稀疏核机培训讲义

# 一 准备工作

## 1.1 环境安装配置

本课程以实践为主，涉及大量工程代码实操。表1是本课程使用的开发环境清单。

表1 开发环境选型列表

| **类别** | **名称** | **版本** | **功能** |
| --- | --- | --- | --- |
| 开发软件 | Visual Studio Code | 1.68.0 | 代码编辑器，可以安装插件以支持更多功能 |
| 包管理器 | Anaconda3 | 2021.05 | 虚拟环境管理，自带python3.8 |
| 第三方库 | numpy | 2.3.0 | 科学计算基础库 |
| pandas | 1.18.0 | 数据分析和操作工具集 |
| matplotlib | 1.4.3 | 数据绘图可视化库 |
| scikit-learn | 3.5.2 | 机器学习数据分析工具集 |

在大型软件工程中需要使用到大量的第三方库。由于库在不同版本间存在API和功能差异，不同库之间存在复杂的依赖关系，将所有项目需要的库环境集中管理时容易出现库间互不兼容的情况。在工程研究实践中常常使用虚拟环境管理，以实现不同工程项目开发环境之间的隔离。

Anaconda3是一款数据科学领域的开源虚拟环境管理工具，相关信息见表2。

表2 Anaconda3信息一览表

| **条目** | **名称** |
| --- | --- |
| 软件名称 | Anaconda3 2021.05 |
| 官方网站 | https://www.anaconda.com/ |
| 官方文档 | https://anaconda.cloud/getting-started |
| 下载地址 | https://repo.anaconda.com/archive/Anaconda3-2021.05-Windows-x86.exe |

一些常用的Anaconda3指令示例如表3所示。遵循官方手册完成安装后，使用指令1创建虚拟环境，使用指令2进入虚拟环境，即可开始第三方库的安装。

表3 Anaconda3指令示例

| **序号** | **功能** | **指令内容** |
| --- | --- | --- |
| 1 | 创建名为py38的python3.8虚拟环境 | conda create --name py38 python=3.8 |
| 2 | 进入py38虚拟环境 | conda activate py38 |
| 3 | 退出虚拟环境 | conda deactivate |
| 4 | 罗列虚拟环境 | conda env list |
| 5 | 删除py38虚拟环境 | conda env remove --name py38 |

机器学习框架的首要设计目标是对开发者的整个工作流进行完整的编程支持。一个常见的机器学习任务一般包含以下流程：

1 **数据处理**：首先，用户需要数据处理API来支持将数据集从磁盘读入，对读取的数据进行预处理，从而可以将数据输入后续的机器学习模型中。

2 **定义模型结构**：完成数据的读取后，用户需要使用模型定义API来定义机器学习模型。这些模型带有模型参数，可以对给定的数据进行推理。

3 **损失函数和优化算法**：模型的输出需要和预期标签进行对比，这个对比差异一般通过损失函数来进行评估。因此，优化器定义API允许用户定义自己的损失函数，并且根据损失来使用各种优化算法来计算梯度，完成对模型参数的更新。

4 **训练**：给定数据集、模型、损失函数和优化器，用户需要使用训练API来定义一个循环从而将数据集中的数据按照小批量的方式读取出来，反复计算梯度来更新模型。该过程称为训练。

5 **测试和调试**：训练过程中，用户需要使用测试API来对当前模型的精度进行评估。当精度达到目标后，训练结束。这一过程中，用户往往需要调试API来完成对模型的性能和正确性进行验证。

Numpy是python语言的科学计算基础库，相关信息见表4。诸如pandas、scikit-image、opencv等第三方库都构建在Numpy的基础之上。

表4 Numpy信息一览表

| **条目** | **名称** |
| --- | --- |
| 软件名称 | numpy 1.18.0 |
| 官方网站 | https://numpy.org/ |
| 官方文档 | https://numpy.org/doc/1.18/index.html |
| 推荐安装方式 | pip install numpy==1.18.0 |

Pandas是python语言的数据分析工具包，相关信息见表5。

表5 Pandas信息一览表

| **条目** | **名称** |
| --- | --- |
| 软件名称 | pandas 1.4.3 |
| 官方网站 | https://pandas.pydata.org/ |
| 官方文档 | https://pandas.pydata.org/pandas-docs/version/1.4/index.html |
| 推荐安装方式 | pip install pandas==1.4.3 |

Matplotlib是python语言的可视化工具包，相关信息见表6。

表6 Matplotlib信息一览表

| **条目** | **名称** |
| --- | --- |
| 软件名称 | matplotlib 3.5.2 |
| 官方网站 | https://matplotlib.org/ |
| 官方文档 | https://matplotlib.org/3.5.2/api/index.html |
| 推荐安装方式 | pip install matplotlib==3.5.2 |

Scikit-learn是python机器学习工具包，相关信息见表7。

表7 Scikit-learn信息一览表

| **条目** | **名称** |
| --- | --- |
| 软件名称 | scikit-learn 1.1 |
| 官方网站 | https://scikit-learn.org/stable/index.html |
| 官方文档 | https://scikit-learn.org/1.1/modules/classes.html |
| 推荐安装方式 | pip install scikit-learn==1.1 |

## 1.2 稀疏核机的学习理论

在之前的学习中，我们研究了许多基于非线性核的学习算法。这种算法的一个最大的局限性是核函数k(xn，xm)必须对所有可能的训练点对xn和xm进⾏求值，这在训练阶段的计算上是不可行的，并且会使得对新的数据点进⾏预测时也会花费过多的时间。本章中，我们会看到具有稀疏（sparse）解的基于核的算法，从⽽对新数据的预测只依赖于在训练数据点的一个子集上计算的核函数。这通常被称为稀疏核机 (sparse kernel machine)。本章重点讨论支持向量机 (support vector machine, SVM)，这就是一种稀疏核机，下面会通过一个例子解释它的名称以及稀疏性来源。这是一种很流行的算法，常常被用来解决分类和回归问题。支持向量机的⼀个重要性质是模型参数的确定对应于一个凸优化问题，因此许多局部解也是全局最优解。

## 1.2.1支持向量机分类问题

## （1）最大边缘分类器

我们首先通过一个二分类问题引出支持向量机，然后再讨论其与核函数的关系。假设线性基函数模型为：

 公式（1）

其中***ϕ***(***x***)表⽰一个固定的特征空间变换，并且我们显式地写出了偏置参数*b*。注意，我们会简要介绍使用核函数表达的对偶形式，这避免了显式地在特征空间中进⾏计算。训练数据集由*N*个输⼊向量***x***1*,…,* ***x****N*组成，对应的目标值为*t*1*,…, tN*，其中**，新的数据点***x***根据*y*(***x***)的符号进行分类。

现阶段，我们假设训练数据集在特征空间中是线性可分的，即根据定义，存在⾄少一个参数w和b的选择方式，使得对于tn = +1的点，函数（1）都满⾜y(xn) > 0，对于tn = −1的点，都有y(xn) < 0，从而对于所有训练数据点，都有tny(xn) > 0。

当然，存在许多能够把类别精确分开的解。在这与感知器一节中的定义相同，在感知器算法中，它能够保证在有限步骤之内找到一个解。然而，它找到的这个解依赖于w和b的（任意的）初始值选择，还依赖于数据点出现的顺序。如果有多个能够精确分类训练数据点的解，那么我们应该尝试寻找泛化错误最小的那个解。支持向量机解决这个问题的方法是：引⼊**边缘（margin）**的概念，这个概念被定义为决策边界与任意样本之间的最⼩距离，如图1所⽰。

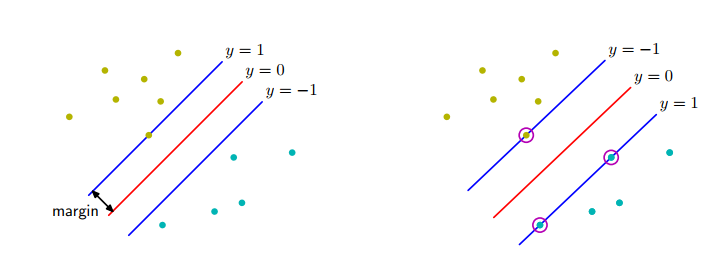


图1: 边缘被定义为决策边界与最近的数据点之间的垂直距离，如左图所⽰。最⼤化边缘会⽣成对决策边界的一个特定的选择，如右图所⽰。这个决策边界的位置由数据点的一个⼦集确定，被称为支持向量，⽤圆圈表⽰。

在⽀持向量机中，决策边界被选为使边缘最大化的那个决策边界。采用最⼤边缘解的动机可以通过计算学习理论（computational learning theory）或者统计学习理论（statistical learning theory）进行理解。然⽽， Tong and Koller（2000）给出了使⽤最大边缘解的⼀个简单的原因。他们考察了⼀个基于生成式⽅法和判别式方法组成的金字塔的分类框架，并且首先使⽤带有共同参数*σ*2的高斯核的Parzen密度估计对每个类别的输⼊向量***x***的分布进行建模。伴随着类别先验，这个分布定义了一个最优的分类错误率决策边界。然⽽，他们没有使用这个最优的决策边界，⽽是通过最小化学习到的模型的错误率来寻找最优的超平⾯。在极限的情况下，可以证明最优超平面是有着最大边缘的超平面。这个结果背后的直观含义是，随着*σ*2的减小，距离超平面较近的点对超平面的控制能力逐渐大于距离较远的点。在极限情况下，超平面会变得与非支持向量的数据点无关。这就是支持向量机稀疏性的来源，对于那些有决定意义的点，我们就称之为**支持向量**，因此这种方法就叫做支持向量机。

在线性分类中，根据图2，点***x***距离由*y*(***x***) = 0定义的超平⾯的垂直距离**。

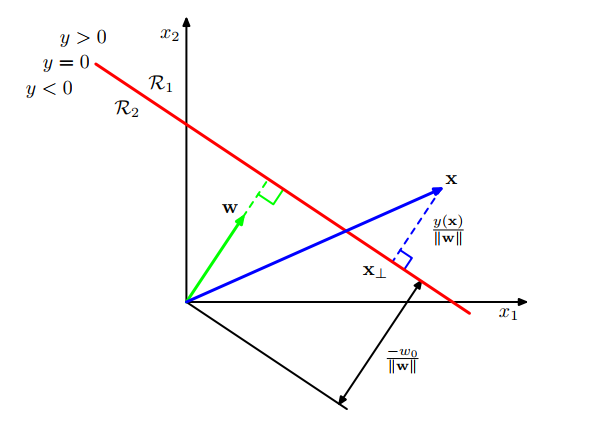


图2: 二维线性判别函数的几何表⽰。决策面（红色）垂直与***w***，它距离原点的偏移量由偏置参数*w*0控制。此外，一个⼀般的点***x***与决策面的有符号的正交距离为y(**x**)/|| **w**||。

其中*y*(***x***)的函数形式由公式（1）给出。此外，我们感兴趣的是那些能够正确分类所有数据点的解，即对于所有的*n*都有，因此点******距离决策面的距离为：

 公式（2）

边缘由数据集里垂直距离最近的点***xn***给出，我们希望最优化参数***w***和*b*，使得这个距离能够最大化。因此，最大边缘解可以通过下式得到：

 公式（3）

其中我们将因子提到了对n的最优化之外，因为w与n无关。直接求解这个最优化问题相当复杂，因此我们要把它转化为一个更容易求解的等价问题。为了完成这件事，我们注意到如果我们进行重新标度以及，那么任意点xn距离决策面的距离不会发生改变。我们可以使⽤这个性质，对于距离决策面最近的点，令：

 公式（4）

在这种情况下，所有的数据点会满足限制：

 公式（5）

这被称为**决策超平面的标准表示**。对于使上式取得等号的数据点，我们说限制被激活（active），对于其他的数据点，我们说限制未激活（inactive）。根据定义，总会存在至少一个激活限制，因为总会有一个距离最近的点，并且一旦边缘被最大化，会有至少两个激活的限制。这样，最优化问题就简化为了最大化，这等价于最小化，因此我们要在限制条件（公式5）下，求解最优化问题：

 公式（6）

公式（6）的因子的引⼊是为了后续计算⽅便。这是二次规划（quadratic programming）问题的一个例⼦，其中我们试图在一组线性不等式的限制条件下最⼩化二次函数。似乎偏置b从最优化问题中消失了。然⽽，它可以通过限制条件隐式地确定，因为这些限制条件要求的改变需要通过b的改变进行补偿。我们稍后会看到它是如何工作的。

为了解决这个限制的最优化问题，我们引入拉格朗日乘数≥ 0。公式（5）中的每个限制条件都对应着一个乘数。从而可得下面的拉格朗日函数：

 公式（7）

其中。注意拉格朗日乘数项前面的负号，因为我们要关于***w***和*b*最⼩化，关于最大化。令**关于***w***和*b*的导数等于零，我们得到了下面两个条件:

 公式（8）

 公式（9）

使用这两个条件从**中消去w和b，就得到了最⼤化边缘问题的对偶表示（dual representation），其中我们要关于a最大化：

 公式（10）

限制条件为：

 公式（11）

 公式（12）

这⾥，核函数被定义为。与之前⼀样，这是一个二次规划问题，其中我们要在不等式限制条件下最优化一个的二次函数。我们会在下一节讨论求解这种二次规划问题的⽅法。

## （2）二次规划问题

M个变量的二次规划问题的求解，通常的时间复杂度为O(M3)。通过将原始问题转化为对偶问题，我们将涉及到M个变量的最⼩化公式（6）的问题转化为了涉及到N个变量的对偶问题公式（10）。对于一组固定的基函数，其中基函数的数量M⼩于数据点的数量N，转化为对偶问题似乎没有什么好处。但是，对偶问题使得模型能够用核函数重新表⽰，因此最⼤边缘分类器可以被高效地应⽤于维数超过数据点个数的特征空间，包括无穷维特征空间。核公式也让核函数正定这一限制条件存在的原因变得更显然，因为这确保了拉格朗日函数有上界，从而使得最优化问题有良好的定义。

为了使用训练过的模型分类新的数据点，我们计算公式（1）定义的y(x)的符号。通过使⽤公式（8）消去w， y(x)可以根据参数和核函数表示，即：

 公式（13）

这种形式的限制的最优化问题须满足Karush-Kuhn-Tucker（KKT）条件。在这个问题中，下面三个性质要成立。

 公式（14）

 公式（15）

 公式（16）

因此对于每个数据点，要么，要么。任何使得的数据点都不会出现在公式（13）的求和式中，因此对新数据点的预测没有作用。剩下的数据点被称为**支持向量（support vector）。由于这些支持向量满足，因此它们对应于特征空间中位于最大边缘超平面内的点，**如上图1所示。这个性质是支持向量机在实际应用中的核心。一旦模型被训练完毕，相当多的数据点都可以被丢弃，只有支持向量被保留。

解决了二次规划问题，找到了a的值之后，注意到支持向量满足，我们就可以确定阈值参数b的值。使⽤公式（13），可得：

 公式(17)

其中*S*表示支持向量的下标集合。虽然我们可以使用任意选择的支持向量******解这个关于*b*的方程，但是我们可以通过下面的⽅式得到⼀个在数值计算上更加稳定的解。首先乘以，使⽤的性质，然后对于所有的支持向量，整理⽅程，解出*b*，可得：

 公式（18）

其中NS是⽀持向量的总数。对于接下来的模型⽐较，我们可以将最⼤边缘分类器⽤带有简单⼆次正则化项的最⼩化误差函数表⽰，形式为：

 公式（19）

其中是⼀个函数，当z ≥ 0时，函数值为零，其他情况下函数值为。这就确保了限制条件公式（5）成⽴。注意，只要正则化参数满⾜λ > 0，那么它的精确值就没有作⽤。

图3给出了一个分类问题的例⼦。分类⽤的模型使⽤支持向量机训练，训练数据是一个简单的⼈工生成的数据集，核函数是⾼斯核。虽然数据点在⼆维空间中显然不是线性可分的，但是它在隐式地由⾮线性核函数定义的⾮线性特征空间中是线性可分的。因此，训练数据点在原始数据空间中被完美地分开了。

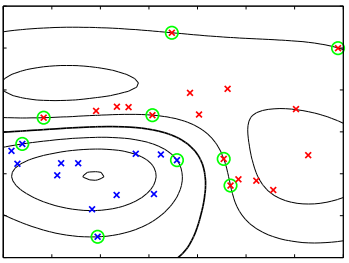


图3: ⼆维空间中来⾃两个类别的⼈⼯⽣成数据的例⼦。图中画出了具有⾼斯核函数的⽀持向量机的得到的常数*y*(***x***)的轮廓线。同时给出的时决策边界、边缘边界以及⽀持向量。

这个例⼦也从几何角度说明了SVM中稀疏性的来源。最大边缘超平面由⽀持向量的位置定义，其他数据点可以自由移动（只要仍然在边缘区域之外）⽽不改变决策边界，因此解与这些数据点⽆关。

## 1.2.2重叠类分布

⽬前为⽌，我们假设训练数据点在特征空间ϕ(x)中是线性可分的。解得的支持向量机在原始输⼊空间x中会对训练数据进⾏精确地划分，虽然对应的决策边界是⾮线性的。然而，在实际中，类条件分布可能重叠，这种情况下对训练数据的精确划分会导致较差的泛化能⼒。因此我们需要⼀种方式修改支持向量机，允许一些训练数据点被误分类。根据公式（7.19），我们看到在可以分开的类别的情况下，我们隐式地使⽤了一个误差函数。当数据点被错误分类时，这个误差函数等于无穷⼤，⽽当数据点被正确分类时，这个误差函数等于零，这样就将模型参数优化为了最⼤化边缘。我们现在修改这种方法，使得数据点允许在边缘边界的“错误侧”，但是增加⼀个惩罚项，这个惩罚项随着与决策边界的距离的增⼤⽽增⼤。对于接下来的最优化问题，令这个惩罚项是距离的线性函数⽐较⽅便。为了完成这一点，我们引⼊**松弛变量（slack variable） ξn ≥ 0，其中n = 1,…,N，每个训练数据点都有一个松弛变量**（Bennett, 1992; Cortes and Vapnik, 1995）。对于位于正确的边缘边界内部的点或者边界上的点， ξn = 0，对于其他点，。因此，对于位于决策边界y(xn) = 0上的点，ξn = 1，并且ξn > 1的点就是被误分类的点。这样，公式（7.5）给出的精确分类的限制条件就被替换为：

 公式(20)

其中松弛变量被限制为满足ξn ≥ 0。 ξn = 0的数据点被正确分类，要么位于边缘上，要么在边缘的正确一侧。 0 < ξn ≤ 1的点位于边缘内部，但是在决策边界的正确一侧。 ξn > 1的点位于决策边界的错误一侧，是被错误分类的点。如图4所示。

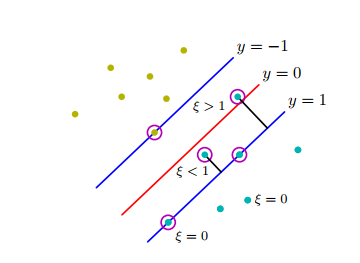


图 4: 松弛变量*ξn ≥* 0的说明。圆圈标记的数据点是⽀持向量。

这种方法有时被描述成放宽边缘的硬限制，得到⼀个软边缘（soft margin），并且允许⼀些训练数据点被错分。注意，虽然松弛变量允许类分布的重叠，但是这个框架对于异常点很敏感，因为误分类的惩罚随着ξ线性增加。

现在我们的目标是最⼤化边缘，同时以⼀种⽐较柔和的⽅式惩罚位于边缘边界错误⼀侧的点。于是，我们最⼩化：

 公式（21）

其中参数*C >* 0控制了松弛变量惩罚与边缘之间的折中。由于任何被误分类的数据点都有*ξn >* 1，因此∑*n ξn*是误分类数据点数量的上界。于是，参数*C*类似于（作用相反的）正则化系数，因为它控制了最⼩化训练误差与模型复杂度之间的折中。在的期限情况下，我们就回到了之前讨论过的用于线性可分数据的支持向量机。

## 1.2.3支持向量机回归问题

我们现在将支持向量机推广到回归问题，同时保持它的稀疏性。在简单的线性回归模型中，我们最⼩化一个正则化的误差函数：

 公式（22）

为了得到稀疏解，二次误差函数被替换为⼀个ϵ-不敏感误差函数（ϵ-insensitive error function）（Vapnik, 1995）。如果预测y(x)和⽬标t之间的差的绝对值⼩于ϵ，那么这个误差函数给出的误差等于零，其中ϵ > 0。 ϵ-不敏感误差函数的一个简单的例子是：

 公式（23）

它在不敏感区域之外，会有一个与误差相关联的线性代价。如图5所⽰。

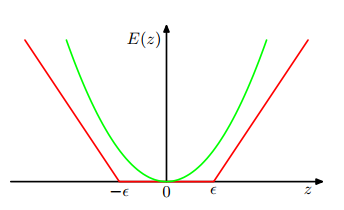


图 5: *ϵ*-不敏感误差函数（红⾊）的图像。在不敏感区域之外，误差函数值随着距离线性增⼤。作为对⽐，同时给出了⼆次误差函数（绿⾊）。

于是我们最⼩化正则化的误差函数，形式为：

 公式（24）

其中y(x)由公式（1）给出。按照惯例，（起着相反作用的）正则化参数被记作C，出现在误差项之前。

与之前⼀样，通过引⼊松弛变量的⽅式，我们可以重新表达最优化问题。对于每个数据点xn，我们现在需要两个松弛变量ξn ≥ 0和，其中ξn > 0对应于tn > y(xn) + ϵ的数据点， 对应于tn < y(xn) − ϵ的数据点，如图6所示。

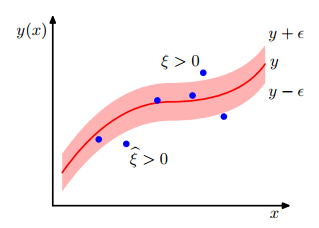


图6: SVM回归的说明。图中画出了回归曲线以及ϵ-不敏感“管道”。同时给出的是松弛变量ξ和的例⼦。对于ϵ-管道上⽅的点， ξ > 0且 = 0，对于ϵ-管道下⽅的点， ξ = 0且 > 0，对于ϵ-管道内部的点， ξ = = 0。

目标点位于ϵ-管道内的条件是yn − ϵ ≤ tn ≤ yn + ϵ，其中yn = y(xn)。引⼊松弛变量使得数据点能够位于管道之外，只要松弛变量不为零即可。对应的条件变为：

 公式（25）

 公式（26）

这样支持向量回归的误差函数就可以写成：

 公式（27）

它必须在限制条件ξn ≥ 0和和公式（25）和公式（26）下进⾏最小化。可以这样做：引⼊拉格朗日乘数an ≥ 0;; µn ≥ 0以及，然后最优化拉格朗日函数：

 公式（28）

我们现在使⽤公式（1）替换y(x)，然后令拉格朗⽇函数关于w, b,ξn和的导数为零，有：

 公式（29）

 公式（30）

 公式（31）

 公式（32）

使用这些结果消去拉格朗日函数中对应的变量，我们看到对偶问题涉及到关于**和**最大化：

 公式（33）

其中我们已经引⼊了核。与之前一样，这是一个具有限制条件的最大化问题。为了找到限制条件，我们注意到和必须成立，因为它们是拉格朗日乘数。并且和以及公式（31）和公式（32）要求且，因此我们又一次得到了盒限制：

 公式（34）

 公式（35）

以及条件公式（30）。

将公式（29）代⼊公式（1），我们看到对于新的输入变量，可以使用下式进⾏预测：

 公式（36）

这又一次被表示为核函数的形式。

对应的Karush-Kuhn-Tucker（KKT）条件说明了在解的位置，对偶变量与限制的乘积必须等于零，形式为：

 公式（37）

 公式（38）

 公式（39）

 公式（40）

根据这些条件，我们能得到一些有用的结果。⾸先，我们注意到如果ϵ + ξn + yn − tn = 0，那么系数an只能非零，这表明数据点要么位于ϵ-管道的上边界上（ξn = 0），要么位于上边界的上方（ξn > 0）。类似地，的⾮零值表示，这些点必须位于ϵ-管道的下边界上或者下边界的下⽅。

此外，两个限制ϵ + ξn + yn − tn = 0和是不兼容的。可以这样证明：将两式相加，注意到ξn和是⾮负的，⽽ξ是严格为正的，因此对于每个数据点xn，或者⾄少⼀个为零，或者都为零。

支持向量是对于由公式（36）给出的预测有贡献的数据点，换句话说，就是那些使得或者成⽴的数据点。这些数据点位于ϵ-管道边界上或者管道外部。管道内部的所有点都有。我们再次得到了一个稀疏解，在预测模型（36）中唯一必须计算的项就是涉及到⽀持向量的项。

参数b可以这样得到：考虑⼀个数据点，满⾜。根据公式（39），⼀定有ξn=0，根据公式（37），一定有ϵ+yn−tn=0。使⽤公式（1），然后求解b，我们有：

 公式（41）

其中我们使用了公式（29）。通过考虑⼀个满足的数据点，我们可以得到一个类似的结果。在实际应⽤中，更好的做法是对所有的这些b的估计进⾏平均。

与分类问题的情形相同，有另一种⽤于回归的SVM的形式。这种形式的SVM中，控制复杂度的参数有一个更加直观的意义（Schölkopf et al., 2000）。特别地，我们不固定不敏感区域ϵ的宽度，⽽是固定位于管道外部的数据点的⽐例ν。这涉及到最大化：

 公式（42）

限制条件为:

 公式（43）

 公式（44）

 公式（45）

 公式（46）

可以证明至多有*νN*个数据点落在不敏感管道外部，⽽至少有*νN*个数据点是支持向量，因此位于管道上或者管道外部。图7说明了使用支持向量机解决回归问题的⼀个例⼦，数据集使⽤的是正弦曲线数据集。这⾥参数*ν*和*C*已经⼿动选择完毕。在实际应⽤中，它们的值通常通过交叉验证的⽅法确定。

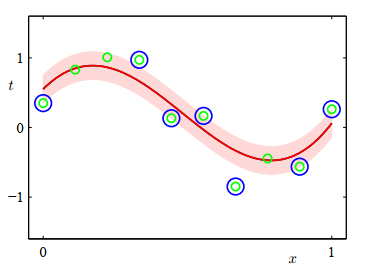


图 7: ν-SVM回归应⽤到⼈⼯⽣成的正弦数据集上的说明， SVM使⽤了⾼斯核。预测分布曲线为红⾊曲线， ϵ-不敏感管道对应于阴影区域。此外，数据点⽤绿⾊表⽰，⽀持向量⽤蓝⾊圆圈标记。