自动化学院

模式识别

15 概率密度函数的非参数估计

第十五章　概率密度函数的非参数估计

15. 1 非参数估计的基本原理

问题：已知样本集 $\mathcal{X}=\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_N\}$，其中样本均从服从 $p(\boldsymbol{x})$ 的总体中独立抽取 (iid,independently identically distribution)，求估计 $\hat{p}(\boldsymbol{x})$，近似 $p(\boldsymbol{x})$。

考虑随机向量 $\boldsymbol{x}$ 落入区域 $\Re$ 的概率 $P=\int_{\Re} p(x)dx$

$\mathcal{X}$中有 k个样本落入区域 $\Re$的概率 $P_k=C_N^k P^k(1-P)^{N-k}$

k的期望值 $E[k]=NP$

k的众数（概率最大的取值）为 $m=[(N+1)P]$

P 的估计 $\hat{P}=\dfrac{k}{N}$ （ k：实际落到 $\Re$中的样本数） $E(k^2)-(NP)^2$ $\therefore E(k^2)=N^2P^2+NP(1-P)$

根据二项分布的性质,变量k/N的均值和方差分别可以计算出来

$$E[k]=NP \quad and \quad \mathrm{var}[k]=NP(1-P)$$

$\mathrm{Var}\left(\frac{k}{N}\right)=E\left[\left(\frac{k}{N}\right)^2\right]-P^2$

$$E\left[\frac{k}{N}\right]=P \quad and \quad \mathrm{var}\left[\frac{k}{N}\right]=E\left[\left(\frac{k}{N}-P\right)^2\right]=\frac{P(1-P)}{N}$$

$=\cancel{P^2}+\frac{NP(1-P)}{N^2}-\cancel{P^2}$

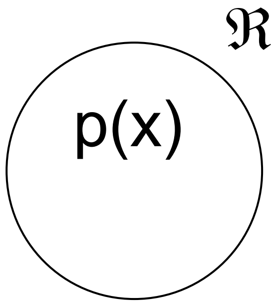
$=\frac{P(1-P)}{N}$

15. 1 非参数估计的基本原理

设 $p(x)$ 在 $\Re$ 内连续，当 $\Re$ 逐渐减小的时候，小到使 $p(x)$ 在其上几乎没有变化时，则

因此 $$P=\int_R p(x)dx=p(x)V \quad x\in\Re$$

其中, $$\hat{p}(x)=\frac{k}{NV} \Leftarrow \boxed{\hat{P}=\frac{k}{N}}$$



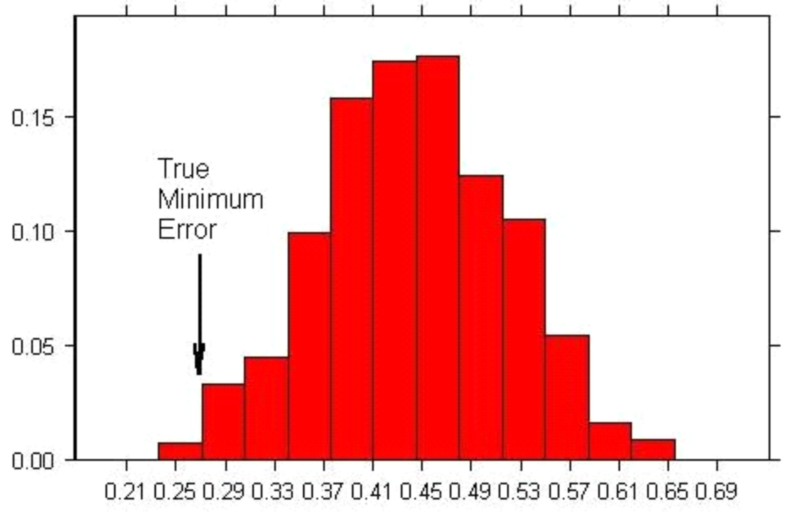
N ：样本总数，

V ：包含 x 的一个小区域 $\Re$ 的体积，有 $V=\int_R dx$

k ：落在此区域中的样本数

$\hat{p}(x)$ 为对 $p(x)$ 在小区域内的平均值的估计。

15.2 直方图方法



直方图方法

非参数概率密度估计的最简单方法

（1）把 x 的每个分量分成 k 个等间隔小窗，（若 $x \in E^d$，则形成 k^d 个小舱）

（2）统计落入各个小舱内的样本数 q_i

（3）相应小舱的概率密度为 $q_i/(NV)$（ N：样本总数， V：小舱体积）

V的选择：过大，估计粗糙；过小，可能某些区域中无样本。

理论结果：

设有一系列包含样本的区域 $\Re_1, \Re_2, \cdots, \Re_n, \cdots$ ，对 $\Re_1$ 采用1个样本进行估计，对 $\Re_2$ 用2个，…。设 $\Re_n$ 包含 k_n 个样本，V_n 为 $\Re_n$ 的体积，$\hat{p}_n(x) = \dfrac{k_n}{nV_n}$ 为 $p(x)$ 的第 n 次估计，有下面的结论：

如果：（1）$\lim\limits_{n\to\infty} V_n = 0$；（2）$\lim\limits_{n\to\infty} k_n = \infty$；（3）$\lim\limits_{n\to\infty} \dfrac{k_n}{n} = 0$

样本个小窗体积应尽可能的小

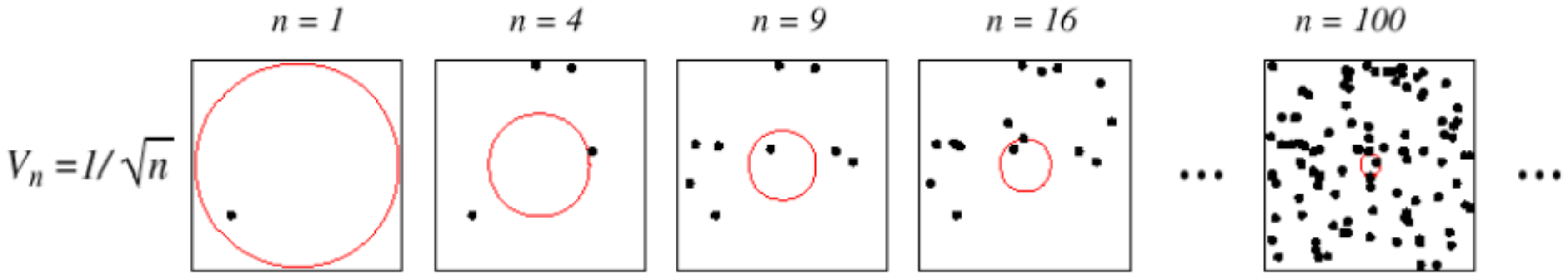
则 $\hat{p}_n(x)$ 收敛于 $p(x)$。随n个区间内样本增多 保证小舱内有充分多的样本

尽管样本增多，每个小舱里样本数又是总样本数中很小的一部分

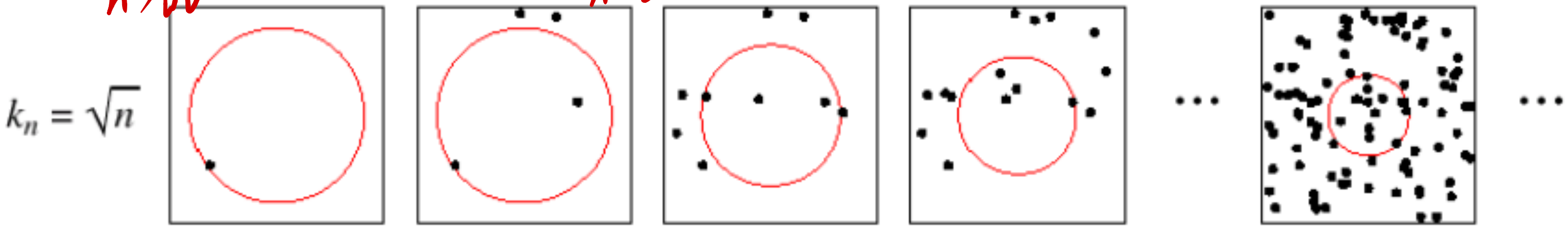
两种选择策略：

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

1\. 选择V_n，（比如 $V_n = \dfrac{1}{\sqrt{n}}$ ），同时对k_n 和$\dfrac{k_n}{n}$ 加限制以保证收敛——Parzen窗法 找一个中心.



2\. 选择 k_n，（比如 $k_n = \sqrt{n}$ ），V_n 为正好包含 x 的 k_n个近邻——k_N 近邻估计 $\lim_{n\to\infty} k_n = \infty$ $\lim_{n\to\infty} \dfrac{k_n}{n} \to 0.$



15.3 Kn近邻估计方法

回顾：

最简单的分段线性分类器：

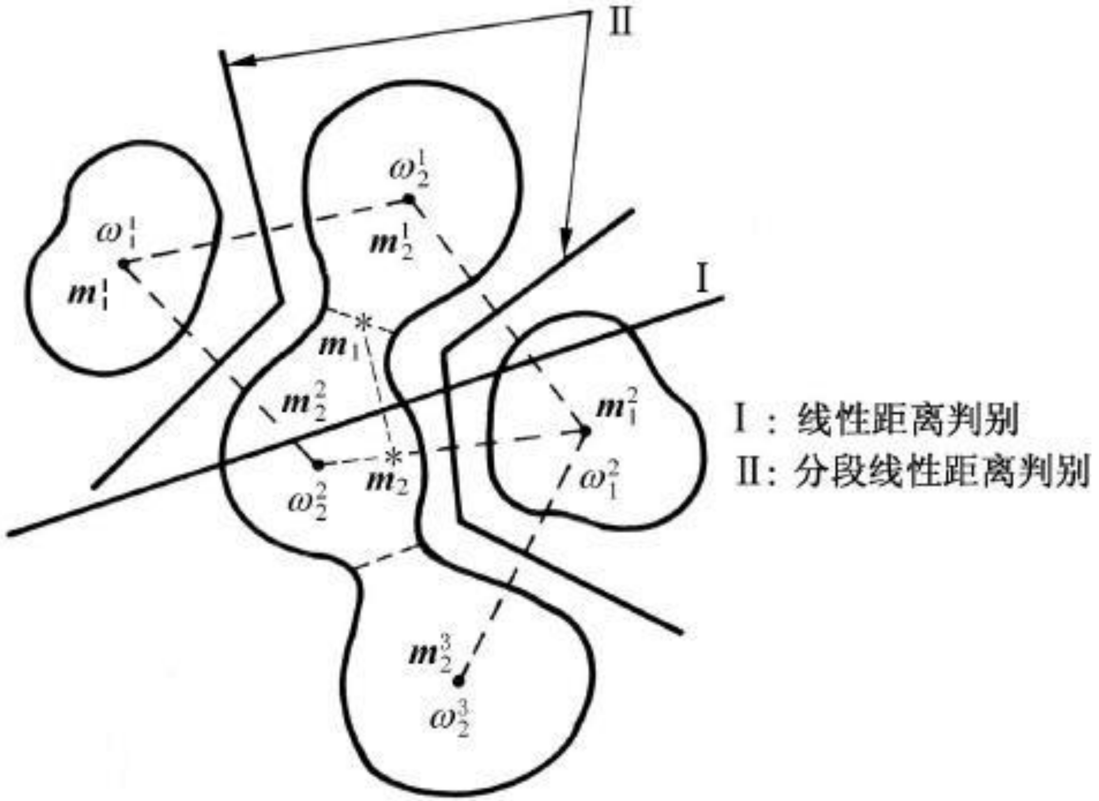
把各类划分为若干子类，

以子类中心作为类别代表

点，考查新样本到各代表

点的距离并将它分到最近

的代表点所代表的类。



极端情况，将所有样本都作为代表点

近邻法 ➔ （Nearest-Neighbor method）

15.3 Kn近邻估计方法

15.3.1 最近邻法

样本集 $S_N = \{(x_1,\theta_1),(x_2,\theta_2),\cdots,(x_N,\theta_N)\}$

x_i：样本，θ_i：类别标号，$\theta_i \in \{1,2,\cdots,c\}$

样本 x_i 与 x_j 之间的距离 $\delta(x_i,x_j)$：比如欧氏距离 $\|x_i - x_j\|$

对未知样本 x，求 S_N 中与之距离最近的样本 x'（类别为 θ'）

$$\delta(x,x') = \min_{j=1,\cdots,N} \delta(x,x_j)$$

则将 x 分到 θ' 类，即 $\hat{\omega}(x) = \theta'$

——最近邻决策（一近邻决策）

15.3.1 最近邻法

一种表达方法：　　　　　i是类别标号，k是样本编号

ω_i 类判别函数 $g_i(x)=\min_k \|x-x_i^k\|$， $x_i^k\in\omega_i$， $k=1,\cdots,N_i$

决策规则： If $g_j(x)=\min_{i=1,\cdots,c} g_i(x)$， then $x\in\omega_j$

定义$\omega_m(x)$　$P(\omega_m|\mathrm{x})=\max_i P(\omega_i|\mathrm{x})$

为什么可以用最近邻法

根据贝叶斯决策，将选择ω_m类作为X的判决

这个贝叶斯条件概率的错误率是

$$P^*(e|\mathrm{x})=1-P(\omega_m|\mathrm{x})$$

让P*(e|x)代表P(e|x)的最小可能值，P*代表P(e)的最小可能值，可得:

$$P^*=\int P^*(e|x)p(x)dx=\int\left(1-P\left(\omega_m|x\right)\right)p\left(x\right)dx$$

15.3.1 最近邻法

最近邻的收敛性

如果$P_n(e)$ 是n个样本的最近邻错误率，并且有

$$P = \lim_{n\to\infty} P_n(e)$$

- ✓ 最近邻法则导致的错误率大于贝叶斯决策方法最小错误率
- ✓ **最近邻规则是一个次优的过程**
  - **•不优于贝叶斯错误率**
  - **•如果样本数目多，**最近邻法则的错误率小于贝叶斯决策方法最小错误率的两倍。

15.3.1 最近邻法

- 最近邻法的错误率（渐近分析）

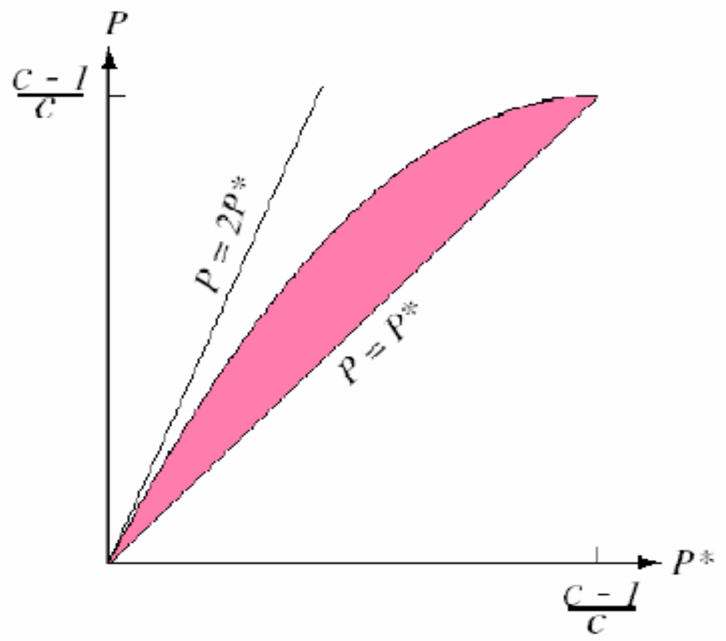
结论： $P^* \le P_1 \le P^*\left(2-\frac{c}{c-1}P^*\right)$

其中： P^*：贝叶斯错误率

P_1：样本无穷多时最近邻法的错误率（渐近平均错误率）

C：类别数

前提：样本集独立同分布



由于一般情况下P*很小，因此又可粗略表示成： $P^* \le P \le 2P^*$

可粗略说最近邻法的渐近平均错误率在贝叶斯错误率的两倍之内。

15.3.1 最近邻法

自习

❖ 点x属于θ类，x的最近邻点x'属于θ'类这样两个概率是互相独立的

$$P\left(\theta,\theta' \mid \mathrm{x},\mathrm{x}'\right)=P\left(\theta \mid \mathrm{x}\right)P\left(\theta' \mid \mathrm{x}'\right)$$

如果我们用最近邻判决规则，我们可能遇到一个错误θ≠θ'，对应的错误率$P_n(e|x,x')$是一个条件概率

$$P_n\left(e \mid \mathrm{x},\mathrm{x}'\right)=1-\sum_{i=1}^{c}P\left(\theta=\omega_i,\theta'=\omega_i \mid \mathrm{x},\mathrm{x}'\right)=1-\sum_{i=1}^{c}P\left(\omega_i \mid \mathrm{x}\right)P\left(\omega_i \mid \mathrm{x}'\right)$$

如果训练样本N趋于无限，空间会被填充满，x等于其最近邻x'的概率为1。

$$\lim_{n\to\infty}P\left(e \mid \mathrm{x},\mathrm{x}'\right)=\lim_{n\to\infty}P\left(e \mid \mathrm{x},\mathrm{x}\right)=\lim_{n\to\infty}P\left(e \mid \mathrm{x}\right)$$

$$P = \lim_{n\to\infty} P_n(e)$$

自习

而在这条件下的平均错误率

P称为渐近平均错误率，是$P_N(e)$在N→∞的极限。

$$P = \lim_{N\to\infty} P_N(e\,|\,\mathrm{x})$$

$$= \lim_{N\to\infty} \int P_N(e\,|\,\mathrm{x})\,p(\mathrm{x})\,dx = \int \lim_{N\to\infty} P_N(e\,|\,\mathrm{x})\,p(\mathrm{x})\,dx$$

$$= \int \left[1 - \sum_{i=1}^{c} P^2(\omega_i\,|\,\mathrm{x})\right] p(\mathrm{x})\,dx$$

$$\boxed{\lim_{n\to\infty} P(\omega_i\,|\,\mathrm{x}') \cong \lim_{n\to\infty} P(\omega_i\,|\,\mathrm{x})}$$

$$\boxed{\lim_{n\to\infty} P(e\,|\,\mathrm{x}) = \lim_{n\to\infty} P(e\,|\,\mathrm{x},\mathrm{x}')} \rightarrow \boxed{P_n(e\,|\,\mathrm{x},\mathrm{x}') = 1 - \sum_{i=1}^{c} P(\omega_i\,|\,\mathrm{x})P(\omega_i\,|\,\mathrm{x}')}$$

贝叶斯错误率的计算式：

$$P^* = \int P^*(e\,|\,\mathrm{x})\,p(\mathrm{x})\,dx$$

$$\boxed{P^*(e\,|\,\mathrm{x}) = 1 - P(\omega_m\,|\,\mathrm{x})} \leftarrow \boxed{P(\omega_m\,|\,\mathrm{x}) = \max_i \left[P(\omega_i\,|\,\mathrm{x})\right]}$$

$$\sum_{i=1}^{c} P^2(\omega_i \mid \mathrm{x}) = P^2(\omega_m \mid \mathrm{x}) + \sum_{i \neq m} P^2(\omega_i \mid \mathrm{x})$$

$$\because P(\omega_i \mid \mathrm{x}) \geq 0,\ \sum_{i \neq m} P(\omega_i \mid \mathrm{x}) = 1 - P(\omega_m \mid \mathrm{x}) = P^*(e \mid \mathrm{x})$$

在i≠m时，所有的P都相等，将使得$\sum_{i=1}^{c} P^2(\omega_i \mid \mathrm{x})$最小

$$P(\omega_i \mid \mathrm{x}) = \begin{cases} \dfrac{P^*(e \mid \mathrm{x})}{c-1} & i \neq m \\ 1 - P^*(e \mid \mathrm{x}) & i = m \end{cases}$$

$1 - 2P^*(e|x) + \frac{c}{c-1} P^{*2}(e|x)$

$$\therefore \quad \sum_{i=1}^{c} P^2(\omega_i \mid \mathrm{x}) \geq \left(1 - P^*(e \mid \mathrm{x})\right)^2 + \frac{P^{*2}(e \mid \mathrm{x})}{c-1}$$

$$and \quad 1 - \sum_{i=1}^{c} P^2(\omega_i \mid \mathrm{x}) \leq 2P^*(e \mid \mathrm{x}) - \frac{c}{c-1} P^{*2}(e \mid \mathrm{x})$$

$$1-\sum_{i=1}^{c} P^2(\omega_i \mid \mathrm{x}) \le 2P^*(e \mid \mathrm{x}) - \frac{c}{c-1} P^{*2}(e \mid \mathrm{x})$$

$$\boxed{P = \int \left[1-\sum_{i=1}^{c} P^2(\omega_i \mid \mathrm{x})\right] p(\mathrm{x})\,dx} \qquad \boxed{P^* = \int P^*(e \mid x)\,p(x)\,dx}$$

$$P \le 2P^* - \int \frac{c}{c-1} P^{*2}(e \mid x)\,p(x)\,dx$$

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

$$P \le 2P^*$$

$$\int P^{*2}(e \mid x)\,p(x)\,dx = E\left(P^{*2}(e \mid x)\right)$$

$$E\left(P^{*2}(e \mid x)\right) = E\left(P^*(e \mid x)\right)^2 + Var\left(P^*(e \mid x)\right)$$

$$\int P^{*2}(e \mid x)\,p(x)\,dx \le E\left(P^*(e \mid x)\right)^2 = P^{*2}$$

$\ge$

15.3.1 最近邻法

自习

$$P_n\left(e \mid \mathrm{x}, \mathrm{x}'\right)=1-\sum_{i=1}^{c} P\left(\omega_i \mid \mathrm{x}\right) P\left(\omega_i \mid \mathrm{x}'\right)$$

若是两类问题，则

贝叶斯错误率：

$$P^*(e \mid X)=\begin{cases}1-P(\omega_1 \mid X) & if\ P(\omega_1 \mid X)>P(\omega_2 \mid X) \\ 1-P(\omega_2 \mid X) & if\ P(\omega_1 \mid X)<P(\omega_2 \mid X)\end{cases}$$

最近邻法错误率：

$$P_N(e \mid X)=1-P^2(\omega_1 \mid X)-P^2(\omega_2 \mid X)$$

$$\Delta P=P_N(e \mid X)-P_N^*(e \mid X)$$

$$\Delta P=P(\omega_1 \mid X)\left[1-P(\omega_1 \mid X)\right]-P^2(\omega_2 \mid X)$$

$$=P(\omega_2 \mid X)\left[P(\omega_1 \mid X)-P(\omega_2 \mid X)\right] \qquad if\ P(\omega_1 \mid X)>P(\omega_2 \mid X)$$

$$\Delta P=P(\omega_2 \mid X)\left[1-P(\omega_2 \mid X)\right]-P^2(\omega_1 \mid X)$$

$$=P(\omega_1 \mid X)\left[P(\omega_2 \mid X)-P(\omega_1 \mid X)\right] \qquad if\ P(\omega_1 \mid X)<P(\omega_2 \mid X)$$

可见在一般情况下△P是大于零的值。只有在$P(\omega_1|x)=1$或$P(\omega_2|x)=1$ 或$P(\omega_1|x)=P(\omega_2|x)=1/2$ 的情况△P=0.

15.3.1 最近邻法

自习

若是两类问题，则

贝叶斯错误率： $$P^*(e|\mathrm{x})=\begin{cases}1-P(\omega_1|\mathrm{x}) & if\ P(\omega_1|\mathrm{x})>P(\omega_2|\mathrm{x})\\ 1-P(\omega_2|\mathrm{x}) & if\ P(\omega_1|\mathrm{x})<P(\omega_2|\mathrm{x})\end{cases}$$

最近邻法错误率： $$P_N(e|\mathrm{x})=1-P^2(\omega_1|\mathrm{x})-P^2(\omega_2|\mathrm{x})$$

$$\Delta P=P_N(e|\mathrm{x})-P_N^*(e|\mathrm{x})$$

$$\boxed{\begin{aligned}&\Delta P=P(\omega_1|X)\left[1-P(\omega_1|X)\right]-P^2(\omega_2|X)\\&=P(\omega_2|X)\left[P(\omega_1|X)-P(\omega_2|X)\right] \qquad if\ P(\omega_1|X)>P(\omega_2|X)\\&\Delta P=P(\omega_2|X)\left[1-P(\omega_2|X)\right]-P^2(\omega_1|X)\\&=P(\omega_1|X)\left[P(\omega_2|X)-P(\omega_1|X)\right] \qquad if\ P(\omega_1|X)<P(\omega_2|X)\end{aligned}}$$

$$\Downarrow$$

$$P_N(e|\mathrm{x})\geq P^*(e|\mathrm{x})$$

15.3.1 最近邻法

自习

$$1-\sum_{i=1}^{c}P^2(\omega_i \mid \mathrm{x})\le 2P^*(e\mid \mathrm{x})-\frac{c}{c-1}P^{*2}(e\mid \mathrm{x})$$

$$\boxed{P=\int\left[1-\sum_{i=1}^{c}P^2(\omega_i\mid \mathrm{x})\right]p(\mathrm{x})d\mathrm{x}}$$

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

$$P_N(e\mid \mathrm{x})\ge P^*(e\mid \mathrm{x})$$

$$P^*\le P\le P^*(2-\frac{C}{C-1}P^*)$$

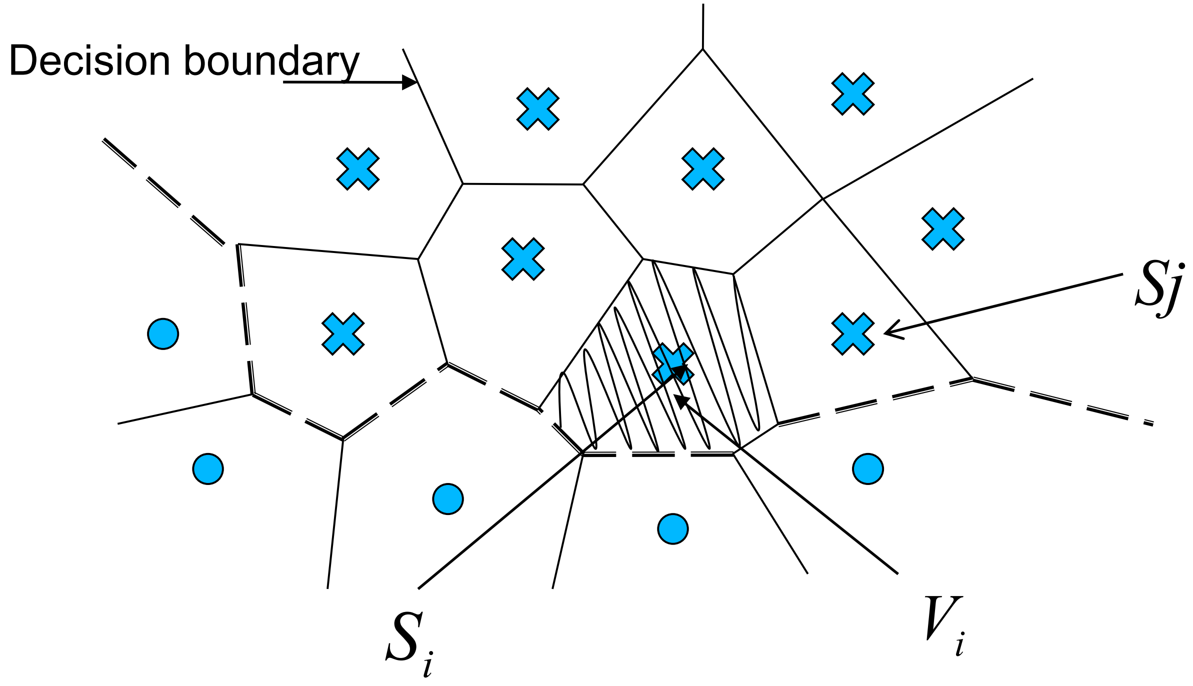
$$P^*\le P\le 2P^*$$

P^*：贝叶斯错误率　P：最近邻法错误率　C：类别数

15.3.1 最近邻法

❖ **Voronoi 网格：决策面来自于训练样本**

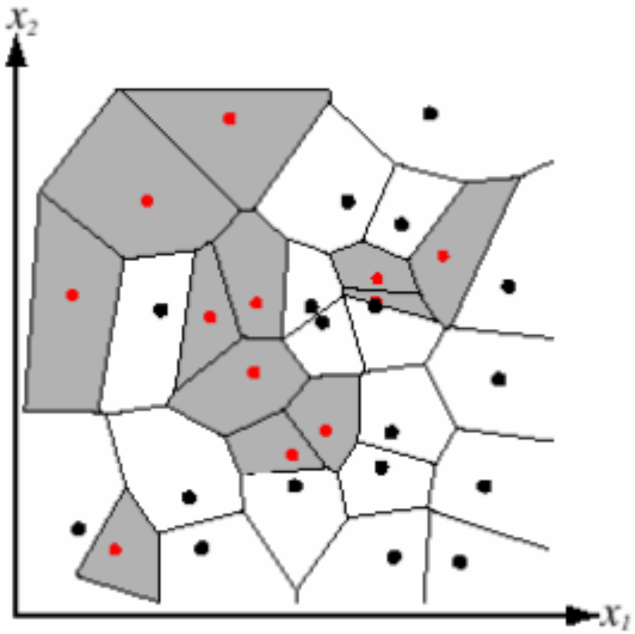
预备知识：Voronoi图，是由一组连接两邻近点直线的垂直平分线组成的连续多边形组成。由此划分的每个区域中包含一个点，对于点pi，它所在区域中的任何一个位置距离p_i点比距离其他点更近。

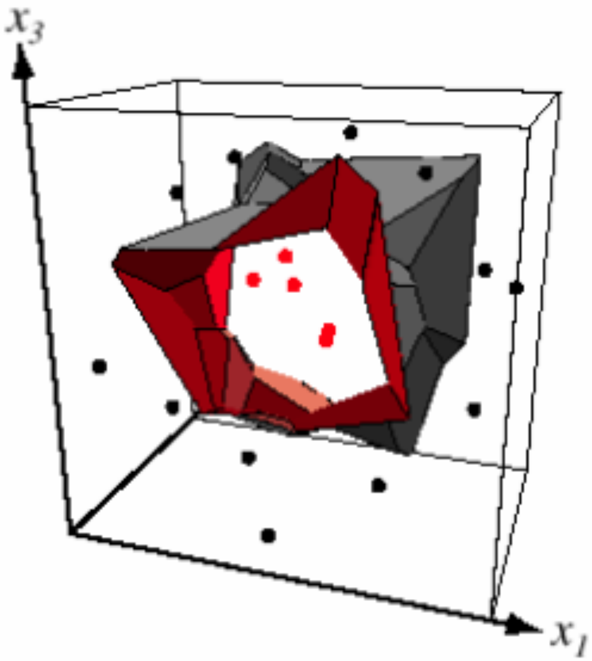


•V_i是一个多边形，任何落入这个多边形的点x距离点S_i都比其他样本点的距离要近。

15. 3. 1 最近邻法

2-dimensions





3-dimensions

✓ 最近邻规则把特征空间分成一个个网格单元结构，称为Voronoi网格,

-每一个单元包含一个训练样本点x’,如果测试样本x落入该单元，x到x’的距离均小于到其他训练样本点的距离

-如果测试样本x落入该单元，则判别为x’所属的类别

15.3.2 k-近邻法（kNN）

最近邻法（一近邻法）的推广：

找出 x的k个近邻，看其中多数属于哪一类，则把 x分到哪一类。

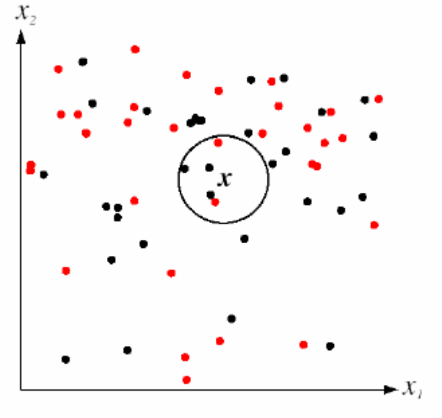
一般表示： c 类 ω_i ，$i=1,\cdots,c$，N 个样本。

$k_i, i=1,\cdots,c$ 为 x 的 k 个近邻中属于 ω_i 的样本数

判别函数：$g_i(x)=k_i$， $i=1,\cdots,c$

决策规则： if $g_j(x)=\max\limits_{i=1,\cdots,c} k_i$ ，then $x\in\omega_j$

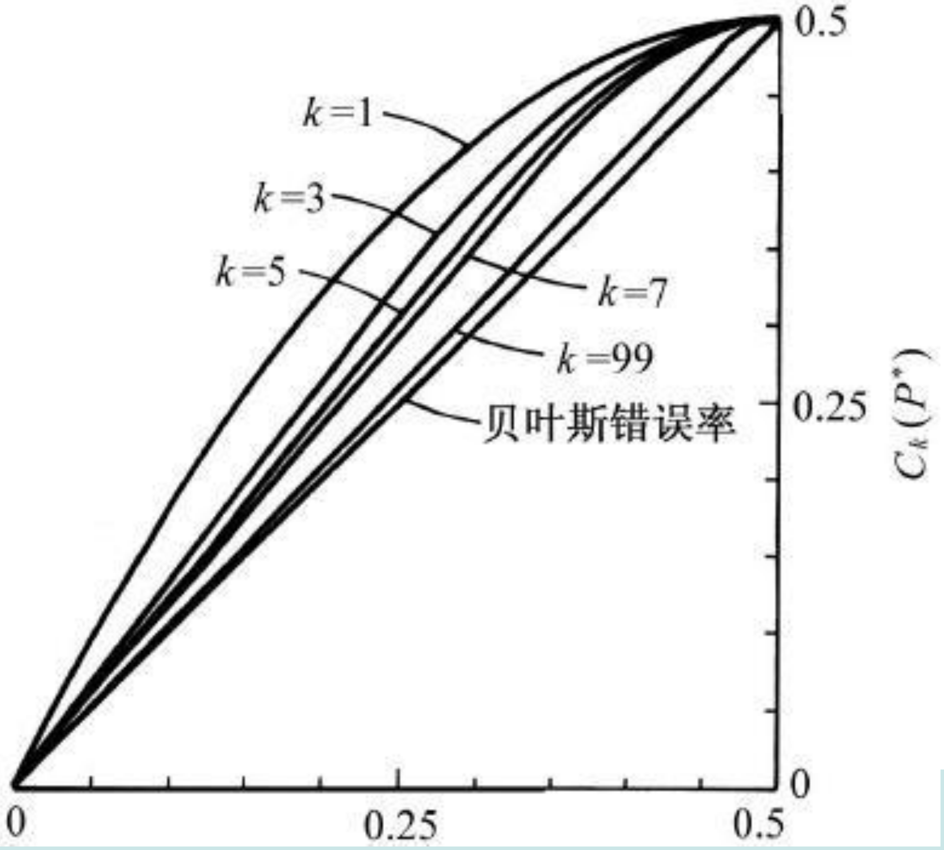
x的分类，是通过统计最邻近的k个样本的属性，用投票法将最常见的类别标示x



15.3.2 k-近邻法（kNN）

渐近平均错误率的界：

N 无穷大时，k 越大,P_k 的上限越低（越靠近下限）。但 k应始终是 N 中的一小部分，保证 k个近邻均充分接近 x。否则这一关系不成立。



一般来说，总有

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

或者简化为 $$P^* \le P_k \le 2P^*$$

15.3.2 k-近邻法（kNN）

自习

✓ 如果k近邻大多数被标记为ω_m类，作出这样选择的概率为（两类问题）

$$\sum_{i=(k+1)/2}^{k}\binom{k}{i}P(\omega_m|\mathrm{x})^{i}\left[1-P(\omega_m|\mathrm{x})\right]^{k-i}$$

可以证明，如果k是奇数，大样本的两类问题的k近邻规则错误率是有界的,它可以用函数$C_k(P^*)$表示, 这里$C_k(P^*)$被定义为关于P *的最小的凹函数

15.3.2 k-近邻法（kNN）

自习

最近邻法条件错误率（两类问题）

$$P_n\left(e\,|\,\mathrm{x},\mathrm{x}'\right)=1-\sum_{i=1}^{c}P\left(\theta=\omega_i,\theta'=\omega_i\,|\,\mathrm{x},\mathrm{x}'\right)=1-\sum_{i=1}^{c}P(\omega_i\,|\,\mathrm{x})P\left(\omega_i\,|\,\mathrm{x}'\right)$$

$$=1-P(\omega_1\,|\,\mathrm{x})P\left(\omega_1\,|\,\mathrm{x}'\right)-P(\omega_2\,|\,\mathrm{x})P\left(\omega_2\,|\,\mathrm{x}'\right)$$

$$=1-\left(1-P(\omega_2\,|\,\mathrm{x})\right)P\left(\omega_1\,|\,\mathrm{x}'\right)-\left(1-P(\omega_1\,|\,\mathrm{x})\right)P\left(\omega_2\,|\,\mathrm{x}'\right)$$

$$=\cancel{1}-\cancel{P}\left(\omega_1\,|\,\mathrm{x}'\right)+P(\omega_2\,|\,\mathrm{x})P\left(\omega_1\,|\,\mathrm{x}'\right)\cancel{-P}\left(\omega_2\,|\,\mathrm{x}'\right)+P(\omega_1\,|\,\mathrm{x})P\left(\omega_2\,|\,\mathrm{x}'\right)$$

$$=P(\omega_1\,|\,\mathrm{x})P\left(\omega_2\,|\,\mathrm{x}'\right)+P(\omega_2\,|\,\mathrm{x})P\left(\omega_1\,|\,\mathrm{x}'\right)$$

$\because N\to\infty$时，有$P\left(\omega_i\,|\,x'\right)\approx P(\omega_i\,|\,x)$

$$\therefore P_{n\to\infty}\left(e\,|\,\mathrm{x},\mathrm{x}'\right)=P(\omega_1\,|\,\mathrm{x})P(\omega_2\,|\,\mathrm{x})+P(\omega_2\,|\,\mathrm{x})P(\omega_1\,|\,\mathrm{x})$$

15.3.2 k-近邻法（kNN）

自习

推广到k近邻，设x属于ω_1，但$k_1 \le \frac{k-1}{2}$，则$k_1 \le k-k_1=k_2$，此时发生误判，

这一事件的概率为$\sum_{i=0}^{(k-1)/2}\binom{k}{j}P(\omega_1|\mathrm{x})^j P(\omega_2|\mathrm{x})^{k-j}$

j

反之，x属于ω_2，发生误判的概率为$\sum_{i=0}^{(k-1)/2}\binom{k}{j}P(\omega_2|\mathrm{x})^j P(\omega_1|\mathrm{x})^{k-j}$

所以有给定x时的条件错误率为

$$P_{k,N}(e|x)=P(\omega_1|x)\sum_{i=0}^{(k-1)/2}\binom{k}{j}P(\omega_1|x)^j P(\omega_2|x)^{k-j}$$

$$+P(\omega_2|x)\sum_{i=0}^{(k-1)/2}\binom{k}{j}P(\omega_2|x)^j P(\omega_1|x)^{k-j}$$

15.3.2 k-近邻法（kNN）

自习

$$P_{k,N}(e|x)=P(\omega_1|x)\boxed{\sum_{j=0}^{(k-1)/2}\binom{k}{j}P(\omega_1|x)^j P(\omega_2|x)^{k-j}}$$

$x\in\omega_1$而决策为

$x\notin\omega_1$的概率

$$+P(\omega_2|x)\boxed{\sum_{j=0}^{(k-1)/2}\binom{k}{j}P(\omega_2|x)^j P(\omega_1|x)^{k-j}}$$

$x\in\omega_2$而决策为

$x\notin\omega_2$的概率

一般化

$$P_{k,N\to\infty}(e|x)=P(\omega_i|x)\sum_{j=0}^{(k-1)/2}\binom{k}{j}P(\omega_i|x)^j\left[1-P(\omega_i|x)\right]^{k-j}$$

$$+\underline{\left[1-P(\omega_i|x)\right]\sum_{j=(k-1)/2}^{k}}\binom{k}{j}P(\omega_i|x)^j P\left[1-P(\omega_i|x)\right]^{k-j}$$

一般化

自习

$$P_{k,N\to\infty}(e|x)=P(\omega_i|x)\sum_{j=0}^{(k-1)/2}\binom{k}{j}P(\omega_i|x)^j\left[1-P(\omega_i|x)\right]^{k-j}$$

$$+\left[1-P(\omega_i|x)\right]\sum_{j=(k-1)/2}^{k}\binom{k}{j}P(\omega_i|x)^j\left[1-P(\omega_i|x)\right]^{k-j}$$

贝叶斯条件错误率为

$$P^*(e|x)=\min\left[P(\omega_1|x),P(\omega_2|x)\right]=\min\left[P(\omega_1|x),1-P(\omega_1|x)\right]$$

组合$\binom{k}{j}=\frac{k!}{j!(k-j)!}=$组合$\binom{k}{k-j}$

$$P_{k,N\to\infty}(e|x)=P^*(e|x)\sum_{j=0}^{(k-1)/2}\binom{k}{j}P^*(e|x)^j\left[1-P^*(e|x)\right]^{k-j}$$

$$+\left[1-P^*(e|x)\right]\sum_{j=0}^{(k-1)/2}\binom{k}{j}P^*(e|x)^{k-j}\left[1-P^*(e|x)\right]^j$$

$$=\sum_{i=0}^{(k-1)/2}\binom{k}{j}\left[\left(P^*\right)^{j+1}\left(1-P^*\right)^{k-j}+\left(P^*\right)^{k-j}\left(1-P^*\right)^{j+1}\right]$$

15.3.2 k-近邻法（kNN）

自习

- 定义一个贝叶斯条件错误率P*(e|x)的函数C_k[P*(e|x)]，C_k[P*(e|x)]为大于 $P_{k,N\to\infty}(e|x)$ 的最小凹函数，那么对于所有x有

K近邻渐进平均错误率

$$P_{k,N\to\infty}(e|\mathrm{x}) \le C_k\left[P^*(e|\mathrm{x})\right]$$

- 因为 $P_{k,N\to\infty}(e|x)$ 随k的增大单调减小，故最小凹函数C_k也随k单调减小，所以有

$$\therefore 有 P = E[P_{k,N\to\infty}(e|\mathrm{x})] \le E\left\{C_k\left[P^*(e|\mathrm{x})\right]\right\} \le C_k\left\{E\left[P^*(e|\mathrm{x})\right]\right\} = C_k\left[P^*\right]$$

$$P^* \le P \le C_k\left[P^*\right] \le C_{k-1}\left[P^*\right] \le \cdots \le C_1\left[P^*\right] \le 2P^*\left(1-P^*\right)$$

贝叶斯错误率

最近邻法和k-近邻法的错误率上下界都是在一倍到两倍贝叶斯决策方法的错误率范围内。

在k >1的条件下，k-近邻法的错误率要低于最近邻法。

在$k\to\infty$的条件下，k-近邻法的错误率等于贝叶斯误差率

得证

15.3.2 k-近邻法（kNN）

问题

① 存储量和计算量

1:1 / 2:2

② 票数接近时风险较大，有噪声时风险加大

③ 有限样本下性能如何？

改进：

① 减少计算量和存储量

② 引入拒绝机制 (票数接近)

③ 根据实际问题修正投票方式

如加权投票，否决票等

如距离加权，考虑样本比例及先验概率等

15.3.3 近邻法的快速算法

近邻法在计算上的问题：
- 需存储所有训练样本
- 新样本需与每个样本做比较

❖ **加速方法:**

- **“部分距离”计算法**
- **“预建立结构 ”算法**

改进的思路：

1) 对样本集进行组织与整理，分群分层，尽可能将计算压缩到在接近测试样本邻域的小范围内，避免盲目地与训练样本集中每个样本进行距离计算。

2) 在原有样本集中挑选出对分类计算有效的样本，使样本总数合理地减少，以同时达到既减少计算量，又减少存储量的双重效果。

15.3.3 近邻法的快速算法

“部分距离” 计算法

$$D(\mathrm{a},\mathrm{b})=\left(\sum_{k=1}^{d}(a_k-b_k)^2\right)^{1/2} \Rightarrow D_r(\mathrm{a},\mathrm{b})=\left(\sum_{k=1}^{r}(a_k-b_k)^2\right)^{1/2}$$

$$r \le d$$

- 一旦其部分的距离大于目前最接近的样本的全欧氏距离(r =d)时,终止距离计算

“预建立结构”计算法

把样本集分级分成多个子集（树状结构）

每个子集（结点）可用较少几个量代表

通过将新样本与各结点比较排除大量候选样本

只有最后的结点（子集）中逐个样本比较，找出近邻

15.3.3 近邻法的快速算法

“预建立结构”计算法

样本集->分集分解构建搜索树->搜索

➢ 基本思想：

• 将样本集按邻近关系分解成树，给出每树的质心所在，以及树内样本至该质心的最大距离。这些树又可形成层次结构，即“树”又分“子树”。

• 待识别样本可将搜索近邻的范围从某一大棵树，逐渐深入到其中的子树，直至树的叶结点所代表的树，确定其相邻关系。搜索过程只需要选择最有可能的那个**根(子节点)，然后只考虑和这个根（子节点）相连的树(子树)上的样本。**

•这种方法着眼于只解决减少计算量，但没有达到减少存储量的要求。

•**如果**结构**合理，可以降低计算时间**

15.3.3 近邻法的快速算法

"预建立结构"计算法

（1）样本集的分级分解构建搜索树

首先将整个样本分成l个子集，每个子集又分为它的l个子集，如此进行若干次就能建立起一个样本集的树形结构。分成子集的原则是该子集内的样本尽可能聚成堆，这可用聚类方法实现。

（2）用树结构表示样本分级:

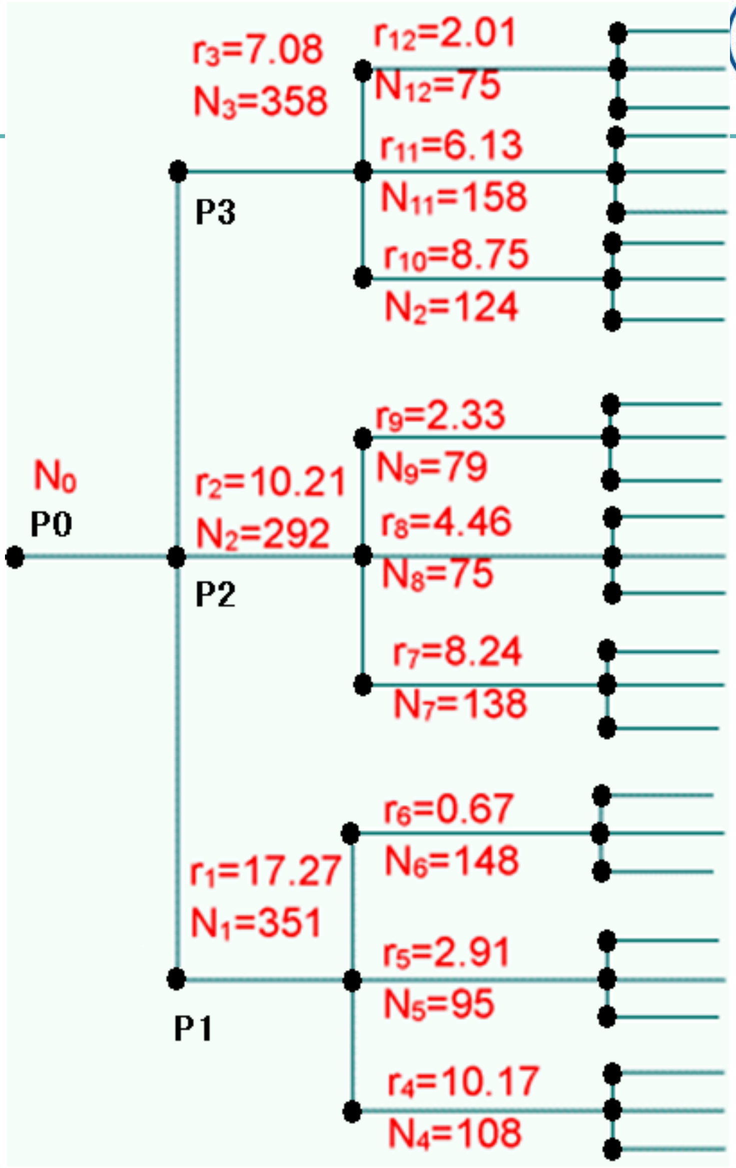
p: 树的一个结点，对应一个样本子集χ_p

N_p : χ_p中的样本数

M_p : χ_p中的样本均值

r_p : 从χ_p中任一样本到M_p的最大距离

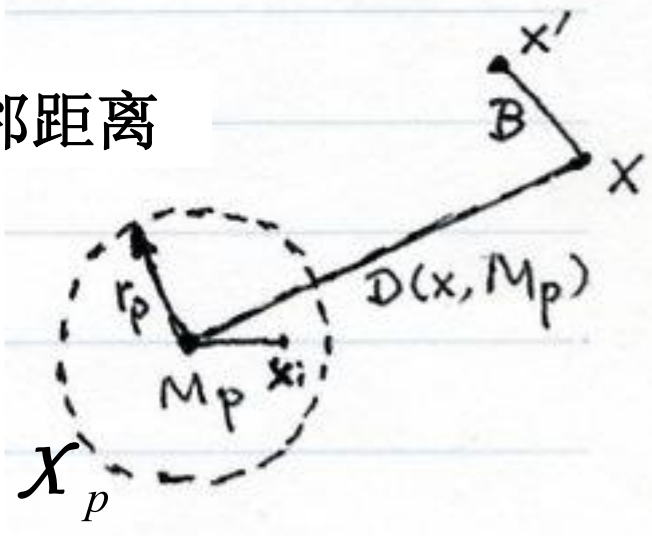
$$r_p = \max_{x_i \in \chi_p} D(x_i, M_p)$$



15.3.3 近邻法的快速算法

规则：

B：当前搜索到的最近邻距离



1. 对新样本 x，结点 X_p

   若 $D(x,M_p) > B + r_p$

   则 x 的近邻不可能在 X_p 中

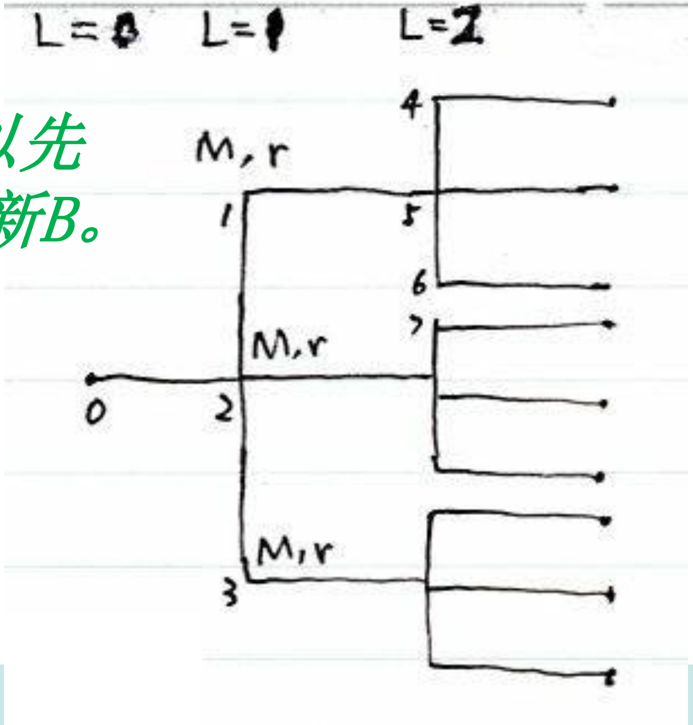
2. 对新样本 x，结点 $\mathcal{X}_p$ 中的样本 $x_i \in X_p$

   +D(x_i,M_p)

   D(x,x_i) 若> $D(x,M_p) > B + D(x_i,M_p)$

   则 x_i 不是 x的最近邻

其中r_p, $D(x_i, M_p)$在训练（建树）过程中可以先计算保存，搜索过程只需计算$D(x, M_p)$或更新B。



两大步：

1. 事先把样本子集划分好(比如用聚类算法)计算并存储 X_p 的 M_p，r_p 及 $D(x_i,M_p)$
2. 用分支定界算法搜索 x 的最近邻

15.3.4 剪辑近邻法

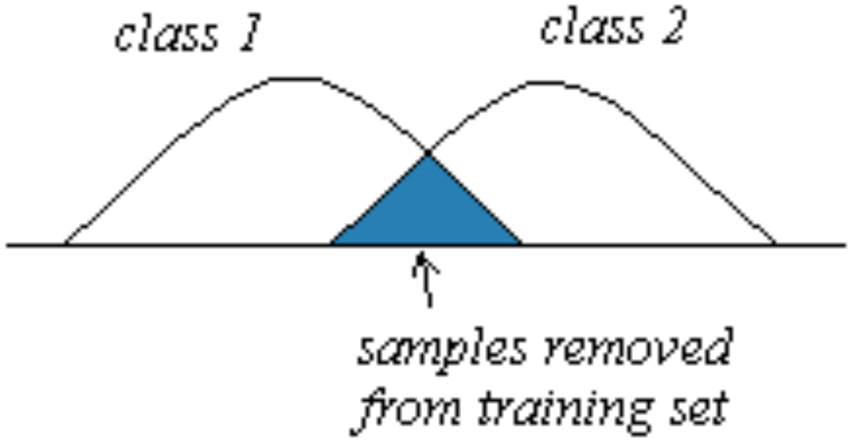
获得更准确的错误率

基本理解：

处在两类交界处或分布重合区的

样本可能误导近邻法决策。

应将它们从样本集中去掉。



基本思路：

考查样本是否为可能的误导样本，

若是则从样本集中去掉——剪辑。

考查方法是通过试分类，认为错分样本为误导样本。

15.3.4 剪辑近邻法

基本做法：

将样本集分为考试集 X^{NT}和参考集 X^{NR}：$X^{N}=X^{NT}\cup X^{NR}$，$X^{NT}\cap X^{NR}=\phi$

剪辑：用 X^{NR} 中的样本对X^{NT} 中的样本进行近邻法分类剪掉

X^{NT} 中被错分的样本，X^{NT} 中剩余样本构成剪辑样本集 X^{NTE}

分类：利用 X^{NTE} 和近邻法对未知样本 x 分类。

思考：

将样本集分为考试集和参考集是为了剪辑的独立性，但既然样本都是独立的，可否考虑下面的做法？

即：对 X^{N} 中每个 x_i ，用所有其他样本对它分类，若分错则剪掉。

训练样本和测试样本没有独立性，会产生一个偏于乐观的估计

15.3.4 剪辑近邻法

错误率分析（渐近错误率）

1. 若用最近邻剪辑，用最近邻分类，则错误率

$$P_1^E(e|x) = \frac{P(e|x)}{2[1-P(e|x)]} \qquad \because P(e|x) \ll 0.5$$

即 $P_1^E(e) \le P(e)$ （$P(e|x)$、$P(e)$是没有剪辑的近邻法的错误率）

当$P(e)$很小时，如 $P(e) < 0.1$，则有 $P_1^E(e) \approx \frac{1}{2}P(e)$

而 $P(e) \le 2P^*$（ P^*为贝叶斯错误率）。

故此时 $P_1^E(e)$ 接近 P^*。

15.3.4 剪辑近邻法

2. 若用 k 近邻剪辑，用最近邻分类，则

$$P_k^E(e|x) = \frac{P(e|x)}{2[1-P_k(e|x)]} < P_1^E(e|x)$$

当 $k \to \infty$ 时 $P_k^E(e)$ 收敛于 P^*（N应更快地趋向∞ ）

3. 多类情况，多类剪辑近邻错误率 $P_{k_c}^E(e|x)$ 小于两类情况

4. 重复剪辑

样本足够多时，可多次重复剪辑，效果更好。

15.3.4 剪辑近邻法

x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8

一种重复剪辑算法——MULTIEDIT: 除本的x_i

（1）（散开）把 X^N随机划分为 s个子集, $X_1,\cdots,X_x$， $s \geq 3$

（2）（分类）用 $X_j\,(j=(i+1)mad(s))$ 对 X_i 中的样本分类

$i=1,\cdots,s$.

（3）（剪辑）去掉(2)中错分的样本

（4）（混合）将剩下的样本合在一起，形成新的 $X^N(X^{NE})$

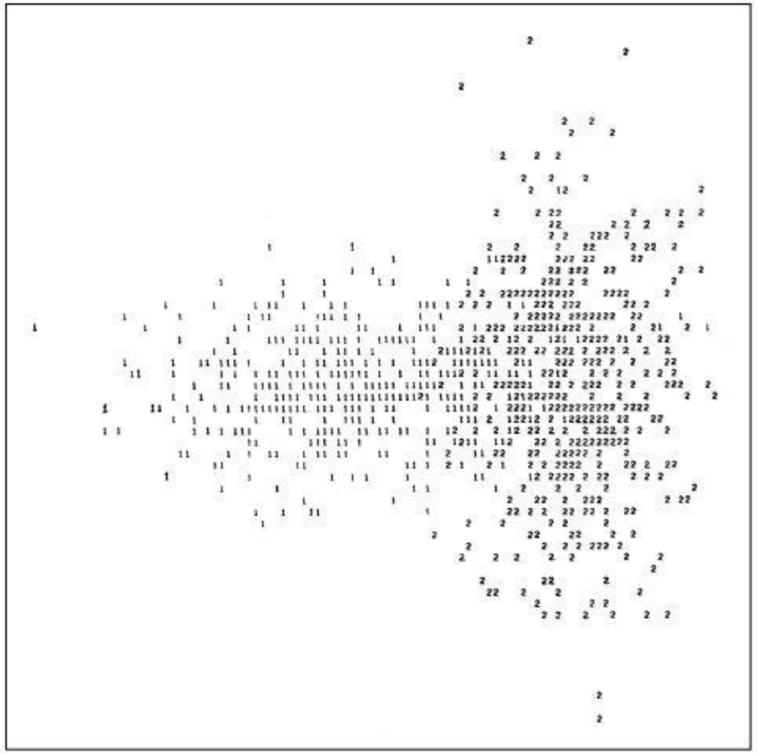
（5）（终止）如果该次迭代都没有样本被剪掉，则停止；否则用新的 X^N转（1）。

算法停止后，用最后的 X^{NE} 作为分类的样本集。

由此可见，每次迭代过程都要重新对现有的样本集进行重新随机划分，保证了剪辑的独立性。

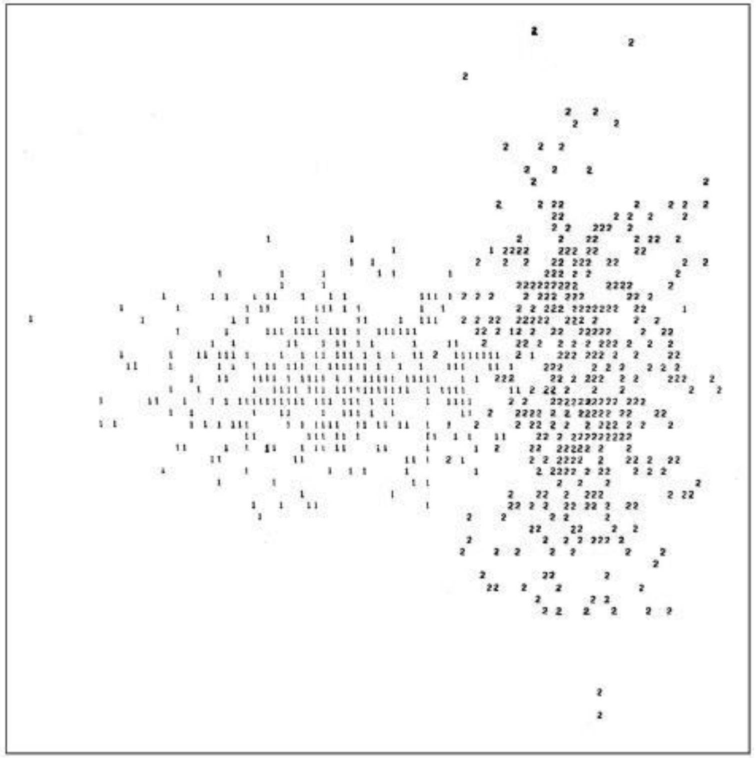
15.3.4 剪辑近邻法

例：



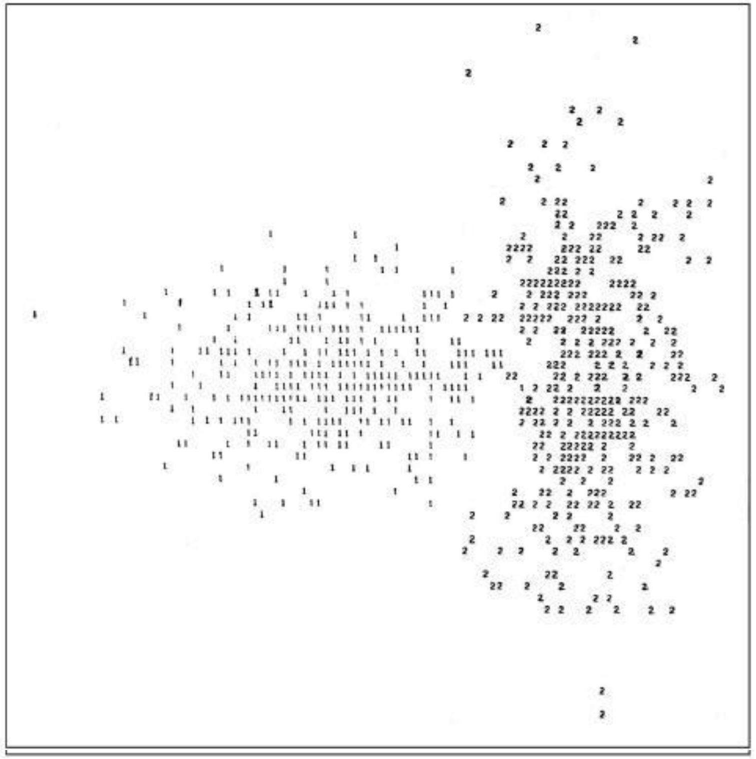
15.3.4 剪辑近邻法

例：



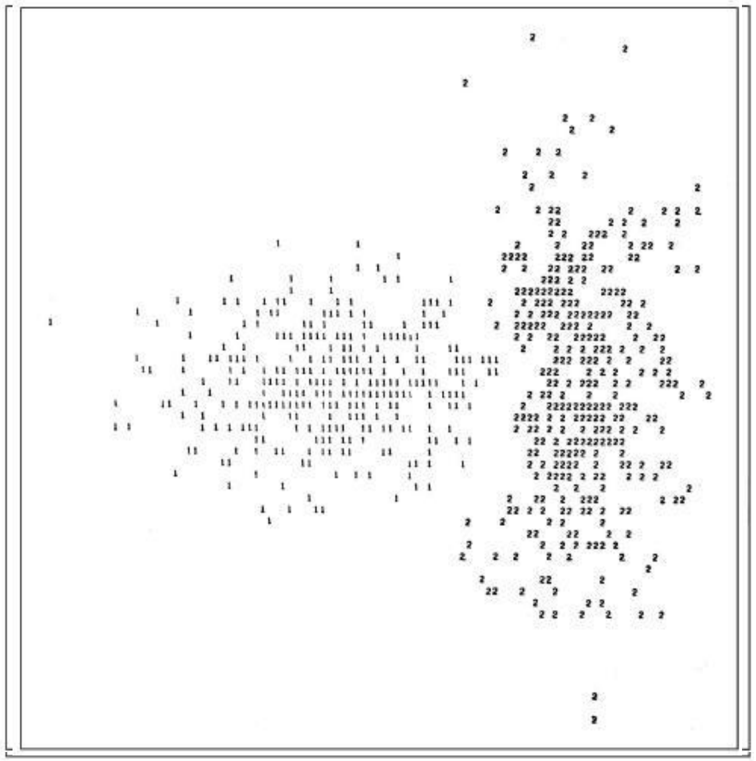
15.3.4 剪辑近邻法

例：



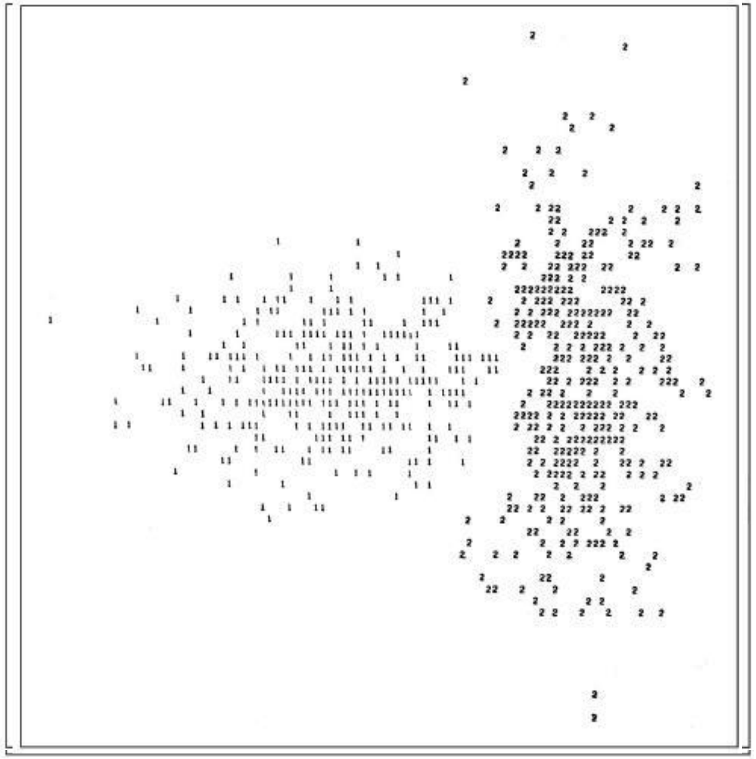
13.6.4 剪辑近邻法

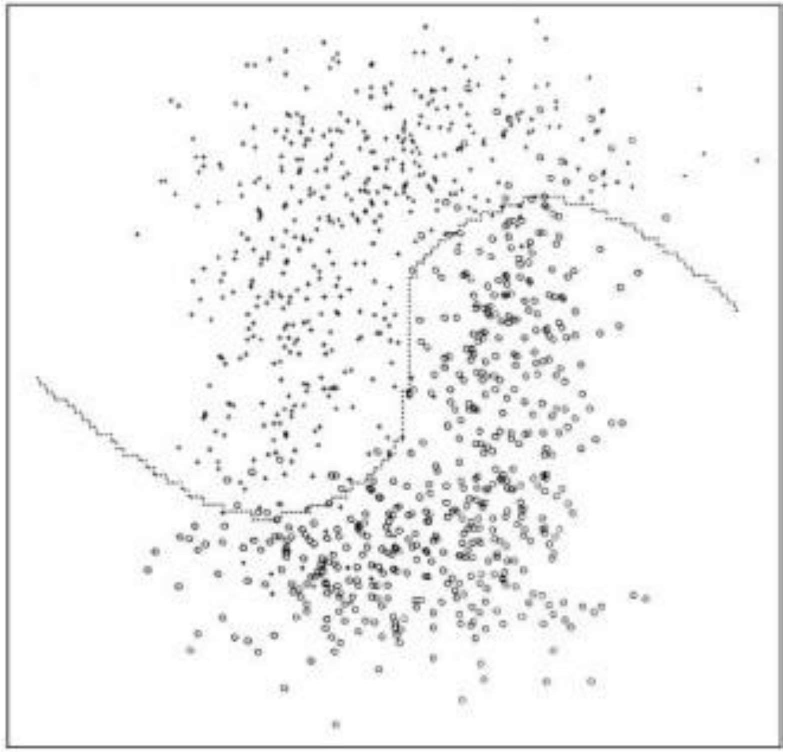
例：



15.3.4 剪辑近邻法

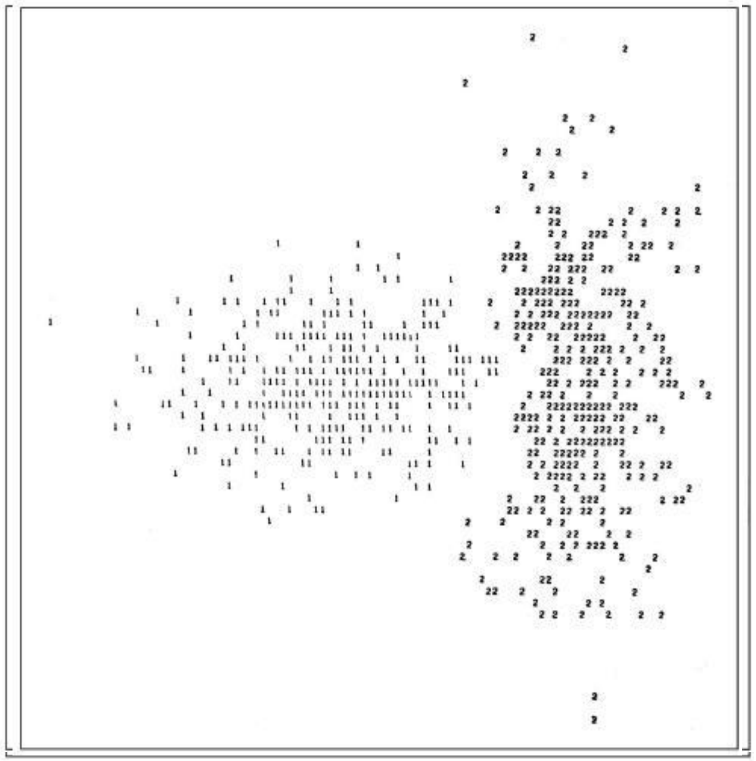
例：

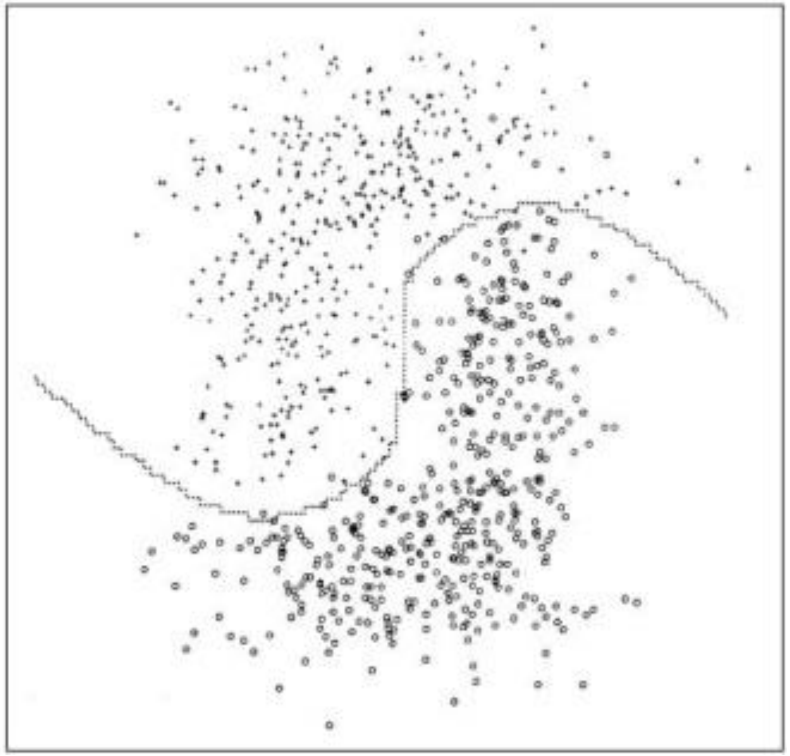




15.3.4 剪辑近邻法

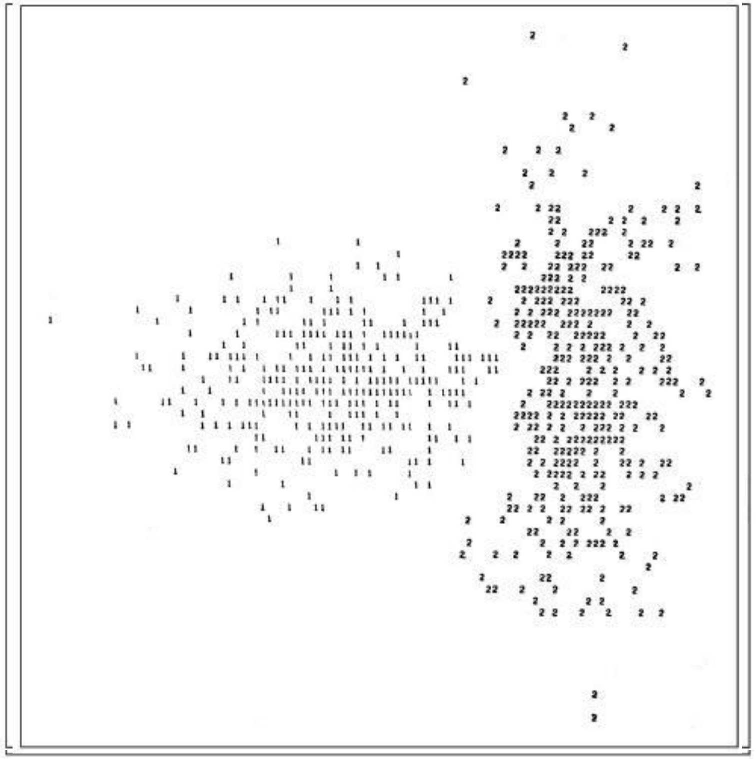
例：

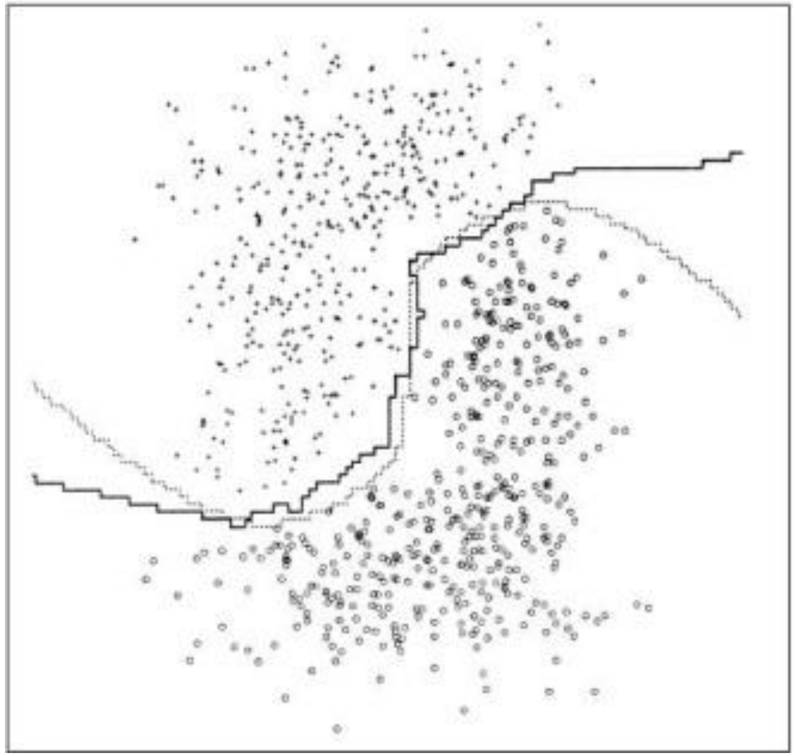




15.3.4 剪辑近邻法

例：





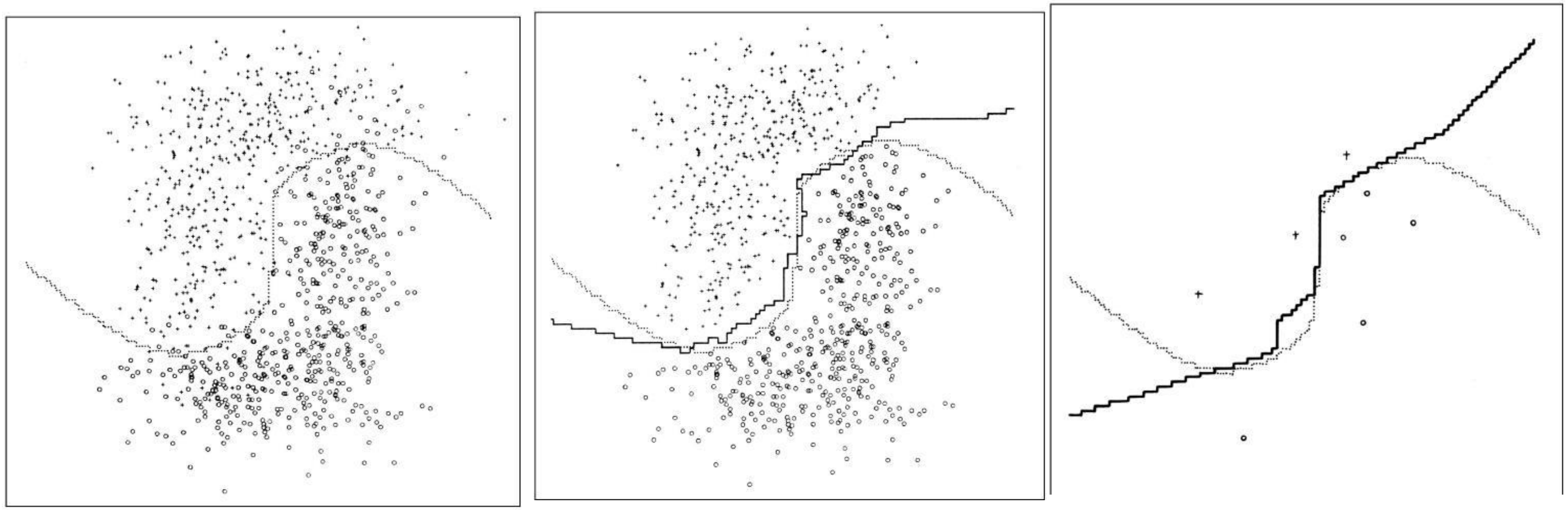
15.3.5 压缩近邻法

主要用以减少计算量

将 X^N分为 X_s和 X_G，开始时 X_s中只有一个样本，X_G中为其余样本。考查 X_G 中每个样本，若用 X_s可正确分类则保留，否则移入 X_s，……最后用 X_s 作分类的样本集。

可与剪辑法配合使用。

例：



15.3.5 压缩近邻法

```
% =====================压缩剪辑近邻算法（Condensing）=====================
% s:  划分的子集数目
% Xn: 当前样本集
% Xcur: 当前样本集经一次迭代后的样本集
% Xi: 当前考试集
% Xr: 当前参考集
% K:  退出控制条件，迭代K次，若没有样本被剪辑掉，则退出
% ===================================================================
clear,close all;
X = [randn(300,2)+ones(300,2);...
     randn(300,2)-ones(300,2);];
X(1:300,3)=1;X(301:600,3)=2;
% ===================================================================
figure, plot(X(1:300,1),X(1:300,2),'r.')
hold on,plot(X(301:600,1),X(301:600,2),'b.')
title('初始样本分布图')
% ===================================================================
s=3;Xcur=X;loop=0;Xold=X;K=5;
```

15. 3. 5 压缩近邻法

```
% ==
% s:
% Xn
% Xc
% Xi
% Xr
% K:
% ==
clea
X =

X(1:
% ==
figu
hold
titl
% ==
s=3;
```

```
while loop<K
    Xn=Xcur;
    Xold=Xcur;
    Xcur=[];
    [row1,col]=size(Xn);
    uu=unifrnd(0,s,row1,1);%产生row1行1列的随机数，随机数的范围在0-s之间
    uu=ceil(uu);%取整，方向是使数据变大
    for i=1:s   %样本随机划分为s个子集
        Xi=Xn((uu==i),:);%test set %Xi为考试集
        r=mod(i+1,s);%取余数
        if r==0
            r=s;
        end
        Xr=Xn((uu==r),:);%reference set%Xr为训练集
        [row,col]=size(Xi);
        j=1;
        while j<=row
            [rClass,jClass]=NNforCondense(Xr,Xi(j,:));%用训练集中的样本对考试集中的样本进行最近邻分类
            if rClass~=jClass%如果类别不同，则从考试集中分类错误的样本去除
                Xi(j,:)=[];
                row=row-1;
            else
                j=j+1;
            end
        end
```

15.3.5 压缩近邻法

```
        Xcur=[Xcur;Xi];
    end
    [oldRow,col]=size(Xold);
    [curRow,col]=size(Xcur);
    if oldRow==curRow
        loop=loop+1;
    else
        loop=0;
    end
end
% ==================================================================
%把当前样本集Xcur中的元素按原类别分类
[row,col]=size(Xcur);
Xcur1=[];Xcur2=[];
tic
for i=1:row
    if Xcur(i,3)==1
        Xcur1=[Xcur1;Xcur(i,1:2)];
    elseif Xcur(i,3)==2
        Xcur2=[Xcur2;Xcur(i,1:2)];
    end
end
time1=toc;
figure, plot(Xcur1(:,1),Xcur1(:,2),'r.')
hold on,plot(Xcur2(:,1),Xcur2(:,2),'b.')
title('剪辑后样本分布图')
```

15.3.5 压缩近邻法

```
    Xcu
    end
    [oldRow,
    [curRow,
    if oldR
        loop
    else
        loop
    end
end
% =========
%把当前样本集
[row,col]=s
Xcur1=[];Xc
tic
for i=1:row
    if Xcur
        Xcu
    elseif X
        Xcu
    end
end
time1=toc;
figure, plo
hold on,plo
title('剪辑
```

```
% ===================Condensing===================================
Xstore=Xcur(1,:);
Xgab=Xcur(2:row,:);
while 1
    Xoldstore=Xstore;
    [row,col]=size(Xgab);
    j=1;
    while j<=row
        [sClass,gClass]=NNforCondense(Xstore,Xgab(j,:));
        if sClass~=gClass
            Xstore=[Xstore;Xgab(j,:)];
            Xgab(j,:)=[];
            row=row-1;
        else
            j=j+1;
        end
    end
    [oldRow,col]=size(Xoldstore);
    [curRow,col]=size(Xstore);
    [gRow,rCol]=size(Xgab);
    if oldRow==curRow | gRow*rCol==0
        break;
    end
end
```

15.3.5 压缩近邻法

```
        Xcur
    end
    [oldRow,
    [curRow,
    if oldR
        loop
    else
        loop
    end
end
% ==========
%把当前样本集
[row,col]=s
Xcur1=[];Xcu
tic
for i=1:row
    if Xcur
        Xcur
    elseif X
        Xcur
    end
end
time1=toc;
figure, plot
hold on,plot
title('剪辑
```

```
% ======
Xstore=Xc
Xgab=Xcur
while 1
    Xolds
    [row,
    j=1;
    while
        [
        i
        e
        end
    end
    [oldRow,col]=size(Xoldstore);
    [curRow,col]=size(Xstore);
    [gRow,rCol]=size(Xgab);
    if oldRow==curRow | gRow*rCol==0
        break;
    end
end
```

```
Xcurstore1=[];Xcurstore2=[];
[curRow,col]=size(Xstore);
for i=1:curRow
    if Xstore(i,3)==1
        Xcurstore1=[Xcurstore1;Xstore(i,1:2)];
    else
        Xcurstore2=[Xcurstore2;Xstore(i,1:2)];
    end
end
figure, plot(Xcurstore1(:,1),Xcurstore1(:,2),'r.')
hold on,plot(Xcurstore2(:,1),Xcurstore2(:,2),'b.')
axis([-4 5,-4 5]);
title('压缩后样本分布图')
```

15.3.5 压缩近邻法

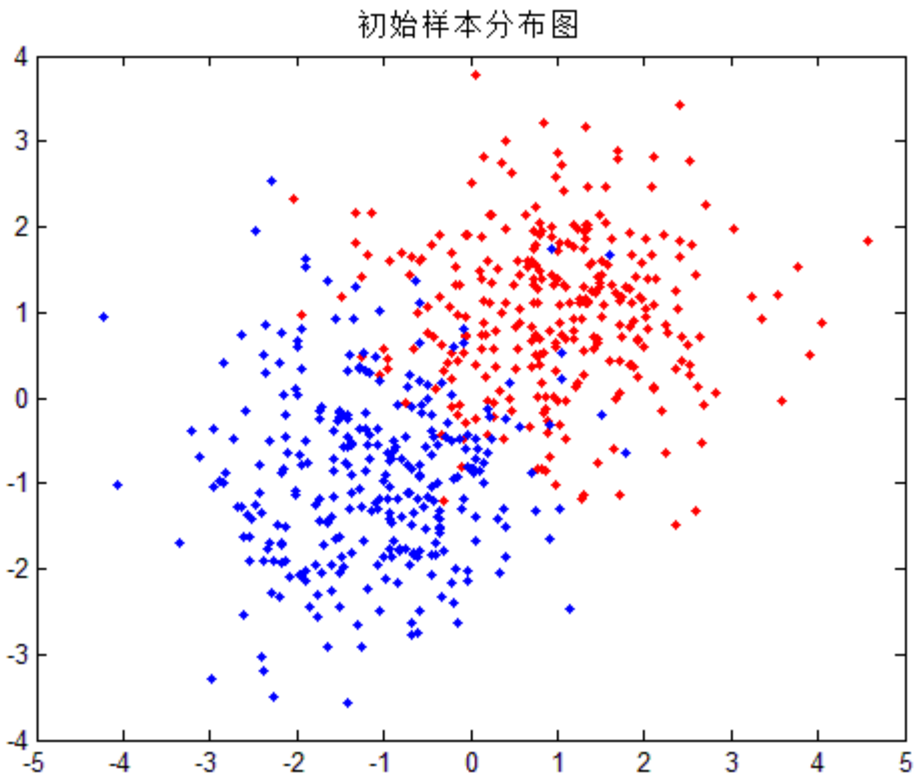
```
    Xcur
    end
    [oldRow,
    [curRow,
    if oldR
        loop
    else
        loop
    end
end
% =========
%把当前样本集
[row,col]=s
Xcur1=[];Xcu
tic
for i=1:row
    if Xcur
        Xcur
    elseif X
        Xcur
    end
end
time1=toc;
figure, plot
hold on,plot
title('剪辑
```

```
% ======
Xstore=Xc
Xgab=Xcur
while 1
    Xolds
    [row,
    j=1;
    while
        [
        i

        e
        end
    end
    [oldRow,col]=s
    [curRow,col]=s
    [gRow,rCol]=si
    if oldRow==cur
        break;
    end
end
```

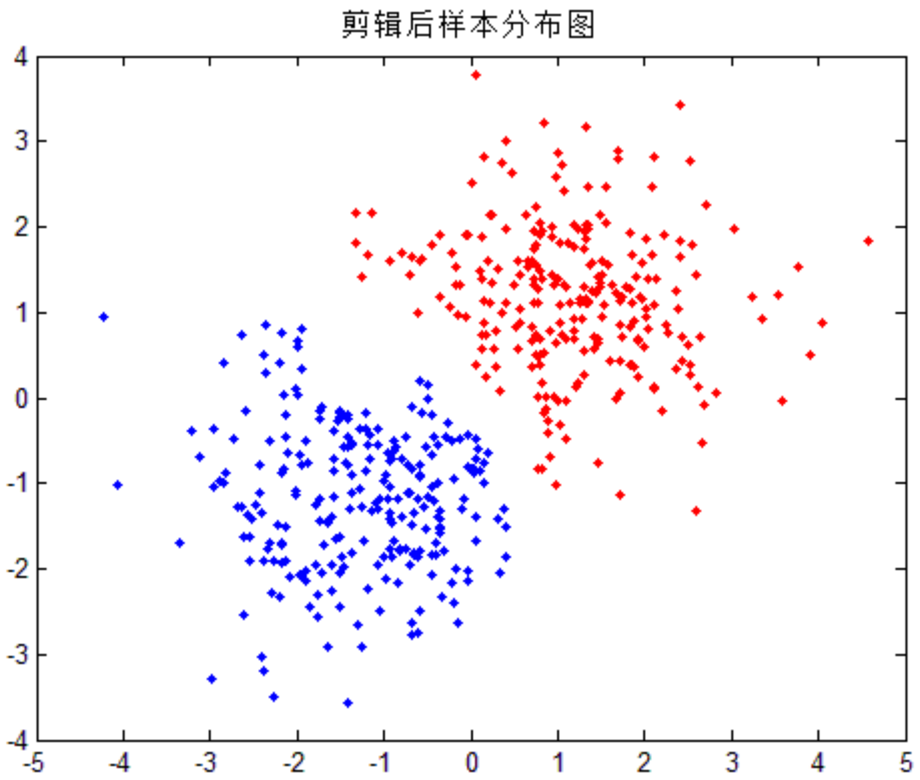
```
Xcurs
[curR
for i
    i

    e

    e
end
figur
hold
axis(
title
```

```
% ----一般近邻算法---
% num:     每类的样本数目
% rClass: 返回值，x在Xr中最近邻的样本类别
% xClass: 返回值，x的样本类别
% =====================================================
function [rClass,xClass]=NNforCondense(Xr,x)

tic
% X = [randn(200,2)+ones(200,2);...
%      randn(200,2)-2*ones(200,2);...
%      randn(200,2)+4*ones(200,2);];
%  x=randn(1,2);%待判样本
[row,col]=size(Xr);
Xdist=zeros(row,1);
for i=1:row
    Xdist(i)=norm(x(1,1:2)-Xr(i,1:2))^2;
end
[Xdist,ind]=sort(Xdist,'ascend');
B=dist(1);
Xnn=Xr(ind(1),:);
rClass=Xnn(1,3);
xClass=x(1,3);
times=toc;
end
```

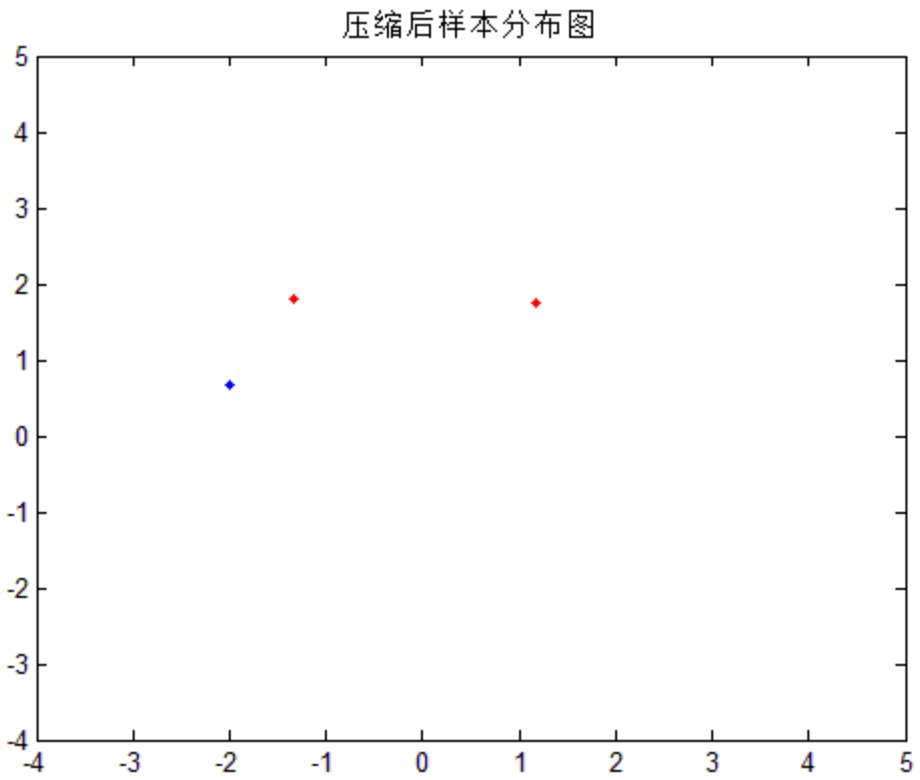
15.3.5 压缩近邻法



15.3.5 压缩近邻法



15.3.5 压缩近邻法



15.3.6 可做拒绝决策的近邻法

由于近邻法决策实际只取决于个别样本，因此有时风险较大，尤其是最近邻法和 k 近邻法当两类近邻数接近时，为此，可考虑引入拒绝决策。

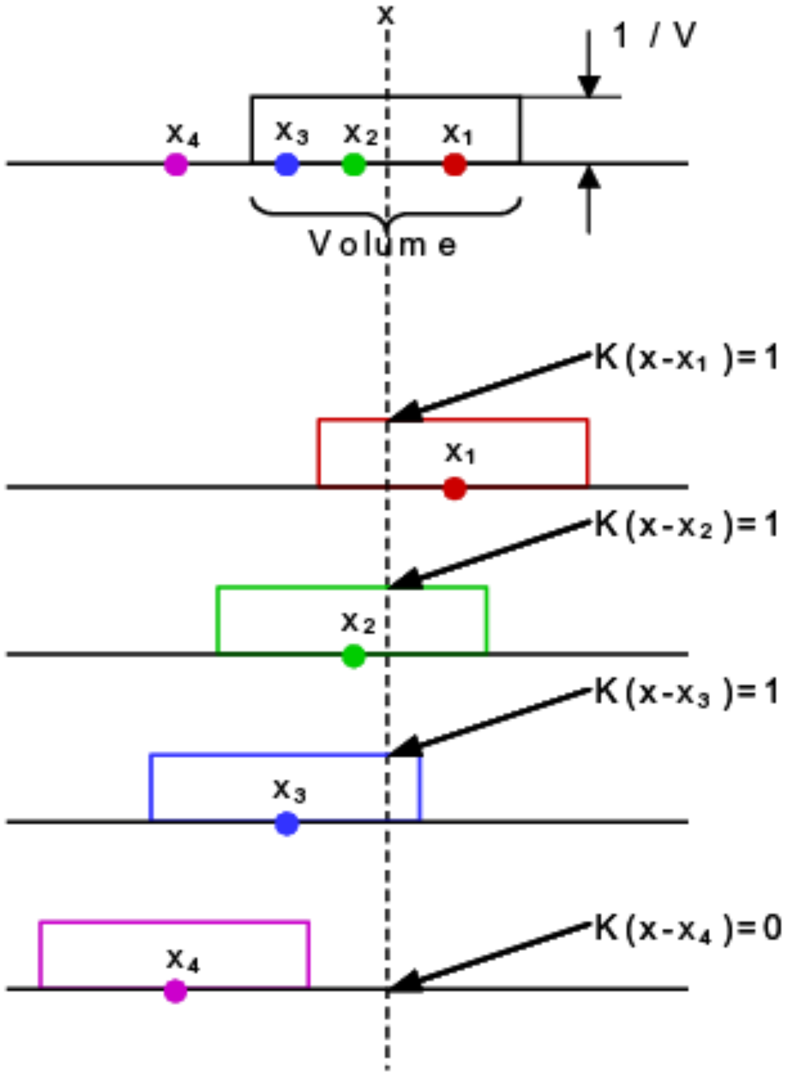
方法很简单： 设某个 $k' > \frac{1}{2}(k+1),\ (k' < k)$

只有当 x的 k个近邻中有大于或等于 k'个属于 ω_i 类时，才决策 $x \in \omega_i$ ，否则拒绝

—— 简单多数 $\Rightarrow$绝对多数

拒绝决策同样可引入改进的近邻法中，比如剪辑近邻法

15.4 Parzen窗法



$$\hat{p}(x)=\frac{1}{N}\sum_{i=1}^{N} k(\boldsymbol{x},\boldsymbol{x}_i)$$

窗函数条件：

$$k(x,x_i)\geq 0$$

$$\int k(x,x_i)dx=1$$

注意到核函数估计和直方图法很相似，但窗的位置是由数据来确定的

15.4 Parzen窗法

常用窗函数：

（1）超立方体窗（方窗）

$$k(x,x_i)=\begin{cases}\dfrac{1}{h^d} & \text{if} \quad \left|x^i-x_i^j\right|\le h/2,\ j=1,2,\cdots,d \\ 0 & \qquad\quad \text{otherwise}\end{cases}$$

h 为超立方体棱长， $V=h^d$

（2）正态窗（高斯窗）

$$k(x,x_i)=\frac{1}{\sqrt{(2\pi)^d\rho^{2d}\,|Q|}}\exp\left\{-\frac{1}{2}\frac{(x-x_i)^TQ^{-1}(x-x_i)}{\rho^2}\right\} \quad (\Sigma=\rho^2Q)$$

一维标准正态： $$k(x,x_i)=\frac{1}{\sqrt{2\pi}}\exp\left\{-\frac{1}{2}(x-x_i)^2\right\}$$

15.4 Parzen窗法

3）超球窗

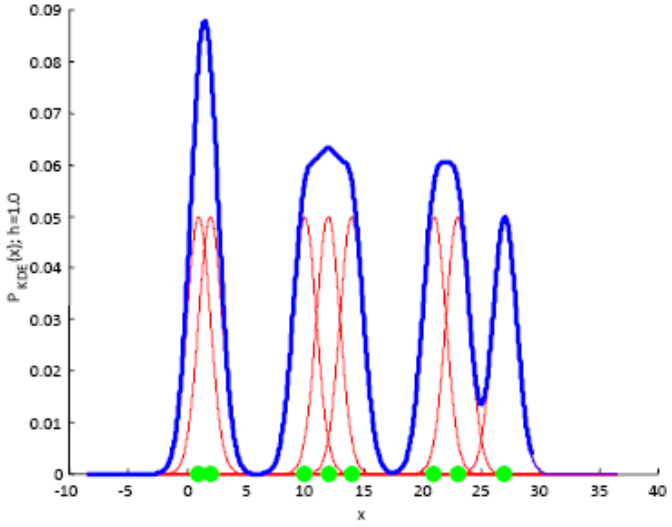
$$k(x,x_i)=\begin{cases} V^{-1} & \text{if} \quad \left\| x-x_i \right\| \le \rho \\ 0 & \text{otherwise} \end{cases}$$

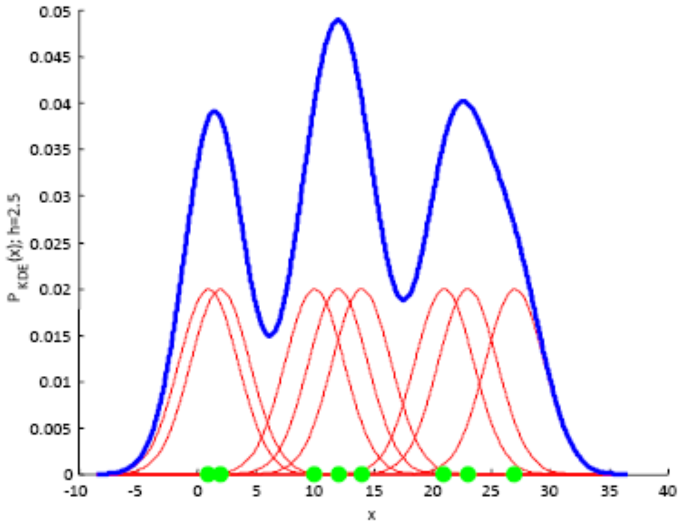
（V ：超球体积，半径 ρ ）

窗宽的选择：

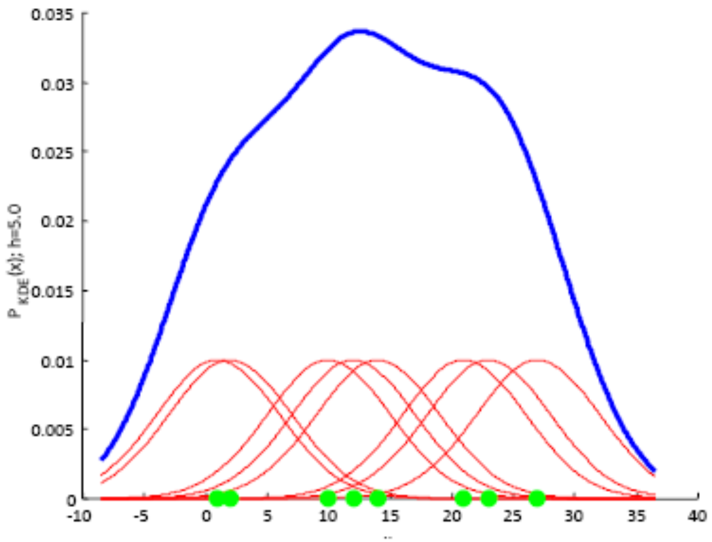
- 样本数少则选大些，样本数多则选小些，

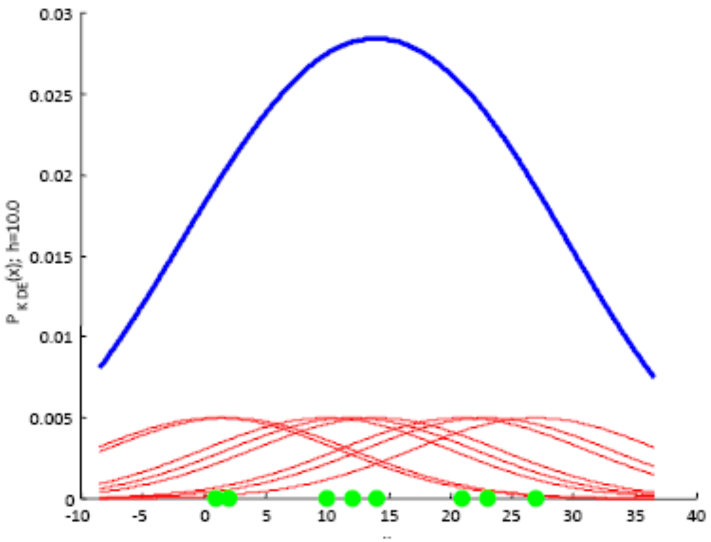
  比如选 $\rho = N^{-\eta/d}$ $\quad \eta \in (0,1)$





15.4 Parzen窗法





- 窗长度ρ对概率密度估计值$p_N(x)$的影响

若ρ太大，$p_N(x)$是$p(x)$的一个平坦、分辨率低的估计，有平均误差

若ρ太小，$p_N(x)$是$p(x)$的一个不稳定的起伏大的估计，有噪声误差

Parzen窗估计的性质：

在满足一定的条件下，估计量 $\hat{p}_N(x)$ 是渐近无偏和平方误差一致的。条件是：

15.4 Parzen窗法

1. 总体密度 $p(x)$ 在x点连续；

2. 窗函数满足以下条件：

$$\varphi(u) \ge 0, \quad \int \varphi(u)du = 1$$ ：窗函数具有密度函数的性质

$$\sup_u \varphi(u) < \infty$$ ：窗函数有界

$$\lim_{\|u\| \to \infty} \varphi(u) \prod_{i=1}^{d} u_i = 0$$ ：窗函数随着距离的增大很快趋于零

3. 窗宽受以下条件约束：

$$\lim_{N \to \infty} V_N = 0$$ ：窗体积随着N的增大而趋于零

$$\lim_{N \to \infty} NV_N = \infty$$ ：但体积减小的速度要低于1/N

$\lim_{N\to\infty} k_N \to \infty$

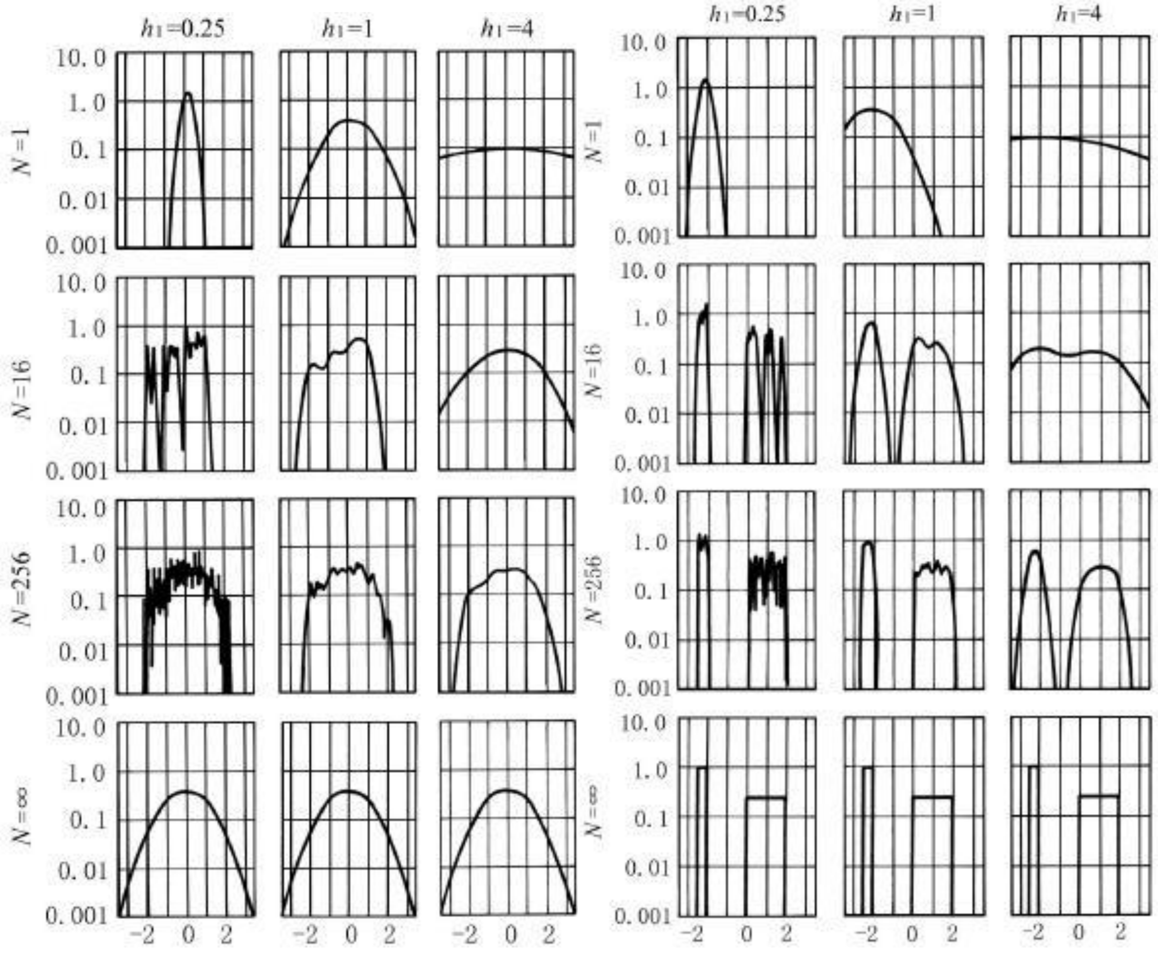
若减小的速度高于 $\frac{1}{N}$

$V > \frac{1}{N}$

15.4 Parzen窗法

举例：

用已知的密度函数产生一系列样本，根据这些样本用Parzen窗法估计概率密度函数，与真实密度函数比较，分析样本数，窗宽等对估计结果的影响。



15. 4 Parzen窗法

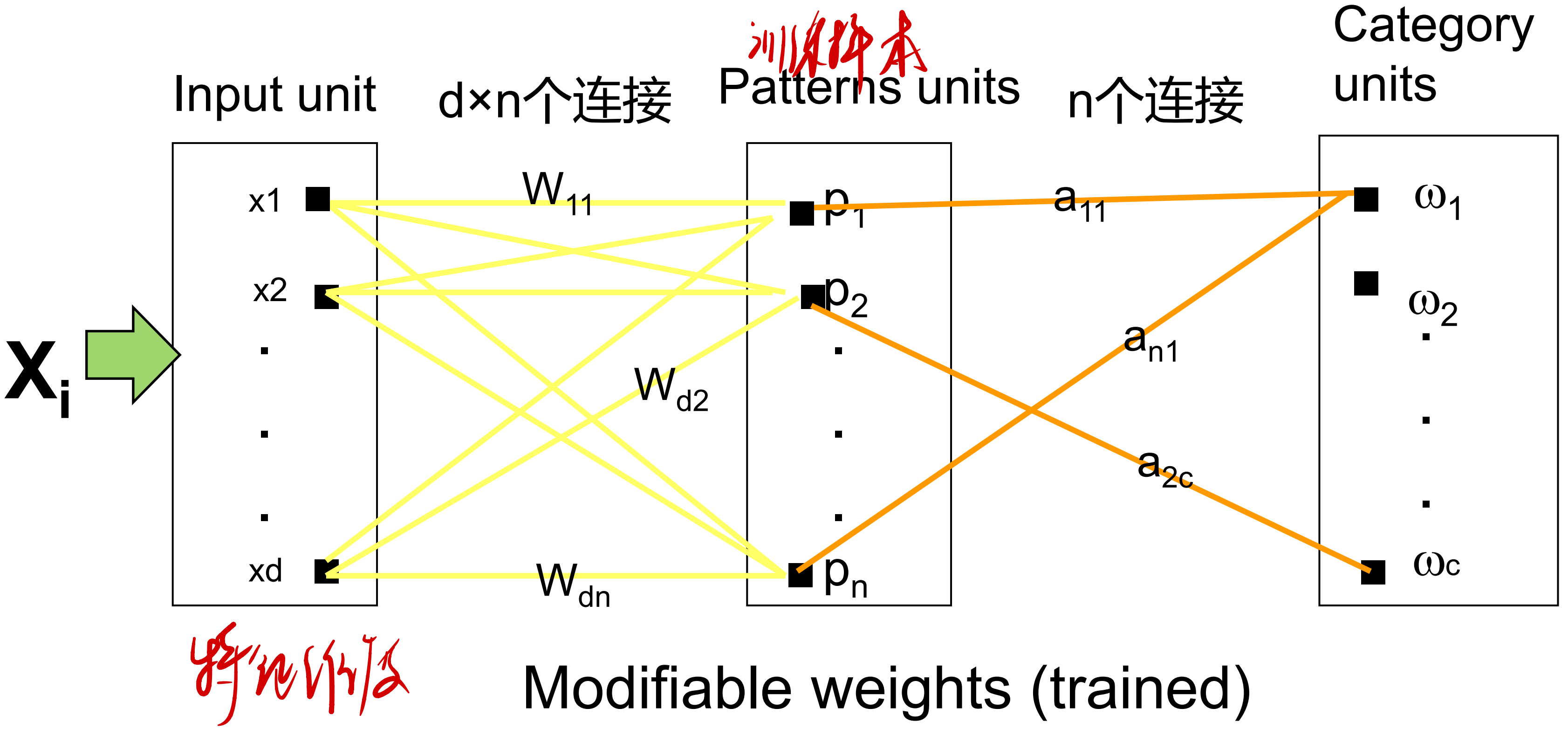
Probabilistic Neural Networks

一种流行的Parzen实现方法

- 概率神经网络（*PNN*）-一种Parzen窗的实现
  - Compute a Parzen estimate based on n patterns
  - Patterns with d features sampled from c classes
  - The input unit is connected to n patterns

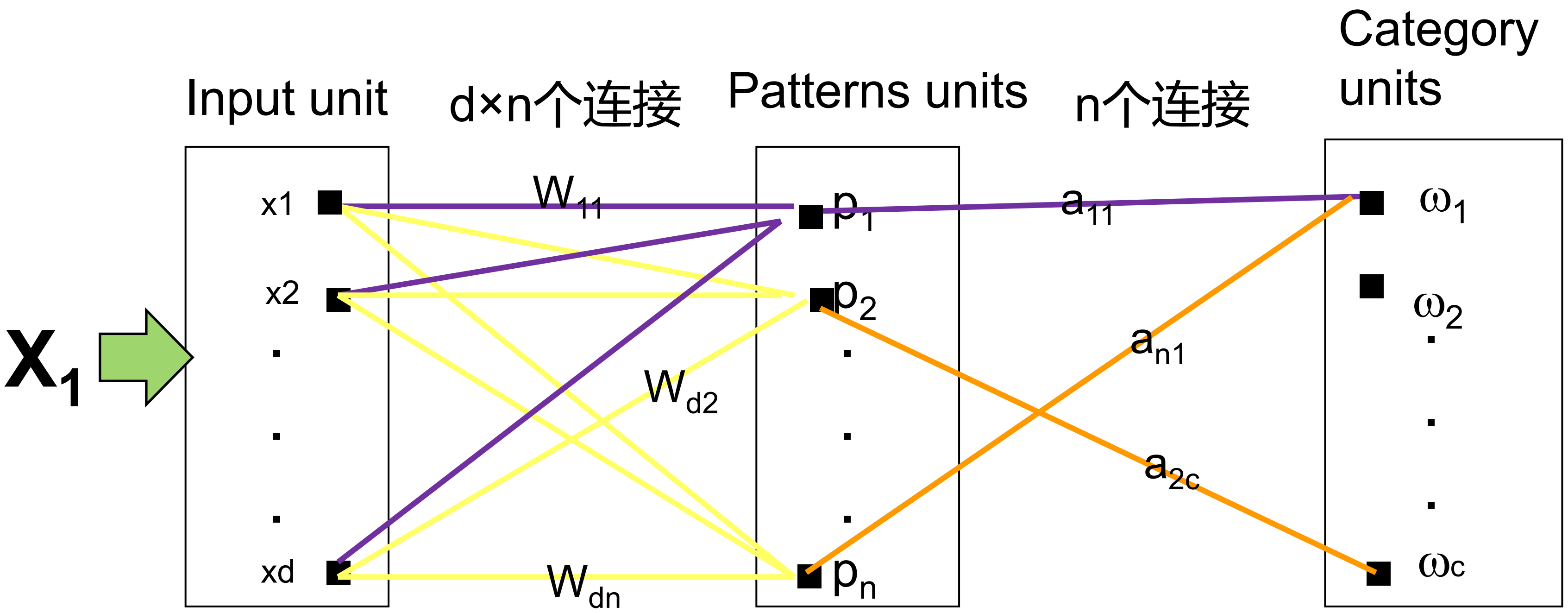
归一化 $x^t x=1$

$x_j \leftarrow \frac{x_j}{\sum_{j=1}^{d} x_j^2}$



Modifiable weights (trained)

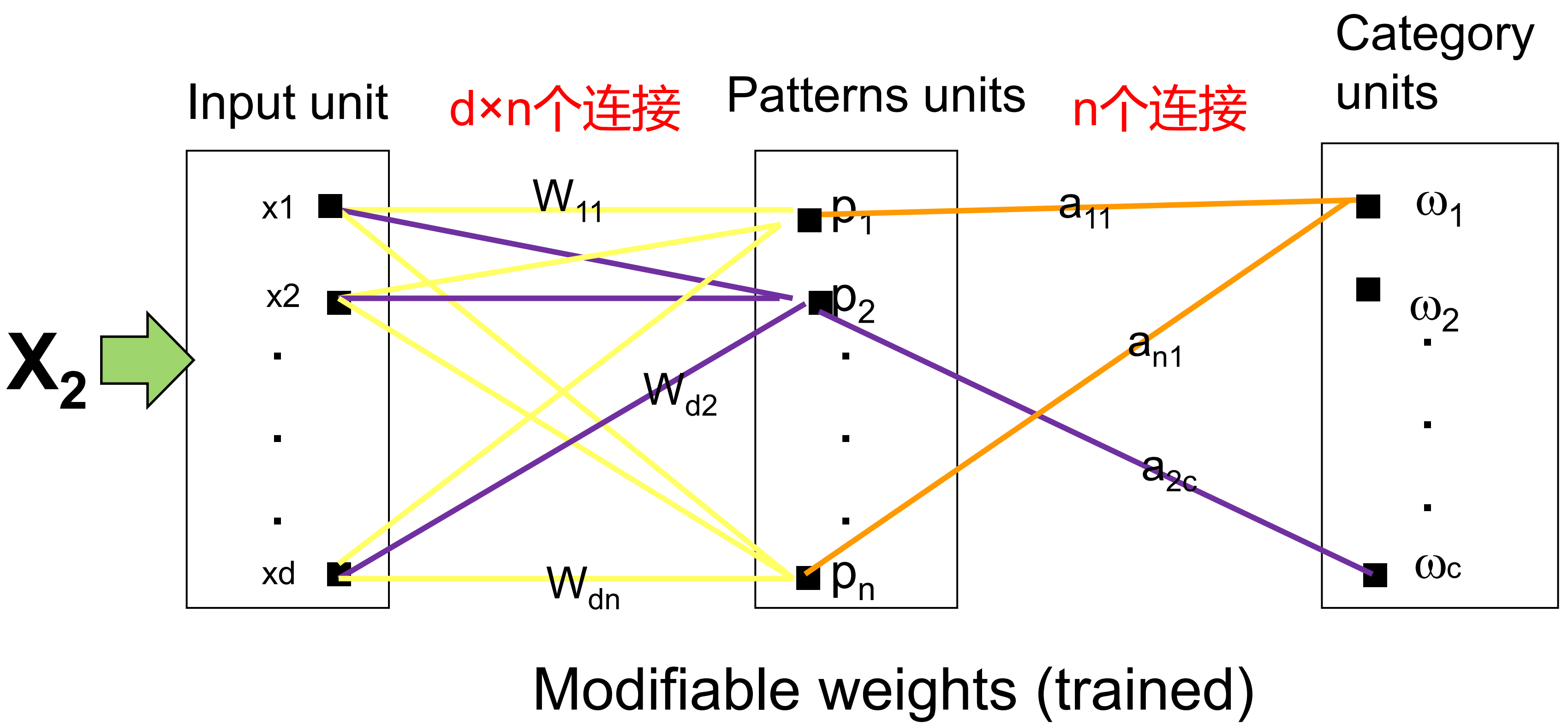
- 概率神经网络（*PNN*）-—种Parzen窗的实现
  - n samples->n patterns
  - d dimensions attribute space
  - c classes



Modifiable weights (trained)

15.4 Parzen窗法

- 概率神经网络（*PNN*）-—种Parzen窗的实现
  - n samples->n patterns
  - d dimensions attribute space
  - c classes



Normalization 先说一下归一化问题

- Patterns are normalized (or scaled) to have unit length, or

$$\sum_{i=1}^{d} x_i^2 = 1$$

- This is done by replacing each feature value by

$$x_j \leftarrow \frac{x_j}{\left(\sum_{i=1}^{d} x_i^2\right)^{1/2}}$$

- Effect of normalization

$$x^t x = 1$$

Normalization Example

Normalize x = [3 4]^t

$(9+16)^{1/2} = 5$

Normalized vector is =[3/5 4/5]^t= [0.6 0.8]^t

Effect of normalization

$$\sum_{i=1}^{d} \mathrm{x}_i^2 = 0.36 + 0.64 = 1$$

$$\mathrm{x}^t\mathrm{x} = \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix} \begin{bmatrix} 0.6 & 0.8 \end{bmatrix} = 1$$

PNN training

Algorithm 1 (PNN training)

1 begin initialize $k \leftarrow 0$, n = # patterns,

$a_{ki} \leftarrow 0$ for k=1,...,N; i=1,...,c; x = test pattern

2 do $k \leftarrow k + 1$

3 $normalize: x_{jk} \leftarrow x_{jk} \Big/ \left(\sum x_{jk}^2\right)^{1/2}, j = 1, \ldots, d$

4 $train: \; w_{jk} \leftarrow x_{jk}$

5 $if \quad x \in \omega_i \quad then \quad a_{ki} \leftarrow 1$

6 $until \;\; k = N$

7 *end*

The a_{ki} correspond to making connections between labeled samples and the corresponding classes

a_{ki}对应于被标示的样本和相应的类之间之间的连接

15.4 Parzen窗法

Activation Function 激活函数

Desired Gaussian

样本

$$\varphi\left(\frac{\mathrm{x}-\mathrm{w}_k}{h_N}\right) \propto e^{-(\mathrm{x}-\mathrm{w}_k)^t(\mathrm{x}-\mathrm{w}_k)/2\sigma^2}$$

窗宽 训练样本.

Contribution of a sample at x
Here we let weights correspond to feature values of sample
样品对x的贡献，在这里，我们让权重对应样本的特征值

$$e^{-(\mathrm{x}-\mathrm{w}_k)^t(\mathrm{x}-\mathrm{w}_k)/2\sigma^2}$$

$$= e^{-\left(\mathrm{x}^t\mathrm{x}+\mathrm{w}_k^t\mathrm{w}_k-2\mathrm{x}^t\mathrm{w}_k\right)/2\sigma^2}$$

$$net_k = \mathrm{w}_k{}^t\mathrm{x}$$

$$= e^{(net_k-1)/\sigma^2}$$

Simplified form due to normalization1

$$\mathrm{x}^t\mathrm{x} = \mathrm{w}_k^t\mathrm{w}_k = 1$$

PNN classication

Algorithm 2 (PNN classification)

begin initialize k = 0; x = test pattern

  do $k \leftarrow k+1$

    $\mathrm{net}_k \leftarrow \mathrm{w}_k^t x$

    if $\mathrm{a}_{ki} = 1$ *then* $g_i \leftarrow g_i + \exp\left[\left(net_k - 1\right)/\sigma^2\right]$

  until $k=N$

return $class \leftarrow \arg\max_i \; g_i(x)$

end

Made possible by appropriate choice of activation function
可以通过选用合适的激活函数

1. 归一化待分类实例*x*;
2. 对每个模式计算内积 $net_k = w_k^t .x$
3. 在有连接的输出层上累加

$$g_i = g_i + \exp\left[\frac{net_k - 1}{\sigma^2}\right]$$

4.最大的响应类别做为最后分类结果

PNN classication

Algorithm 2 (PNN classification)

1 begin initialize k = 0; x = test pattern

2 do $k \leftarrow k+1$

3 $\text{net}_k \leftarrow \text{w}_k^t x$

4 if $\text{a}_{ki} = 1$ *then* $g_i \leftarrow g_i + \exp\left[\left(net_k - 1\right)/\sigma^2\right]$

5 until $k=N$

6 *return* $class \leftarrow \arg\max_i \; g_i\left(x\right)$

7 *end*

Made possible by appropriate choice of activation function
可以通过选用合适的激活函数

Output is the sum of the activation functions g_i corresponding to the labeled samples of that class
每个类上的输出是对应于标记为该类别的样本的激活函数g_i的总和,选最大的**和值**对应的类作为判决结果.

Producing the classifier

$$h(c) = \arg\max_{c} P(\mathrm{c}|\mathrm{x}) = \arg\max_{c} \frac{p(\mathrm{x}|\mathrm{c})P(\mathrm{c})}{p(\mathrm{x})}$$

$$= \arg\max_{c} p(\mathrm{x}|\mathrm{c})P(\mathrm{c}) = \arg\max_{c} \frac{k_{\mathrm{c}}(bin_{\mathrm{x}})}{N_{\mathrm{c}}V(bin_{\mathrm{x}})} \underbrace{\frac{N_c}{N}}_{p(c)}$$

$$= \arg\max_{c} \frac{k_{\mathrm{c}}(bin_{\mathrm{x}})}{NV(bin_{\mathrm{x}})} = \arg\max_{c} k_{\mathrm{c}}(bin_{\mathrm{x}})$$

与k有关，与kc无关.

分类器 $h(c)$ = 使得后验概率 $P(\mathrm{c}|\mathrm{x})$ 最大的那个类

根据贝叶斯规则计算后验概率

$$P(\mathrm{c}|\mathrm{x}) = \frac{\text{类概率密度函数}p(\mathrm{x}|\mathrm{c}) \times \text{类c的先验概率}P(\mathrm{c})}{\text{x的概率密度}p(\mathrm{x})}$$

Producing the classifier

$$h(c) = \arg\max_{c} P(\mathrm{c}|\mathrm{x}) = \arg\max_{c} \frac{p(\mathrm{x}|\mathrm{c})P(\mathrm{c})}{p(\mathrm{x})}$$

$$= \arg\max_{c} p(\mathrm{x}|\mathrm{c})P(\mathrm{c}) = \arg\max_{c} \frac{H_c(\mathrm{x})}{N_{\mathrm{c}}V}\frac{N_c}{N}$$

$$= \arg\max_{c} \frac{H_c(\mathrm{x})}{NV(bin_{\mathrm{x}})} = \arg\max_{c} H_c(\mathrm{x})$$

$$= \arg\max_{c} \sum_{j=1}^{N} \varphi\left(\frac{\mathrm{x}-\mathrm{x}_j \mid \mathrm{x}_j \in c}{h_N}\right) = \arg\max_{c} g_{\mathrm{c}}(\mathrm{x})$$

分类器 $h(c)=$ 使得后验概率$P(\mathrm{c}|\mathrm{x})$最大的那个类

=使得激励函数累加值最大=判决函数g(x|c)最大

思考

PNN网络的缺陷?

Normalization
归一化

- Patterns are normalized (or scaled) to have unit length, or

$$\sum_{i=1}^{d} x_i^2 = 1$$

- This is done by replacing each feature value by

$$x_j \leftarrow \frac{x_j}{\left(\sum_{i=1}^{d} x_i^2\right)^{1/2}}$$

- Effect of normalization

$$x^T x = 1$$

Activation Function 激活函数

Desired Gaussian

$$\varphi\left(\frac{\mathrm{x}-\mathrm{w}_k}{h_N}\right) \propto e^{-(\mathrm{x}-\mathrm{w}_k)^T(\mathrm{x}-\mathrm{w}_k)/2\sigma^2}$$

Contribution of a sample at x
Here we let weights correspond to feature values of sample
样本对x的贡献，在这里，我们让权重对应样本的特征值

$$e^{-(\mathrm{x}-\mathrm{w}_k)^T(\mathrm{x}-\mathrm{w}_k)/2\sigma^2}$$

$$= e^{-\left(\mathrm{x}^T\mathrm{x}+\mathrm{w}_k^T\mathrm{w}_k-2\mathrm{x}^T\mathrm{w}_k\right)/2\sigma^2}$$

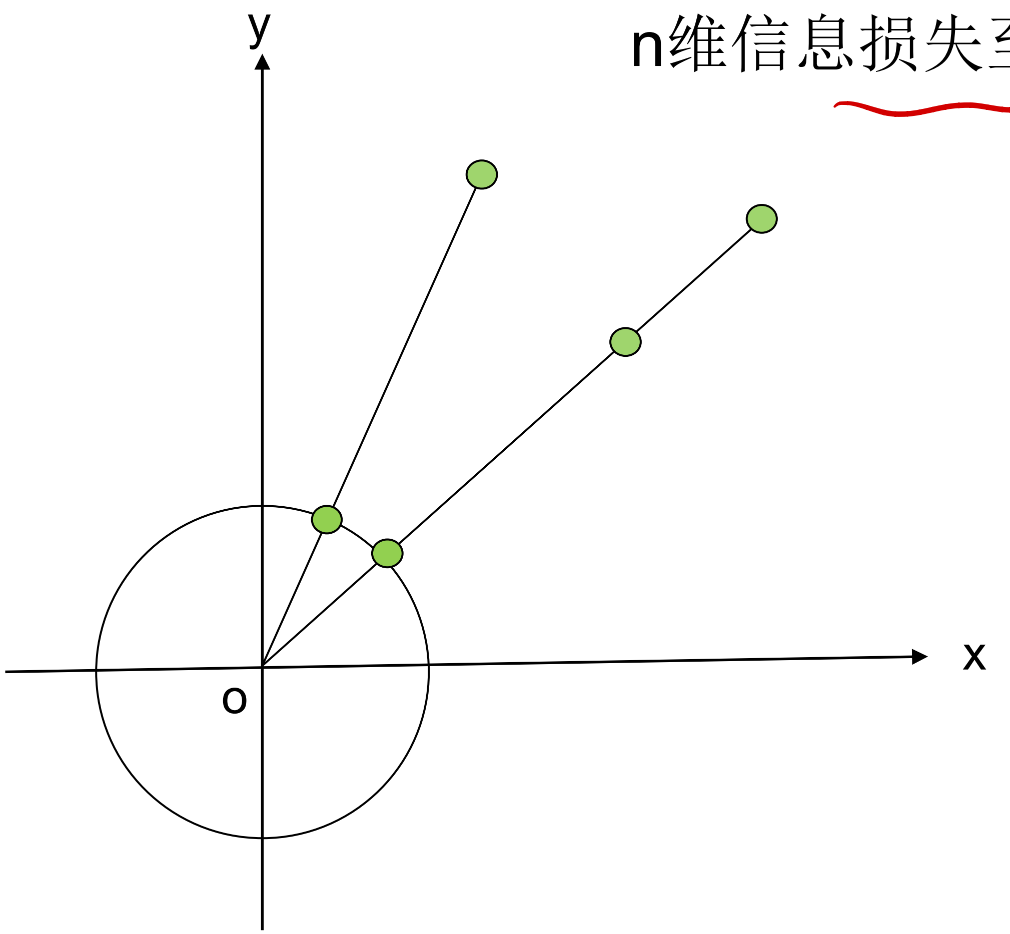
$$net_k = \mathrm{w}_k{}^T\mathrm{x}$$

$$= e^{(net_k-1)/\sigma^2}$$

Simplified form due to normalization1

$$\mathrm{x}^T\mathrm{x} = \mathrm{w}_k^T\mathrm{w}_k = 1$$

n维信息损失至(n-1)维信息



Activation Function 激活函数

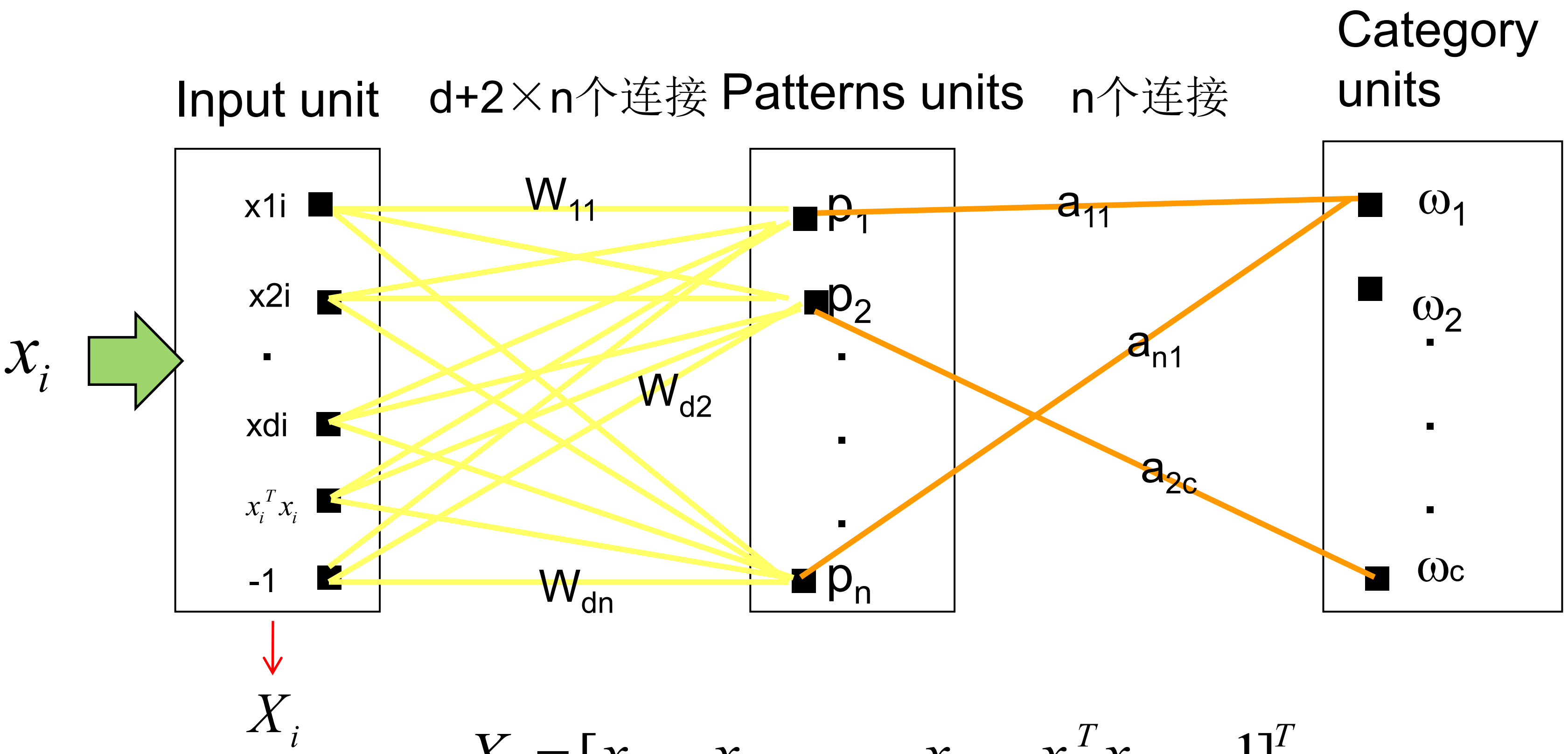
$$\varphi\left(\frac{x-w_k}{h_N}\right) \propto e^{-(x-w_k)^T(x-w_k)/2\sigma^2}$$

$$= e^{(2x^T w_k - x^T x - w_k{}^T w_k)/2\sigma^2}$$

$$= e^{(2x^T x_k - x^T x - x_k{}^T x_k)/2\sigma^2}$$

$$w_k{}^T X = 2x^T x_k - x^T x - x_k{}^T x_k$$

Probabilistic Neural Networks



$$X_i = [x_{1i} \quad x_{2i} \quad \ldots \quad x_{di} \quad x_i^T x_i \quad -1]^T$$

Modifiable weights (trained)

PNN training

Algorithm 1 (PNN training)

$$
\begin{array}{l}
\text{begin initialize } k \leftarrow 0,\ \ \text{n} = \#\text{ patterns}, \\
\qquad \text{a}_{ki} \leftarrow 0 \ \ \text{for } k=1,\ldots,N;\ \ i=1,\ldots,c;\ \ \text{x} = \text{test pattern} \\
\qquad \text{do } \ k \leftarrow k+1 \\
\qquad \mathit{train}: \\
\qquad\quad \mathit{if}\ \ j \le d \qquad \mathit{then}\ w_{jk} \leftarrow 2 \times x_{jk} \\
\qquad\quad \mathit{if}\ \ j = d+1 \ \ \mathit{then}\ w_{jk} \leftarrow x_k^T x_k \\
\qquad\quad \mathit{if}\ \ j = d+2 \ \ \mathit{then}\ w_{jk} \leftarrow \ -1 \\
\qquad \mathit{if}\ \ x_k \in \omega_i \ \ \mathit{then}\ \ a_{ki} \leftarrow 1 \\
\qquad \mathit{until}\ \ k = N \\
\mathit{end}
\end{array}
$$

PNN classication 用PNN分类

$$w_k = [2\times x_{1k} \quad 2\times x_{2k} \quad \dots \quad 2\times x_{dk} \quad x_k{}^T x_k \quad -1]^T$$

$$X = [x_1 \quad x_2 \quad \dots \quad x_d \quad -1 \quad x^T x]^T$$

$$w_k{}^T X = 2x^T x_k - x^T x - x_k{}^T x_k$$

Algorithm 2 (PNN classification)

1 begin initialize k = 0; x = test pattern

2 do $k \leftarrow k+1$

3 $\mathrm{net}_k \leftarrow \mathrm{w}_k^t x$

4 if $\mathrm{a}_{ki} = 1$ *then* $g_i \leftarrow g_i + \exp\left[net_k / 2\sigma^2\right]$

5 until $k=N$

6 *return* $class \leftarrow \arg\max_i \; g_i(x)$

7 *end*

Experimental result 实验结果

Data1:二维正态分布 mean = [0,0]; SIGMA = [1 0;0 1]; num = 7000;

Data2:二维正态分布 mean = [200,0]; SIGMA = [1 0;0 1]; num = 7000;

待测样本均值	[0,0]	[50,0]	[100,0]	[150,0]	[200,0]
Matlab自带pnn归一化法正确率	50.48%	0%	50.56%	100%	100%
本方法正确率	100%	100%	100%	100%	100%

测试样本集均为5000个点，改进方法同matlab自带pnn对比得到正确率

ending