子午线

学习笔记

作者：leekarry
组织：果壳
时间：May 24, 2019
版本：0.1

ElegantLaTeX Program

我很快，快到时间都会变慢。而我一慢，时间就会过得飞快。

目　录

第 I 部分 I

机器学习

第 1 章 机器学习概论

1.1 基本概念

统计学习的特点

统计学习 (statistical learning) 是关于计算机基于数据构建概率统计模型并运用模型对数据进行预测与分析的一门学科。

- 统计学习以计算机及网络为平台；
- 统计学习以数据为研究对象；
- 统计学习的目的是对数据进行预测与分析；
- 统计学习以方法为中心；
- 统计学习是概率论、统计学、信息论、计算理论、最优化理论及计算机科学等多个领域的交叉学科。

统计学习的对象

统计学习的对象是数据 (data)。它从数据出发，提取数据的特征，抽象出数据的模型，发现数据中的知识，又回到对数据的分析与预测中去。**统计学习关于数据的基本假设是同类数据具有一定的统计规律性，这是统计学习的前提**。

统计学习的目的

统计学习用于对数据进行预测与分析，特别是对未知新数据进行预测与分析。对数据的预测与分析是通过构建概率统计模型实现的。统计学习总的目标就是考虑学习什么样的模型和如何学习模型，以使模型能对数据进行准确的预测与分析，同时也要考虑尽可能地提高学习效率。

统计学习的方法

统计学习的方法是基于数据构建统计模型从而对数据进行预测与分析。

- 监督学习 (supervised learning)
- 非监督学习 (unsupervised learning)
- 半监督学习 (semi-supervised learning)
- 强化学习 (reinforcement learning)

实现统计学习方法的步骤如下：

(1) 得到一个有限的训练数据集合

(2) 确定包含所有可能的模型的假设空间，即学习模型的集合

(3) 确定模型选择的准则，即学习的策略

(4) 实现求解最优化模型的算法,即学习的算法
(5) 通过学习方法选择最优模型
(6) 利用学习的最优模型对新数据进行预测或分析

统计学习的研究

统计学习方法 (statistical learning method),旨在开发新的学习方法。

统计学习理论 (statistical learning theory),旨在探求统计学习方法的有效性与效率。

统计学习应用 (application of statistical learning),旨在将统计学习方法应用到实际问题中去,解决实际问题。

统计学习的重要性

统计学习是处理海量数据的有效方法;统计学习是计算机智能化的有效手段;统计学习是计算机科学习发展的一个重要组成部分。

1.2 统计学习三要素

(1) 模型:统计学习首先要考虑的问题是学习什么样的模型
(2) 策略:有了模型的假设空间,统计学习接着需要考虑的是按照什么样的准则学习或选择最优的模型
(3) 算法:算法是指学习模型的具体计算方法

1.3 模型评估与模型选择

统计学习的目的是使学到的模型不仅对已知数据而且对未知数据都能有很好的预测能力。不同的学习方法会给出不同的模型。当损失函数给定时,基于损失函数的模型的训练误差(training error)和模型的测试误差(test error)就自然成为学习方法评估的标准。测试误差反映了学习方法对未知的测试数据集的预测能力——泛化能力。

1.4 正则化与交叉验证

模型选择的典型方法是正则化 (regularization)。正则化是结构风险最小化策略的实现,是在经验风险上加一个正则化项 (regularizer) 和罚项 (penalty term)

$$\min_{f\in F} \quad \frac{1}{2}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{1.1}$$

另一种常用的模型选择方法是交叉验证 (cross validation) 训练集用来训练模型验证集用来选择模型测试集用来对模型进行评估

1. 简单交叉验证

2. S 折交叉验证
3. 留一交叉验证

1.5 生成模型与判别模型

监督学习方法又可以分为生成方法 (generative approach) 和判别方法 (discriminative approach)。所学到的模型分别称为生成模型和判别模型。

生成方法由数据学习联合分布 $P(X,Y)$,然后求出条件概率分布 $P(Y|X)$ 作为预测的模型,即生成模型:

$$P(Y|X) = \frac{P(X,Y)}{P(X)} \tag{1.2}$$

这样的方法之所以称为生成方法,是因为模型表示了给定输入 X 产生输出 Y 的生成关系。

判别方法由数据直接学习决策函数 $f(X)$ 或者条件概率分布 $P(Y|X)$ 作为预测的模型,即判模型。判别方法关心的是对给定的输入 X,应该预测什么样的输出 Y。

生成方法的特点:生成方法可以还原出联合概率分布 P(X|Y),而判别方法则不能;生成方法的学习收敛速度更快,即当样本容量增加的时候,学到的模型可以更快地收敛于真实模型;当存在隐变量时,仍可以用生成方法学习,此时判别方法就不能用。

判别方法的特点:判别方法直接学习的是条件概率 P(Y|X) 或决策函数 $f(X)$,直接面对预测,往往学习的准确率更高;由于直接学习 P(Y|X) 或 $f(X)$,可以对数据进行各种程度上的抽象、定义特征并使用特征,因此可以简化学习问题。

1.6 频率学派和贝叶斯学派的参数估计

频率学派与贝叶斯学派的区别

简单地说,频率学派与贝叶斯学派探讨**不确定性**这件事的出发点与立足点同。频率学派从" 自然" 角度出发,试图直接为" 事件" 本身建模,即事件 A 在独立重复试验中发生的频率趋于极限 p,那么这个极限就是该事件发生的概率。贝叶斯学派并不从试图刻画" 事件" 本身,而从" 观察者" 角度出发。贝叶斯学派并不试图说" 事件本身是随机的",或者" 世界的本体带有某种随机性",而只是从" 观察者知识不完备" 这一出发点开始,构造一套在贝叶斯概率论的框架下可以对不确定知识做出推断的方法。体现在参数估计中,频率学派认为参数是客观存在,不会改变,虽然未知,但却是固定值;贝叶斯学派则认为参数是随机值,因此参数也可以有分布。

频率学派的参数估计

极大似然估计 (Maximum Likelihood Estimate,MLE),也叫最大似然估计。若总体 X 属离散型 (连续型与此类似),其分布律 $P\{X = x\} = p(x;\theta), \theta \in \Theta$ 的形式为已知,θ 为待估参数,Θ 是 θ 的取值范围,设 $X_1, X_2, \ldots, X_n$ 是来自 X 的样本,则 $X_1, X_2, \ldots, X_n$ 的联合概率分

布为

$$\prod_{i=1}^{n} p(x;\theta) \tag{1.3}$$

设 $x_1, x_2, \ldots, x_n$ 是相应的样本值，则

$$L(\theta) = L(x_1, x_2, \ldots, x_n; \hat{\theta}) = \underset{\theta \in \Theta}{argmax} \prod_{i=1}^{n} p(x;\theta) \tag{1.4}$$

贝叶斯学派的参数估计

最大后验估计 (Maximum a Posteriori estimation,MAP)，它与极大似然估计最大的区别就是，它考虑了参数本身的分布，也就是先验分布。最大后验估计是根据经验数据获得对难以观察的量的点估计。可以看作规则化的最大似然估计。假设 x 为独立同分布的采样，θ 为模型参数，p 为我们所使用的模型。那么最大似然估计可以表示为

$$\hat{\theta}_{MLE}(x) = \underset{\theta}{argmax}\, p(x|\theta) \tag{1.5}$$

现在，假设 θ 的先验分布为 g。通过贝叶斯理论，对于 θ 的后验分布如下式所示：

$$p(\theta|x) = \frac{p(x|\theta)g(\theta)}{\int_{\theta \in \Theta} p(x|\theta')g(\theta')d\theta'} \tag{1.6}$$

分母为 x 的边缘概率与 θ 无关，因此最大后验等价于使分子最大，故目标函数为

$$\hat{\theta}_{MAP}(x) = \underset{\theta}{argmax}\, p(x|\theta)g(\theta) \tag{1.7}$$

第 2 章 回归的线性模型

2.1 线性基函数模型

回归问题的最简单模型是输入变量的线性组合

$$y(\boldsymbol{x}, \boldsymbol{w}) = w_0 + w_1 x_1 + \ldots w_D x_D$$

其中 $x = (x_1, \ldots, x_D)^T$。这通常被简单地称为**线性回归 (linear regression)**。这个模型的关键性质是它是参数 $w_0, \ldots, w_D$ 的一个线性函数。但是，它也是输入变量 x_i 的一个线性函数，这给模型带来了极大的局限性。因此，我们扩展模型的类别：将输入变量的固定的非线性函数进行线性组合，形式为

$$y(\boldsymbol{x}, \boldsymbol{w}) = w_0 + \sum_{y=1}^{M-1} w_j \phi_j(\boldsymbol{x}) = \sum_{j=0}^{M-1} w_j \phi_j(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x})$$

其中 $\phi_j(\boldsymbol{x})$ 被称为基函数 (basis function)。

1. 高斯基函数

$$\phi_j(x) = exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$

2. sigmoid 基函数

$$\phi_j(x) = \frac{1}{1 + exp\left(\frac{x-\mu_j}{s}\right)}$$

3. 傅里叶基函数

最大似然与最小平方

最小平方的几何描述

顺序学习

正则化最小平方

多个输出

2.2 偏置-方差分解

目前为止，我们对于回归的线性模型的讨论中，我们假定了基函数的形式和数量都是固定的。如果使用有限规模的数据集来训练复杂的模型，那么使用最大似然法，或者等价地使用最小平方法，会导致严重的过拟合问题。正如前面所说，过拟合现象确实是最大似

然方法的一个不好的性质。但是当我们在使用贝叶斯方法对参数进行求和或者积分时，过拟合现象不会出现。从贝叶斯观点讨论模型的复杂度之前，从频率学家的观点考虑一下模型的复杂度的问题——偏置-方差折中 (bias-variance trade-off)。

最优的预测由条件期望 (记作 $h(x)$) 给出，即

$$h(x) = \mathbb{E}[t|x] = \int tp(t|x)dt \tag{2.1}$$

平方损失函数的期望可以写成

$$\begin{aligned}\mathbb{E}[L] &= \iint \{y(x) - h(x)\}^2 p(x,t)dxdt \\ &= \int \{y(x) - h(x)\}^2 p(x)dx + \iint \{h(x) - t\}^2 p(x,t)dxdt\end{aligned} \tag{2.2}$$

考虑第一项的被积函数，对于一个特定的数据集 D，它的形式为

$$\{y(x;D) - h(x)\}^2 \tag{2.3}$$

由于这个量与特定的数据集 D 相关，因此我们对所有的数据集取平均。如果我们在括号内加上然后减去 $\mathbb{E}_D[y(x;D)]$，然后展开，我们有

$$\begin{aligned}&\{y(x;D) - \mathbb{E}_D[y(x;D)] + \mathbb{E}_D[y(x;D)] - h(x)\}^2 \\ &\quad = \{y(x;D) - \mathbb{E}_D[y(x;D)]\}^2 \\ &\quad + \{\mathbb{E}_D[y(x;D)] - h(x)\}^2 \\ &\quad + 2\{y(x;D) - \mathbb{E}_D[y(x;D)]\}\{\mathbb{E}_D[y(x;D)] - h(x)\}\end{aligned} \tag{2.4}$$

现在关于 D 求期望，然后注意到**最后一项等于零**，可得

$$\begin{aligned}&\mathbb{E}_D[\{y(x;D) - h(x)\}^2] \\ &\quad = \underbrace{\{\mathbb{E}_D[y(x;D)] - h(x)\}^2}_{(\text{偏置})^2} + \underbrace{\mathbb{E}_D[\{y(x;D) - \mathbb{E}_D[y(x;D)]\}^2]}_{\text{方差}}\end{aligned} \tag{2.5}$$

将式 2.5 代入式 2.2 中，就得到了对于期望平方损失的分解

$$\text{期望损失} = \text{偏置}^2 + \text{方差} + \text{噪声} \tag{2.6}$$

其中

$$\text{偏置}^2 = \int \{\mathbb{E}_D[y(x;D)] - h(x)\}^2 p(x)dx \tag{2.7}$$

$$\text{方差} = \int \mathbb{E}_D[\{y(x;D) - \mathbb{E}_D[y(x;D)]\}^2]p(x)dx \tag{2.8}$$

$$\text{噪声} = \iint \{h(x) - t\}^2 p(x,t)dxdt \tag{2.9}$$

我们的目标是最小化期望损失，它可以分解为 (平方) 偏置、方差和一个常数噪声项的和。对于非常灵活的模型来说，偏置较小，方差较大。对于相对固定的模型来说，偏置较大，方差较小。有着最优预测能力的模型是在偏置和方差之前取得最优的平衡的模型。

2.3 贝叶斯线性回归

2.4 贝叶斯模型比较

2.5 证据近似

2.6 固定基函数的局限性

第 3 章 分类的线性模型

3.1 判别函数

3.2 概率生成式模型

3.3 概率判别式模型

3.4 拉普拉斯近似

3.5 贝叶斯 logistic 回归

第 4 章 神经网络

在上两章，我们考虑了由固定基函数的线性组合构成的回归模型和分类模型。我们看到，这些模型具有一些有用的分析性质和计算性质，但是它们的实际应用被维数灾难问题限制了。为了将这些模型应用于大规模的问题，有必要根据数据调节基函数。

支持向量机是这样解决这个问题的：首先定义以训练数据点为中心的基函数，然后在训练过程中选择一个子集。支持向量机的一个优点是，虽然训练阶段涉及到非线性优化，但是目标函数是凸函数，因此最优化问的解相对很直接，并且通常随着数据规模的增加而增多。相关向量机也选择固定基函数集合的一个子集，通常会生成一个相当稀疏的模型。与支持向量机不同，相关向量机也产生概率形式的输出，嘎然这种输出的产生会以训练阶段的非凸优化为代价。

另一种方法是事先固定基函数的数量，但是允许基函数可调节。换名话，就是使用参数形式的基函数，这些参数可以在训练阶段调节。在模式识别中，这种类型的最成功的模型是有前馈神经网络，也被称为多层感知器 (multilayer perceptron)。与具有同样泛化能力的支持向量机相比，最终的模型会相当简洁，因此计算的速度更快。这种简洁性带来的代价就是，与相关向量机一样，构成了网络训练根基的似然函数不再是模型参数的凸函数。然而，在实际应用中，考察模型在训练阶段消耗的计算资源是很有价值的，这样做会得到一个简洁的模型，它可以快速地处理新数据。

首先，我们考虑神经网络的函数形式，包括基函数的具体参数，然后我们讨论使用最大似然框架确定神经网络参数的问题，这涉及到非线性最优化问题的解。这种方法需要计算对数似然函数关于神经网络参数的导数，我们会看到这些导数可以使用误差反向传播 (error backpropagation) 的方法高效地获得。我们还会说明误差反向传播的框架如何推广到计算其他的导数，例如 Jacobian 矩阵和 Hessian 矩阵。接下来，我们讨论神经网络训练的正则化和各种方法，以及方法之间的关系。我们还会考虑神经网络模型的一些扩展。特别地，我们会描述一个通用的框架，用来对条件概率密度建模。这个框架被称为混合密度网络 (mixture density network)。最后，我们讨论神经网络的贝叶斯观点。

4.1 前馈神经网络

深度前馈网络 (deep feedforward network) 也叫做前馈神经网络 (feedforward neural network) 或者多层感知机 (multilayer perceptron,MLP)，是典型的深度学习模型。前馈神经网络的目标是近似某个函数 f^*。前馈网络定义了一个映射 $y = f(x;\theta)$，并且学习参数 θ 的值，使它能够得到最佳的函数近似。

前馈神经网络之所以被称作网络，是因为它们通常用许多不同函数复合在一起来表示。该模型与一个有向无环图相关联，而图描述了函数是如何复合在一起的。例如，我们有三个函数 $f^{(1)}, f^{(2)}, f^{(3)}$ 连接在一个链上以形成 $f^{(3)}(f^{(2)}(f^{(1)}(x)))$。

在神经网络训练过程中,我们让 $f(x)$ 去匹配 $f^*(x)$ 的值。训练数据为我们提供了在不同训练点上取值的、含有噪声的 $f^*(x)$ 近似实例。每个样本 x 都伴随着一个标签 $y \approx f^*(x)$。训练样本直接指明了输出层在每一点 x 上必须做什么;它必须产生一个接近 y 的值。但是训练数据并没有直接指明其他层应该怎么做。学习算法必须决定如何使用这些层来诞生想要的输出,但是训练数据并没有说每个单独的层应该做什么。相反,学习算法必须决定如何使用这些层来最好地实现 f^* 的近似。因为训练数据并没有给出这些层中的每一层所需的输出,所以这些层被称为隐藏层 (hidden layer)。

4.2 网络训练

4.3 误差反向传播

4.4 Hessian 矩阵

4.5 神经网络的正则化

4.6 混合密度网络

4.7 贝叶斯神经网络

第 5 章 组合模型

5.1 贝叶斯模型平均

5.2 委员会

5.3 提升方法

提升 (boosting) 方法是一种常用的统计学习方法，应用广泛且有效。在分类问题中，它通过改变训练样本的权重，学习多个分类器，并将这些分类器进行线性组合，提高分类的性能。

提升方法基于这样一种思路：对于一个复杂任务来说，将多个专家的判断进行适当的综合所得出的判断，要比其中任何一个专家单独的判断好。提升方法就是从弱学习算法出发，反复学习，得到一系列弱分类器 (又称基本分类器)，然后组合这些弱分类器，构成一个强分类器。对提升方法来说，有两个问题需要回答：

1. 在每一轮如何改变训练数据的权值或概率分布；
2. 如果将弱分类器组合成一个强分类器。

AdaBoost 的做法是，提高那些被前一轮弱分类器错误分类样本的权值，而降低那些被正确分类样本的权值。这样一来，那些没有得到正确分类的数据，由于其权值的加大而受到后一轮的弱分类器的更大关注。于是，分类问题被一系列的弱分类器“分而治之”。至于第 2 个问题，即弱分类器的组合，AdaBoost 采取加权表决的方法。具体地，加大分类误差率小的弱分类器的权值，使其在表决中起较大作用，减小分类误差率大的弱分类器的权值，使其在表决中起较小的作用。AdaBoost 的巧妙之外就在于它将这些想法自然且有效地实现在一种算法里。

AdaBoost 算法

输入：训练数据集 $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$；弱学习算法；

输出：最终分类器 $G(x)$

(1) 初始化训练数据的权值分布

$$D_1 = (w_{11}, \ldots, w_{1i}, \ldots, w_{1N}), w_{1i} = \frac{1}{N}, i = 1, 2, \ldots, N \tag{5.1}$$

假设训练数据集具有均匀的权值分布，即每个训练样本在基本分类器的学习中作用相同，这一假设保证第 1 步能够在原始数据上学习基本分类器 $G_1(x)$。

(2) 对 $m = 1, 2, \ldots, M$

a. 使用具有权值分布 D_m 的训练数据集学习，得到基本分类器

$$G_m(x)X \rightarrow \{-1, +1\} \tag{5.2}$$

b. 计算 $G_m(x)$ 在训练数据集上的分类误差率

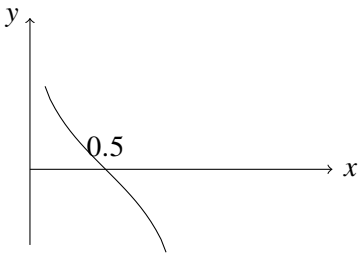
$$e_m = \sum_{i=1}^{N} P(G_m(x_i) \neq y_i) = \sum_{i=1}^{N} w_{mi} I(G_m(x_i) \neq y_i) \tag{5.3}$$

这表明，$G_m(x)$ 在加权的训练数据集上的分类误差率是被 $G_m(x)$ 误分类样本的权值之和。

c. 计算 $G_m(x)$ 的系数

$$a_m = \frac{1}{2} log \frac{1 - e_m}{e_m} \tag{5.4}$$

a_m 表示 $G_m(x)$ 在最终分类器中的重要性。



由上图可知，当 $e_m \leqslant \frac{1}{2}$ 时，$a_m \geqslant 0$，并且 a_m 随着 e_m 的减小而增大，所以**分类误差率越小的基本分类器在最终分类器中的作用越大**。

d. 更新训练数据集的权值分布

$$\begin{aligned} D_{m+1} &= (w_{m+1,1}, \ldots, m_{m+1,i}, \ldots, w_{m+1,N}) \\ w_{m+1,i} &= \frac{w_{mi}}{Z_m} exp(-a_m y_i G_m(x_i)), i = 1, 2, \ldots, N \end{aligned} \tag{5.5}$$

这里，Z_m 是规范化因子

$$Z_m = \sum_{i=1}^{N} w_{mi} exp(-a_m y_i G_m(x_i)) \tag{5.6}$$

它使 D_{m+1} 成为一个概率分布。式 5.5 可以写成

$$w_{m+1,i} = \begin{cases} \frac{w_{mi}}{Z_m} e^{-a_m}, & G_m(x_i) = y_i \\ \frac{w_{mi}}{Z_m} e^{a_m}, & G_m(x_i) \neq y_i \end{cases} \tag{5.7}$$

由此可知，**被基本分类器 $G_m(x)$ 误分类样本的权值得以扩大，而被正确分类样本的权值却得以缩小**。因此，误分类样本在下一轮学习中起更大的作用。不改变所给的训练数据，而不断改变训练数据权值的分布，使得训练数据在基本分

类器的学习中起不同的作用，这是 AdaBoost 的一个特点。

(3) 构建基本分类器的线性组合

$$f(x) = \sum_{m=1}^{M} a_m G_m(x) \tag{5.8}$$

得到最终分类器

$$G(x) = sign(f(x)) = sign\left(\sum_{m=1}^{M} a_m G_m(x)\right) \tag{5.9}$$

线性组合 $f(x)$ 实现 M 个基本分类器的加权表决。系数 a_m 表示了基本分类器 $G_m(x)$ 的重要性，这里，所有 a_m 之和并不为 1.$f(x)$ 的符号决定实例 x 的类，$f(x)$ 的绝对值表示分类的确信度。利用基本分类器的线性组合构建最终分类器是 AdaBoost 的另一个特点。

AdaBoost 算法的训练误差分析

AdaBoost 最基本的性质是它能在学习过程中不断减少训练误差，即在训练数据集上的分类误差率。

AdaBoost 算法的解释

AdaBoost 算法还有另一个解释，即可以认为 AdaBoost 算法是模型为加法模型、损失函数为指数函数、学习算法为前向分步算法时的二类分类学习方法

5.4 基于树的模型

5.5 条件混合模型

5.6 logistic 模型的混合

第 6 章 支持向量机

支持向量机(support vector machines,SVM)是一种二类分类模型。它的基本模型是定义在特征空间上的间隔最大的线性分类器,间隔最大使它有别于感知机;支持向量机还包括核技巧,这使它成为实质上的非线性分类器。支持向量机的学习策略就是间隔最大化,可形式化为一个求解凸二次规划(convex quadratic programming)的问题,也等价于正则化的合页损失函数的最小化问题。支持向量机的学习算法是求解凸二次规划的最优化算法。

支持向量机学习方法包含构建由简至繁的模型:线性可分支持向量机(linear support vector machine in linearly separable case)、线性支持向量机(linear support vector machine)及非线性支持向量机(non-linear support vector machine)。简单模型是复杂模型的基础,也是复杂模型的特殊情况。当训练数据线性可分时,通过软件间隔最大化(hard margin maximization),学习一个线性的分类器,即线性可分支持向量机,又称为硬间隔支持向量机;当训练数据近似线性可分时,通过软件间隔最大化(soft margin maximization),也学习一个线性的分类器,即线性支持向量机,又称谓软间隔支持向量机;当训练数据线性不可分时,通过使用核技巧(kernel trick)及软间隔最大化,学习非线性支持向量机。

当输入空间为欧氏空间或离散集合、特征空间为希尔伯特空间时,核函数(kernel function)表示将输入从输入空间映射到特征空间得到的特征向量之间的内积。通过使用核函数可以学习非线性支持微量机,等价于隐式地在高维的特征空间中学习线性支持向量机。这样的方法称为核技巧。核方法(kernel method)是比支持向量机更为一般的机器学习方法。

Cortes 与 Vapnik 提出线性支持向量机,Boser、Guyon 与 Vapnik 又引入核技巧,提出非线性支持向量机。

6.1 间隔与支持向量

给定训练样本集 $D = \{(x_1, y_1), \ldots, (x_m, y_m)\}, y_i \in \{-1, +1\}$,分类学习最基本的想法是基于训练集 D 在样本空间中找到一个划分超平面,将不同类别的样本分开。如图 6.1

样本空间中任意点 $\boldsymbol{x}$ 到超平面 $(\boldsymbol{w}, b)$ 的距离可写为

$$\gamma = \frac{y(w^T x + b)}{\|w\|} = \frac{y f(x)}{\|w\|}$$

注:$y f(x)$ 相当于 $|f(x)|$。

假设超平面 $(\boldsymbol{w}, b)$ 能将训练样本正确分类,即对于 $(x_i, y_i) \in D$,若 $y_i = +1$,则有 $\boldsymbol{w}^T x_i + b > 0$;若 $y_i = -1$,则有 $\boldsymbol{w}^T x_i + b < 0$。令

$$\begin{cases} \boldsymbol{w}^T x_i + b \geq +1, & y_i = +1; \\ \boldsymbol{w}^T x_i + b \leq -1, & y_i = -1. \end{cases} \tag{6.1}$$

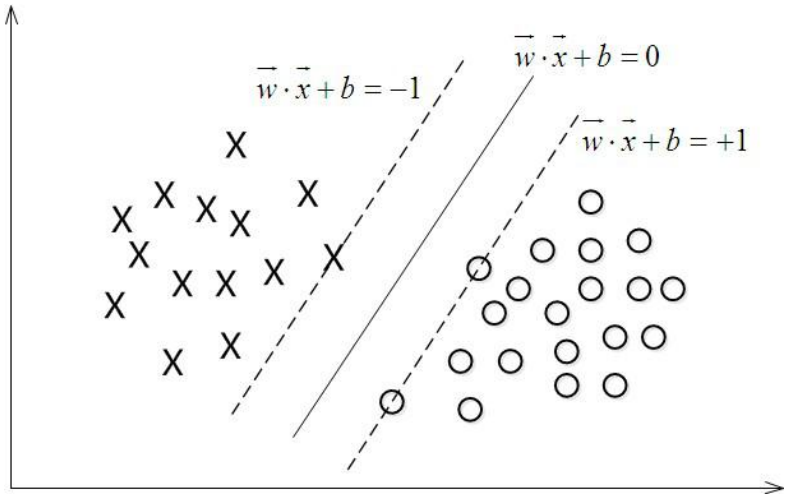


图 6.1

距离超平面最近的这几个训练样本点使式 6.1 的等号成立，它们被称为“**支持向量**(support vector)”，两个异类支持向量到超平面的距离之和为

$$\gamma = \frac{2}{\|w\|}, \tag{6.2}$$

它被称为“**间隔**”(margin)。

欲找到具有“最大间隔”的划分超平面，也就是要找到能满足式 6.1 中约束的参数 $\boldsymbol{w}$ 和 b，使得 γ 最大，即

$$\begin{aligned} min \quad & \frac{1}{2}\|w\|^2, \\ s.t. \quad & y_i(w^T x_i + b) \geq 1, \quad i = 1, \ldots, m. \end{aligned} \tag{6.3}$$

这就是支持向量机的基本型。

6.2 对偶问题

注意到式 6.3 本身是一个凸二次规划问题，能直接用现成的优化计算包求解，但我们可以有更高效的办法。由于这个问题的特殊结构，还可以通过拉格朗日对偶性变换到对偶变量的优化问题，即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，这就是线性可分条件下支持向量机的对偶算法，这样做的优点在于：

1. 对偶问题往往更容易求解；
2. 可以自然的引入核函数，进而推广到非线性分类问题。

该问题的拉格朗日函数可写为

$$L(w, b, a) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} a_i(y_i(w^T x_i + b) - 1) \tag{6.4}$$

然后令

$$\theta(\boldsymbol{w}) = \max_{a_i \geq 0} L(\boldsymbol{w}, b, a) \tag{6.5}$$

具体写出来，目标函数变成了

$$\min_{\boldsymbol{w},b}\ \theta(\boldsymbol{w}) = \min_{\boldsymbol{w},b}\max_{a_i\geq 0}\ L(\boldsymbol{w},b,a) = p^* \tag{6.6}$$

这里用 p^* 表示这个问题的最优值，且和最初的问题是等价的。如果直接求解，那么一上来便得面对 w 和 b 两个参数，而 a_i 以是不等式约束，这个求解过程不好做。考虑对偶问题

$$\min_{\boldsymbol{w},b}\ \theta(\boldsymbol{w}) = \max_{a_i\geq 0}\ \min_{\boldsymbol{w},b}\ L(\boldsymbol{w},b,a) = d^* \tag{6.7}$$

原始问题通过满足 KKT 条件，已经转化成了对偶问题。而求解这个对偶问题，分为 3 个步骤

1. **让 $L(\boldsymbol{w},b,a)$ 关于 $\boldsymbol{w}$ 和 b 最小化**

   首先固定 a，要让 L 关于 w 和 b 最小化，分别对 w 和 b 求偏导，令其等于 0；

$$\begin{aligned}\frac{\partial L}{\partial \boldsymbol{w}} &= \|\boldsymbol{w}\| - \sum_{i=1}^{n} a_i y_i x_i \boldsymbol{w}^T = 0 \Rightarrow \boldsymbol{w} = \sum_{i=1}^{n} a_i y_i x_i \\ \frac{\partial L}{\partial b} &= \sum_{i=1}^{n} a_i y_i = 0 \Rightarrow \sum_{i=1}^{n} a_i y_i = 0\end{aligned} \tag{6.8}$$

   将以上结果代入之前的 L，得到

$$\begin{aligned}L(\boldsymbol{w},b,a) &= \frac{1}{2}\sum_{i,j=1}^{n} a_i a_j y_i y_j x_i^T x_j - \sum_{i,j=1}^{n} a_i a_j y_i y_j x_i^T x_j - b\sum_{i=1}^{n} a_i y_i + \sum_{i=1}^{n} a_i \\ &= \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i,j=1}^{n} a_i a_j y_i y_j x_i^T x_j\end{aligned} \tag{6.9}$$

2. **求对 a 的极大**

   求对 a 的极大，即是关于对偶问题的最优化问题。经过上一个步骤的求解，得到的拉格朗日函数式子已经没有了变量 w 和 b，只有 a。从上面的式子得到

$$\begin{aligned}\max_{a}\quad & \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i,j=1}^{n} a_i a_j y_i y_j x_i^T x_j \\ s.t.\quad & a_i \geq 0, i = 1,\ldots,n \\ & \sum_{i=1}^{n} a_i y_i = 0\end{aligned} \tag{6.10}$$

   这样，求出了 a_i，从而根据

$$\begin{aligned}\boldsymbol{w}^* &= \sum_{i=1}^{n} a_i y_i x_i \\ b^* &= -\frac{max\ \boldsymbol{w}^{*T} x_i + min\ \boldsymbol{w}^{*T} x_i}{2}\end{aligned} \tag{6.11}$$

即可求出 w, b，最终得出分离超平面和分类决策函数。

3. **利用 SMO 算法求解对偶问题中的拉格朗日乘子**

在求得 $L(w, b, a)$ 关于 w 和 b 最小化和对 a 的极大之后，最后一步便是利用 SMO 算法求解对偶问题中的拉格朗日乘子

6.3 序列最小最优算法

接上一小节，讨论支持向量机学习的实现问题。讲述其中的序列最小最优化 (sequential minimal optimization,SMO) 算法。

SMO 算法是一种启发式算法，其基本思路是：**如果所有变量的解都满足此最优化问题的 KKT 条件，那么这个最优化问题的解就得到了**。因为 KKT 条件是该最优化问题的充分必要条件。否则，选择两个变量，固定其他变量，针对这两个变量构建一个二次规划问题。这个二次规划问题关于这两个变量的解应该更接近原始二次规划问题的解，因为这会使得原始二次规划问题的目标函数值变得更小。重要的是，这时子问题可以通过解析方法求解，这样就可以大大提高整个算法的计算计算速度。子问题有两个变量，一个是违反 KKT 条件最严重的那一个，另一个由约束条件自动确定。如此，SMO 算法将原问题不断分解为子问题并对子问题求解，进而达到求解原问题的目的。

SMO 算法要解如下凸二次规划的对偶问题：

$$\min_{a} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j y_i y_j K(x_i, x_j) - \sum_{i=1}^{N} a_i \tag{6.12}$$

$$s.t. \quad C \geq a_i \geq 0, \quad i = 1, \ldots, n \tag{6.13}$$

$$\sum_{i=1}^{N} a_i y_i = 0 \tag{6.14}$$

注意，子问题的两个变量中只有一个是自由变量。假设 a_1, a_2 为两个变量，$a_3, a_4, \ldots, a_N$ 固定，那么由等式约束 6.14 可知

$$a_1 = -y_1 \sum_{i=2}^{N} a_i y_i \tag{6.15}$$

如果 a_2 确定，那么 a_1 也随之确定。所以子问题中同时更新两个变量。

整个 SMO 算法包括两个部分：**求解两个变量二次规划的解析方法和选择变量的启发式方法**。

于是 SMO 的最优化问题 6.12 的子问题可以写成

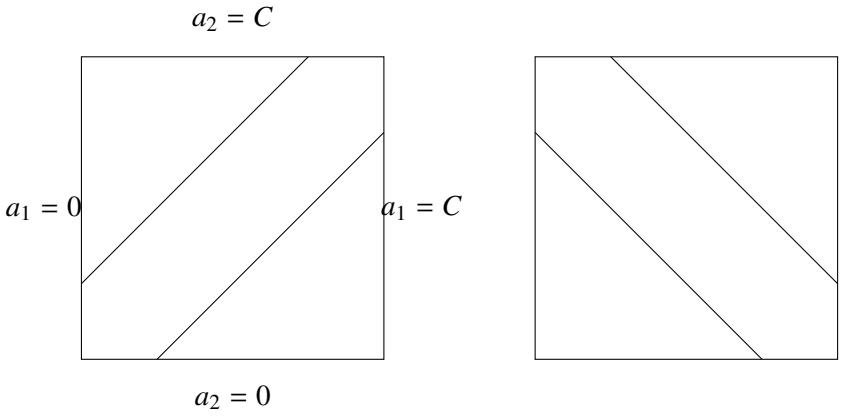
$$\min_{a_1,a_2} W(a_1,a_2)=\frac{1}{2}K_{11}a_1^2+\frac{1}{2}K_{22}a_2^2+y_1y_2K_{12}a_1a_2 -(a_1+a_2)+y_1a_1\sum_{i=3}^{N}y_ia_iK_{i1}+y_2a_2\sum_{i=3}^{N}y_ia_iK_{i2} \tag{6.16}$$

$$s.t\ a_1y_1+a_2y_2=-\sum_{i=3}^{N}y_ia_i=\varsigma \tag{6.17}$$

$$0\le a_i\le C,\ i=1,2 \tag{6.18}$$

其中，$K_{ij}=K(x_i,x_j),i,j=1,2,\ldots,N,\varsigma$ 是常数。

为了求解两个变量的二次规划问题 6.16，首先分析约束条件，然后在此约束条件下求极小。由于只有两个变量 (a_1,a_2)，约束可以用二维空间中的图形表示，如图所示



不等式约束使得 (a_1,a_2) 在盒子 $[0,C]\times[0,C]$ 内，等式约束使 (a_1,a_2) 在平行盒子的对角线的直线上。因此要求的是目标函数在一条平行于对角线线的线段上的最优值。这使得两个变量的最优化问题成为实质上的单变量的最优化问题，不妨考虑为变量 a_2 的最优化问题。

假设问题 6.16 的初始可行解为 a_1^{old},a_2^{old}，最优解为 a_1^{new},a_2^{new}，并且假设在沿着约束方向未经剪辑时 a_2 的最优解为 $a_2^{new,unc}$。

引进记号

$$g(x)=\sum_{i=1}^{N}a_iy_iK(x_i,x)+b \tag{6.19}$$

$$v_i=\sum_{j=3}^{N}y_ia_iK(x_i,x_j)=g(x_i)-\sum_{j=1}^{2}a_jy_jK(x_i,x_j)-b,\ i=1,2 \tag{6.20}$$

目标函数可写成

$$W(a_1,a_2)=\frac{1}{2}K_{11}a_1^2+\frac{1}{2}K_{22}a_2^2+y_1y_2K_{12}a_1a_2-(a_1+a_2)+y_1a_1v_1+y_2a_2v_2 \tag{6.21}$$

令

$$E_i=g(x_i)-y_i=\left(\sum_{j=1}^{N}a_jy_jK(x_j,x_i)+b\right)-y_i,\quad i=1,2 \tag{6.22}$$

当 $i=1,2$ 时，E_i 为函数 $g(x)$ 对输入 x_i 的预测值与真实输出 y_i 之差。由 $a_1y_1=\varsigma-a_2y_2$ 及 $y_i^2=1$，可将 a_1 表示为

$$a_1=(\varsigma-y_2a_2)y_1 \tag{6.23}$$

代入式 6.21，得到只是 a_2 的函数的目标函数，对 a_2 求导数

$$\frac{\partial W}{\partial a_2}=K_{11}a_2+K_{22}a_2-2K_{12}a_2-K_{11}\varsigma y_2+K_{12}\varsigma y_2+y_1y_2-1-v_1v_2+y_2v_2 \tag{6.24}$$

令其为 0，得到

$$\begin{aligned}(K_{11}+K_{22}-2K_{12})a_2=&y_2(y_2-y_1+\varsigma K_{11}-\varsigma K_{12}+v_1-v2)\\ =&y2\left[y_2-y_1+\varsigma K_{11}-\varsigma K_{12}+\left(g(x_1)-\sum_{j=1}^{2}y_ja_jK_{1j}-b\right)\right.\\ &\left.\left(g(x_2)-\sum_{j=1}^{2}y_ja_jK_{2j}-b\right)\right]\end{aligned} \tag{6.25}$$

将 $\varsigma=a_1^{old}y_1+a_2^{old}y_2$ 代入，得到

$$\begin{aligned}(K_{11}+K_{22}-2K_{12})a_2^{new,unc}&=y_2((K_{11}+K_{22}-2K_{12})a_2^{old}y_2+y_2-y_1+g(x_1)-g(x_2))\\ &=(K_{11}+K_{22}-2K_{12})a_2^{old}+y_2(E_1-E_2)\end{aligned} \tag{6.26}$$

将 $\eta=K_{11}+K_{22}-2K_{12}$ 代入，于是得到

$$a_2^{new,unc}=a_2^{old}+\frac{y_2(E_1-E_2)}{\eta} \tag{6.27}$$

要使其满足不等式的约束必须将其限制在区间 $[L,H]$ 内，从而得到 a_2^{new} 的表达式

$$a_2^{new}=\begin{cases}H, & a_2^{new,unc}>H\\ a_2^{new,unc}, & L\le a_2^{new,unc}\le H\\ L, & a_2^{new,unc}<L\end{cases} \tag{6.28}$$

如果 $y_1\neq y_2$

$$L=max(0,a_2^{old}-a_1^{old}),\quad H=min(C,C+a_2^{old}-a_1^{old})$$

如果 $y_1 = y_2$

$$L = max(0, a_2^{old} + a_1^{old} - C), \quad H = min(C, a_2^{old} + a_1^{old})$$

变量的选择方法

SMO 算法在每个子问题中选择两个变量优化，其中至少一个变量是违反 KKT 条件的。

1. 第 1 个变量的选择

   SMO 称选择第 1 个变量的过程为外层循环。外层循环在训练样本中选取违反 KKT 条件最严重的样本点，并将其对应的变量作为第 1 个变量。具体地，检验训练样本点 (x_i, y_i) 是否满足 KKT 条件，即

$$a_i = 0 \Leftrightarrow y_i g(x_i) \geq 1 \tag{6.29}$$

$$C > a_i > 0 \Leftrightarrow y_i g(x_i) = 1 \tag{6.30}$$

$$a_i = C \Leftrightarrow y_i g(x_i) \leq 1 \tag{6.31}$$

2. 第 2 个变量的选择

   SMO 称选择第 2 个变量的过程为内层循环，假设在外层循环中已经找到第 1 个变量 a_1，现在要在内层循环中找到第 2 个变量 a_2。第 2 个变量选择的标准是希望能使 a_2 有足够大的变化。由式 6.27 知，a_2^{new} 是依赖于 $|E_1 - E_2|$ 的，加了加快计算速度，一种简单的做法是选择 a_2，使其对应的 $|E_1 - E_2|$ 最大。

3. 计算阈值 b 和差值 E_i

6.4 核函数

SVM 处理线性可分的情况，而对于非线性的情况，SVM 的处理方法是选择一个核函数 $K < \cdot, \cdot >$，通过将数据映射到高维空间，来解决在原始空间中线性不可分的问题。

此外，用对偶形式表示学习器的优势在于该表示中可调参数的个数不依赖输入属性的个数，通过使用恰当的核函数来替代内积，可以隐式得将非线性的训练数据映射到高维空间，而不增加可调参数的个数。

在线性不可分的情况下，支持向量机首先在低维空间中完成计算，然后通过核函数将输入空间映射到高维牲空间，最终在高维特征空间中构造出最优分离超平面，从而把平面上本身不好分的非线性数据分开。

而在我们遇到核函数之前，如果用原始的方法，那么在用线性学习器学习一个非线性关系，需要选择一个非线性特征集，并且将数据写成新的表达形式，这等价于应用一个固定的非线性映射，将数据映射到特征空间，在特征空间中使用线性学习器，因此考虑的假设集是这种类型的函数：

$$f(x) = \sum_{i=1}^{N} w_i \phi_i(x) + b \tag{6.32}$$

这里 $\phi : X \rightarrow F$ 是从输入空间到某个特征空间的映射，这意味着建立非线性学习器分为

两步：

1. 首先使用一个非线性映射将数据变换到一个特征空间 F
2. 然后在特征空间使用线性学习器分类

而由于对偶形式就是线性学习器的一个重要性质，这意味着假设可以表达为训练点的线性组合，因此决策规则可以用测试点和训练点的内积来表示：

$$f(x) = \sum_{i=1}^{l} a_i y_i < \phi(x_i), \phi(x) > +b \tag{6.33}$$

如果有一种方式可以在特征空间中直接计算内积 $< \phi(x_i), \phi(x) >$，就像在原始输入点的函数中一样，就有可能将两个步骤融合到一起建立一个非线性的学习器，这样直接计算法的方法称为**核函数方法**

定义 6.1. Kernel

核是一个函数 K，对所有 $x, z \in X$，满足 $K(x, z) =< \phi(x), \phi(z) >$，这里 ϕ 是从 X 到内积特征空间 F 的映射。 ♣

下面举一个核函数把低维空间映射到高维空间的例子：

我们考虑核函数 $K(v_1, v_2) =< v_i, v_2 >^2$，即“内积平方”，这里 $v_1 = (x_1, y_1), v_2 = (x_2, y_2)$ 是二维空间中的两个点。这个核函数对应着一个二维空间到三维空间的映射，它的表达式是：

$$P(x, y) = (x^2, \sqrt{2}xy, y^2)$$

可以验证，

$$\begin{aligned} < P(v_1), p(v_2) > &=< (x_1^2, \sqrt{2}x_1y_1, y_1^2), (x_2^2, \sqrt{2}x_2y_2, y_2^2) > \\ &= x_1^2x_2^2 + 2x_1x_2y_1y_2 + y_1^2y_2^2 \\ &= (x_1x_2 + y_1y_2)^2 \\ &=< v_1, v_2 >^2 \\ &= K(v_1, v_2) \end{aligned}$$

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。

核函数的本质

1. 实际中，我们会经常遇到线性不可分的样例，此时，我们的常用做法是把特征映射到高维空间中去
2. 但进一步，如果凡是遇到线性不可分的样例，一律映射到高维空间，那么这个维度大小会高到可怕的。那咋办呢？
3. 此时，核函数就隆重登场了，核函数的价值在于它虽然也是讲特征进行从低维到高维的转换，但核函数绝就绝在它事先在低维上进行计算，而将实质上的分类效果表现在了高维上，也就避免了直接在高维空间中的复杂计算。

几个核函数：

- 多项式核 $K(x_1, x_2) = (< x_1, x_2 > +R)^d$
- 高斯核 $K(x_1, x_2) = exp(-\|x_1 - x_2\|^2/2\sigma^2)$
- 线性核 $K(x_1, x_2) =< x_1, x_2 >$

6.5 软间隔与正则化

6.6 支持向量回归

6.7 核方法

第 7 章 混合模型和 EM 算法

7.1 一般形式的 EM 算法

EM 算法是一种迭代算法，1977 年由 Dempster 等人总结提出的，用于含有隐变量 (hidden variable) 的概率模型参数的极大似然估计，或极大后验概率估计。EM 算法的每次迭代由两步组成：E 步，求期望 (expectation)；M 步，求极大 (maximization)。所以这一算法称为期望极大算法 (expection maximization)，简称 EM 算法。

概率模型有时既含有观测变量 (observable variable)，又含有隐变量或潜在变量 (latent variable)。如果概率模型的变量都是观测变量，那么给定数据，可以直接用极大似然估计法，或贝叶斯估计法估计模型参数。但是，当模型含有隐变量时，就不能简单地使用这些估计方法。EM 算法就是含有隐变量的概率模型参数的极大似然估计法，或极大后验概率估计法。仅讨论极大似然估计，极大后验概率估计与其类似。

EM 算法的导出

为什么 EM 算法能近似实验对观测数据的极大似然估计呢？下面通过近似求解观测数据的对数似然函数的极大化问题来导出 EM 算法。

我们面对一个含有隐变量的概率模型，目标是极大化观测数据 (不完全数据)Y 关于参数 θ 的对数似然函数，即极大化

$$\begin{aligned} L(\theta) &= logP(Y|\theta) = log\sum_{Z} P(Y,Z|\theta) \\ &= log\left(\sum_{Z} P(Y|Z,\theta)P(Z|\theta)\right) \end{aligned} \tag{7.1}$$

注意到这一极大化的主要困难是式 7.1 中有未观测数据并有包含和 (或积分) 的对数。

事实上，EM 算法是通过迭代逐步极大化 $L(\theta)$ 的。假设在第 i 次迭代后 θ 的估计值是 $\theta^{(i)}$。我们希望新估计值 θ 能使 $L(\theta)$ 增加，即 $L(\theta) > L(\theta^{(i)})$，并逐步达到极大值。为此，考虑两者的差：

$$L(\theta) - L(\theta^{(i)}) = log\left(\sum_{Z} P(Y|Z,\theta)P(Z|\theta)\right) - logP(Y|\theta^{(i)}) \tag{7.2}$$

利用 Jensen 不等式[1](Jensen inequality) 得到其下界：

$$\begin{aligned} L(\theta) - L(\theta^{(i)}) &= log\left(\sum_Z P(Y|Z,\theta^{(i)})\frac{P(Y|Z,\theta)P(Z|\theta)}{P(Y|Z,\theta^{(i)})}\right) - logP(Y|\theta^{(i)}) \\ &\geqslant \sum_Z P(Y|Z,\theta^{(i)})log\frac{P(Y|Z,\theta)P(Z|\theta)}{P(Y|Z,\theta^{(i)})} - logP(Y|\theta^{(i)}) \\ &= \sum_Z P(Y|Z,\theta^{(i)})log\frac{P(Y|Z,\theta)P(Z|\theta)}{P(Y|Z,\theta^{(i)})P(Y|\theta^{(i)})} \end{aligned} \tag{7.3}$$

令

$$B(\theta,\theta^{(i)}) = L(\theta^{(i)}) + \sum_Z P(Y|Z,\theta^{(i)})log\frac{P(Y|Z,\theta)P(Z|\theta)}{P(Y|Z,\theta^{(i)})P(Y|\theta^{(i)})} \tag{7.4}$$

则

$$L(\theta) \geqslant B(\theta,\theta^{(i)}) \tag{7.5}$$

即函数 $B(\theta,\theta^{(i)})$ 是 $L(\theta)$ 的一个下界，因此**任何可以使 $B(\theta,\theta^{(i)})$ 增大的 θ，也可以使 $L(\theta)$ 增大**。为了使 $L(\theta)$ 尽可能大的增大，选择 $\theta^{(i+1)}$ 使 $B(\theta,\theta^{(i)})$ 达到极大，即

$$\theta^{(i+1)} = arg\ \max_\theta B(\theta,\theta^{(i)}) \tag{7.6}$$

现在求 $\theta^{(i+1)}$ 的表达式。省去对 θ 的极大化而言是常数的项，有

$$\begin{aligned} \theta^{(i+1)} &= arg\ \max_\theta \left(L(\theta^{(i)}) + \sum_Z P(Y|Z,\theta^{(i)})log\frac{P(Y|Z,\theta)P(Z|\theta)}{P(Y|Z,\theta^{(i)})P(Y|\theta^{(i)})}\right) \\ &= arg\ \max_\theta \left(\sum_Z P(Y|Z,\theta^{(i)})log(P(Y|Z,\theta)P(Z|\theta))\right) \\ &= arg\ \max_\theta \left(\sum_Z P(Y|Z,\theta^{(i)})log(P(Y,Z|\theta))\right) \\ &= arg\ \max_\theta \underline{Q(\theta,\theta^{(i)})} \end{aligned} \tag{7.7}$$

式 7.7 等价于 EM 算法的一次迭代，即求 Q 函数及其极大化。**EM 算法是通过不断求解下界的极大化逼近求解对数似然函数极大化的算法**。

定义 7.1. Q 函数

完全数据的对数似然函数 $logP(Y,Z|\theta)$ 关于在给定观测数据 Y 和当前参数 $\theta^{(i)}$ 下对未观测数据 Z 的条件概率分布 $P(Z|Y,\theta^{(i)})$ 的期望称为 Q 函数。即

$$Q(\theta,\theta^{(i)}) = E_Z[logP(Y,Z|\theta)|Y,\theta^{(i)}] \tag{7.8}$$

[1]这里用到的是 $log\sum_j \lambda_j y_j \geqslant \sum_j \lambda_j log y_j$

EM 算法

输入：观测变量数据 Y，隐变量数据 Z，联合分布 $P(Y,Z|\theta)$，条件分布 $P(Z|Y,\theta)$；

输出：模型参数 θ

1. 选择参数的初值 $\theta^{(0)}$，开始迭代；
2. E 步：记 $\theta^{(i)}$ 为第 i 次迭代参数 θ 的估计值，在第 i+1 次迭代的 E 步，计算

$$\begin{aligned} Q(\theta,\theta^{(i)}) &= E_Z[logP(Y,Z|\theta)|Y,\theta^{(i)}] \\ &= \sum_Z logP(Y,Z|\theta)P(Z|Y,\theta^{(i)}) \end{aligned} \tag{7.9}$$

这里，$P(Z|Y,\theta^{(i)})$ 是在给定观测数据 Y 和当前的参数估计 $\theta^{(i)}$ 下隐变量数据 Z 的条件概率分布；

3. M 步：求使 $Q(\theta,\theta^{(i)})$ 极大化的 θ，确定第 i+1 次迭代的参数的估计值 $\theta^{(i+1)}$

$$\theta^{(i+1)} = arg\ \max_\theta Q(\theta,\theta^{(i)}) \tag{7.10}$$

4. 重复第 (2)、(3) 步，直到收敛。

EM 算法的收敛性

EM 算法提供一种近似计算含有隐变量概率模型的极大似然估计的方法。EM 算法的最大优点是简单性和普适性。我们自然地要问：EM 算法得到的估计序列是否收敛？如果收敛，是否收敛到全局最大值或局部极大值？

定理 7.1

设 $P(Y|\theta)$ 为观测数据的似然函数，$\theta^{(i)}(i=1,2,\dots)$ 为 EM 算法得到的参数估计序列，$P(Y|\theta^{(i)})(i=1,2,\dots)$ 为对应的似然函数序列，则 $P(Y|\theta^{(i)})$ 是单调递增的，即

$$P(Y|\theta^{(i+1)}) \geqslant P(Y|\theta^{(i)}) \tag{7.11}$$

设 $L(\theta) = logP(Y|\theta)$ 为观测数据的对数似然函数，$L(\theta^{(i)})(i=1,2,\dots)$ 为对应的对数似然函数序列。

1. 如果 $P(Y|\theta)$ 有上界，则 $L(\theta^{(i)}) = logP(Y|\theta^{(i)})$ 收敛到某一值 L^*
2. 在函数 $Q(\theta,\theta')$ 与 $L(\theta)$ 满足一定条件下，由 EM 算法得到的参数估计序列 $\theta^{(i)}$ 的收敛值 θ^* 是 $L(\theta)$ 的稳定点 ♡

7.2 EM 的另一种观点

7.3 K 均值聚类

7.4 混合高斯

EM 算法的一个重要应用是高斯混合模型的参数估计。高斯混合模型应用广泛，在许多情况下，EM 算法是学习高斯混合模型 (Gaussian misture model) 的有效方法。

定义 7.2. 高斯混合模型

高斯混合模型是指具有如下形式的概率分布模型：

$$P(y|\theta) = \sum_{k=1}^{K} \alpha_k \phi(y|\theta_k) \tag{7.12}$$

其中，α_k 是系数，$\alpha_k \geqslant 0, \sum_{k=1}^{K} \alpha_k = 1$；$\phi(Y|\theta_k)$ 是高斯分布密度，$\theta_k = (\mu_k, \sigma_k^2)$，

$$\phi(y|\theta_k) = \frac{1}{\sqrt{2\pi}\sigma_k} exp\left(-\frac{(y-\mu_k)^2}{2\sigma_k^2}\right) \tag{7.13}$$

称为第 k 个模型。 ♣

高斯混合模型参数估计的 EM 算法

假设观测数据 $y_1, y_2, \ldots, y_N$ 由高斯混合模型生成，

$$P(y|\theta) = \sum_{k=1}^{K} \alpha_k \phi(y|\theta_k) \tag{7.14}$$

其中，$\theta = (\alpha_1, \alpha_2, \ldots, \alpha_K; \theta_1, \theta_2, \ldots, \theta_K)$。我们用 EM 算法估计高斯混合模型的参数 θ

1. 明确隐变量，写出完全数据的对数似然函数

   隐变量由 γ_{jk} 表示，其定义如下：

$$\gamma_{jk} = \begin{cases} 1, & \text{第 j 个观测来自第 k 个模型} \\ 0, & \text{否则} \end{cases} \tag{7.15}$$

   γ_{jk} 是 0-1 随机变量。那么完全数据是

$$(y_j, \gamma_{j1}, \gamma_{j2}, \ldots, \gamma_{jK}), \qquad j = 1, 2, \ldots, K$$

于是可以写出完全数据的似然函数

$$\begin{aligned}P(y,\gamma|\theta) &= \prod_{j=1}^{N} P(y_j,\gamma_{j1},\gamma_{j2},\ldots,\gamma_{jK}|\theta)\\ &= \prod_{k=1}^{K}\prod_{j=1}^{N}[\alpha_k\phi(y|\theta_k)]^{\gamma_{jk}}\\ &= \prod_{k=1}^{K}\alpha_k^{n_k}\prod_{j=1}^{N}[\phi(y|\theta_k)]^{\gamma_{jk}}\\ &= \prod_{k=1}^{K}\alpha_k^{n_k}\prod_{j=1}^{N}\left[\frac{1}{\sqrt{2\pi}\sigma_k}exp\left(-\frac{(y-\mu_k)^2}{2\sigma_k^2}\right)\right]^{\gamma_{jk}}\end{aligned}$$

式中，$n_k = \sum_{j=1}^{N}\gamma_{jk}$，$\sum_{k=1}^{K} n_k = N$。(意思是选择第 k 个高斯模型的次数)
那么，完全数据的对数似然函数为

$$logP(y,\gamma|\theta) = \sum_{k=1}^{K}\left\{n_k log\alpha_k + \sum_{j=1}^{N}\gamma_{jk}\left[log\left(\frac{1}{\sqrt{2\pi}}\right) - log\sigma_k - \frac{1}{2\sigma_k^2}(y_j-\mu_k)^2\right]\right\}$$

2. EM 算法的 E 步：确定 Q 函数

$$\begin{aligned}Q(\theta,\theta^{(i)}) &= E[logP(y,\gamma|\theta)|y,\theta^{(i)}]\\ &= E\left\{\sum_{k=1}^{K}\left\{n_k log\alpha_k + \sum_{j=1}^{N}\gamma_{jk}\left[log\left(\frac{1}{\sqrt{2\pi}}\right) - log\sigma_k - \frac{1}{2\sigma_k^2}(y_j-\mu_k)^2\right]\right\}\right\}\\ &= \sum_{k=1}^{K}\left\{\overbrace{n_k log\alpha_k}^{\text{常数}} + \sum_{j=1}^{N}E(\gamma_{jk})\left[log\left(\frac{1}{\sqrt{2\pi}}\right) - log\sigma_k - \frac{1}{2\sigma_k^2}(y_j-\mu_k)^2\right]\right\}\end{aligned}$$

这里需要计算 $E(\gamma_{jk}|y,\theta)$，记为 $\hat{\gamma}_{jk}$

$$\begin{aligned}\hat{\gamma}_{jk} &= E(\gamma_{jk}|y,\theta) = P(\gamma_{jk}=1|y,\theta)(\text{二项分布})\\ &= \frac{P(\gamma_{jk}=1|y,\theta)}{\sum_{k=1}^{K}P(\gamma_{jk}=1|y,\theta)}\\ &= \frac{P(y_j|\gamma_{jk}=1,\theta)P(\gamma_{jk}=1|\theta)}{\sum_{k=1}^{K}P(y_j|\gamma_{jk}=1,\theta)P(\gamma_{jk}=1|\theta)}\\ &= \frac{\alpha_k\phi(y_j|\theta_k)}{\sum_{k=1}^{K}\alpha_k\phi(y_j|\theta_k)},\quad j=1,2,\ldots,N;k=1,2,\ldots,K\end{aligned}$$

$\hat{\gamma}_{jk}$ 是在当前模型参数下第 j 个观测数据来自第 k 个分模型的概率，称为分模型 k 对观测数据 y_j 的响应度。代入式中得

$$Q(\theta, \theta^{(i)}) = \sum_{k=1}^{K} \left\{ n_k log\alpha_k + \sum_{j=1}^{N} \hat{\gamma}_{jk} \left[log\left(\frac{1}{\sqrt{2\pi}}\right) - log\sigma_k - \frac{1}{2\sigma_k^2}(y_j - \mu_k)^2 \right] \right\} \tag{7.16}$$

3. 确定 EM 算法的 M 步迭代的 M 步是求函数 $Q(\theta, \theta^{(i)})$ 对 θ 的极大值，即求新一轮迭代的模型参数：

$$\theta^{(i+1)} = arg\ \max_{\theta} Q(\theta, \theta^{(i)})$$

分别对 μ_k, σ_k^2 求偏导并令其为 0，即可得到 $\hat{\mu}_k, \hat{\sigma}_k$。求 $\hat{a}_k$ 是在 $\sum_{k=1}^{K} = 1$ 条件下求偏导并令其为 0 得到。

$$\hat{\mu}_k = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk} y_j}{\sum_{j=1}^{N} \hat{\gamma}_{jk}}, \quad k = 1, 2, \ldots, K \tag{7.17}$$

$$\hat{\sigma}_k^2 = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk}(y_j - \mu_k)^2}{\sum_{j=1}^{N} \hat{\gamma}_{jk}}, \quad k = 1, 2, \ldots, K \tag{7.18}$$

$$\hat{\alpha}_k = \frac{n_k}{N} = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk}}{N}, \quad k = 1, 2, \ldots, K \tag{7.19}$$

$$\tag{7.20}$$

4. 重复以上计算，直到对数似然函数值不再有明显的变化为止。

现将估计高斯混合模型参数的 EM 算法总结如下

输入：观测数据 $y_1, y_2, \ldots, y_N$，高斯混合模型；

输出：高斯混合模型参数。

(1) 取参数的初始值开始迭代

(2) E 步：依据当前模型参数，计算分模型 k 对观测数据 y_j 的响应度

$$\hat{\gamma}_{jk} = \frac{\alpha_k \phi(y_j|\theta_k)}{\sum_{k=1}^{K} \alpha_k \phi(y_j|\theta_k)}, \quad j = 1, 2, \ldots, N; k = 1, 2, \ldots, K$$

(3) M 步:计算新一轮迭代的模型参数

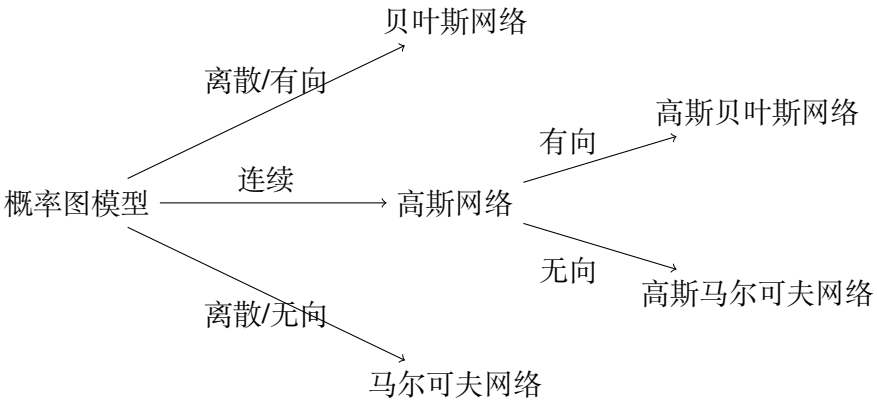
$$\hat{\mu}_k = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk} y_j}{\sum_{j=1}^{N} \hat{\gamma}_{jk}}, \quad k = 1, 2, \ldots, K$$

$$\hat{\sigma}_k^2 = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk} (y_j - \mu_k)^2}{\sum_{j=1}^{N} \hat{\gamma}_{jk}}, \quad k = 1, 2, \ldots, K$$

$$\hat{\alpha}_k = \frac{n_k}{N} = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk}}{N}, \quad k = 1, 2, \ldots, K$$

(4) 重复以上计算,直到对数似然函数值不再有明显的变化为止。

第 8 章 概率图模型



8.1 贝叶斯网络

8.2 条件独立

8.3 马尔科夫随机场

8.4 图模型中的推断

8.5 隐马尔可夫模型

隐马尔可夫模型 (hidden Markov model,HMM) 是可用于标注问题的统计学习模型,描述由隐藏的马尔可夫链随机生成观测序列的过程,属于生成模型。本节首先介绍隐马尔可夫模型的基本概念,然后分别叙述隐马尔可夫模型的概率计算算法、学习算法以及预测算法。隐马尔可夫模型在语音识别、自然语言处理、生物信息、模式识别等领域有着广泛的应用。

隐马尔可夫模型的基本概念

隐马尔可夫模型是关于时序的概率模型,描述由一个隐藏的马尔可夫链随机生成不可观测的状态随机序列,再由各个状态生成一个观测而产生观测随机序列的过程。隐藏的马尔可夫链随机生成的状态的序列,称为状态序列 (state sequence);每个状态生成一个观测,而由此产生的观测的随机序列,称为观测序列 (observation sequence)。序列的每一个位置又可以看作是一个时刻。隐马尔可夫模型由初始概率分布、状态转移概率分布以及观测概率分布确定。隐马尔可夫模型定义如下:

设 Q 是所有可能的状态的集合，V 是所有可能的观测的集合。

$$Q = \{q_1, q_2, \ldots, q_N\}, \quad V = \{v_1, v_2, \ldots, v_M\} \tag{8.1}$$

设 N 是所有可能的状态数，M 是可能的观测数。
I 是长度为 T 的状态序列，O 是对应的观测序列。

$$I = \{i_1, i_2, \ldots, i_T\}, \quad O = \{o_1, o_2, \ldots, o_T\} \tag{8.2}$$

A 是状态转移概率矩阵：

$$A = \left[a_{ij}\right]_{N\times N} \tag{8.3}$$

其中，

$$a_{ij} = P(i_{t+1} = q_j | i_t = q_i), i = 1, 2, \ldots, N; j = 1, 2, \ldots, N \tag{8.4}$$

是在时刻 t 处于状态 q_i 的条件下在时刻 $t + 1$ 转移到状态 q_j 的概率。
B 是观测概率矩阵：

$$B = \left[b_j(k)\right]_{N\times N} \tag{8.5}$$

其中，

$$b_j(k) = P(o_t = v_k | i_t = q_j), k = 1, 2, \ldots, M; j = 1, 2, \ldots, N \tag{8.6}$$

是在时刻 t 处于状态 q_j 的条件下生成观测 v_k 的概率。π 是初始状态概率向量：

$$\pi = (\pi_i) \tag{8.7}$$

其中，

$$\pi_i = P(i_1 = q_i), i = 1, 2, \ldots, N \tag{8.8}$$

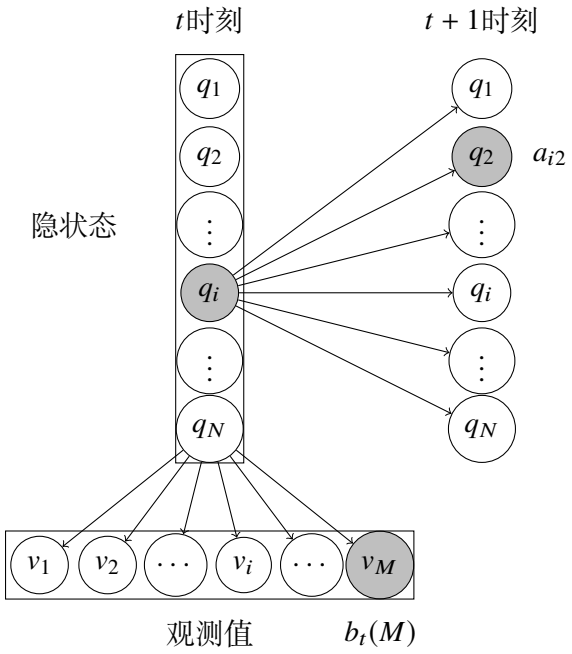
是时刻 $t = 1$ 处于状态 q_i 的概率。

隐马尔可夫模型由初始状态概率向量 π、状态转移概率矩阵 A 和观测概率矩阵 B 决定。π 和 A 决定状态序列，B 决定观测序列。因此，隐马尔可夫模型 λ 可以用三元符号表示，即

$$\lambda = (A, B, \pi) \tag{8.9}$$

A, B, π 称为隐马尔可夫模型的三要素。从定义可知，隐马尔可夫模型作了两个基本假设：

1. 齐次马尔可夫性假设，即假设隐藏的马尔可夫链在任意时刻 t 的状态只依赖于其前一时刻的状态，与其他时刻的状态及观测无关，也与时刻 t 无关。
2. 观测独立性假设，即假设任意时刻的观测只依赖于该时刻的马尔可夫链的状态，与其他观测及状态无关。



隐马尔可夫模型的 3 个基本问题

1. 概率计算问题。给定模型 $\lambda = (A, B, \pi)$ 和观测序列 $O = (o_1, o_2, \ldots, o_T)$，计算在模型 λ 下观测序列 O 出现的概率 $P(O|\lambda)$
2. 学习问题。已知观测序列 $O = (o_1, o_2, \ldots, o_T)$，估计模型 $\lambda = (A, B, \pi)$ 参数，使得在该模型下观测序列概率 $P(O|\lambda)$ 最大。即用极大似然估计的方法估计参数。
3. 预测问题。也称为解码（decoding）问题。已知模型 $\lambda = (A, B, \pi)$ 和观测序列 $O = (o_1, o_2, \ldots, o_T)$，求对给定观测序列条件概率 $P(I|\lambda)$ 最大的状态序列 $I = (i_1, i_2, \ldots, i_T)$。即给定观测序列，求最有可能的对应的状态序列。

问题一：概率计算算法

直接计算法

给定模型 $\lambda = (A, B, \pi)$ 和观测序列 $O = (o_1, o_2, \ldots, o_T)$，计算在模型 λ 下观测序列 O 出现的概率 $P(O|\lambda)$。最直接的方法是按概率公式直接计算。

$$\begin{aligned} P(O|\lambda) &= \sum_I P(O|I, \lambda) P(I|\lambda) \\ &= \sum_{i_1, i_2, \ldots, i_T} \pi_{i_1} b_{i_1}(o_1) a_{i_1 i_2} \ldots a_{i_{T-1} i_T} b_{i_T}(o_T) \end{aligned} \tag{8.10}$$

利用公式计算量大，这种算法不可行。

前向算法

首先定义前现概率

定义 8.1. 前向概率

给定隐马尔可夫模型 λ，定义到时刻 t 部分观测序列为 $O=(o_1,o_2,\ldots,o_t)$ 且状态为 q_i 的概率为前向概率，记作

$$\alpha_t(i)=P(o_1,o_2,\ldots,o_t,i_t=q_i|\lambda) \tag{8.11}$$

可以递推地求得前向概率 $\alpha_t(i)$ 及观测序列概率 $P(O|\lambda)$

输入：隐马尔可夫模型 λ，观测序列 O；

输出：观测序列概率 $P(O|\lambda)$

(1) 初值

$$\alpha_1(i)=\pi_i b_i(o_1),\qquad i=1,2,\ldots,N \tag{8.12}$$

(2) 递推，对 $t=1,2,\ldots,T-1$

$$\alpha_{t+1}(i)=\left[\sum_{j=1}^{N}\alpha_t(j)a_{ji}\right]b_i(o_{t+1}) \tag{8.13}$$

(3) 终止

$$P(O|\lambda)=\sum_{i=1}^{N}\alpha_T(i) \tag{8.14}$$

前向算法实际是基于“状态序列的路径结构”递推计算 $P(O|\lambda)$ 的算法。前身算法的高效的关键是其局部计算前向概率，然后利用路径结构将前向概率“递推”到全局，得到 $P(O|\lambda)$.

后向概率

定义 8.2. 后向算法

给定隐马尔可夫模型 λ，定义在时刻 t 状态为 q_i 的条件下，从 $t+1$ 到 T 的部分观测序列为 $o_{t+1},o_{t+2},\ldots,o_T$ 的概率为后向概率，记作

$$\beta_t(i)=P(o_{t+1},o_{t+2},\ldots,o_T|i_t=q_i,\lambda) \tag{8.15}$$

可以用递推的方法求得后向概率 $\beta_t(i)$ 及观测序列概率 $P(O|\lambda)$

输入：隐马尔可夫模型 λ，观测序列 O；

输出：观测序列概率 $P(O|\lambda)$

(1) 初值

$$\beta_T(i)=1,\qquad i=1,2,\ldots,N \tag{8.16}$$

(2) 递推，对 $t=T-1,T-2,\ldots,1$

$$\beta_t(i)=\sum_{j=1}^{N}a_{ij}b_j(o_{t+1})\beta_{t+1}(j),\quad i=1,2,\ldots,N \tag{8.17}$$

(3) 终止

$$P(O|\lambda)=\sum_{i=1}^{N}\pi_i b_i(o_1)\beta_1(i) \tag{8.18}$$

利用前向概率和后向概率的定义可以将观测序列概率 $P(O|\lambda)$ 统一写成

$$P(O|\lambda)=\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j),\quad t=1,2,\ldots,T-1 \tag{8.19}$$

一些概率与期望值的计算

1. 给定模型 λ 和观测 O,在时刻 t 处于状态 q_j 的概率。记 $\gamma_t(i)$
2. 给定模型 λ 和观测 O,在时刻 t 处于状态 q_j 且在时刻 $t+1$ 处于状态 q_j 的概率。记 $\xi_t(i,j)$
3. 一些有用的期望值

问题二:学习算法

隐马尔可夫模型的学习,根据训练数据是包括观测序列和对应的状态序列还是只有观测序列,可以分别由监督学习与非监督学习实现。

监督学习算法

假设训练数据包含 S 个长度相同的观测序列和对应的状态序列 $\{(O_1,I_1),\ldots,(O_S,I_S)\}$,那么可以利用**极大似然法**来估计隐马尔可夫模型的参数。具体方法如下。

1. 转移概率 a_{ij} 的估计

   设样本中时刻 t 处于状态 i 时刻 $t+1$ 转移到状态 j 的频数为 A_{ij},那么状态转移概率 a_{ij} 的估计是

$$\hat{a}_{ij}=\frac{A_{ij}]}{\sum_{j=1}^{N}A_{ij}},\qquad i=1,2,\ldots,N;j=1,2,\ldots,N \tag{8.20}$$

2. 观测概率 $b_j(k)$ 的估计

   设样本中状态为 j 并观测为 k 的频数是 B_{jk},那么状态为 j 观测为 k 的概率 $b_j(k)$ 的估计是

$$\hat{b}_j(k)=\frac{B_{jk}}{\sum_{k=1}^{M}B_{jk}},\qquad j=1,2,\ldots,k=1,2,\ldots,M \tag{8.21}$$

3. 初始状态概率 π_i 的估计 $\hat{\pi}_i$ 为 S 个样本中初始状态为 q_i 的频率

Baum-Welch 算法

假设训练数据包含 S 个长度相同的观测序列 $\{(O_1, I_1), \ldots, (O_S, I_S)\}$ 而没有对应的状态序列，目标是学习隐马尔可夫模型 $\lambda = (A, B, \pi)$ 的参数。我们将观测序列数据看作观测数据 O，状态序列数据看作不可观测的隐数据 I，那么隐马尔可夫模型事实上是一个含有隐变量的概率模型

$$P(O|\lambda) = \sum_I P(O|I, \lambda)P(I|\lambda) \tag{8.22}$$

它的参数学习可以由 EM 算法实现。

1. 确定完全数据的对数似然函数
   所有观测数据写成 $O = (o_1, o_2, \ldots, o_T)$，所有隐数据写成 $I = (i_1, i_2, \ldots, i_T)$ 完全数据是 $(O, I) = (o_1, o_2, \ldots, o_T, i_1, i_2, \ldots, i_T)$。完全数据的对数似然函数是 $logP(O, I|\lambda)$
2. EM 算法的 E 步：求 Q 函数 $Q(\lambda, \bar{\lambda})$

$$E_I[logP(O, I|\lambda)|O, \bar{\lambda}] = \sum_I logP(O, I|\lambda)P(I|O, \bar{\lambda}) \tag{8.23}$$

$$Q(\lambda, \bar{\lambda}) = \sum_I logP(O, I|\lambda)P(O, I|\bar{\lambda}) \tag{8.24}$$

$P(I|O, \bar{\lambda}) = P(I, O|\bar{\lambda})/P(O|\bar{\lambda})$ 省略了对 λ 而言的常数因子。

$$P(O, I|\lambda) = \pi_{i_1} b_{i_1}(o_1) a_{i_1 i_2} \ldots a_{i_{T-1} i_T} b_{i_T}(o_T) \tag{8.25}$$

于是函数 $Q(\lambda, \bar{\lambda})$ 可以写成：

$$\begin{aligned} Q(\lambda, \bar{\lambda}) = & \sum_I log\pi_{i_1} P(O, I|\bar{\lambda}) \\ & + \sum_I \left(\sum_{t=1}^{T-1} log a_{i_t i_{t+1}} \right) P(O, I|\bar{\lambda}) \\ & + \sum_I \left(\sum_{t=1}^{T} log b_{i_t}(o_t) \right) P(O, I|\bar{\lambda}) \end{aligned} \tag{8.26}$$

3. EM 算法的 M 步：极大化 $Q(\lambda, \bar{\lambda})$ 求模型参数 A, B, π
   由于要极大化的参数在式 8.26 中单独地出现在 3 个项中，所以只需对各项分别极大化。注意到

$$\sum_{i=1}^{N} \pi_i = 1, \sum_{j=1}^{N} a_{ij} = 1, \sum_{k=1}^{M} b_j(k) = 1,$$

利用拉格朗日乘子法，求得

$$\pi_i = \frac{P(O, i_1 = i|\bar{\lambda})}{P(O|\bar{\lambda})} = \gamma_1(i) \tag{8.27}$$

$$a_{ij} = \frac{\sum\limits_{t=1}^{T-1} P(O, i_t = i, i_{t+1} = j|\bar{\lambda})}{\sum\limits_{t=1}^{T-1} P(O, i_t = i|\bar{\lambda})} = \frac{\sum\limits_{t=1}^{T-1} \xi_t(i, j)}{\sum\limits_{t=1}^{T-1} \gamma_t(i)} \tag{8.28}$$

$$b_j(k) = \frac{\sum\limits_{t=1}^{T} P(O, i_t = j|\bar{\lambda}) I(o_t = v_k)}{\sum\limits_{t=1}^{T} P(O, i_t = j|\bar{\lambda})} = \frac{\sum\limits_{t=1, o_t = v_k}^{T} \gamma_t(j)}{\sum\limits_{t=1}^{T} \gamma_t(j)} \tag{8.29}$$

问题三：预测算法

近似算法

近似算法的想法是，在每个时刻 t 选择在该时刻最有可能出现的状态 i_t^*，从而得到一个状态序列 $I^* = (i_1^*, i_2^*, \ldots, i_T^*)$，将它作为预测的结果。近似算法的优点是计算简单，其缺点是不能保证预测的状态序列整体是最有可能的状态序列，因为预测的状态序列可能有实际不发生的部分。

维特比算法

维特比算法实际是用动态规划解隐马尔可夫模型预测问题，即用动态规划 (dynamic programming) 求概率最大路径 (最优路径)。这时一条路径对应着一个状态序列。

首先导入两个变量 δ 和 ψ。定义在时刻 t 状态为 i 的所有单个路径 $i_1, i_2, \ldots, i_t$ 中概率最大值为

$$\begin{aligned} \delta_{t+1} &= \max_{i_1, i_2, \ldots, i_t} P(i_t = i, i_{t-1}, \ldots, i_1, o_{t+1}, \ldots, o_1|\lambda) \\ &= \max_{1 \leqslant j \leqslant N} [\delta_t(j) a_{ji}] b_i(o_{t+1}), \quad i = 1, 2, \ldots, N \end{aligned} \tag{8.30}$$

定义在时刻 t 状态为 i 的所有单个路径 $i_1, i_2, \ldots, i_t$ 中概率最大的路径的第 $t-1$ 个结点为

$$\psi_t(i) = arg \max_{1 \leqslant j \leqslant N} [\delta_{t-1}(j) a_{ji}], \quad i = 1, 2, \ldots, N \tag{8.31}$$

(维特比算法)

输入：模型 $\lambda = (A, B, \pi)$，观测序列 $O = (o_1, o_2, \ldots, o_T)$；

输出：最优路径 $I^* = (i_1^*, i_2^*, \ldots, i_T^*)$

1. 初始化

$$\delta_1 = \pi_i b_i(o_1),$$

$$\psi_1(i) = 0, \qquad i = 1, 2, \ldots, N$$

2. 递推。对 $t = 2, 3, \ldots, T$

$$\delta_{t+1} = \max_{1 \leqslant j \leqslant N} [\delta_t(j) a_{ji}] b_i(o_{t+1})$$

$$\psi_t(i) = arg \max_{1 \leqslant j \leqslant N} [\delta_{t-1}(j) a_{ji}], \quad i = 1, 2, \ldots, N$$

3. 终止

$$P^* = \max_{1 \leqslant i \leqslant N} \delta_T(i)$$

$$i_T^* = arg \max_{1 \leqslant i \leqslant N} \delta_T(i)$$

4. 最优路径回溯。对 $t = T - 1, T - 2, \ldots, 1$

$$i_t^* = \psi_{t+1}(i_{t+1}^*)$$

求得最佳路径 $I^* = (i_1^*, i_2^*, \ldots, i_T^*)$

8.6 条件随机场

第 9 章 采样方法

第 10 章 近似推断

第 11 章 连续潜在变量

11.1 主成分分析

主成分分析,或者称为 PCA,是一种被广泛使用的技术,应用的领域包括维度降低、有损数据压缩、特征抽取「数据可视化。它也被称为 Karhunen-Loeve 变换。

有两种经常使用的 PCA 的定义,它们会给出同样的算法。PCA 可以被定义为数据在低维线性空间上的正交投影,这个线性空间被称为主空间 (principal subspace),使得投影数据的方差被最大化。等价地,它也可以被定义为使得平均投影代价最小的线性投影。平均投影代价是指数据点和它们的投影之间的平均平方距离。

考察一组观测数据集 $\{x_n\}$,其中 $n = 1, \ldots, N$,因此 x_n 是一个 D 维欧几里德空间中的变量。我们的目标是将数据投影到维度 $M < D$ 的空间中,同时最大化投影数据的方差。

样本集合

$$X = (x_1\ x_2\ \ldots x_N)^T_{N\times p} = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1p} \\ x_{21} & x_{22} & \ldots & x_{2p} \\ \ldots & \ldots & \ldots & \ldots \\ x_{11} & x_{12} & \ldots & x_{np} \end{pmatrix}_{N\times p} \tag{11.1}$$

样本均值

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x_n = \frac{1}{N}(x_1\ x_2\ \ldots\ x_n)\begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}_{N\times 1} = \frac{1}{N}X^T I_N \tag{11.2}$$

样本方差

$$\begin{aligned} S &= \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})(x_n - \bar{x})^T \\ &= \frac{1}{N}(x_1 - \bar{x}\ x_2 - \bar{x}\ \ldots\ x_N - \bar{x})\begin{pmatrix} (x_1 - \bar{x})^T \\ (x_2 - \bar{x})^T \\ \vdots \\ (x_N - \bar{x})^T \end{pmatrix} \\ &= \frac{1}{N}X^T\overbrace{(I_N - \frac{1}{N}I_N I_N^T)}^{H_N-\text{中心矩阵}}\cdot(I_N - \frac{1}{N}I_N I_N^T)^T \cdot X \\ &= \frac{1}{N}X^T H X \end{aligned} \tag{11.3}$$

中心矩阵 H 有以下良好的性质

$$H = I_N - \frac{1}{N} I_N I_N^T \tag{11.4}$$

$$H^T = H \tag{11.5}$$

$$H^n = H \tag{11.6}$$

最大方差形式

首先，考虑在一维空间 ($M = 1$) 上的投影。我们可心使用 D 维向量 u_1 定义这个空间的方向。不失一般性，我们假定选择一个单位向量，从而 $u_1^T u_1 = 1$(注意，我们只对 u_1 的方向感兴趣，而对 u_1 本身的大小不感兴趣)。这样，每个数据点 x_n 被投影到一个标题值 $x_1^T x_n$ 上。投影数据的均值是 $u_1^T \bar{x}$。投影数据的方差为

$$\begin{aligned} J &= \frac{1}{N} \sum_{n=1}^{N} \{u_1^T x_n - u_1^T \bar{x}\}^2 \\ &= u_1^T \underbrace{\sum_{n=1}^{N} \frac{1}{N} (x_n - \bar{x}) \cdot (x_n - \bar{x})^T}_{S} u_1 \\ &= u_1^T S u_1 \end{aligned} \tag{11.7}$$

S 是数据的协方差矩阵，优化问题变成

$$\begin{aligned} \hat{u_1} &= arg\,max\, u_1^T S u_1 \\ s.t\ \ & u_1^T u_1 = 1 \end{aligned} \tag{11.8}$$

利用拉格朗日乘子法，求偏导并令其为零，我们看到驻点满足

$$S u_1 = \lambda_1 u_1 \tag{11.9}$$

表明 u_1 一定是 S 的一个特征向量，这个特征向量被称为第一主成分。

我们可以用一种增量的方式定义额外的主成分，方法为：在所有与那些已经考虑过的方向正交的所有可能的方向中，将新的方向选择为最大化投影方差的方向。

总结一下，主成分分析涉及到计算数据集的均值 $\bar{x}$ 和协方差矩阵 S，然后寻找 S 的对应于 M 个最大特征值的 M 个特征向量。

最小误差形式

引入 D 维基向量的一个完整的单位正交集合 $\{u_i\}$，其中 $i = 1, \ldots, D$，且满足

$$u_i^T u_j = \delta_{ij} \tag{11.10}$$

由于基是完整的，因此每个数据点可以精确地表示为基向量的一个纯性组合，即

$$x_n = \sum_{i=1}^{D} a_{ni} u_i \tag{11.11}$$

其中，系数 a_{ni} 对于不同的数据点来说是不同的。这对应于将坐标系旋转到了一个由 $\{u_i\}$ 定义的新坐标系，原始的D个分量 $\{x_{n1}, \ldots, x_{nD}\}$ 被替换为一个等价的集合 $\{a_{n1}, \ldots, a_{nD}\}$。我们有 $a_{nj} = x_n^T u_j$，因此不失一般性

$$x_n = \sum_{i=1}^{D} (x_n^T u_i) u_i \tag{11.12}$$

然而，我们的目标是使用限定数量 $M < D$ 个变量的一种表示方法来近似数据点，这对应于在低维子空间上的一个投影。不失一般性，M 维线性子空间可以用前 M 个基向量表示，因此我们可以用下式来近似每个数据点 x_n

$$\widetilde{x}_n = \sum_{i=1}^{M} z_{ni} u_i + \sum_{i=M+1}^{D} b_i u_i \tag{11.13}$$

其中 $\{z_{ni}\}$ 依赖于特定的数据点，而 $\{b_i\}$ 是常数，对于所有数据点都相同。我们可以任意选择 $\{u_i\}, \{z_{ni}\}, \{b_i\}$，从而最小化由维度降低所引入的失真。作为失真的度量，我们使用原始数据点与它的近似点 $\widetilde{x}_n$ 之间的平方距离，在数据集上取平均。因此我们的目标是最小化

$$J = \frac{1}{N} \sum_{n=1}^{N} \|x_n - \widetilde{x}_n\|^2 = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=M+1}^{D} (x_n^T u_i - \bar{x}^T u_i)^2 = \sum_{i=M+1}^{D} u_i^T S u_i \tag{11.14}$$

剩下的任务是关于 $\{u_i\}$ 对 J 进行最小化。同样利用拉格朗日乘子法能够得到

$$S u_i = \lambda_i u_i \tag{11.15}$$

PCA 的应用

11.2 概率 PCA

前一节讨论的 PCA 的形式所基于的是将数据线性投影到比原始数据空间维度更低的子空间内。PCA 也可以被视为概率潜在变量模型的最大似然解。PCA 的这种形式，被称为概率 PCA(probabilistic PCA)，与传统的 PCA 相比，会带来如下几个优势。

- 概率 PCA 表示高斯分布的一个限制形式，其中自由参数的数量可以受到限制，同时仍然使得模型能够描述数据集的主要的相关关系。
- 我们可以为 PCA 推导一个 EM 算法，这个算法在只有几个主要的特征向量需要求出的情况下，计算效率比较高，并且避免了计算数据协方差的中间步骤。

- 概率模型与 EM 的结合使得我们能够处理数据集里缺失值的问题。
- 概率 PCA 混合模型可以用一种有理有据的方式进行形式化，并且可以使用 EM 算法进行训练。
- 概率 PCA 构成了 PCA 的贝叶斯方法的基础，其中主子空间的维度可以自动从数据中找到。
- 似然函数的存在使得直接与其他的概率密度模型进行对比成为可能。相反，传统的 PCA 会给接近主子空间的数据点分配一个较低的重建代价，即使这些数据点的位置距离训练数据任意远。
- 概率 PCA 可以被用来对类条件概率密度建模，因此可以应用于分类问题。
- 概率 PCA 模型可以用一种生成式的方式运行，从而可以按照某个概率分布生成样本。

最大似然 PCA

用于 PCA 的 EM 算法

11.3 核 PCA

11.4 非线性隐变量模型