模式识别笔记

陈鸿峥

2020.01*

目录

*Build 20200108

1 简介

机器学习侧重于处理的算法，而模式识别则包括了数据预处理、实际运算和数据输出的完整过程。

- 模式识别：涵盖的范围广，包括特征提取、特征选择、降维、各种分类器等。
- 机器学习：主要是讲学习，更多关于分类器如何训练模型，而不涉及特征方面的知识。

良好特征的四个特点：

- 可区别性（不同类）
- 可靠性（同类）
- 独立性（特征之间）
- 参数少（复杂性）

一个对象的所有特征参数组成特征向量。同样需要从高维测量空间（样本）中提取特征映射到低维特征空间。

模式识别分为两类

- 结构/句法模式识别
- 统计/神经网络模式识别

模式识别系统过程如下

传感器 → 分割 → 特征提取 → **分类器** → 后处理

设计循环

采集 → 选择基本特征 → 选择模型 → 训练分类 → 评价

2 贝叶斯决策论

2.1 离散变量

处于类别ω_i并具有特征值x，有后验概率[1]（给特征判类别）

$$\mathbb{P}(\omega_i \mid x) = \frac{p(x \mid \omega_i)\mathbb{P}(\omega_i)}{p(x)}$$

[1]通常用$p(\cdot)$代表概率密度函数（连续变量），用$\mathbb{P}(\cdot)$代表概率质量函数（离散变量）

即

$$posterior = \frac{likelihood \times prior}{evidence}$$

无论什么情况，当我们观察到特定的x，对于二分类问题有错误率

$$\mathbb{P}(error \mid x) = \begin{cases} \mathbb{P}(\omega_1 \mid x) & \text{决策}\omega_2 \\ \mathbb{P}(\omega_2 \mid x) & \text{决策}\omega_1 \end{cases} = \min[\mathbb{P}(\omega_1 \mid x), \mathbb{P}(\omega_2 \mid x)]$$

分段函数的分界点成为决策边界。

平均错误概率可表示为

$$\mathbb{P}(error) = \int_{-\infty}^{\infty} \mathbb{P}(error, x)\ \mathrm{d}x = \int_{-\infty}^{\infty} \mathbb{P}(error \mid x)\, p(x)\ \mathrm{d}x$$

注意$p(x)$是证据，可以看为是固定分布（常量）。

定理 1 (贝叶斯决策/最小错误率准则). 若$P(\omega_1 \mid x) > P(\omega_2 \mid x)$，则判定类别为$\omega_1$；否则判为$\omega_2$。或有等价判别$P(x \mid \omega_1)p(\omega_1) > P(x \mid \omega_2)p(\omega_2)$。依照这种准则可以获得最小错误率，即$P(error \mid x) = \min[P(\omega_1 \mid x), P(\omega_2 \mid x)]$

Neyman-Pearson准则是限定某一类别w_i的误差率不能超过一个常数，但会导致总的误差率提升。

2.2 连续变量

考虑特征向量$\mathbf{x} \in \mathbb{R}^d$（$\mathbb{R}^d$称为特征空间），令$\{\omega_1, \ldots, \omega_c\}$表示有限的$c$个类别集，$\{\alpha_1, \ldots, \alpha_a\}$表示有限的$a$种可能采取的行为集，损失函数(loss)$\lambda(\alpha_t \mid \omega_j)$描述类别状态为$\omega_j$时采取行动$\alpha_i$的风险。 $p(\mathbf{x} \mid \omega_j)$表示在真实类别为$\omega_j$的条件下$\mathbf{x}$的概率密度函数，$P(\omega_j)$表示类别处于状态$\omega_j$时的先验概率，后验概率$P(\omega_j \mid \mathbf{x})$则通过贝叶斯公式

$$P(\omega_j \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_j)P(\omega_j)}{p(\mathbf{x})}$$

计算得到，证据变为

$$p(\mathbf{x}) = \sum_{j=1}^{c} p(\mathbf{x} \mid \omega_j)P(\omega_j)$$

与行动α_i相关联的风险(risk)为

$$R(\alpha_i \mid \mathbf{x}) = \sum_{j=1}^{c} \lambda(\alpha_i \mid \omega_j)\mathbb{P}(\omega_j \mid \mathbf{x})$$

进而得到总损失

$$R = \int R(\alpha(\mathbf{x}) \mid \mathbf{x})p(\mathbf{x})\ \mathrm{d}\mathbf{x}$$

因此得到连续情形下的贝叶斯决策论：

定理 2. 为最小化R，计算条件概率

$$R(\alpha_i \mid \mathbf{x}) = \sum_{j=1}^{c} \lambda(\alpha_i \mid \omega_j)\mathbb{P}\left(\omega_j \mid \mathbf{x}\right),\ \forall i = 1, \ldots, a$$

选择α_i使得$R(\alpha_i \mid \mathbf{x})$最小，进而最小化总的风险即称为贝叶斯风险，记为$R^*$

2.2.1 二类分类

对称损失/0-1损失

$$\lambda(\alpha_i \mid \omega_j) = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases} \qquad i, j = 1, 2, \ldots, c$$

有条件风险

$$R(\alpha_1 \mid \mathbf{x}) = \lambda_{11}P(\omega_1 \mid \mathbf{x}) + \lambda_{12}P(\omega_2 \mid \mathbf{x})$$
$$R(\alpha_2 \mid \mathbf{x}) = \lambda_{21}P(\omega_1 \mid \mathbf{x}) + \lambda_{22}P(\omega_2 \mid \mathbf{x})$$

可得贝叶斯决策

$$\frac{p(\mathbf{x} \mid \omega_1)}{p(\mathbf{x} \mid \omega_2)} > \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \frac{P(\omega_2)}{P(\omega_1)}$$

2.3 正态密度

- 连续单变量正态函数

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right] \sim N(\mu, \sigma^2)$$

有期望和方差

$$\mu = \mathbb{E}\left(x\right) = \int_{-\infty}^{\infty} xp(x)\,\mathrm{d}x$$
$$\sigma^2 = \mathbb{E}\left((x-\mu)^2\right) = \int_{-\infty}^{\infty} (x-\mu)^2 p(x)\,\mathrm{d}x$$

- d维多元高斯分布

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \sim N(\boldsymbol{\mu}, \Sigma)$$

其中$\boldsymbol{\mu}$为d维均值向量，Σ维$d \times d$协方差矩阵，同时

$$\boldsymbol{\mu} = \mathbb{E}\left(\mathbf{x}\right) = \int \mathbf{x}p(\mathbf{x})\,\mathrm{d}\mathbf{x}$$
$$\Sigma = \mathbb{E}\left((\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\right) = \int (\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} p(\mathbf{x})\,\mathrm{d}\mathbf{x}$$

每一元素为

$$\mu_i = \mathbb{E}\left(x_i\right)$$
$$\sigma_{ij} = \mathbb{E}\left((x_i - \mu_i)(x_j - \mu_j)\right)$$

注意协方差矩阵Σ通常是对称且半正定的。但这里我们严格限定Σ为正定的，使得Σ的行列式是一个正数。

服从正态分布的随机变量的线性组合都是一个正态分布。特别地，若$p(\mathbf{x}) \sim N(\boldsymbol{\mu}, \Sigma)$，$A$是$d \times k$的矩阵，且$\mathbf{y} = A^{\mathrm{T}}\mathbf{x}$是一$k$维向量，则

$$p(\mathbf{y}) \sim N(A^{\mathrm{T}}\boldsymbol{\mu}, A^{\mathrm{T}}\Sigma A)$$

协方差用于计算数据沿任何方向或任意子空间的分散程度。

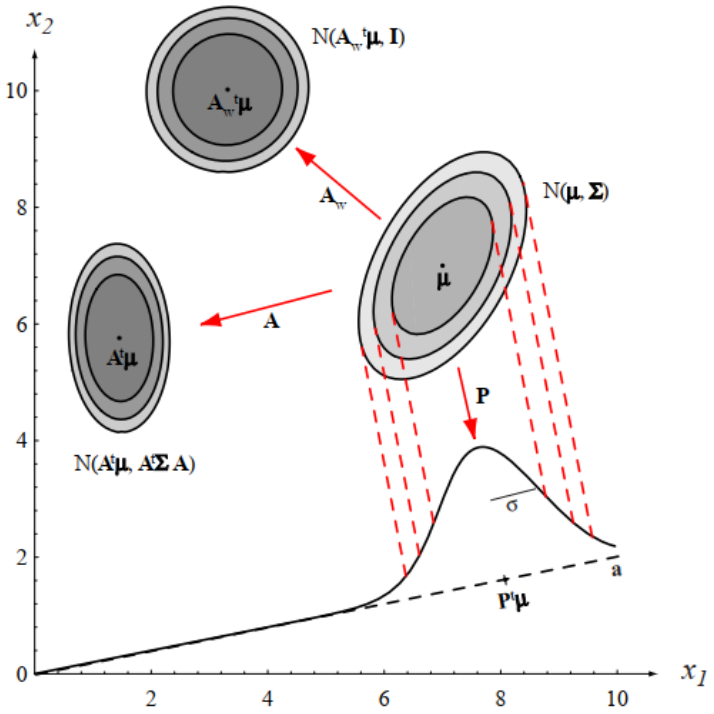


Figure 2.8: The action of a linear transformation on the feature space will convert an arbitrary normal distribution into another normal distribution. One transformation, $\mathbf{A}$, takes the source distribution into distribution $N(\mathbf{A}^t\boldsymbol{\mu}, \mathbf{A}^t\boldsymbol{\Sigma}\mathbf{A})$. Another linear transformation — a projection $\mathbf{P}$ onto line $\mathbf{a}$ — leads to $N(\mu, \sigma^2)$ measured along $\mathbf{a}$. While the transforms yield distributions in a different space, we show them superimposed on the original $x_1 - x_2$ space. A whitening transform leads to a circularly symmetric Gaussian, here shown displaced.

某个分布的协方差矩阵与单位阵I成比例，若定义矩阵Φ，其列向量是Σ的正交本征向量，Λ为与相应本征值对应的对角矩阵，变换

$$A_w = \Phi\Lambda^{-1/2}$$

将使变换后的分布的协方差矩阵成为**单位阵**[2]，此变换称为白化变换。

从$\mathbf{x}$到$\boldsymbol{\mu}$的平方马氏(Mahalanobis)距离定义为（即正态分布中的指数部分）

$$r^2 = (\mathbf{x} - \boldsymbol{\mu})^{\mathrm{T}}\Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu})$$

[2]这种表示方法是$\mathrm{Cov}\left(A_w^{\mathrm{T}}\right) = I$

可以证明与一Mahalanobis距离r对应的超椭球体体积为

$$V = V_d |\Sigma|^{1/2} r^d$$

其中V_d是一个d维单位超球体的体积

$$V_d = \begin{cases} \pi^{d/2}/(d/2)! & d\text{为偶数} \\ 2^d \pi^{(d-1)/2} \left(\frac{d-1}{2}\right)!/d! & d\text{为奇数} \end{cases}$$

因此对于一给定维数，样本的离散程度直接随$|\Sigma|^{1/2}$而变化。

最小误差概率分类可用判别函数获得

$$g_i(\mathbf{x}) = \ln p(\mathbf{x} \mid \omega_i) + \ln P(\omega_i)$$

如果$p(\mathbf{x} \mid \omega_i) \sim N(\boldsymbol{\mu}, \Sigma)$，则可以求得

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^{\mathrm{T}} \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) - \frac{d}{2} \ln 2\pi - \frac{1}{2} \ln |\Sigma_i| + \ln P(\omega_i)$$

3 极大似然与贝叶斯参数估计

3.1 极大似然估计

假设样本集$\mathcal{D}$中有n个样本$\mathbf{x}_1, \ldots, \mathbf{x}_n$，由于这些样本均独立抽取，故

$$p(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k \mid \boldsymbol{\theta})$$

这里的$\boldsymbol{\theta}$为参数向量。

定义对数似然为

$$\ell(\boldsymbol{\theta}) = \ln p(\mathcal{D} \mid \boldsymbol{\theta})$$

进而

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta})$$

求解最大似然估计值的必要条件为

$$\nabla_{\boldsymbol{\theta}} \ell = 0$$

而最大后验(maximum a posteriori, MAP)则是使$\ell(\boldsymbol{\theta})p(\boldsymbol{\theta})$取最大值的参数向量$\boldsymbol{\theta}$，注意这里最好**先乘起来再取对数**。

高斯分布若均值和协方差矩阵均未知，则最大似然估计结果为

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k$$
$$\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k - \hat{\boldsymbol{\mu}})(\mathbf{x}_k - \hat{\boldsymbol{\mu}})^{\mathrm{T}} \approx \mathbb{E}\left((\mathbf{x} - \hat{\boldsymbol{\mu}})(\mathbf{x} - \hat{\boldsymbol{\mu}})^{\mathrm{T}}\right)$$

注意上述对方差的估计是有偏的估计。

而**样本协方差矩阵**的无差估计如下

$$C = \frac{1}{n-1}\sum_{k=1}^{n}(\mathbf{x}_k - \hat{\mu})(\mathbf{x}_k - \hat{\mu})^{\mathrm{T}}$$

3.2 贝叶斯参数估计

在最大似然估计方法中，将需要估计的参数向量$\boldsymbol{\theta}$看作一个确定而未知的参数，而在贝叶斯方法中，我们将参数向量$\boldsymbol{\theta}$本身看作一个随机变量，已有的训练样本可以使我们把对于$\boldsymbol{\theta}$的初始密度估计转为后验概率密度。

将训练样本依据类别归到c个次样本集$\mathcal{D}_1, \ldots, \mathcal{D}_c$中，结合先验，贝叶斯公式可表成

$$P(\omega_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_i, \mathcal{D}_i)P(\omega_i)}{\sum_{j=1}^{c} p(\mathbf{x} \mid \omega_j, \mathcal{D}_j)P(\omega_j)}$$

已知训练样本$\mathcal{D}$，这些样本都从固定但未知的概率密度函数$p(\mathbf{x})$中独立抽取，要求根据这些样本估计$p(\mathbf{x} \mid \mathcal{D})$，即贝叶斯学习的核心问题。

得到贝叶斯估计的核心公式

$$p(\mathbf{x} \mid \mathcal{D}) = \int p(\mathbf{x} \mid \boldsymbol{\theta})p(\boldsymbol{\theta} \mid \mathcal{D})\,\mathrm{d}\boldsymbol{\theta}$$

根据贝叶斯公式

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathcal{D} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}}$$

再有样本独立性假设

$$p(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k \mid \boldsymbol{\theta})$$

当没有观测样本时，$p(\boldsymbol{\theta} \mid \mathcal{D}^0) = p(\boldsymbol{\theta})$，反复应用上述公式有概率密度函数$p(\boldsymbol{\theta}), p(\boldsymbol{\theta} \mid \mathbf{x}_1), p(\boldsymbol{\theta} \mid \mathbf{x}_1, \mathbf{x}_2)$等，实际上即增量学习(incremental learning)。

3.3 Fisher线性判别

PCA方法寻找的是用来有效表示的主轴方向，而判别分析方法(discriminant analysis)寻找的是用来有效**分类**的方向。

考虑将d维空间中的数据点投影到一条直线上，以最大限度区分各类数据点的投影方向。假设有一组n个d维的样本$\mathbf{x}_1, \ldots, \mathbf{x}_n$分属两个不同类别，其中$n_1$个样本的子集$\mathcal{D}_1$属于$\omega_1$，$n_2$个样本的子集$\mathcal{D}_2$属于$\omega_2$。对$\mathbf{x}$中各个成分做线性组合，得到点积[3]$y = \mathbf{w}^{\mathrm{T}}\mathbf{x}$，进而全部$n$个样本产生$n$个结果$y_1, \ldots, y_n$相应属于$\mathcal{Y}_1$和$\mathcal{Y}_2$。

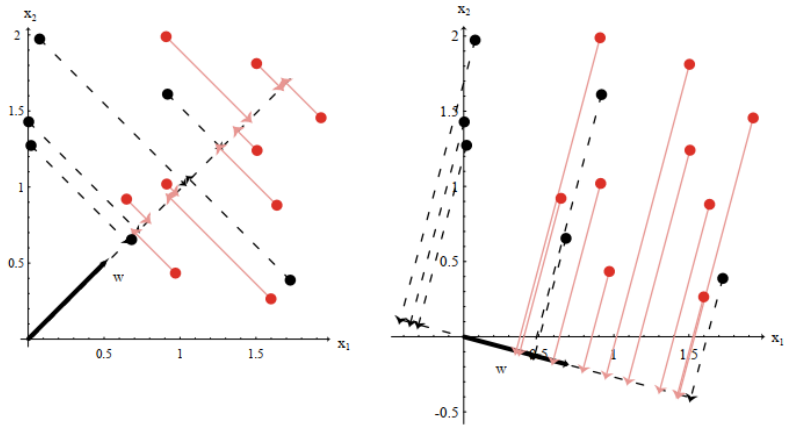


Figure 4.27: Projection of samples onto two different lines. The figure on the right shows greater separation between the red and black projected points.

如何确定最佳的直线方向$\mathbf{w}$得到最佳的分类效果，一个衡量标准即样本均值的差。设原样本第i个类别的均值为$\mathbf{m}_i$，则投影后的样本均值为$\tilde{\mathbf{m}}_i = \mathbf{w}^{\mathrm{T}}\mathbf{m}_i$。样本均值之差为

$$|\tilde{m}_1 - \tilde{m}_2| = |\mathbf{w}^{\mathrm{T}}(\mathbf{m}_1 - \mathbf{m}_2)|$$

可以通过$\mathbf{w}$幅值方法来得到任意大小的均值之差，但这样子没有意义。因此定义类别ω_i的类内散布(scatter)/方差如下

$$\tilde{s}_i^2 = \sum_{y \in \mathcal{Y}_i} (y - \tilde{m})^2$$

这样$1/n(\tilde{s}_1^2 + \tilde{s}_2^2)$即为全部数据总体方差的估计，$\tilde{s}_1^2 + \tilde{s}_2^2$称为投影样本的总类内散布。

故Fisher线性可分性准则要求在投影$y = \mathbf{w}^{\mathrm{T}}\mathbf{x}$下

$$\max_{\mathbf{w}} J(\mathbf{w}) := \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

即均值差尽可能大，同时类内方差尽可能小。

定义类内散布矩阵S_i和总类内散布矩阵S_W如下

$$S_i = \sum_{\mathbf{x} \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^{\mathrm{T}}$$

$$S_W = S_1 + S_2$$

[3]回忆高中知识，点积相当于做投影，将$\mathbf{x}$往直线$\mathbf{w}$上投影

可求得

$$\tilde{s}_i^2 = \mathbf{w}^{\mathrm{T}} S_i \mathbf{w}$$

$$\tilde{s}_1^2 + \tilde{s}_2^2 = \mathbf{w}^{\mathrm{T}} S_W \mathbf{w}$$

而投影样本均值之差

$$S_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^{\mathrm{T}}$$

$$(\tilde{\mathbf{m}}_1 - \tilde{\mathbf{m}}_2)^2 = \mathbf{w}^{\mathrm{T}} S_B \mathbf{w}$$

其中S_B称为总类间散布矩阵。 S_W和S_B都是对称且半正定的。

进而原来的准则函数可写成

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}} S_B \mathbf{w}}{\mathbf{w}^{\mathrm{T}} S_W \mathbf{w}}$$

称为广义瑞利商，可证令其最大化

$$S_B \mathbf{w} = \lambda S_W \mathbf{w}$$

一般情况下

$$\arg\max_{\mathbf{w}} J(\mathbf{w}) = S_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$$

4 非参数技术

4.1 概率密度的估计

向量$\mathbf{x}$落在区域$\mathcal{R}$的概率为

$$P = \int_{\mathcal{R}} p(\mathbf{x}') \, \mathrm{d}\mathbf{x}'$$

即P是概率密度函数$p(\mathbf{x})$平滑后的版本。若假设$p(\mathbf{x})$是连续的，且区域$\mathcal{R}$足够小，以致于在这个区间中p几乎没有变化，则

$$\int_{\mathcal{R}} p(\mathbf{x}') \, \mathrm{d}\mathbf{x}' \approx p(\mathbf{x}) V$$

其中$\mathbf{x}$为一个点，而V为区域$\mathcal{R}$所包含的体积。可以用下述公式作为一个估计

$$p(\mathbf{x}) \approx \frac{k/n}{V}$$

即从n个服从$p(\mathbf{x})$的独立同分布样本落在$\mathcal{R}$中的有k个。

为了估计$\mathbf{x}$的概率密度函数，构造一系列包含$\mathbf{x}$的区域$\mathcal{R}_1, \mathcal{R}_2, \ldots$，第一个区域用1个样本，第二个区域用2个，以此类推。 V_n为区域$\mathcal{R}_n$的体积，k_n为落在区间$\mathcal{R}_n$中的样本个数，而$p_n(\mathbf{x})$表示对$p(\mathbf{x})$的第n次估计：

$$p_n(\mathbf{x}) = \frac{k_n/n}{V_n}$$

若要求$p_n(\mathbf{x})$能够收敛到$p(\mathbf{x})$，则下面3个条件必须满足：

- $\lim_{n\to\infty} V_n = 0$

- $\lim_{n\to\infty} k_n = \infty$
- $\lim_{n\to\infty} k_n/n = 0$

第一个条件保证区域均匀收缩和$p(\cdot)$在点$\mathbf{x}$处连续的情况下，区间平滑了的P/V能够收敛到$p(\mathbf{x})$。第二个条件只有在$p(\mathbf{x}) \neq 0$才有意义，保证频率之比能够收敛到概率P。最后一个条件说明虽然最后落在小区域$\mathcal{R}_n$中的样本数目非常大，但是这么多样本在全体样本中所占的比例非常小。这里通常考虑均方意义下的收敛[4]。

4.2 Parzen窗方法

假设区间$\mathcal{R}_n$是d维超立方体，h_n为一条边长度，体积为

$$V_n = h_n^d$$

定义窗函数

$$\varphi(\mathbf{u}) = \begin{cases} 1 & |\mathbf{u}_j| \leq 1/2,\ j = 1, \ldots, d \\ 0 & \text{其他} \end{cases}$$

这样$\varphi(\mathbf{u})$就表示一个中心在原点的单位超立方体。若$\mathbf{x}_i$落在中心点为$\mathbf{x}$的立方体V_n中，那么

$$\varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right) = 1$$

否则为0，进而可解析表达超立方体样本个数

$$k_n = \sum_{i=1}^{n} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right)$$

代入估计式有

$$p_n(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{V_n} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right)$$

均值的收敛性

$$\begin{aligned} \bar{p}_n(\mathbf{x}) &= \mathbb{E}\left(p_n(\mathbf{x})\right) \\ &= \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}\left(\frac{1}{V_n} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right)\right) \\ &= \int \frac{1}{V_n} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right) p(\mathbf{v})\, \mathrm{d}\mathbf{v} \\ &= \int \delta_n(\mathbf{x} - \mathbf{v}) p(\mathbf{v})\, \mathrm{d}\mathbf{v} \end{aligned}$$

4.3 k_n近邻估计

最佳的窗函数的选择是个问题，因此一种可行的方案是让体积成为训练样本的函数，而不是硬性规定窗函数为样本个数的某个函数。

[4] https://en.wikipedia.org/wiki/Convergence_of_random_variables

比如说可以取$k_n = \sqrt{n}$，有下列迭代过程。

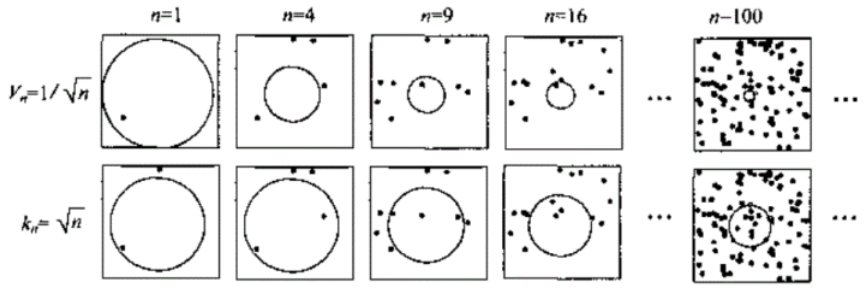


图 4-2　估计某一点处的概率密度函数有两种最基本的方法。这里，我们假设这个点位于图中所示的正方形的中心。第一行表示的方法是从一个以目标样本点为中心的较大的区域开始，根据某个函数，例如 $V_n=1/\sqrt{n}$，逐渐的缩小区域面积。第二种方法如第二行所示。这一方法缩小区域面积的方式是依赖于样本点的。例如，令区域必须包括 $k_n=\sqrt{n}$ 个样本点。这两种情况中的序列都是随机变量，它们一般会收敛，这样就能估计出测试样本点处的真正的概率密度函数

4.4　最近邻规则

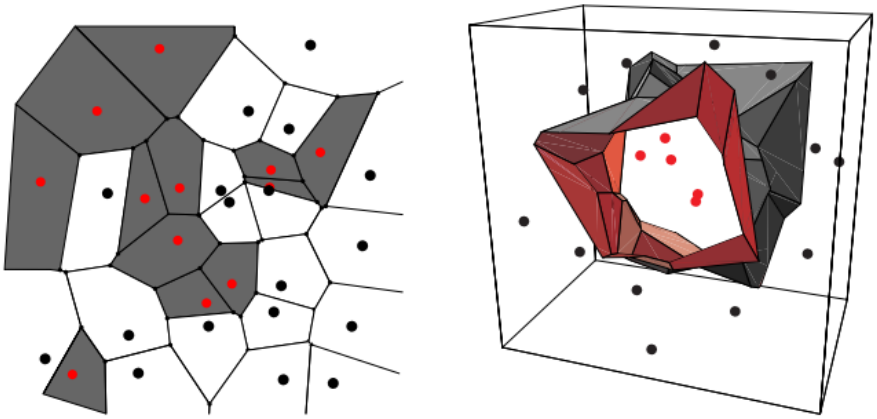


Figure 4.13: In two dimensions, the nearest-neighbor algorithm leads to a partitioning of the input space into Voronoi cells, each labelled by the category of the training point it contains. In three dimensions, the cells are three-dimensional, and the decision boundary resembles the surface of a crystal.

定义$\omega_m(\mathbf{x})$为

$$P(\omega_m \mid \mathbf{x}) = \max_i P(\omega_i \mid \mathbf{x})$$

令$P^*(e \mid \mathbf{x})$表示$P(e \mid \mathbf{x})$的最小可能值，P^*为$P(e)$的最小可能值，则根据贝叶斯风险

$$P^*(e \mid \mathbf{x}) = 1 - P(\omega_m \mid \mathbf{x})$$

有总的误差率

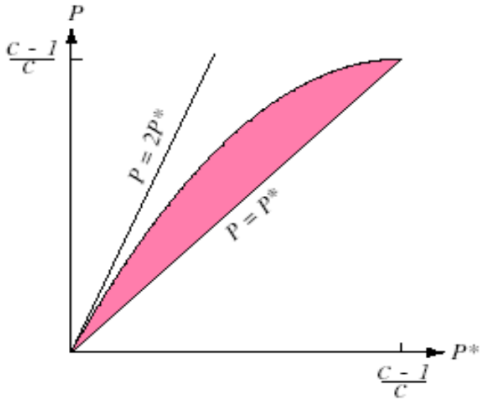
$$P^* = \int P^*(e \mid \mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}$$

记n个样本的平均误差率为$P_n(e)$，且

$$\begin{aligned} P &= \lim_{n\to\infty} P_n(e) \\ &= \lim_{n\to\infty} \int P_n(e \mid \mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x} \\ &= \int \left[1 - \sum_{i=1}^{c} P^2(\omega_i \mid \mathbf{x})\right] p(\mathbf{x})\,\mathrm{d}\mathbf{x} \end{aligned}$$

可以证明

$$P^* \le P \le P^*\left(2 - \frac{c}{c-1}P^*\right)$$



进而推广有k近邻规则(KNN)。

5 线性判别函数

5.1 线性判别函数和判定面

判别(discriminant)函数是指由$\mathbf{x}$的各个分量线性组合而成的函数

$$g(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0$$

这里$\mathbf{w}$是权向量，w_0称为阈权值(threshold)或偏置。

对于二类线性分类器来说，$g(\mathbf{x}) > 0$则判为ω_1，否则判为ω_2。方程$g(\mathbf{x}) = 0$定义了一个判定面，将归类于ω_1和ω_2的点分开来。当$g(\mathbf{x})$是线性的，这个平面称为超平面。

判别函数是特征空间某点$\mathbf{x}$到超平面距离的代数度量（注意到垂直平行特性）

$$\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$$

其中$\mathbf{x}_p$是$\mathbf{x}$在超平面H上的投影向量，r是相应的算术距离，为正则$\mathbf{x}$在H正侧，否则在H负侧。由于$g(\mathbf{x}_p) = 0$，有

$$g(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 = r\,\|\mathbf{w}\|$$

或

$$r = \frac{g(\mathbf{x})}{\|\mathbf{w}\|}$$

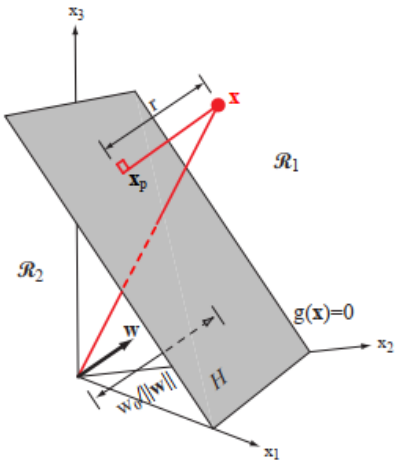


Figure 5.2: The linear decision boundary H, where $g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0 = 0$, separates the feature space into two half-spaces $\mathcal{R}_1$ (where $g(\mathbf{x}) > 0$) and $\mathcal{R}_2$ (where $g(\mathbf{x}) < 0$).

5.2 广义线性判别函数

线性判别函数可写成

$$g(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i$$

系数w_i是权向量$\mathbf{w}$的分量，通过加入另外的项（$\mathbf{w}$的各对分量之间的乘积），得到二次判别函数

$$g(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d}\sum_{j=1}^{d} w_{ij} x_i x_j$$

因$x_i x_j = x_j x_i$，不失一般性假设$w_{ij} = w_{ji}$。

继续加入更高次项（如$w_{ijk}x_i x_j x_k$）可得到多项式判别函数，可看作对某一判别函数$g(x)$做级数展开，然后取截尾逼近，意味着某一广义线性判别函数

$$g(\mathbf{x}) = \sum_{i=1}^{d} a_i y_i(\mathbf{x})$$

或

$$g(\mathbf{x}) = \mathbf{a}^{\mathrm{T}}\mathbf{y}$$

特别地，线性判别函数可写成

$$g(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i = \sum_{i=0}^{d} w_i x_i$$

令$\mathbf{y} = \begin{bmatrix} 1 & \mathbf{x} \end{bmatrix}^{\mathrm{T}}$为**增广特征向量**，$\mathbf{a} = \begin{bmatrix} w_0 & \mathbf{w} \end{bmatrix}$为增广权向量，注意这里是将偏置放在了第0位。

对于一个样本$\mathbf{y}_i$，若$\mathbf{a}^{\mathrm{T}}\mathbf{y}_i > 0$就标记为$\omega_1$，若$\mathbf{a}^{\mathrm{T}}\mathbf{y}_i < 0$就标记为$\omega_2$。这样可以通过一种规范化(normalization)方法来简化两类样本训练过程，即对于属于ω_2的样本，用负号表示而不是标记ω_2（将属于ω_2的样本用$\begin{bmatrix} -1 & -y_i \end{bmatrix}^{\mathrm{T}}$表示）。这样就可以直接寻找一个对所有样本都有$\mathbf{a}^{\mathrm{T}}\mathbf{y}_i > 0$的权向量$\mathbf{a}$，这样的向量称为分离(separating)向量或解向量。

求解权向量的过程相当于确定权空间中一点，每个样本都对应解向量的可能位置给出限制。 $\mathbf{a}\mathbf{y}_i^{\mathrm{T}}$确定了一个穿过权空间原点的超平面，$\mathbf{y}_i$为其法向。解向量若存在，则一定在每个超平面的正侧。由于是不等式约束，故解不唯一，要加入其他约束条件。一种方法是找到一个单位长度的权向量，使得从样本到分类平面最小距离达到最大；另一种方法则在所有i中寻找满足$\mathbf{a}^{\mathrm{T}}\mathbf{y}_i \geq b$的具有最小长度的权向量，这里的$b$是被称为边沿裕量(margin)/间隔的正常数。

求解$\mathbf{a}^{\mathrm{T}}\mathbf{y}_i > 0$的解所采用的方法是：定义一个准则函数$J(\mathbf{a})$，当$\mathbf{a}$是解向量时，$J(\mathbf{a})$最小，因此可将其简化为一个标量函数极小化问题，通常用梯度下降法解决。

$$\mathbf{a}(k+1) = \mathbf{a}(k) - \eta(k)\nabla J(\mathbf{a}(k))$$

其中η为正的比例因子，即学习率。

假设准则函数可以通过二阶展开近似

$$J(\mathbf{a}) \approx J(\mathbf{a}(k)) + \nabla J^{\mathrm{T}}(\mathbf{a} - \mathbf{a}(k)) + \frac{1}{2}(\mathbf{a} - \mathbf{a}(k))^{\mathrm{T}} H (\mathbf{a} - \mathbf{a}(k))$$

即当选择

$$\eta(k) = \frac{\|\nabla J\|^2}{\nabla J^T H \nabla J}$$

时，可使$J(\mathbf{a}(k+1))$最小化。

牛顿法

$$\mathbf{a}(k+1) = \mathbf{a}(k) - H^{-1}\nabla J$$

5.3 感知器

考虑构造线性不等式$\mathbf{a}^{\mathrm{T}}\mathbf{y}_i > 0$的准则函数，用错分的样本作为准则函数不好因为分段常数不利于梯度搜索。更好的选择是感知器(perceptron)准则函数

$$J_p(\mathbf{a}) = \sum_{\mathbf{y} \in \mathcal{Y}} (-\mathbf{a}^{\mathrm{T}}\mathbf{y})$$

这里$\mathcal{Y}(\mathbf{a})$是被$\mathbf{a}$错分的样本集。由几何上知，$J_p(\mathbf{a})$是与错分样本到判决边界距离之和成正比的。对上式

求导有

$$\nabla J_p = \sum_{\mathbf{y}\in\mathcal{Y}}(-\mathbf{y})$$

得到梯度下降迭代公式

$$\mathbf{a}(k+1) = \mathbf{a}(k) + \eta(k)\sum_{\mathbf{y}\in\mathcal{Y}_k}\mathbf{y}$$

5.4 最小平方误差

设Y为$n\times\hat{d}$的矩阵，n为样本数，$\hat{d}=d+1$为维度，第i行是向量$\mathbf{y}_i^{\mathrm{T}}$，令$\mathbf{b}=\begin{bmatrix}b_1 & \cdots & b_n\end{bmatrix}^{\mathrm{T}}$，目标是找到权重$\mathbf{a}$满足

$$Y\mathbf{a}=\mathbf{b}$$

定义误差向量

$$\mathbf{e}=Y\mathbf{a}-\mathbf{b}$$

使误差向量长度平方最小化，即最小化平方误差和(MSE)准则函数

$$J_s(\mathbf{a}) = \|Y\mathbf{a}-\mathbf{b}\|^2 = \sum_{i=1}^{n}(\mathbf{a}^{\mathrm{T}}\mathbf{y}_i - b_i)^2$$

计算梯度有

$$\nabla J_s = 2Y^{\mathrm{T}}(Y\mathbf{a}-\mathbf{b})$$

令其为0有

$$Y^{\mathrm{T}}Y\mathbf{a} = Y^{\mathrm{T}}\mathbf{b}$$

当方阵$Y^{\mathrm{T}}Y$非奇异时，有唯一解

$$\mathbf{a} = (Y^{\mathrm{T}}Y)^{-1}Y^{\mathrm{T}}\mathbf{b} = Y^{\dagger}\mathbf{b}$$

这里的$\hat{d}\times n$矩阵

$$Y^{\dagger}(Y^{\mathrm{T}}Y)^{-1}Y^{\mathrm{T}}$$

称为Y的伪逆矩阵。注意到若Y为方阵且非奇异，则这个伪逆矩阵即为Y的逆矩阵。还应注意到$Y^{\dagger}Y=I$，但通常$YY^{\dagger}\neq I$。

MSE与Fisher线性判别函数的解是一样的。

5.5 支持向量机

支持向量机(support vector machine, SVM)通过一个足够高维的非线性映射$\varphi(\cdot)$，将两类数据用超平面进行分割。假设每个模式$\mathbf{x}_k$变换到$\mathbf{y}_k=\varphi(\mathbf{x}_k)$，则问题在于选择$\varphi(\cdot)$。对$n$个模式中的每一个$k=1,\ldots,n$，根据模式属于$\omega_1$或$\omega_2$，分别令$z_k=\pm1$，增广空间$\mathbf{y}$上的判别函数是

$$g(\mathbf{y}) = \mathbf{a}^{\mathrm{T}}\mathbf{y}$$

这里权向量和变换后的模式向量都是增广的（取$a_0 = w_0, y_0 = 1$），则这样的分割超平面保证

$$z_k g(\mathbf{y}_k) \geq 1,\ k = 1, \ldots, n$$

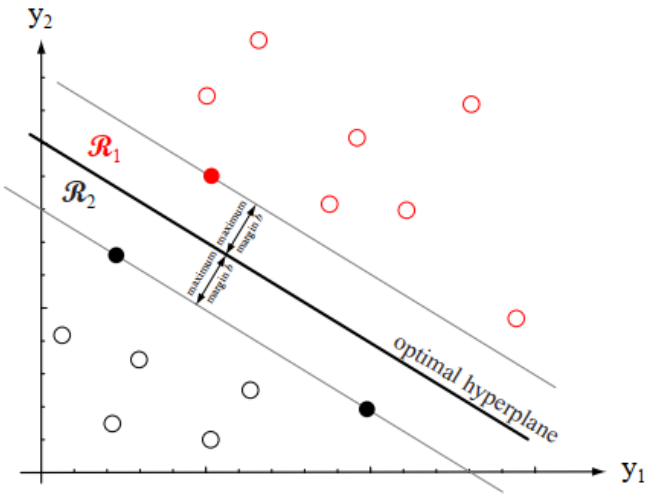


Figure 5.19: Training a Support Vector Machine consists of finding the optimal hyperplane, i.e., the one with the maximum distance from the nearest training patterns. The support vectors are those (nearest) patterns, a distance b from the hyperplane. The three support vectors are shown in solid dots.

训练一个支持向量机的目标是找到一个具有最大间隔(largest margin)的分割平面，若间隔越大则得到的分类器也越好。从超平面到变换后的模式$\mathbf{y}$的距离是$|g(\mathbf{y})|/\|\mathbf{a}\|$（即做投影），若正的间隔$b$存在，则推出

$$\frac{z_k g(\mathbf{y}_k)}{\|\mathbf{a}\|} \geq b,\ k = 1, \ldots, n$$

目标即找一个使得b最大化的权向量$\mathbf{a}$。由于解向量可以任意伸缩且保持超平面不变，故有限制条件$b\|a\| = 1$，即其确定的是$\|\mathbf{a}\|$的极小值。

支持向量是使上式等号成立的模式向量，即支持向量是最靠近超平面的，也是最难分类的样本/对求解分类任务最富有信息的模式。

目标为极小化$\|\mathbf{a}\|$，构造拉格朗日函数

$$L(\mathbf{a}, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{a}\|^2 - \sum_{k=1}^{n} \alpha_k [z_k \mathbf{a}^{\mathrm{T}} \mathbf{y}_k - 1]$$

可用KKT条件改写为

$$L(\boldsymbol{\alpha}) = \sum_{k=1}^{n} \alpha_i - \frac{1}{2} \sum_{k,j}^{n} \alpha_k \alpha_j z_k z_j \mathbf{y}^{\mathrm{T}} \mathbf{y}_k$$

$$s.t. \sum_{k=1}^{n} z_k \alpha_k = 0,\ \alpha_k \geq 0,\ k = 1, \ldots, n$$

进行求解（先用约束消元，逐一令偏导为0求解）。

6　多层神经网络

三层神经网络：输入层、隐含层、输出层，也称多层感知器(multilayer perceptron, MLP)

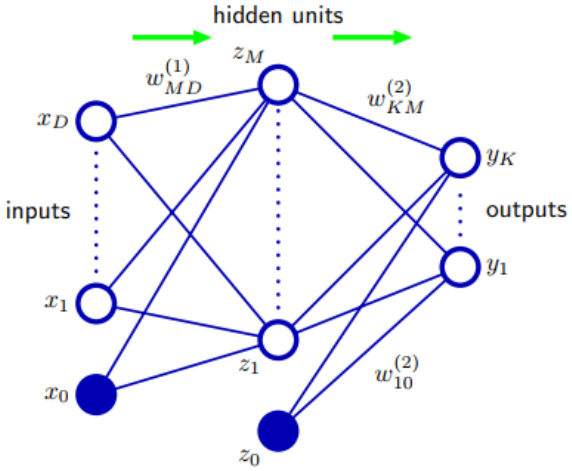


Figure 5.1 Network diagram for the two-layer neural network corresponding to (5.7). The input, hidden, and output variables are represented by nodes, and the weight parameters are represented by links between the nodes, in which the bias parameters are denoted by links coming from additional input and hidden variables x_0 and z_0. Arrows denote the direction of information flow through the network during forward propagation.

前馈运算如下，判别函数$y_k(\mathbf{x}, \mathbf{w})$为每个输出单元产生的信号

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right)$$

任何从输入到输出的连续映射函数都可以用一个三层非线性网络实现，只要有足够的隐单元M、适当的非线性函数和权值。

最小化误差函数

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} \|\mathbf{y}\left(\mathbf{x}_n, \mathbf{w}\right) - \mathbf{t}_n\|^2$$

隐含层到输出层的权重更新公式为

$$\Delta w_{kj} = \eta \delta_k y_j = \eta(t_k - z_k) f'(net_k) y_j$$

输入层到隐含层的权重更新公式为

$$\Delta w_{ji} = \eta x_i \delta_j = \eta \left[\sum_{k=1}^{c} w_{kj}\delta_k\right] f'(net_j) x_i$$

sigmoid函数

$$f(\text{net}) = a\tanh(b \cdot \text{net}) = a\left[\frac{1 - \mathrm{e}^{-b\cdot\text{net}}}{1 + \mathrm{e}^{-b\cdot\text{net}}}\right] = \frac{2a}{1 + \mathrm{e}^{-b\cdot\text{net}}} - a$$

7 随机方法

假设给定多个变量s_i，其中每个变量数值都取两个离散值之一，记为± 1。优化问题为确定N个s_i的合适取值，使下述代价函数或能量函数最小

$$E = -\frac{1}{2}\sum_{i,j=1}^{N} w_{ij} s_i s_j$$

其中w_{ij}是对称的，取值可正可负。

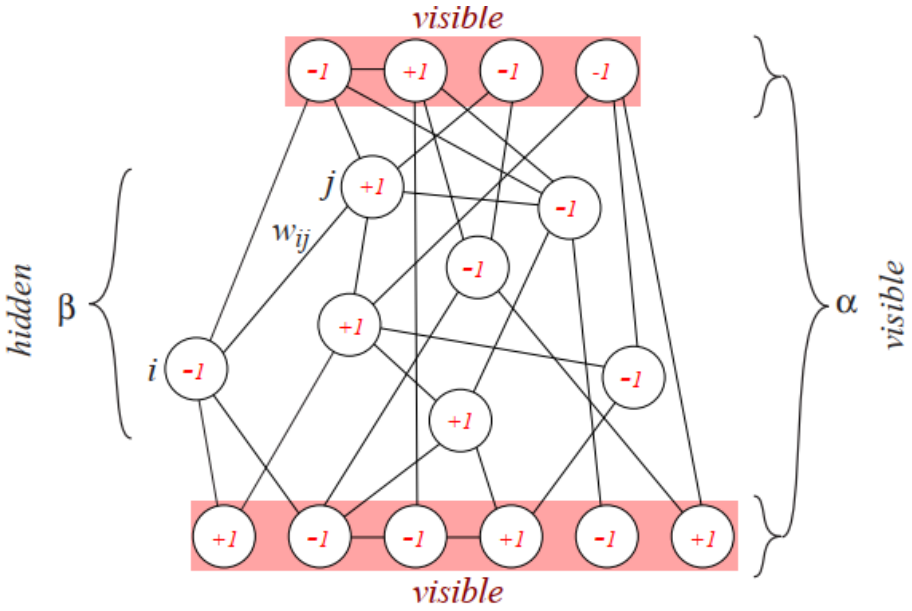


Figure 7.1: The class of optimization problems of Eq. 1 can be viewed in terms of a network of nodes or units, each of which can be in the $s_i = +1$ or $s_i = -1$ state. Every pair of nodes i and j is connected by bi-directional weights w_{ij}; if a weight between two nodes is zero then no connection is drawn. (Because the networks we shall discuss can have an arbitrary interconnection, there is no notion of layers as in multilayer neural networks.) The optimization problem is to find a configuration (i.e., assignment of all s_i) that minimizes the energy described by Eq. 1. The state of the full network is indexed by an integer γ, and since here there are 17 binary nodes, γ is bounded $0 \le \gamma < 2^{17}$. The state of the visible nodes and hidden nodes are indexed by α and β, respectively and in this case are bounded $0 \le \alpha \le 2^{10}$ and $0 \le \beta < 2^{7}$.

7.1 模拟退火

一个系统具有能量E_γ通过下式给出

$$P(\gamma) = \frac{\mathrm{e}^{-E_\gamma/T}}{Z(T)}$$

其中分子为Boltzmann因子，而Z是一个归一化常量/分配(partition)函数

$$Z(T) = \sum_{\gamma'} \mathrm{e}^{-E_{\gamma'}/T}$$

为Boltzmann因子对所有构型的求和。

Algorithm 1 模拟退火(Simulated Annealing)

将网络随机初始化，并设一个高的初始温度$T(1)$

随机选择节点i，设其状态为$s_i = +1$，计算该构型下系统总能量E_a

改变其道候选状态不$s_i = -1$，系统总能量为E_b

if $E_b < E_a$ **then**

接受此次状态改变

else

以$\mathrm{e}^{-\Delta E_{ab}/T}$概率接受改变，其中$\Delta E_{ab} = E_b - E_a$

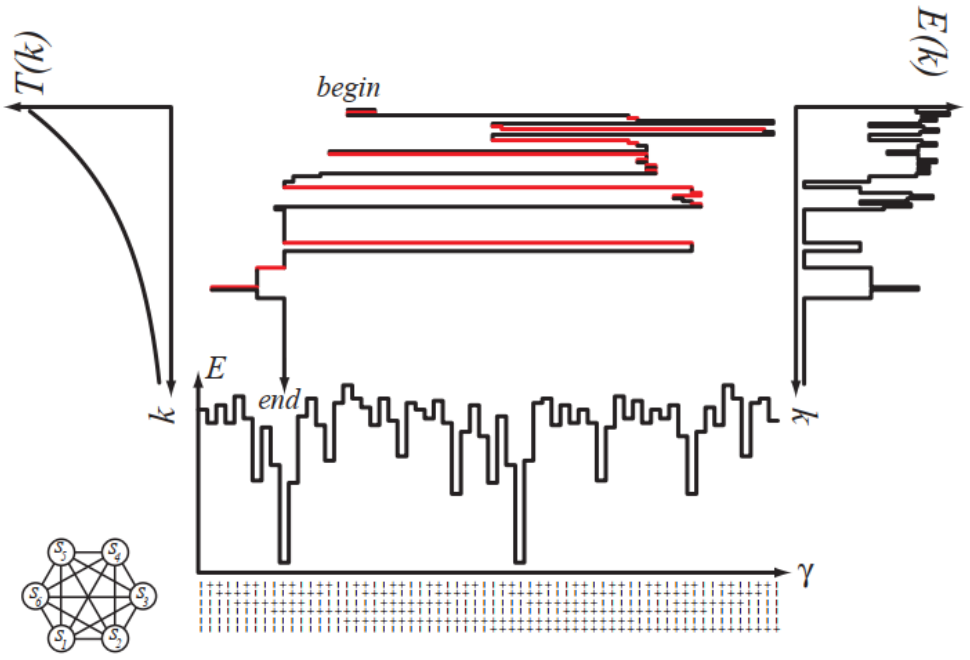


Figure 7.3: Stochastic simulated annealing (Algorithm 1) uses randomness, governed by a control parameter or "temperature" $T(k)$ to search through a discrete space for a minimum of an energy function. In this example there are $N = 6$ variables; the $2^6 = 64$ configurations are shown at the bottom along as a column of + and -. The plot of the associated energy of each configuration given by Eq. 1 for randomly chosen weights. Every transition corresponds to the change of just a single s_i. (The configurations have been arranged so that adjacent ones differ by the state of just a single node; nevertheless most transitions corresponding to a single node appear far apart in this ordering.) Because the system energy is invariant with respect to a global interchange $s_i \leftrightarrow -s_i$, there are two "global" minima. The graph at the upper left shows the annealing schedule — the decreasing temperature versus iteration number k. The middle portion shows the configuration versus iteration number generated by Algorithm 1. The trajectory through the configuration space is colored red for transitions that increase the energy; late in the annealing such energetically unfavorable (red) transitions are rarer. The graph at the right shows the full energy $E(k)$, which decreases to the global minimum.

7.2 Boltzmann学习

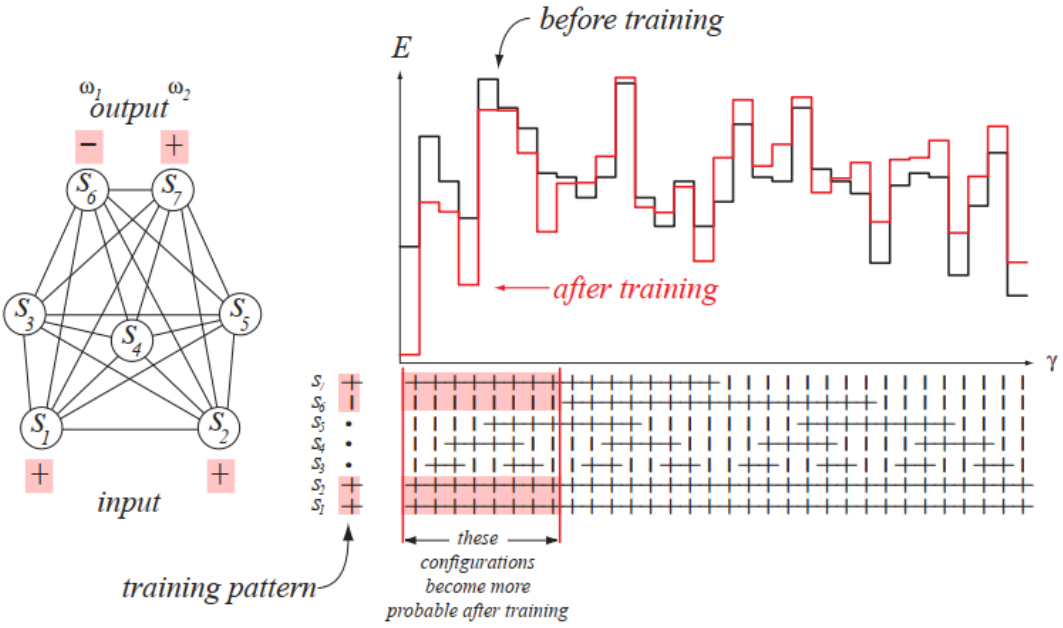


Figure 7.8: The fully connected seven-unit network at the left is being trained via the Boltzmann learning algorithm with the input pattern $s_1 = +1$, $s_2 = +1$, and the output values $s_6 = -1$ and $s_7 = +1$, representing categories ω_1 and ω_2, respectively. All $2^5 = 32$ configurations with $s_1 = +1$, $s_2 = +1$ are shown at the right, along with their energy (Eq. 1). The black curve shows the energy before training; the red curve shows the energy after training. Note particularly that after training all configurations that represent the full training pattern have been lowered in energy, i.e., have become more probable. Consequently, patterns that do not represent the training pattern become *less* probable after training. Thus, after training, if the input pattern $s_1 = +1$, $s_2 = +1$ is presented and the remaining network annealed, there is an increased chance of yielding $s_6 = -1$, $s_7 = +1$, as desired.