MLAPP-C7

7.线性回归

7.1 引言

线性回归是统计学和（监督）机器学习中的“主力”。当它与核方法或者其他形式的基函数拓展相配合时，它可以对非线性关系进行建模，关于这一点我们将在后面介绍。

7.2 模型说明

正如我们在1.4.5节所讨论的，线性回归的形式为：

 （7.1）

如果将上式中的**x**替换为关于输入的非线性函数，线性回归模型可以对非线性关系进行建模。也就是说，我们使用：

 （7.2）

这被称为**基函数拓展**（basis function expansion）。（值得注意的是，该模型关于参数**w**依然是线性的，所以它被称为线性回归；关于这一点的重要性将在后面体现。）一个简单的例子就是多项式基函数，其模型的形式为：

 （7.3）

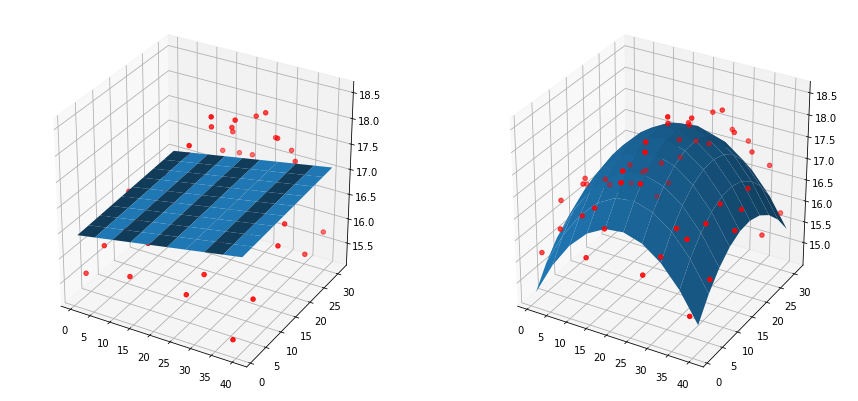


图7.1 2维数据中的线性回归。垂直坐标轴表示温度，水平轴表示房间中的坐标。（a）拟合曲线的形式为。（b）温度数据利用二次函数进行拟合。图形由程序**surfaceFitDemo**生成。

图1.18说明了改变*d*的影响：增加自由度*d*允许我们创造更加复杂的函数。我们也可以将线性回归应用在多个输入的情况，举例来说，考虑温度是一个关于坐标的函数。图7.1（a）绘制了，图7.1（b）绘制了。

如果我们的输入具备类别属性，其取值具有*K*种可能，我们需要使用哑编码的方式对输入进行编码（见2.3.2节）。如果所有的输入都是类别属性的，则模型被称为**方差分析**（analysis of variance, anova）。

7.3 最大似然估计（最小二乘法）

估计一个统计模型参数的最常用的方式是计算参数的最大似然估计或者MLE（见3.2.3节），定义为：

 （7.4）

一般情况下，我们假设训练样本是服从独立同分布的（简称iid）。这就意味着，我们可以将对数似然函数写成：

 （7.5）

相较于最大化对数似然函数，我们一般等价地使用最小化**负对数似然函数**（negative log likelihood, NLL）：

 （7.6）

使用NLL作为优化目标一般是比较方便的，因为许多优化工具包在设计的时候都是寻找函数的最小值而非最大值。

现在让我们将MLE方法应用在线性回归的情况。将高斯分布的定义引入，我们发现对数似然的结果为：

 （7.7）

 （7.8）

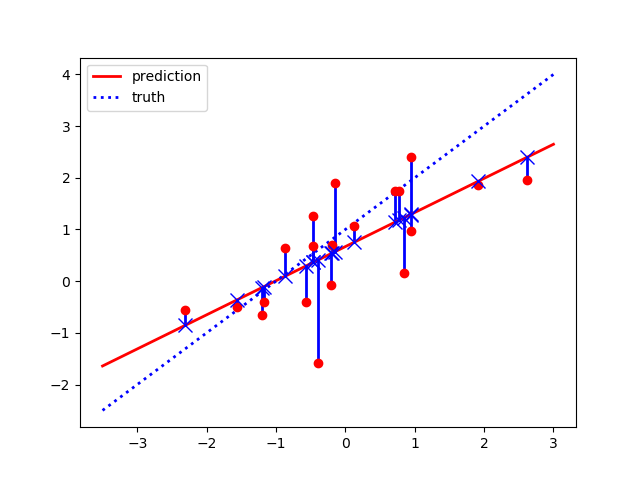
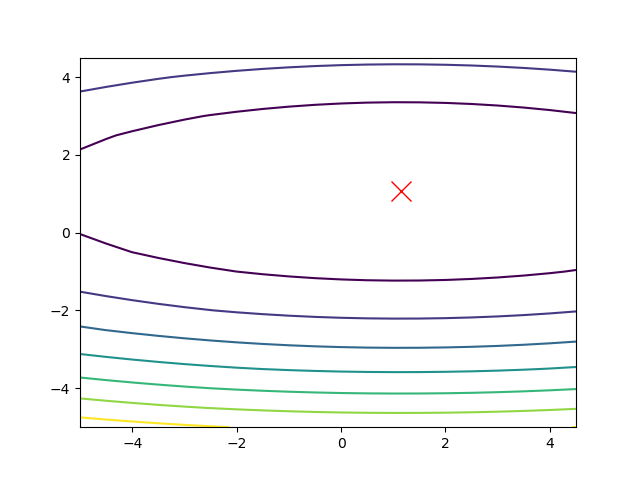
RSS表示**残差平方和**（residual sum of squares），定义为：

 （7.9）

RSS又被称为**误差平方和**（sum of squared errors，SSE），SSE/N被称为**均方差**（mean squared error, MSE）。它又可以写成残差向量的*l*2**范数**（norm）的平方：

 （7.10）

其中。

（a） （b）

图7.2. （a）在线性最小二乘中，我们尝试最小化每个训练样本（红色实心点）到它的近似点（蓝色×）的距离的平方，也就是最小化蓝色垂直线段的长度和。红色的对角线代表，它是通过最小二乘方法得到的拟合曲线。需要注意的是，那些残差直线与拟合的直线并非垂直关系。图像由程序**residualsDemo**生成。（b）相同案例下的RSS误差等高线。红色×代表MLE，**w**=（1.39,1.02）（**译者注：**在程序中我们使用了随机数，所以每次的结果可能不一样，此处重点关注误差的等高线）。图像由程序**contoursSSEdemo**生成。

我们发现参数**w**的MLE是让RSS最小的值，所以这种方法又被称为**最小二乘**（least squares）。图7.2（a）说明了这种方法。训练样本（*x*i, *y*i）以红色实心点显示，估计值以蓝色的×显示，残差以蓝色的垂直线段表示。我们的目标是找到这样的参数，使得平方差最小（即垂直的蓝色线段长度最小）。

在图7.2（b），我们绘制了线性回归例子中的NLL表面。我们发现它是一个“碗状”曲面，具有唯一的最小值，即我们需要推导的MLE。（更加重要的是，就算我们使用基函数拓展，比如多项式拟合，这一点也是正确的，因为NLL依然是参数**w**的线性函数，哪怕它与输入**x**呈非线性关系。）

7.3.1 MLE的推导

首先我们将NLL进行形式上改写，从而更方便求导：

 （7.11）

其中：

 （7.12）

为矩阵的**平方和**（sum of squares），同时

 （7.13）

使用4.10的结果，我们发现NLL关于**w**的梯度为：

 （7.14）

令上式等于0，得到：

 （7.15）

这被称为**正规方程**（normal equation）。上述线性方程组对应的解被称为**普通最小二乘**（ordinary least squares, OLS）解，等于：

 （7.16）

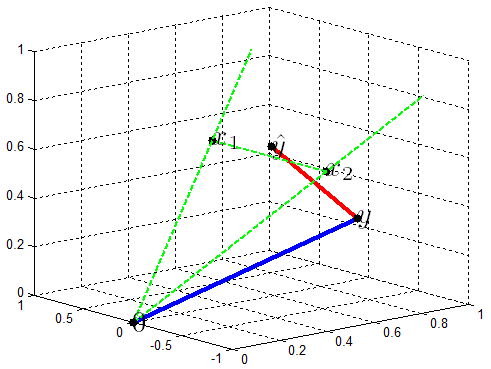


图7.3 最小二乘法的几何解释，其中N=3,D=2。和为3维空间中的向量；它们一起定义了一个2维平面。**y**也是3维空间中的向量，但不在2维平面中。**y**在该平面的正交投影为。红色直线为残差，其范数就是我们需要最小化的目标。为了可视化的效果，所有向量都已被转换为单位向量。（图像来自于原书）

7.3.2 几何解释

上述方程有一个简洁的几何解释，我们现在就解释这一点。假设*N*>*D*，也就是说我们的样本数量大于特征的数量。设计矩阵**X**的列定义了一个维度为*D*的空间，该空间为*N*维空间中的子空间（**译者注：**以3维空间为例，假设我们在三维空间中任意选择2个向量，这2个向量线性无关，显然由这2个向量可以构成一个平面的空间，我们称这个2维平面空间为3维空间的子空间）。令**X**的第*j*列为向量，且。（注意不要与向量混淆，表示第*i*个样本。）类似地，**y**为空间中的一个向量。举个例子，假设我们有*N*=3个样本，且*D*=2：

， （7.17）

这些向量在图7,3中给出说明。

我们需要在这个线性子空间中找到一个向量，使其尽可能靠近向量**y**，也就是说，我们希望找到满足下式的向量：

 （7.18）

既然，那么存在权重向量**w**，使得：

 （7.19）

为了最小化残差的范数，我们希望残差向量垂直于**X**的每一列，所以对于*j*=1:*D*都成立。所以：

 （7.20）

所以向量**y**的投影向量为：

 （7.21）

这被称为向量**y**在矩阵**X**构成的列空间（**译者注**：即矩阵的列向量构成的空间）中的**正交投影**（orthogonal projection）。投影矩阵被称为**帽子矩阵**（hat matrix），因为它为“向量**y**带上了帽子”。

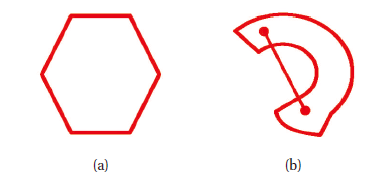


图7.4 （a）凸集的定义。（b）非凸集的定义

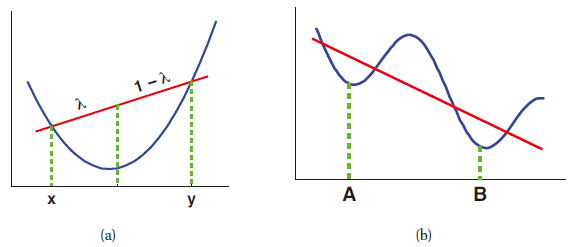


图7.5 （a）凸函数的示意图。我们发现(x,f(x))与(y,f(y))的连线处在函数的上方。（b）既不是凸函数也不是凹函数。**A**为局部最优解，**B**为全局最优解。（图形来自于原书）

7.3.3 凸函数

当我们讨论最小二乘时，我们提到NLL是具有唯一最小值的“碗状”。类似这样的函数具有一个专业术语：**凸函数**（convex）。凸函数在机器学习中十分重要。

让我们给出更加精确的定义。我们称集合是**凸**（convex）的，如果对于任意，我们有：

 （7.22）

也就是说，如果我们在之间绘制一条直线，那么直线上的所有点都应该在集合中。图7.4（a）给出了凸集的说明，图7.4（b）给出了非凸集的说明。

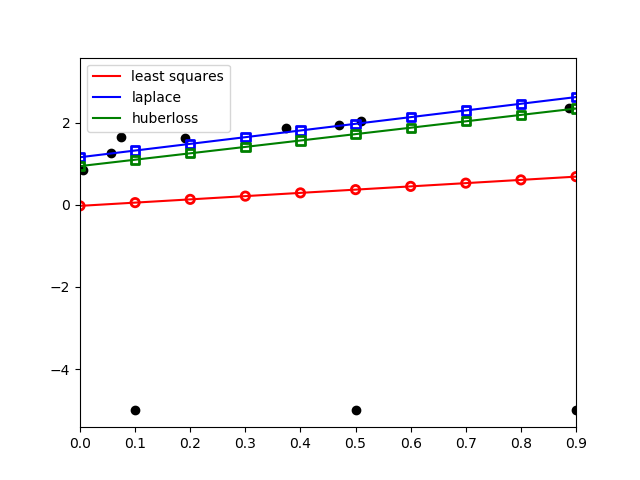
函数被称为是凸函数，如果它的**上图**（epigraph, 函数上方所有点构成的集合）定义了一个凸集。等价地，函数被称为是凸函数，如果它定义在凸集上且对于任意的，，我们有：

 （7.23）

图7.5给出了一个1维空间中的例子。一个函数被称为是**严格凸**（strictly convex）的，如果不等号是严格成立的。一个函数是**凹**（concave）的，如果是凸函数。标量凸函数的例子的有。标量凹函数的例子包括。

直觉上，一个（严格的）凸函数具备一个“碗状”形状，所以具备一个唯一的最优解，对应着“碗”的底部。所以它的二阶导数在任何地方都是正的，即。一个二阶连续的可微，多元函数*f*是凸函数，如果它的海森矩阵对任意都是正定的。在机器学习中，函数*f*通常就是指NLL。

如果模型的NLL是凸函数，那会是一件非常好的事情。因为这就意味着我们总可以找到全局最优解MLE。我们会在本书的后面内容看到许多例子。然而，许多我们感兴趣的模型并没有凹的似然函数（**译者注**：NLL不是凸函数）。在这种情况下，我们将会讨论推导出参数的局部最优估计值的方法。

（a） （b）

图7.6 （a）鲁棒线性回归的说明。图形由程序**linregRobustDemoCombined**生成。（b）*l*2,*l*1和huber损失函数的示意图（图形来自于原书）。

7.4 健壮的线性回归

在回归模型中，我们经常使用期望为0，方差为固定值的高斯分布对噪音进行建模，其中。在这种情况下，最大似然等价于最小化残差的平方和。然而，如果数据中存在**异常点**（outliers），这将会导致一个较差的拟合，图7.6（a）给出了说明（异常点在图形的底部。）。因为我们使用误差的平方作为目标函数，所以那些离拟合曲线更远的异常点对模型训练的影响要比离曲线近的点更大。

如果需要对异常点更具有鲁棒性，一种方式是改变响应变量服从高斯分布的假设，选择一种拥有**肥尾**（heavy tails）的分布。这类分布将给那些异常点更高的似然值，而不需要改变拟合的曲线去适应那些异常值。

一种可选的分布为2.4.4节介绍的拉普拉斯分布。如果我们使用该分布作为对观测值分布的建模，我们得到如下的似然：

 （7.24）

上式之所以具有鲁棒性是因为我们将高斯分布中的替换成。为简单起见，我们假设*b*是固定的值。令为第*i*个预测值的残差。NLL的形式如下：

 （7.25）

不幸地是，上式是一个非线性目标函数，所以很难进行优化。幸运地是，使用下面的**分离变量**（split variable）技巧。我们可以将其转换为含线性约束的线性目标函数。首先我们定义：

 （7.26）

然后添加线性不等式约束和。现在含约束的目标函数变成

 （7.27）

上式是一个**线性规划**（linear program）的例子，包含*D*+2*N*个未知量和3*N*个约束。

既然这是一个凸优化问题，那么它有一个唯一的解。为了解决这个线性规划问题，我们必须首先将它写成如下的标准形式：

 （7.28）

在当前例子中，

。上式可以通过任何线性规划求解包解决。图7.6（a）给出在实践中的例子。

一种替代基于拉普拉斯分布的NLL目标函数的方法是使用Huber loss作为优化目标，定义为：

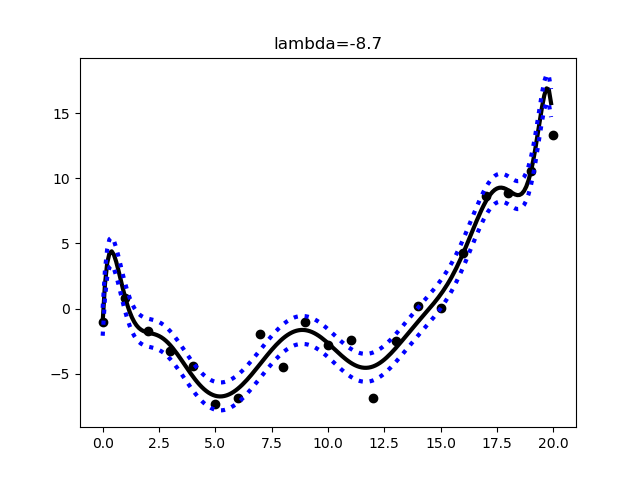
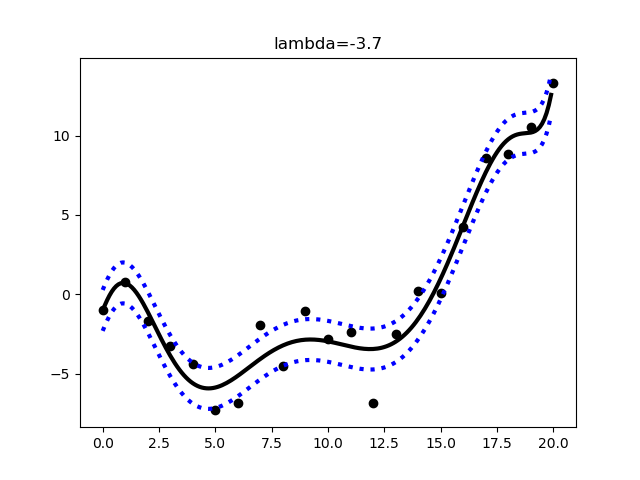
 （7.29）

当误差小于时，这种方法与*l*2方法等价，对于大于的误差，这种方法等价于*l*1方法。图7.6（b）给出了案例。基于的事实，我们知道这种损失函数的优势在于它处处可微。同时我们也可以发现该函数的一阶导数是连续函数，因为函数的两部分的导数在处匹配，即。因此优化Huber损失比使用拉普拉斯损失要快，因为我们可以使用标准的平滑优化方法（比如伪牛顿法）代替线性规划。

图7.6（a）给出了Huber损失函数的说明，结果与使用概率的方法比较相似。（事实上，结果表明Huber方法也具有一个概率方面的解释，尽管它显得不那么自然。）

7.5 岭回归

ML估计的一个问题在于其可能导致过拟合。本节，我们讨论一种缓解这个问题的方式，使用基于高斯先验分布的MAP估计。为了简单起见，我们使用高斯似然函数，而非一个健壮的似然函数。

（a） （b）

图7.7 阶数为14的多项式拟合，N=21。图中我们使用了*l*2正则。数据在生成的过程中加入的噪音的方差。图中的误差区间表示噪音的标准差，不难发现随着拟合曲线光滑程度的提高，误差区间变得更宽，这是因为我们将数据的变化更多地归咎于噪音。图形由程序**linregPolyVsRegDemo**生成。

7.5.1 基本思想

MLE容易过拟合的原因在于它挑选出最符合训练数据的参数；但如果数据是含噪的，函数会变得十分复杂。一个简单的例子，假设我们基于*N*=21个数据点，使用最小二乘方式去训练一个阶数为14的多项式。最终的曲线非常“扭曲”，正如7.7（a）所示。对应的最小二乘系数（除了w0）如下：

17.868, 24.993, -68.929, -570.145, -113.651, 6276.655, 1847.918, -26612.568, -4994.926, 51692.265, 5335.092, -46475.953, -2019.027, 15677.365

我们发现存在很多绝对值特别大的正数和负数。这种失衡正好使曲线以正确的方式“摆动”，以便几乎完美地对数据进行插值。但这种情况是不稳定的：如果数据发生很小的变化，那最终的系数将改变很大。

通过使用期望为0的高斯先验分布，我们可以鼓励模型的参数变小，所以最终的曲线会变化平滑：

 （7.30）

其中控制着先验分布的强度。对应的MAP估计问题将演变成：

 （7.31）

（读者）可以证明，上式等价于最小化如下目标函数：

 （7.32）

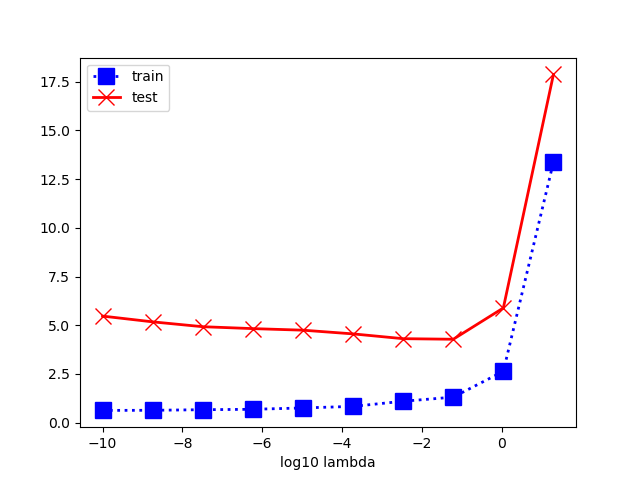
其中，表示向量w的二范数的平方。所以上式的第一项就是前面介绍的MSE/NLL，第二项为复杂度惩罚项，其中。上式对应的解为：

 （7.33）

这个技术被称为**岭回归**（ridge regression）或者含**惩罚项的最小二乘**（penalized least squares）。一般情况下，为模型参数添加一个高斯先验分布，从而鼓励参数变得很小，这种技术被称为*l*2**正则**（regularization）或者**权重衰减**（weight decay）。值得注意的是，偏置项w0不被考虑正则，因为它只影响函数的高度，而非它的复杂度。通过惩罚权重的幅值的和，我们可以确保函数变得简单（因为当**w**=**0**时，最终的函数将会是一条直线，即最简单的函数，对应着一个常数。）

我们在图7.7中说明了这个思想，我们发现增加可以导致一个平滑的函数。最终的系数也变得更小。比如说，如果令，我们得到的系数为：

8.576, -0.443, 4.902, 15.102, -11.491, -4.625, -4.686, -4.118, 2.622, 0.760, 6.591, 2.159, 1.549, 1.126



（a）

图7.8

在图7.8（a）中，我们绘制了在训练集和测试集上MSE与的关系。我们发现，当我们增加时（模型的受约束程度更多），在训练集上的误差增加。对于测试集，我们发现了形似U的曲线，其中模型先过拟合后欠拟合。一般使用交叉验证的方式选择，正如图7.8（b）所示。在7.6.4节，我们将讨论一个更具备概率解释的方式去选择。

本书我们将讨论不同的先验分布。每一个都对应着不同形式的**正则**（regularization）。在防止过拟合方面，这项技术被广泛使用。

7.5.2 计算中的数值稳定性问题

有趣地是，岭回归不仅在统计方面的效果比较好，同时在数值上也更加容易拟合，因为相较于，条件数更好（所以更有可能是可逆的），至少对于合适的比较大的值而言，这一点是成立的。

然而，出于数值稳定性的考虑，对矩阵求逆依然是应该避免的。现在我们描述一个拟合岭回归模型的有用的技巧（我们也可以将这种方法延伸到一般最小二乘估计上），这种技巧在数值计算上更具有稳定性。我们假设参数的先验分布为，其中为精度矩阵。在岭回归的情况下，。为了避免惩罚*w*0项，我们应该先对数据进行中心化。

首先使用来自先验分布的“虚拟数据”对原始数据进行数据增强：

 （7.34）

其中为矩阵的**切比雪夫分解**（Cholesky decomposition）。我们发现为（*N*+*D*）×*D*的矩阵，其中额外的行代表来自于先验分布的伪数据。

现在我们展示基于拓展后的数据的NLL与含惩罚项的基于原始数据的NLL是等价的：

 （7.35）

 （7.36）

 （7.37）

 （7.38）

 （7.39）

所以MAP估计为：

 （7.40）

现在令

 （7.41）

为的**QR分解**（QR decomposition），其中Q为正交矩阵（意味着），R为上三角矩阵。我们有：

 （7.42）

所以：

 （7.43）

需要注意的是R是一个容易求逆的上三角矩阵。这给了我们一个计算岭回归参数估计值的方法，在这种方法中我们可以避免对求逆。

我们可以使用这个技术去求解MLE，只需要计算非增强矩阵X的QR分解，使用原始的数据y。这是解决最小二乘问题的方法。需要注意的是，计算一个*N*×*D*的矩阵的QR分解的时间复杂度为*O*(*ND*2)，并且在数值计算上也是稳定的。

如果，我们首先应该进行矩阵的SVD分解。特别地，令为矩阵X的SVD分解，其中，，S为*N*×*N*的对角矩阵。现在令**Z**=**US**为*N*×*N*的矩阵。然后，我们重写岭回归估计值为：

 （7.44）

换句话说，我们可以将原本的*D*维的向量**x***i*换成*N*维的向量**z***i*，然后向之前那样求解估计值。然后，再通过映射矩阵**V**，将*N*维的解映射到*D*维空间。在几何上，我们正在向一个新的坐标系旋转，在这个坐标系中，除了前*N*的坐标值之外，其他都为0。这并不影响最终的解，因为球状高斯先验分布在旋转过程中是不变的。上式的计算时间复杂度为*O*(*DN*2)。

7.5.3 与PCA的联系\*

本节，我们将讨论岭回归与PCA（12.2节介绍）之间的一个有趣的联系，这将解释为什么岭回归的效果会更好。

令为矩阵X的SVD分解。根据式7.44，我们有：

 （7.45）

所以，在训练集上岭回归模型的预测结果为：

 （7.46）

 （7.47）

其中：

 （7.48）

其中为矩阵**X**的奇异值。所以

 （7.49）

相反，最小二乘法得到的预测值为：

 （7.50）

如果相对于比较小，那么方向对结果的预测将不会有太大影响。从这一点来看。我们定义模型自由度（degress of freedom）的有效数为：

 （7.51）

当，模型的自由度为*D*，，自由度趋向于0。

让我们尝试理解为什么这种行为是令人满意的。在7.6节，我们将表明，如果使用关于参数**w**的均匀先验分布，那么。所以我们对**w**估计值最不确定的那个方向，对应于矩阵特征值最小的那个特征向量所指的方向，如图4.1所示。进一步地，我们将在12.2.3节展示奇异值的平方就是矩阵的特征值。所以，小的奇异值对应后验方差大的方向。这些方向是岭回归收缩最厉害的方向。

这个过程在图7.9中给出说明。水平方向的*w*1通过数据没有办法较好地确定下来（即后验方差较大），但垂直方向的*w*2可确定度比较高。所以接近，但是在很大程度上向先验的期望0值移动（对比图4.14（c），该图说明了不同精度的传感器的融合。）在这种方式下，确定度小的参数在大小上被收缩至0。这被称为**收缩**（shrinkage）。

有一个相关但不同的技术被称为**主元回归**（principal components regression）。其思想为：首先使用PCA将维度降到*K*维，然后使用这些低维特征作为回归的输入。然而，这项技术在预测精度上并不如岭回归好。理由在于主元回归方法只使用前*K*个维度的特征，剩下的*D*-*K*个维度的特征被完全忽略。相反，岭回归对所有维度使用一个“柔和的”加权方法。

7.5.4 大数据的正则化效果

正则化是最常用的避免过拟合的方式。然而，另一个不经常使用的方式是使用大量的数据。直觉上，我们拥有的数据量越多，模型的训练效果更好。所以我们期望随着*N*的增加，测试集上的误差将衰减到一个稳定的值。

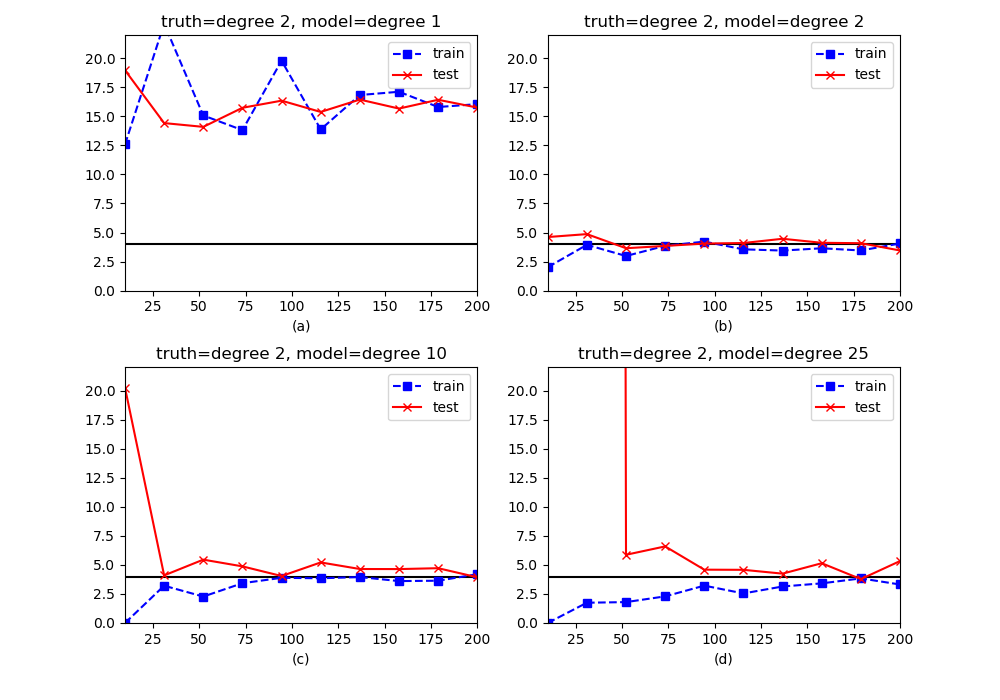


图7.10 训练集和测试集上的MSE和训练样本数量的关系，数据由一个2阶多项式产生，高斯噪音的方差为4。我们通过使用不同阶数的多项式来拟合数据。（a）阶数为1.（b）阶数为2.（c）阶数为10.（d）阶数为25。对于样本数量小的情况，阶数为25的多项式的测试误差高于阶数为2的多项式的测试误差，原因是过拟合，但当数据量增加时，这种差距会消失。同时需要注意的是，阶数为1的多项式因为过于简单，所以就算有足够的训练数据，其测试误差都很高。 图形由程序**linregPolyVsN**生成。

图7.10给出了说明，其中我们绘制了随着数据量*N*的增加，通过不同阶数的多项式拟合训练出来的模型，在测试集上的均方差。（误差与训练集大小的关系曲线被称为**学习曲线**（learning curve））。测试集上的误差的稳定状态由两部分组成：一个是所有模型都必须具备的不可减少的部分，它是由于数据生成过程中的固有的多样性（被称为**本底噪声**（noise floor））；另一部分是由于数据生成过程（“真相”）与模型自身的不相符：即所谓的**结构误差**（structural error）。

在图7.10中，数据真实的生成过程为2阶多项式，我们尝试用阶数为1,2和25的多项式去拟合生成的数据。称3个模型为1，2，和25。我们发现模型2和25的结构误差为0，因为两个模型都可以捕获到真实的数据生成过程。然而，1的结构误差很大，它的误差稳定值明显高于本底噪声。

对于任何具有足够表达力以捕捉真相的模型（也就是说结构误差很小），当，测试误差将稳定在本底噪音。然而，对于简单的模型，到达稳定值的速度更快，因为需要估计的参数更少。特别地，对于数量有限的训练集，在我们实际估计的参数值与我们能够估计的最好的参数值之间会存在一定的不符。这被称为**近似误差**（approximation error），当，这个误差将趋向于0，但对于简单的模型，趋于0的速度更快。图7.10给出了说明。

在大数据领域，简单模型的效果可以惊人的好。然而，研究更加复杂的模型也是有原因的，因为当我们的数据量比较少时，总会存在一些问题。举例来说，即使在web搜索这样一个数据丰富的领域中，只要我们想进行一些个性化，任何给定用户可用的数据量就会再次变得很小（相对于问题的复杂度而言）。在这种情况下，我们可能想要同时学习多个相关的模型，这被称为多任务学习。这将允许我们从拥有大量数据的任务中“借用统计强度”，并将其与拥有少量数据的任务共享。我们将在本书后面讨论这些方法。

7.6 贝叶斯线性回归

尽管岭回归是一种计算参数点估计的有效方式，有些时候我们也想计算关于参数**w**和的后验分布。为了简单起见，我们首先假设噪音的方差已知，所以将注意力集中在计算。在7.6.3节中，我们会再考虑一般的情况，即计算。我们在讨论中假设似然函数是一个高斯分布模型。对一个鲁棒的似然函数进行贝叶斯推断也是可能的，但需要更加高级的技术。

7.6.1 计算后验分布

在线性回归中，似然函数的形式为：

 （7.52）

 （7.53）

其中为偏置项。如果输入已经进行中心化处理，即对于每个*j*满足，那么输出的期望值为负数或者正数的可能性是一样的。所以我们为添加一个不合适先验，形式为，然后对它进行积分得到：

 （7.54）

其中为输出的经验期望值。为了符号上简便，我们假设输出已经被中心化，使用代替。

上述高斯似然函数的先验分布依然是一个高斯分布，使用符号表示为。使用贝叶斯法则（公式4.125），后验分布表示为：

 （7.55）

 （7.56）

 （7.57）

 （7.58）

如果，，并且定义，那么后验期望将变成岭估计。这是因为高斯分布的期望和众数是一样的。

为了深入了解后验分布（而不仅仅是它的众数），让我们考虑一个1维的例子：

 （7.59）

其中“真实”的参数值为,。在图7.11中，我们绘制了先验分布，似然函数，后验分布以及从后验预测分布中采样得到的一些数据。特别地，最右侧绘制了函数，其中x的范围为[-1,1]，为从参数后验分布中得到的一个采样值。一开始，我们从先验分布中进行采样（第1行），我们的预测结果分布在整个数据空间，因为我们的先验分布是一个均匀分布。在我们看到一个数据点时（第2行），我们的后验分布被相应的似然函数所约束，我们的预测值也更加接近观测值。然而，我们发现后验分布具有一个类似于山岭的形状，意味着可能存在很多种解，即不同的斜率/截距组合。这一点也与我们的直觉是一致的，因为我们没有办法从一个数据点中唯一地确定出2个参数。在我们看到第2个数据点时（第3行），后验分布变得更加集中，我们的预测结果几乎都具有相似的斜率和截距。在观察到20个数据点时（最后1行），后验分布必然变成了以真实值（白色×）为中心的delta函数。（这个估计值之所以能够收敛于真实值，是因为观测到的数据来自于真实的模型，而且贝叶斯估计是一个一致性估计函数，章节6.4.1关于这一点给出了更多讨论。）

7.6.2 计算后验预测

It’s tough to make predictions, especially about the future. ——Yogi Berra

在机器学习领域，相较于对模型参数的估计，我们更关心对预测值的估计。使用式4.126，我们可以很容易地发现在测试点**x**的后验预测分布依然是一个高斯分布：

（7.60）

预测值的方差与两项相关：观测值噪音的方差和参数的方差。后者转换为关于观测值的方差，这种转换取决于x与训练集D的距离。图7.12（b）说明了这一点，图中我们发现当我们远离训练集时，误差条变得越来越大，说明不确定度也更大。这在主动学习之类的应用中十分重要，在这种应用中，我们不仅需要对我们知道的进行建模，还需要对我们不知道的进行建模。相反，点估计具有大小一致的误差条（图7.12（a）说明），因为：

（7.63）

7.6.3 当未知时的贝叶斯推理\*

本节，我们将应用4.6.3节的结论来解决在线性回归模型中计算的问题。这是对7.6.1节结果的一般化，在7.6.1节中我们假设是已知的。当我们使用一个非信息先验分布时，我们会发现与频率学派的一些有趣的联系。

7.6.3.1 共轭先验

像之前介绍的一样，似然函数的形式为：

（7.64）

类似于4.6.3节，我们可以发现一些具备如下形式的自然的共轭先验：

（7.65）

根据先验分布和似然函数，我们可以得到后验分布具备如下的形式：

（7.69）

关于和的解释与已知的情况下是一致的。关于的解释也是符合直觉的，因为它只是对计数进行了更新。关于可以解释为：它是先验平方和b0，加上经验平方和，再加上一个因为参数w先验分布误差引起的项。

后验边缘分布为：

（7.74）

我们将在7.6.3.3节利用这些公式给出一个有效的例子。

类似于章节4.6.3.6，后验预测分布是一个学生t分布。特别地，给定m个新的测试输入，我们有：

（7.76）

预测方差具备2个部分：由观测值噪音引起的，由参数w的不确定引起的。后一项还取决于测试数据与训练集的距离。

通常情况下，我们设置，对应于一个关于的非信息先验，令，其中g为任意正数。这被称为Zellner’s g-prior。这里的g扮演者类似于岭回归中的角色。然而，先验协方差矩阵正比于而不是I。这就确保了后验分布对输入的尺度不变性。

接下来，我们将发现，如果我们使用一个无信息先验分布，在给定N个测量值的情况下，后验分布的精度为。单元信息先验（unit information prior）被定义为包含与一个样本一样的信息。为了为线性回归模型创建一个单元信息先验，我们需要使用，等价于g=N时的g-prior。

7.6.3.2 无信息先验

通过考虑共轭g-prior的无信息极限，我们可以得到一个无信息先验分布，对应于。这等价于一个不合适的NIG先验分布，其中，得到。

可选的方案是，我们可以从一个半共轭先验开始，对每一项我们单独地使用它的无信息极限，得到。这等于一个不合适的NIG先验，其中。对应的后验分布为：

（7.77）

关于权重的边缘分布为：

（7.83）

其中，为MLE。我们在下面讨论这些公式所隐含的意义。

7.6.3.3 贝叶斯推理与频率学派推理一致性的例子\*

（半共轭）无信息先验分布的使用是一件十分有趣的事情，因为最终导致的后验分布等价于通过频率学派得到的结果（见4.6.3.9）。特别地，根据式7.83，我们有：

（7.84）

这就是MLE的采样分布，由下式给定：

（7.85）

其中：

（7.86）

为估计参数的标准差。（见6.2节对采样分布的讨论。）因此，频率学派的置信区间和贝叶斯学派的边缘可信度区间在这种情况下是一样的。

作为一个例子，考虑一个caterpillar数据集（数据集中数据的意义与我们现在需要展示的没有关联。）我们可以计算出后验期望和标准差，以及回归系数的95%的可信度区间（credible intervals,CI）（使用式7.84）。结果在表7.2中进行展示。很容易发现，95%的可信度区间与使用标准的频率学派方法计算出来的95%置信区间是一样的（见程序linregBayesCaterpillar）。

我们也可以使用边缘后验分布来计算系数是否在很大程度上区别于0。一种非正式的方式（不使用决策论）是检测它的95%可信区间是否不包含0。根据表7.2，我们发现系数0,1,2,4,5的CIs在这种测量基准下都十分重要（译者注：区间内不包含0），所以我们给他们标注了\*。我们也可以很容易地验证这些结果与通过频率学派软件包计算出来的结果是一致的，这些软件包在5%的水平计算p值。

尽管贝叶斯和频率学派的计算结果的一致性可能对某些读者很有吸引力，但回忆6.6节中介绍的频率学派推理中充斥的困顿。同时需要注意的是，MLE在N<D时甚至不存在，所以频率学派推理在这种情况下是失效的。但贝叶斯推理依然奏效，尽管它需要使用一个合适的先验分布。

7.6.4 线性回归的EB（证据程序）

截止目前，我们都是假设先验分布是已知的。本节，我们将介绍一种经验贝叶斯程序，实现对超参数的选择。更加精确地表述，我们选择使边缘似然最大，其中为观测噪音的精度，为先验分布的精度。这被称为证据程序（evidence procedure）。13.7.4节将讨论更多的算法细节。

证据程序提供了一种替代交叉验证的方案。举例来说，在图7.13（b）中，我们绘制了不同值下的对数边缘似然，以及基于优化器找到的最大值。我们发现，在这个例子中，我们得到的结果与5-CV交叉验证的结果一样，如图7.13（a）所示。（在两种方法中，我们保持不变，从而使两种方法具备可比性。）

相较于交叉验证，证据程序在实践中的主要优势变得十分明显，这一点我们会在13.7节重点介绍，彼时，我们将会为每个特征赋予一个不同值，从而实现对先验分布的一般化。这种方法可以用于特征选择，使用一种被称为自动关联决定（automatic relevancy determination,ARD）的技术。相反，我们不可能通过使用交叉验证实现D个不同超参的调节。

证据程序在比较不同模型方面也有用，因为它提供了一个关于证据的好的近似：

（7.87）

对进行积分而不是将它设置为任意值，这一点是十分重要的，理由在5.3.2.5节有讨论。的确，这种方法被我们用来评估图5.7和5.8中的多项式回归模型。对于一个“更贝叶斯”的方法，我们将在21.5.2节介绍，在这种方法中我们需要对的不确定度进行建模，而不是计算它的点估计。