方式解释SVM模型，使得我们对SVM的理解可以放到机器学习模型的一般框架：

即SVM的最佳模型为与训练数据拟合得最好的最简单的模型，亦被称为结构风险最小化，与之对应的训练集上的损失函数值和被称为经验风险。同时同Logistic回归模型类似，SVM的正则项也可以取L1正则。并且L1正则从特征层面上可以得到稀疏解，起到特征选择的作用。

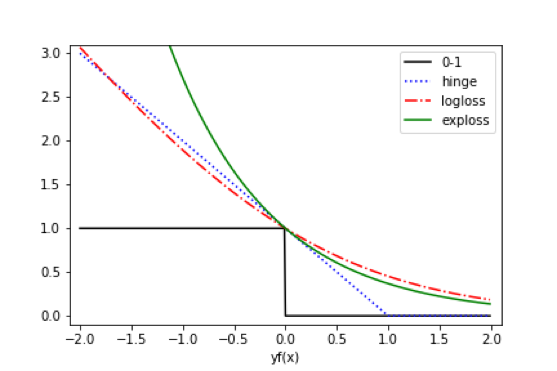


图4-6 分类任务的代理损失函数。

## **标题2** 4.4 核方法

前面我们讲到了线性可分SVM的硬间隔最大化和软间隔最大化的算法，当数据不完全线性可分时，线性模型表现不好。本节我们探讨SVM如何处理线性不可分的数据，重点讲述核方法将线性模型非线性化，从而能处理线性不可分数据。

### **4.4.1** 核技巧

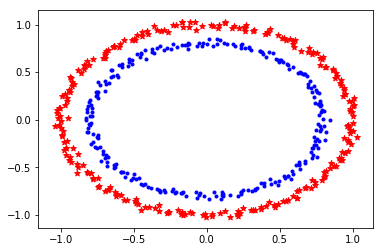
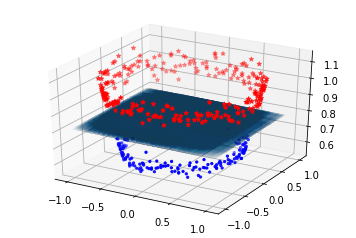
当数据不可分时，我们可以将数据升维到高维空间。在这个高维空间，数据也许就变得线性可分，这样我们可以用线性分类器就能实现分类任务。

例如，图4-7（a）给出了2维空间的两类数据，一类为外圈圆形数据为红色的星形数据点，另一类为内圈圆形数据为蓝色点状数据点。在这个2维平面上，我们找不到一个直线（2维空间中的线性模型为直线）可以将两类圆形数据完全分开。但将原始数据变换到如图4-7（b）所示的3维空间*,*其中，即表示2维空间中样本点到原点的距离。由于两类数据的半径不同，加入后，可以轻松地用一个平面（3维空间中的线性模型为平面）将两类数据分开。从上述例子可以看出，对于一维不线性可分的数据，我们将其映射到了两维以后，就变成了线性可分的数据。

假设我们用映射将映射成高维特征向量，则在特征空间中分类超平面对应的模型可表示为：

|  |  |
| --- | --- |
|  | （4-25） |

由于该模型在高维特征空间仍是线性模型，所以我们可以完全照搬第4.2节中线性SVM模型的推导过程，只需将原来的换成即可。

  （a） （b）

图4-7 2维空间线性不可分圆形数据扩展到3维空间，变得线性可分。

因此，SVM原问题的最小化优化目标可写为：

|  |  |
| --- | --- |
|  | （4-26） |

得到对偶变量的优化问题：

|  |  |
| --- | --- |
|  | （4-27） |

计算出便能求出，从而得到分类判别函数：

|  |  |
| --- | --- |
|  | （4-28） |

从上述推导过程可以看出，我们将线性模型中的原始特征的内积换成映射后特征的内积，就完成了将线性模型非线性化。看起来似乎这样我们就已经完美解决了线性不可分SVM的问题了。但在有些情况下，特征空间维数可能很高（甚至无穷维）才能将数据变得线性可分，而在高维空间直接计算特征空间的点积通常是困难的。

为了避开这个障碍，核函数隆重登场！

假设是一个从低维的输入空间X（欧氏空间的子集或者离散集合）到高维的希尔伯特空间的映射。那么如果存在函数，对于任意 X，都有：

|  |  |
| --- | --- |
|  | （4-29） |

那么我们就称为核函数。

这样，在推导SVM过程中高维特征内积 分别用核函数替代即可。核函数的计算是在低维特征空间计算的，避免了上面提到的在高维维度空间计算内积的计算量大的问题。也就是说，我们可以享受在高维特征空间线性可分的红利，却避免了高维特征空间繁重的内积计算量。这种技巧被称为核技巧（Kernel trick）。

### **4.4.2** 核函数构造

我们还需要解决一个问题，即核函数的存在性判断和如何构造。既然我们不关心高维度空间的表达形式，那么怎么才能判断一个函数是否是核函数呢？

定理：令为输入空间，是定义在X X上的对称函数，则是核函数，当且仅当对任意数据，如下核矩阵总是半正定的：

事实上，只要一个函数对应的矩阵是半正定，总可以找到一个与之对应的映射。也就是说任何一个核函数都隐式定义了一个再生核希尔伯特空间（Reproducing Kernel Hilbert Space，RKHL）。

通过前面讨论可知，我们希望样本映射到高维特征空间后是线性可分的，因此特征空间的好坏对支持向量机的性能至关重要。在核方法中，特征空间由核函数决定，因此核函数的选择对支持向量机的性能很重要。

常用的核函数有：

• 线性核函数

线性核函数对应线性SVM，表达式为：

|  |  |
| --- | --- |
|  | （4-30） |

这样我们可以将线性SVM和核化的SVM统一起来，区别仅仅在于线性可分SVM用的是线性核函数。

• 多项式核函数

多项式核函数表达式为：

|  |  |
| --- | --- |
|  | （4-31） |

其中为多项式核函数的参数。越大，多项式阶数越高，决策边界越不平滑。

• 高斯核函数

高斯核函数也被称为径向基核函数（Radial Basis Function，RBF），它是最主流的核函数：

|  |  |
| --- | --- |
|  | （4-32） |

其中为高斯核函数的宽度的倒数。越大，决策边界越不平滑。

需要注意的是，计算RBF核函数的值需要用，如果各维特征的取值范围不同，会影响特征在欧氏距离中的权值，从而影响RBF核函数的值。所以RBF核SVM需要对特征取值范围进行缩放（正则函数也需要对特征进行去量纲）。

对RBF核SVM，正则参数和RBF核函数参数共同控制模型复杂度。参数为核函数宽度的倒数，确定单个训练样本的影响范围，较小的值表示核函数宽度较宽，样本的影响范围“远”，此时决策函数相对平滑。参数是正则化参数，其值越小，表示训练误差在目标函数中的权重较低，可以接受决策函数分错一部分样本，决策边界更平滑。

图4-8给出了在鸢尾花分类任务上（只取了前2维特征），不同参数值的RBF核SVM分类器的交叉验证精度。可以看出，模型对参数非常敏感。如果太大，支持向量影响区域仅包括支持向量本身，此时即使控制进行正则也无法防止过拟合。当非常小时，模型过于受限，不能捕获数据的复杂性，支持向量的影响区域将包括整个训练集，此时模型将类似于一个线性模型。对于中间值，可以看到在和的对角线上可以找到好的模型。平滑模型（较低的值）可以通过增加正确分类每个样本点（较大的值）的重要性而变得更复杂，从而对角线对应的模型性能良好。对于的一些中间值，当变得非常大时，模型的性能基本相同。这提示我们不必通过强制减弱训练集的重要性来实现正则，RBF核的宽度本身就是一个很好的结构正则器。在实践中，我们可以用较低的值来简化决策函数，从而使用内存更少、预测速度更快。Scikit-Learn中默认的，其中为特征的维数，为训练样本组成的输入特征矩阵，为方差计算函数。

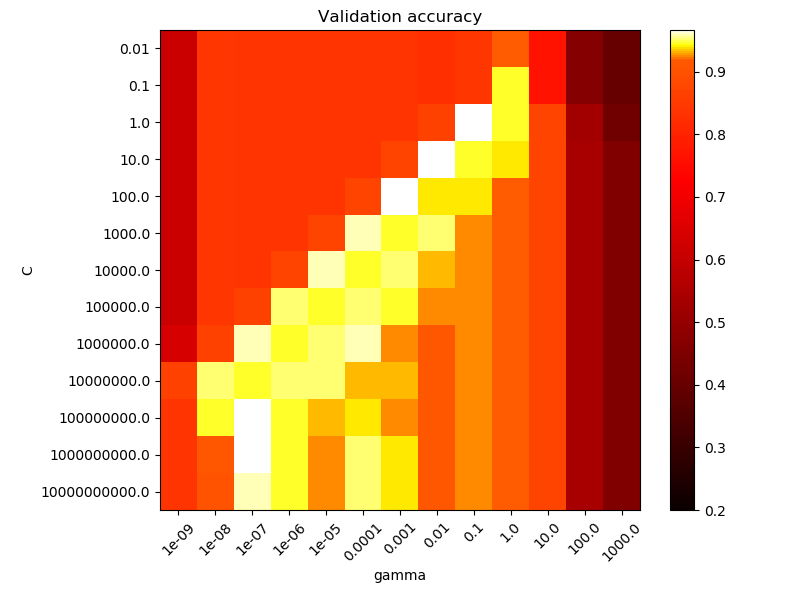


图4-8 RBF核SVM中参数和的影响。

此外，核函数还可以通过函数组合得到，例如：

如果和是核函数，则下列函数也是核函数：

* ，其中 和；
* ；
* ，其中为的多项式；
* 。

## **标题2** 4.5 **SVM**优化求解：**SMO**

### **4.5.1 SMO**算法原理

SVM的对偶问题：

是一个二次规划问题，可用通用的二次规划算法求解。然而该对偶问题的未知数规模与训练样本数目相同，这会在样本数目比较多时开销很大。为了避开这个障碍，人们通过利用问题本身的特性，提出了很多高效的算法。John Platt于1998年提出序列最小化优化算法（Sequential Minimal Optimization，SMO），是最快的二次规划优化算法，特别针对线性SVM和数据稀疏时性能更优。关于SMO详细资料请参考文献 [6]。

SMO算法将原始问题的求解个参数二次规划问题分解成多个二次规划问题求解，每个子问题只需要求解2个参数。根据坐标轴下降（在SVM的对偶问题中，我们求函数的极大值，应该是坐标轴上升）算法，我们应该固定其他参数，每次更新一个参数。但在SVM中，并不是完全独立，而是具有约束

因此一个改变，另一个也要随之变化以满足条件。所以在SMO算法中我们每次需要选择一对变量 一起修改。

SMO在整个二次规划的过程中：

• 选取一对参数

• 固定向量的其他元素，对求最优解，获得更新后的。

SMO不断执行这两个步骤直至收敛。

将SVM对偶问题的最大化优化目标转化为最小化优化目标：

|  |  |
| --- | --- |
|  | （4-33） |

假设我们选取的两个需要优化的参数为，剩下的 固定，作为常数处理。将SVM优化问题进行展开就可以得到（去掉与无关的项）。

|  |  |
| --- | --- |
|  | （4-34） |

令，且

|  |  |
| --- | --- |
|  | （4-35） |

则目标函数可简写为：

|  |  |
| --- | --- |
|  | （4-36） |

根据约束条件

可得到

|  |  |
| --- | --- |
|  | （4-37） |

式（4-37）两边同乘以，且，

|  |  |
| --- | --- |
|  | （4-38） |

将式（4-38）结论且代入式（4-36），消除，得到仅仅包含的式子：

我们需要对这个一元函数进行求极值，对的一阶导数为0得到：

整理得到：

|  |  |
| --- | --- |
|  | （4-39） |

因为SVM对数据点的预测值为：

，

则式（4-35）中以及的值可以表示成：

|  |  |
| --- | --- |
|  | （4-40） |

根据式（4-38）的结论，两边乘以，得到

|  |  |
| --- | --- |
|  | （4-41） |

将式（4-38）、式（4-41）代入（4-40），得到

将上述的值代入式（4-39），得到

等式右边的是更新前的值，记为，并且记预测值和真值之间的差异为，则上式可写成

令，得到的更新公式为

|  |  |
| --- | --- |
|  | （4-42） |

这样我们就得到了通过旧的获取新的 的表达式，可以通过得到。

SMO这种解析求解方法避免了二次规划数值解法的复杂迭代过程。不仅大大节省了计算时间，而且不会牵涉到迭代法造成的误差积累。

### **4.5.2** 对原始解进行修剪

上面我们通过对一元函数求极值的方式得到的最优是未考虑约束条件下的最优解。下面我们将公式（4-38）中的记为，即

但是在SVM中的是有约束的，即：

又由于只能取值1或者，这样在和形成的盒子里面，并且两者的关系直线的斜率只能为1或者，也就是说的关系直线平行于和形成的盒子的对角线，如图4-9所示。

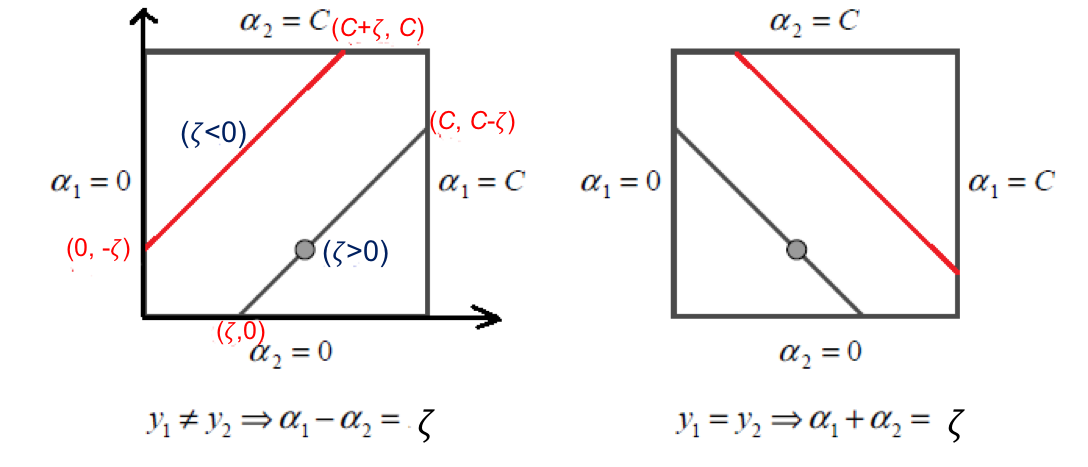


图4-9 SVM系数修剪。

由于的关系被限制在盒子里的一条线段上，所以两变量的优化问题实际上仅仅是一个变量的优化问题。不妨我们假设最终是的优化问题。当 时（4-9左图），线性限制条件可以写成：，根据的正负可以得到不同的上下界，因此统一表示成：

* 下界：
* 上界：

当 时（4-8右图），限制条件可写成: ，上下界表示成：

* 下界：
* 上界：

根据得到的上下界，我们可以得到修剪后的 ：

|  |  |
| --- | --- |
|  | （4-43） |

得到了 我们便可以根据 得到 :

|  |  |
| --- | --- |
|  | （4-44） |

### **4.5.3** 选择

SMO算法需要选择合适的两个变量做迭代，其余的变量做常量来进行优化，那么怎么选择这两个变量呢？

在SMO迭代的两个步骤中，只要中有一个违背了KKT条件，这一轮迭代完成后，目标函数的值必然会减小。通常而言，KKT条件违背的程度越大，迭代后的优化效果越明显，降幅越大。

与梯度下降类似，我们要找到使之优化程度最大的方向（变量）进行优化。所以SMO先选取违背KKT条件程度最大的变量，那么第二个变量应该选择使目标函数值减少最快的变量。SMO使用了一个启发式的方法，当确定了第一个变量后，选择使两个变量对应样本之间最大的变量作为第二个变量。直观来说，更新两个差别很大的变量，比起相似的变量，会带给目标函数更大的变化。

SMO算法称选择第一个变量为外层循环，这个变量需要选择在训练集中违反KKT条件最严重的样本点。对于每个样本点，要满足的KKT条件：

所以

一般来说，我们首先选择违反这个条件的点。如果这些支持向量都满足KKT条件，再选择违反的点和的点。

SMO算法称选择第二个变量为内层循环，假设我们在外层循环已经找到了，第二个变量的选择标准是让有足够大的变化（回忆：）。由于确定时，也确定了，所以要想最大，只需要在为正时，选择最小的作为；在为负时，选择最大的作为，可以将所有的保存下来加快迭代。

如果内存循环找到的点不能让目标函数有足够的下降，可以采用遍历支持向量点来做,直到目标函数有足够的下降，如果所有的支持向量做都不能让目标函数有足够的下降，可以跳出循环，重新选择。

### **4.5.4** 更新截距项

当我们更新了一对 之后都需要重新计算阈值，因为关系到的计算，关系到下次优化的时候误差的计算。

为了使得被优化的样本都满足KKT条件，当 不在边界，即，根据KKT条件可知相应的数据点为支持向量，满足：

|  |  |
| --- | --- |
|  | （4-45） |

式（4-45）两边同时乘上，由于，得到 ，进而得到：

由于，所以上式的前两项可以写成：

所以

|  |  |
| --- | --- |
|  | （4-46） |

当 ，同理可以得到的表达式：

|  |  |
| --- | --- |
|  | （4-47） |

当 和都有效的时候他们是相等的，即。

当两个乘子都在边界上，且时，和之间的值就是和KKT条件的阈值一致，SMO选择他们的中点作为新的阈值：

|  |  |
| --- | --- |
|  | （4-48） |

得到了我们需要更新：

其中S是所有支持向量的集合。

### **4.5.5 SMO**小结

SMO算法是一个迭代优化算法。在每一个迭代步骤中，算法首先选取两个待更新的向量，此后分别计算它们的误差项，并根据上述结果计算出和。最后再根据SVM的定义计算出偏移量。对于误差项而言，可以根据、和的增量进行调整，而无需每次重新计算。

**算法4-1：SMO**

1. 随机数初始化向量权重，，并计算偏移；
2. 初始化误差项；

3．选取两个向量作为需要调整的点；

4．令，其中；

5．如果, 令；

如果, 令。

6．令

7．利用更新的和，修改和的值。

8．如果达到终止条件，则停止算法；否则，转3。

SMO算法的终止条件是所有向量均满足KKT条件：

或者目标函数增长率小于某个阈值：

与通常的分解算法比较，尽管SMO可能需要更多的迭代次数，但每次迭代的计算量比较小，所以该算法表现出快速收敛性，且不需要存储核矩阵，也没有矩阵运算。

## **标题2** 4.6 支持向量回归（**SVR**）

### **4.6.1** 不敏感损失函数

在SVM分类模型中，我们的目标函数是让最小，同时每个训练样本尽量远离自己类别一边的支持向量，即。加入松弛变量时，目标函数是，对应的约束条件变成：。

在回归模型中，优化目标函数可以继续和SVM分类模型保持一致为，但是约束条件是让训练集中的每个点尽量拟合到一个线性模型。回归任务中我们通常采用均方误差（L2损失）作为损失函数，但在SVM中我们定义一个常量，对于某一个点，

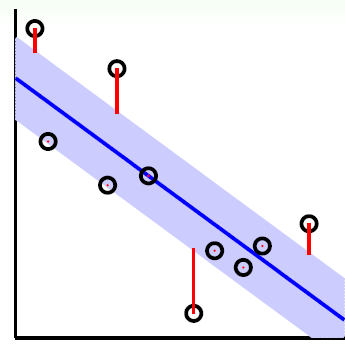


图4-10 SVR中的不敏感损失函数。

如果，则完全没有损失；

如果，则对应的损失为。

这个损失函数称为不敏感损失，如图4-10所示。在蓝色条带里面的点的损失为0，但是外面的点的是有损失的，损失大小为红色线段的长度。

## **标题2** 6.2 **SVR**

根据4.6.1小节定义的不敏感损失函数，得到SVR的目标函数：

|  |  |
| --- | --- |
|  | （4-49） |

和SVM分类模型相似，回归模型也可以对每个样本加入松弛变量。但是由于我们这里用的是绝对值，所以实际上是两个不等式，即两边都需要松弛变量。我们定义为，则我们SVM回归模型的损失函数度量在加入松弛变量之后变为：

|  |  |
| --- | --- |
|  | （4-50） |
|  |  |

依然和SVM分类模型相似，我们可以用拉格朗日乘子法得到拉格朗日函数：

|  |  |
| --- | --- |
|  | （4-51） |

首先将拉格朗日函数对于的偏导数：

|  |  |
| --- | --- |
|  |  |

将上面4个式子代入式（4-51），对 进行消元，最终得到对偶形式为：

对目标函数取负号，求最小值可以得到和SVM分类模型类似的求极小值的目标函数如下：

|  |  |
| --- | --- |
|  | （4-52） |

对应的KKT条件为：

|  |  |
| --- | --- |
|  | （4-53） |

根据可以看出，只有时，才能取非0值。换句话说，仅当样本落在间隔中，才能取非0值。此外，和不能同时成立，所以至少一个为0。这些样本我们称为支持向量。

求出后，将代入，得到回归函数为

|  |  |
| --- | --- |
|  | （4-54） |

如果采用核函数，回归函数为

|  |  |
| --- | --- |
|  | （4-55） |

## **标题2** 4.7 **SVM**案例分析**—Otto**商品分类

本节我们以Kaggle 2015年举办的奥拓商品分类竞赛数据为例，采用线性SVM和 RBF核的SVM实现商品分类。数据集描述请见3.7节。

在Scikit-Learn中，线性SVM模型可用或（参数）实现。其中采用liblinear工具包实现优化，采用工具包libsvm优化实现。SVM是非概率模型，所以不支持预测类别的概率输出。如果需要支持预测概率输出，在生成实例时参数需设置。RBF核的SVM在Scikit-Learn中用（参数）实现。

由于本案例训练样本数目较多（61878个样本），而SVM，尤其是RBF核SVM，在样本数目很多（如超过1万个样本）时，训练速度非常慢，因此我们不用交叉验证（）来对模型超参数进行调优，而是直接将数据分成80%做训练，20%做校验实现。这在Scikit-Learn中，可用函数实现。

线性SVM中，我们采用 L2正则，不同模型超参数对应模型在校验集上的性能（正确率）如图4-11（a）所示（原始特征），得到最佳超参数的值为，最佳正确率为0.765837。由于不支持概率输出，无法计算logloss，因此没有将其对测试集的预测结果提交Kaggle。

对RBF核SVM，我们采用TFIDF特征，正则项采用 L2正则，模型超参数包括正则参数和RBF核函数的核宽度。越小，决策边界越平滑；越小，决策边界也越平滑。设置超参数 的搜索范围为，的搜索范围为 。不同超参数对应模型在校验集上的性能（正确率）如图4-11（b）所示。由于当，最大值发生在处，为搜索范围的最右侧，需要继续扩大的搜索范围。不过继续扩大后，性能开始下降（0.790805）。因此得到最佳超参数组合为，对应正确率为0.817065，比线性SVM的性能（0.765837）好得多。

|  |  |
| --- | --- |
|  |  |
| （a） | （b） |

图4-11 Otto商品分类数据集上线性SVM模型和RBF核SVM模型验证集上的正确率。

支持预测概率输出，因此我们将RBF核SVM对测试集的预测结果提交Kaggle，得到其Private Leaderboard分数为：0.48877（排名1245位），比对应 Logistic线性回归模型的性能（0.63319）好得多，可能Otto商品分类任务是一个比较复杂的任务，非线性模型的性能更好。

## **标题2** 4.8 **SVM**案例分析**—共享单车骑行量预测**

这节我们在共享单车数据集上，采用RBF核的SVR实现共享单车骑行量的预测。数据说明和特征工程同2.7节相同。

在Scikit-Learn中，RBF核的SVM回归模型用（参数）实现，正则项采用 L2正则，模型超参数包括正则参数和RBF核函数的核宽度。从图4-8和图4-11（b）看出，超过1时效果很差，所以我们设置超参数 的搜索范围为。当较小时，最佳的的值更大，因此设置的搜索范围为 。不同超参数对应模型通过交叉验证得到均方误差如图4-12所示，得到最佳超参数的值为。该模型在测试集上均方根误差为689.787410，远小于线性的岭回归模型的均方根误差（785.595791），非线性的支持向量回归模型比线性回归模型性能好。

我们进一步扩大和的搜索范围，不同超参数下模型的性能如表4-1所示。从表中可以看出，回归任务中超参数和对模型性能的影响类似图4-8所示的分类任务。除了，其余情况下，不同对应的最佳模型性能相似，只是但最佳的不同。实际应用中，如果训练时间有限，可以固定（如采用Scikit-Learn的默认值），只对参数进行调优。在本案例中，，此时最佳的，虽然交叉验证集上的性能不是最佳，但在测试集上取得了最好的性能。

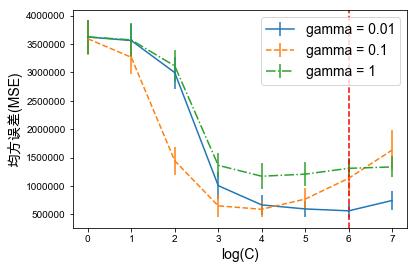


图4-11 共享单车数据集上RBF核SVR模型测试集的均方根上误差。

表4-1. 共享单车骑行数据集上不同超参数对应的SVR模型的性能

|  |  |  |  |
| --- | --- | --- | --- |
| gamma | 最佳的*C* | 交叉验证估计的均方误差 | 测试集上的均方根误差 |
| 0.001 |  | **560313.114361** | 704.857552 |
| 0.01 |  | 563809.767457 | 689.787410 |
| 0.03 |  | 573371.02247 | **670.161404** |
| 0.1 |  | 591020.388826 | 678.809423 |
| 1 |  | 1172568.501617 | 1075.563903 |

## **标题2** 4.9 **SVM**小结

SVM算法是一个很优秀的算法，在集成学习和深度神经网络算法没有表现出优越性之前，SVM占据着统治地位。在当前大数据时代的大样本背景下，SVM因其在大样本时超大的计算量，热度有所下降。

SVM算法的主要优点有：

1．解决高维特征的分类问题和回归问题很有效，在特征维度大于样本数时依然有好的效果。

2．仅仅使用一部分支持向量来做超平面的决策，无需依赖全部数据。

3．结合核函数使用，可以灵活地解决非线性的分类和回归问题。

4．对小样本情况尤其适用，分类准确率高，泛化能力强。

SVM算法的主要缺点有：

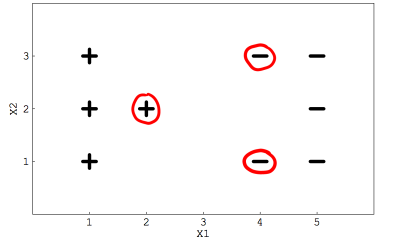
1．当样本量非常大和核函数映射维度非常高时，计算量过大。

2．核函数选择没有通用标准。

3．SVM对噪声数据敏感。

## **标题2** 4.9 习题

1. 假设您正在使用线性SVM分类器解决如下图所示的2类分类问题，其中一些点用红色圆圈圈出，代表支持向量。以下哪些选项是正确的？



（A）如果从数据中删除任一红色圆圈点，决策界限会改变；

（B）如果从数据中删除任一红色圆圈点，决策界限不会改变；

（C）如果从数据中删除任一非红色圆圈点，则决策边界会改变；

（D）如果从数据中删除任一非红色圆圈点，则决策边界不会改变。

1. SVM的有效性取决于：

（A）核函数选择

（B）核函数的参数

（C）软边距参数

（D）以上所有

1. 当出现下述哪种情况时，SVM性能不佳？

（A）数据线性可分

（B）数据干净

（C）数据有噪声

1. 假设您在SVM中使用具有高值的RBF核函数。这意味着什么？

（A）模型将考虑离超平面很远的点；

（B）模型将只考虑离超平面很近的点；

（C）模型不受点到超平面距离的影响。

（D）以上都不对

1. 假设您采用一个线性SVM模型来处理某个任务，并且知道这个SVM模型是欠拟合的。下列哪些方法可以提升该模型性能？

（A）减少训练样本

（B）增加训练样本

（C）增加特征

（D）减少特征

（E）增加参数

（F）减少参数

1. 如果我使用数据集的所有特征，在训练集的准确率达到100％，而验证集的准确率为70％，那么我应该注意什么？

（A）模型是欠拟合的；

（B）模型是完美的；

（C）模型是过拟合的。

1. 参照SVM的核技巧处理方式，将线性的Logistic回归模型转换成核Logistic回归模型。请分析核Logistic回归模型是否存在稀疏支持向量，在实际应用中为什么很少见到核Logistic回归模型。
2. 参照SVM的核技巧处理方式，将线性的岭回归模型转换成核岭回归模型。
3. 请用线性SVM和RBF核的SVM对3.9节的第10题的数据进行建模，并比较这些模型与Logistic回归模型的性能。
4. 请用线性SVR和RBF核的SVR对2.9节的第6题的数据进行建模，并比较这些模型与线性回归模型的性能。