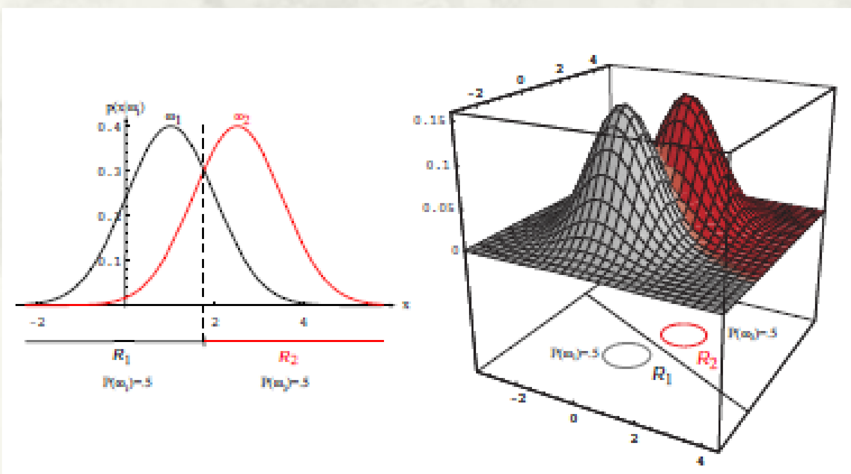
机器学习
Machine Learning

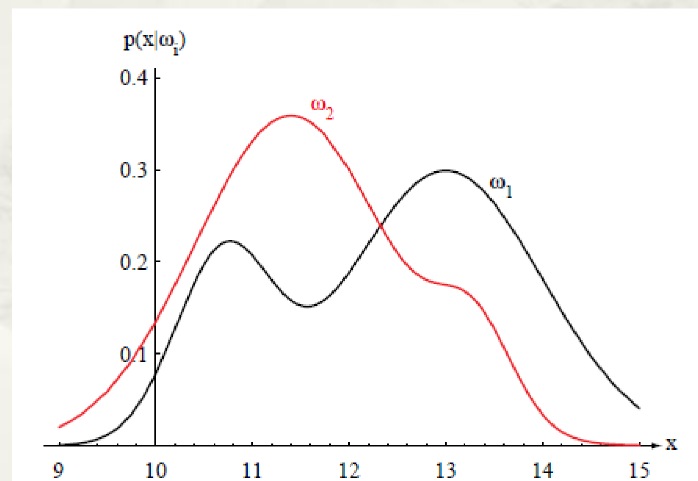
（四）非参数学习

何劲松
中国科学技术大学

6. 非参数估计方法

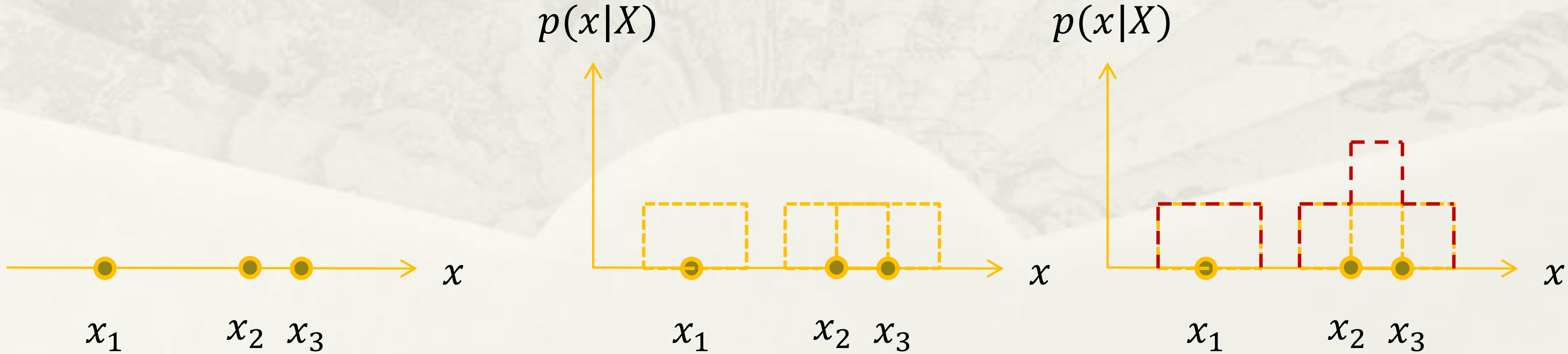
- 参数估计方法的前提是假设类条件概率密度函数$p(x|\omega_i)$的形式已经知道。
- 在多数模式分类问题中，往往不知道$p(x|\omega_i)$的形式，常见的函数形式并不适合实际的真正密度分布。
- 或者说，经典的参数估计大多适合用于平滑变化和单峰突出的密度分布，只有一个极大值。实际概率分布却是多峰的。





6.1 非参数估计的基本概念

- 非参数估计的任务是利用抽样试验的训练集 X估计样本x的总体概率分布密度$p(x|X)$。如果训练集X 中的样本数目n足够大，那么$p(x|X)$应该非常接近真实的分布密度$p(x)$ 。
- 完成这种训练的基本思想是：每个样本$x_k(k=1,2,\ldots,n)$对总体概率分布密度$p(x)$都有一定贡献。把n个训练样本的贡献叠加起来，就得到总体分布密度估计。$\hat{p}(x)=p(x|X)$
- 样本数目n越多，估计的曲线越平滑，也越接近真实的概率分布曲线。
- 例如矩形窗函数：（也可以是内插函数  ）



6.2 非参数估计方法

令样本x落入区域$\mathcal{R}$里的概率p为

$$p = \int_{\mathcal{R}} p(x)dx \qquad (1)$$

当彼此独立地抽取n个试验，得到n 个训练样本$x_1, x_2, \dots, x_n$，它们分别以$p(x_1), p(x_2), \dots, p(x_n)$出现。

其中有k个样本落入这个区域$\mathcal{R}$里的概率服从二项式分布。即

$$p_k = \binom{n}{k} p^k (1-p)^{n-k} \qquad (2)$$

式中

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!} \qquad (3)$$

由于x_k是随机抽取的，所以落入区域$\mathcal{R}$里的数目k也是随机的。它的期望值

$$E(k) = np \qquad (4)$$

所以p的一个估计为

$$\hat{p} \approx \frac{k}{n} \qquad (5)$$

如果区域$\mathcal{R}$足够小，以至于概率密度$p(x)$在$\mathcal{R}$里可以近似地认为恒定不变。那么由(1)式可得

$$p = \int_{\mathcal{R}} p(x)dx \approx p(x) \cdot V \tag{6}$$

其中V为区域$\mathcal{R}$所占有的空间体积。

利用(5)和(6)，可有

$$\hat{p} = \hat{p}(x)\,V \approx \frac{k}{n} \tag{7}$$

所以

$$\hat{p}(x) \approx \frac{k/n}{V} \tag{8}$$

对于(8)式，

A. 若区域$\mathcal{R}$的体积V是固定的，训练样本n越来越多，且假定k/n随n的增大而收敛，那么我们就只能得到$p(x)$在$\mathcal{R}$上的一个平均估计，即

$$\frac{\hat{p}}{V} = \frac{\int_{\mathcal{R}} p(x)dx}{\int_{\mathcal{R}} dx}$$

B.若训练样本数目n是固定的，使$\mathcal{R}$的区域不断缩小，即，令趋于0，
就会发生2种情况：

① $\mathcal{R}$中不再有任何样本；

②碰巧$\mathcal{R}$里有1个或n个重合的样本。

这两种情况分别对应$\hat{p}(x)=0$和$\hat{p}(x)=\infty$，显然应该避免。为此要加限制条件。即

<1> $\lim\limits_{n\to\infty} V_n = 0$

<2> $\lim\limits_{n\to\infty} k_n = \infty$

<3> $\lim\limits_{n\to\infty} \left(\frac{k_n}{n}\right) = 0$

满足上述三个条件的区域序列的选择方法，存在2个基本技术途径：

其一：把包含x点的区域序列$\{\mathcal{R}_n\}$选为训练样本数目n的函数，并且对应空间体积V_n随n增大而减小。

例如，令$V_n = \frac{V_1}{\sqrt{n}}$

⇒ Parzen窗函数法的思想

其二，把k_n选为训练样本数目n的函数。

例如，令$k_n = \sqrt{n}$

然后选择包含x点的区域体积，使之不断增大，直到正好包含k_n个样本。那么该区域体积可用作x点的密度估计。

⇒ k_n近邻法的思想

6.3 Parzen窗法

假定围绕x点的区域$\mathcal{R}_n$为一超立方体，其边长为h_n，d为特征空间的维数。则

$$V_n = {h_n}^d \tag{9}$$

为考查x_k是否落入超立方体内，则要检查向量$x - x_k$的每一个分量是否小于$^{h_n}/_2$。
定义窗函数

$$\phi(u) = \begin{cases} 1, & |u_j| \le \frac{1}{2}, j = 1,2, \ldots, d \\ 0, & 其它 \end{cases} \tag{10}$$

可见，$\phi(u)$的体积为1.
若令

$$u = \frac{x - x_k}{h_n} \tag{11}$$

则窗函数变为

$$\phi\left(\frac{x - x_k}{h_n}\right) = \begin{cases} 1, & |x - x_k|_j \le \frac{1}{2}, j = 1,2, \ldots, d \\ 0, & 其它 \end{cases} \tag{12}$$

显然

$$k_n = \sum_{k=1}^{n} \phi\left(\frac{x - x_k}{h_n}\right) \tag{13}$$

于是

$$\hat{p}_n(x) = \frac{1}{n}\sum_{k=1}^{n}\frac{1}{V_n}\phi\left(\frac{x - x_k}{h_n}\right) \tag{14}$$

且满足

$$\phi(u) \geq 0,\ \int \phi(u)du = 1$$

以及

$$\int \hat{p}_n(x)dx = \int \frac{1}{n}\sum_{k=1}^{n}\frac{1}{V_n}\phi\left(\frac{x - x_k}{h_n}\right)dx = \frac{1}{n}\sum_{k=1}^{n}\int \phi(u)du = \frac{1}{n}\cdot n = 1$$

(14)式即为Parzen窗估计方法。

注意：h_n对$\hat{p}(x)$有重大影响。

6.4 窗函数的选择

(1)矩形窗函数

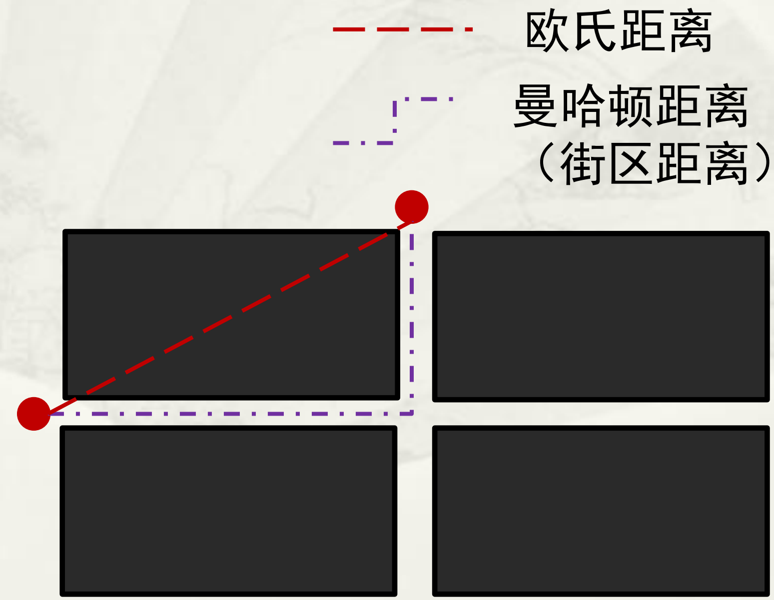
$$\phi(u)=\begin{cases}1, & |u_j|\le\frac{1}{2}, j=1,2,\ldots,d\\ 0, & \text{其它}\end{cases}$$

(2)正态窗函数

$$\phi(u)=\frac{1}{\sqrt{2\pi}}exp\left\{-\frac{1}{2}u^2\right\}$$

(3)指数窗函数

$$\phi(u)=exp\{-|u|\}$$

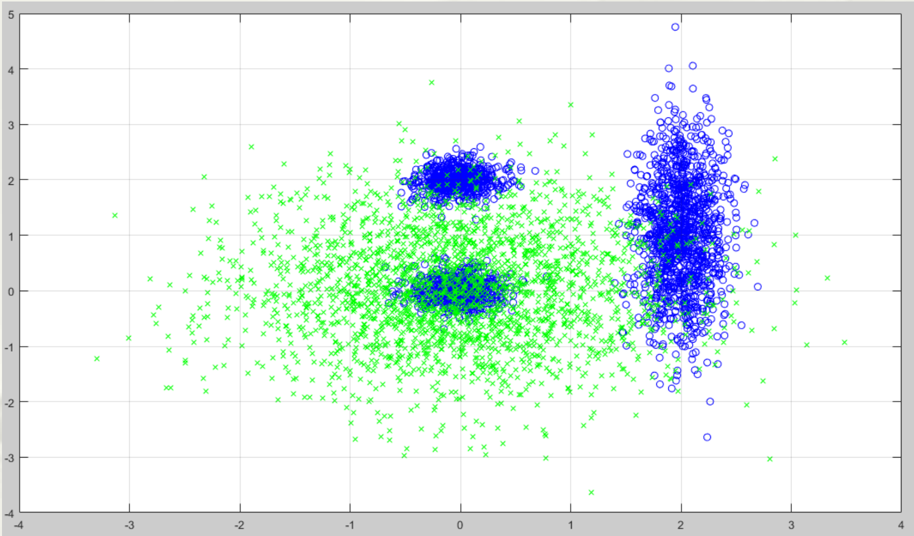


6.5 Parzen窗函数法程序实现

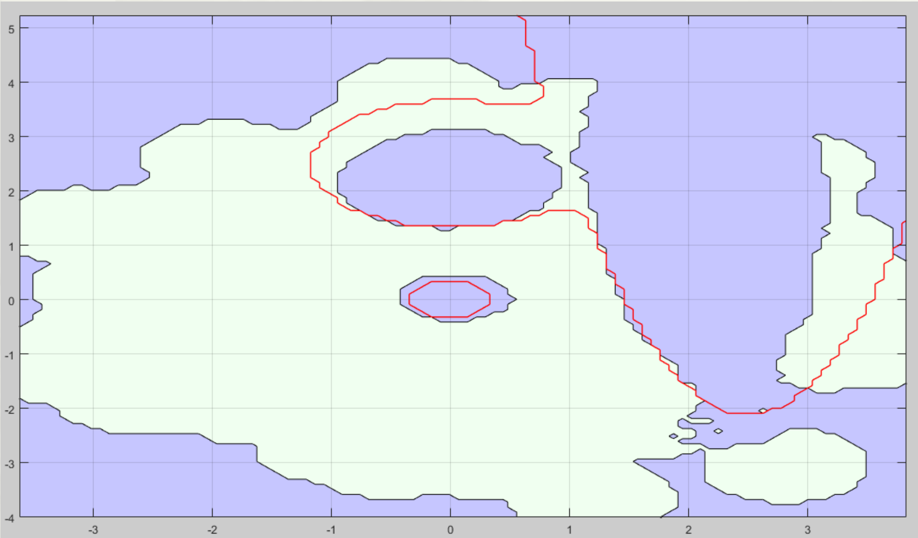
```
function D = parzen(train_features, train_targets, hn, region)
% Classify using the Parzen windows algorithm
% Inputs:
%           features        - Train features
%           targets         - Train targets
%           hn              - Normalizing factor for h
%           region          - Decision region vector: [-x x -y y number_of_points]
% Outputs
%           D               - Decision sufrace

N                           = region(5);     %Number of points on the grid
x                           = ones(N,1) * linspace (region(1),region(2),N);
y                           = linspace (region(3),region(4),N)' * ones(1,N);
Uc     = unique(train_targets);
V                           = zeros(length(Uc), N, N);
x_i     = train_features;
for j = 1:length(Uc),
    indices = find(train_targets == Uc(j));
    P(j)    = length(indices)/size(x_i,2);
    n                       = length(indices);
    for i = 1:n,
        temp        = (x - x_i(1,indices(i))).^2 + (y - train_features(2,indices(i))).^2;
        V(j,:,:)    = squeeze(V(j,:,:)) + phi(temp./hn);
        if (i/50 == floor(i/50)),
            disp(['Finished ' num2str(i) ' iterations out of ' num2str(n) ' iterations.'])
        end
    end
        V(j,:,:) = V(j,:,:) / sum(sum(squeeze(V(j,:,:))));
end
D = (squeeze(V(1,:,:))*P(1) < squeeze(V(2,:,:))*P(2));
%END Parzen
```

```
function p = phi(val)
%The window function for the Parzen window
p = (abs(val) <= 0.5);
```

Parzen窗法的决策面



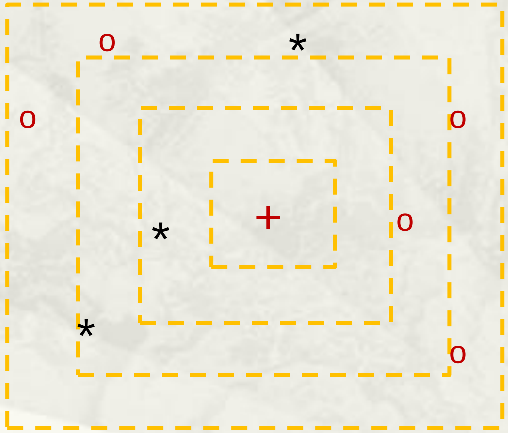
数据

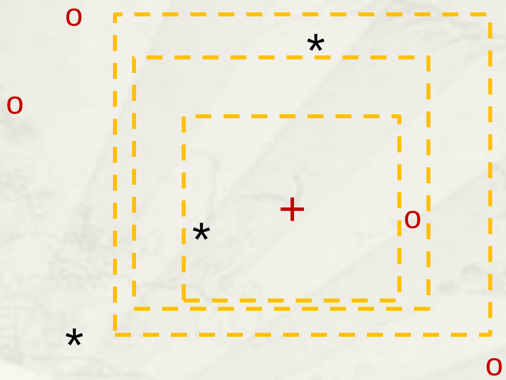


决策面
（图中红色线条为Bayes决策面）

6.6 Kn近邻法

- Parzen窗估计中存在一个具体问题是体积序列$V_1, V_2,\ldots, V_n$的选择问题。
- k_n近邻法的基本思想是使体积为数据的函数，而不是样本数目n的函数。





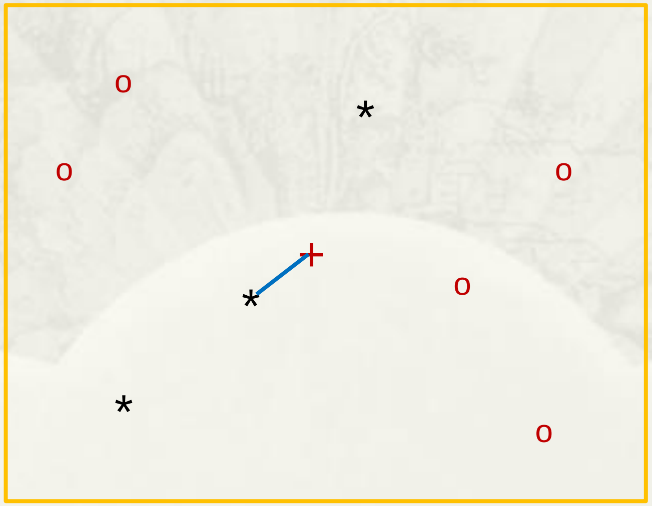
6.7 最近邻决策规则

- 最初的近邻法由Cover和Hart于1968年提出。至今仍是模式分类非参数法中最重要的方法之一。

假定有C个模式类别$\omega_1, \omega_2, \ldots, \omega_i, \ldots, \omega_C$。训练样本集有$N$个样本，每个样本记做$x_j, j = 1,2, \ldots, N$。若

$$d(x, x_k) = \min_j \left[d(x, x_j) \right]$$

且$x_k \in \omega_j$，则判$x \in \omega_j$



6.8 *k*NN决策规则

假定有C个模式类别$\omega_1, \omega_2, \ldots, \omega_i, \ldots, \omega_C$。训练样本集有$N$个样本，每个样本记做$x_j, j = 1,2, \ldots, N$。

记未知样本为x。计算x与所有样本的距离，得到x与已知样本集合中样本的距离集合

$$D = \{d(x, x_k) | k = 1,2, \ldots, N\}$$

或，记做

$$D = \{d_1, d_2, \ldots, d_N\}$$

将D按从小到大排序，取前k个最小元素及对应样本的类别标签，构成集合$\Omega(k)$。若在$\Omega(k)$中出现最多的类别为ω_j，则判

$$x \in \omega_j$$

6.9 KNN算法实现

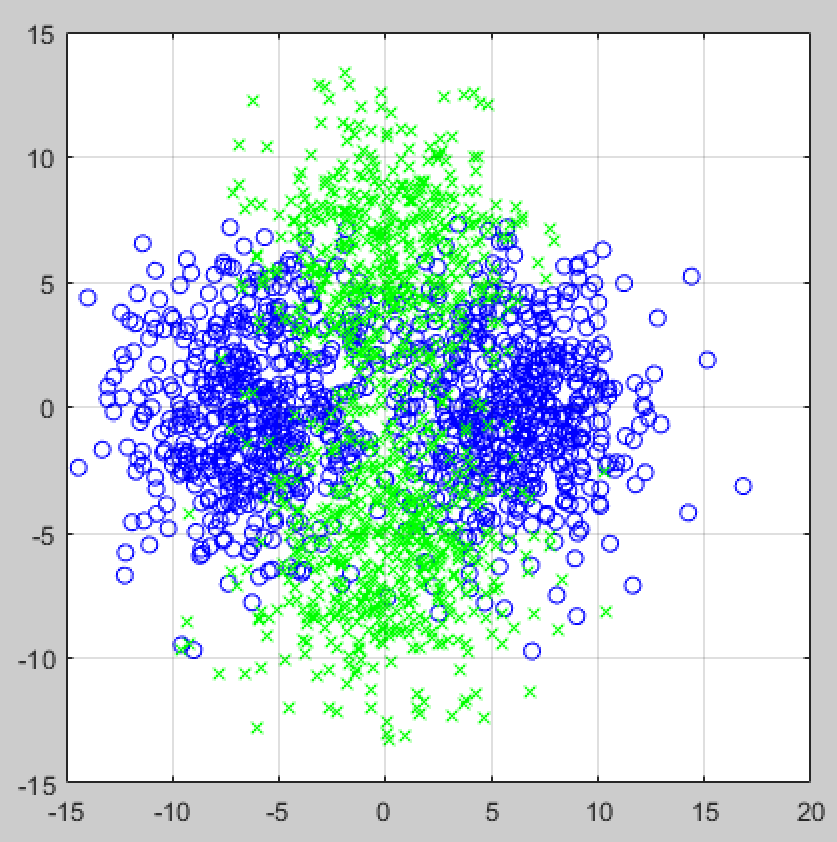
```
function D = Nearest_Neighbor(train_features, train_targets, Knn, region)

% Classify using the Nearest neighbor algorithm
% Inputs:
%   features    - Train features
%   targets - Train targets
%   Knn     - Number of nearest neighbors
%   region  - Decision region vector: [-x x -y y number_of_points]
%
% Outputs
%   D           - Decision suface

L           = length(train_targets);
N           = region(5);
x           = linspace (region(1),region(2),N);
y           = linspace (region(3),region(4),N);

D           = zeros(N);

if (L < Knn),
   error('You specified more neighbors than there are points.')
end

y_dist  = (ones(N,1) * train_features(2,:) - y'*ones(1,L)).^2;

for i = 1:N,
    if (i/50 == floor(i/50)),
      disp(['Finished ' num2str(i) ' lines out of ' num2str(N) ' lines.'])
   end

   x_dist = ones(N,1)  * (train_features(1,:)-x(i)).^2;
    dist   = abs(x_dist + y_dist);
   [sorted_dist, indices] = sort(dist');
   k_nearest = train_targets(indices(1:Knn,:));
   if (Knn > 1),
      D(:,i)     = (sum(k_nearest) > Knn/2)';
   else
      D(:,i)      = (k_nearest > 0)';
   end

end
```

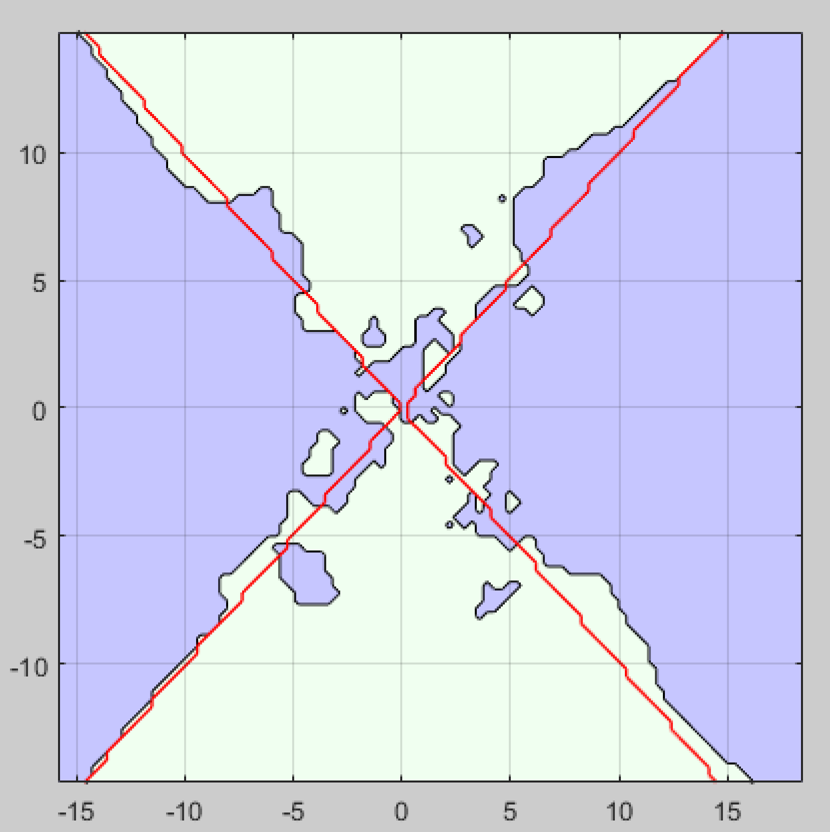
kNN决策规则的错误率

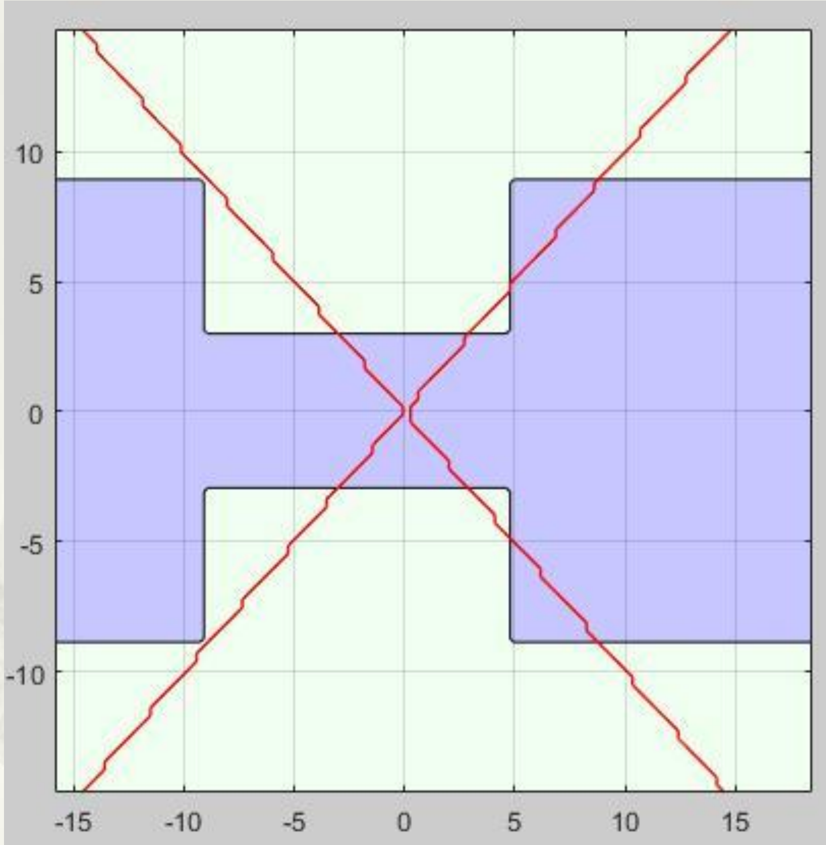
$$P^* \le P_e \le P^*\left(2 - \frac{C}{C-1}P^*\right)$$

其中，P^*表示Bayes错误率，C表示类别数目。

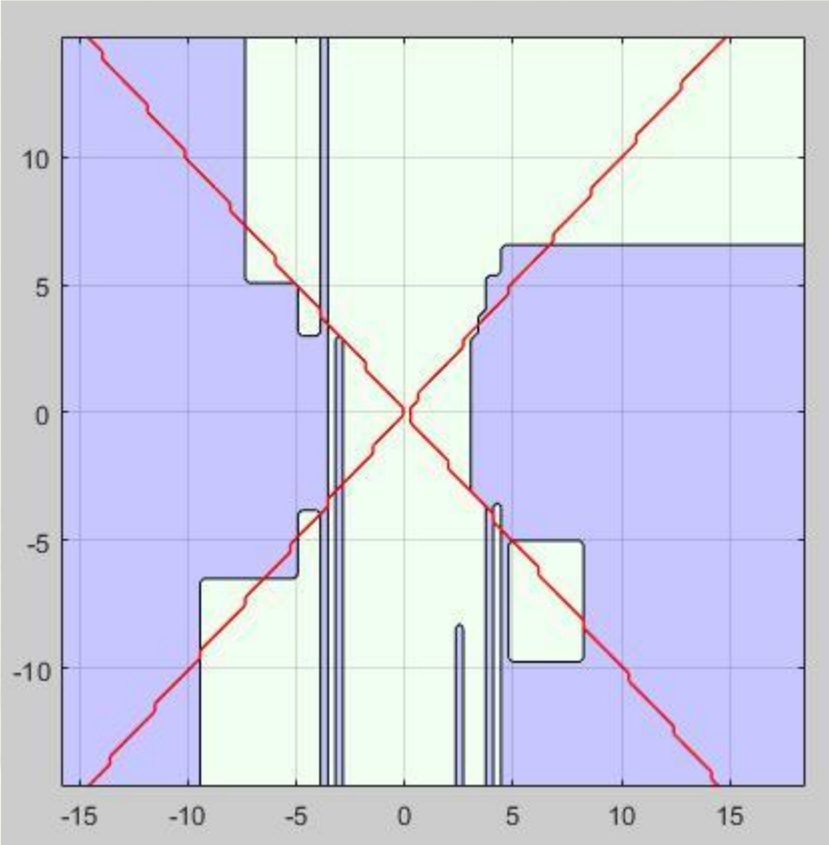
也可粗略地表示为$P^* \le P_e \le 2\,P^*$







ID3



C4.5

6.10 剪辑近邻法

设N个样本分成C类，并用集合$\mathbb{X}^N=\left\{\mathbb{X}_1{}^{N_1},\mathbb{X}_2{}^{N_2},\ldots,\mathbb{X}_C{}^{N_C}\right\}$表示，其中每一类表示为$\mathbb{X}_i{}^{N_i}=\{x_i{}^k\}$，$i=1,2,\ldots C$，$k=1,2,\ldots,N_i$。

方法的第一步：
利用已知集$\mathbb{X}^N$中的样本进行预分类，并剪辑掉被错分类的样本。留下的样本构成剪辑样本集$\mathbb{X}^{NE}$ 。显然， $\mathbb{X}^{NE}$中的样本数比$\mathbb{X}^N$中的少。称之为剪辑。

方法的第二步：
利用剪辑样本集$\mathbb{X}^{NE}$近邻规则对未知样本x进行分类。

6.11 压缩近邻法

压缩近邻法旨在进一步缩短计算时间和降低存储要求。

对于$\mathbb{X}^N$，把第1个样本放入STORE中，其余放入GARBBAG。

Step 1.
用当前STORE中样本以1NN对的第i个样本分类。
若正确，则该样本仍送回GARBBAG；否则放入STORE。
对GARBBAG中样本重复以上过程。

Step 2.
若GARBBAG中所有样本在上述检验中没有一个样本从GARBBAG中转入STORE，
或者GARBBAG= $\{\emptyset\}$，算法终止。
否则，转Step 1。

7. 错误率（再一次讨论）

错误率是分类问题固有复杂性的一种度量，是比较各种模式识别方案好坏的标准。

已经讨论过，对于二类问题用最小错误率Bayes决策规则时，其错误率为

$$\begin{aligned}\mathrm{P}(e) &= \int_{-\infty}^{t} p(x|\omega_2)p(\omega_2)dx + \int_{t}^{+\infty} p(x|\omega_1)p(\omega_1)dx \\ &= P(\omega_2)P_2(e) + P(\omega_1)P_1(e)\end{aligned}$$

理论上，只要知道先验概率和类条件概率密度，就可以计算错误率。然而，当是多维向量时，需要计算多重积分，往往变得十分复杂，实际上难以进行。因此，

（1）按理论公式计算；（某些简单情况下）
（2）计算错误率上限；（稍微复杂一些的情况下）
（3）通过实验进行评估。

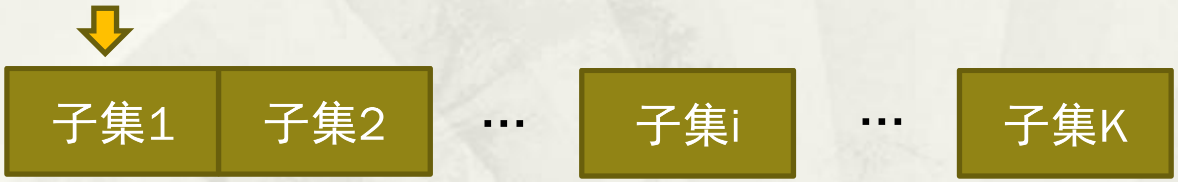
训练集和测试集（用iris数据示例）

1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
5	5.0	3.6	1.4	0.2	Iris-setosa
6	5.4	3.9	1.7	0.4	Iris-setosa
7	4.6	3.4	1.4	0.3	Iris-setosa
8	5.0	3.4	1.5	0.2	Iris-setosa
9	4.4	2.9	1.4	0.2	Iris-setosa
10	4.9	3.1	1.5	0.1	Iris-setosa
11	5.4	3.7	1.5	0.2	Iris-setosa
12	4.8	3.4	1.6	0.2	Iris-setosa
13	4.8	3.0	1.4	0.1	Iris-setosa
14	4.3	3.0	1.1	0.1	Iris-setosa
15	5.8	4.0	1.2	0.2	Iris-setosa
16	5.7	4.4	1.5	0.4	Iris-setosa
17	5.4	3.9	1.3	0.4	Iris-setosa
18	5.1	3.5	1.4	0.3	Iris-setosa
19	5.7	3.8	1.7	0.3	Iris-setosa
20	5.1	3.8	1.5	0.3	Iris-setosa
21	5.4	3.4	1.7	0.2	Iris-setosa
22	5.1	3.7	1.5	0.4	Iris-setosa
23	4.6	3.6	1.0	0.2	Iris-setosa
24	5.1	3.3	1.7	0.5	Iris-setosa
25	4.8	3.4	1.9	0.2	Iris-setosa
26	5.0	3.0	1.6	0.2	Iris-setosa
27	5.0	3.4	1.6	0.4	Iris-setosa
28	5.2	3.5	1.5	0.2	Iris-setosa
29	5.2	3.4	1.4	0.2	Iris-setosa
30	4.7	3.2	1.6	0.2	Iris-setosa
31	4.8	3.1	1.6	0.2	Iris-setosa
32	5.4	3.4	1.5	0.4	Iris-setosa
33	5.2	4.1	1.5	0.1	Iris-setosa
34	5.5	4.2	1.4	0.2	Iris-setosa
35	4.9	3.1	1.5	0.1	Iris-setosa
36	5.0	3.2	1.2	0.2	Iris-setosa
37	5.5	3.5	1.3	0.2	Iris-setosa
38	4.9	3.1	1.5	0.1	Iris-setosa
39	4.4	3.0	1.3	0.2	Iris-setosa
40	5.1	3.4	1.5	0.2	Iris-setosa
41	5.0	3.5	1.3	0.3	Iris-setosa
42	4.5	2.3	1.3	0.3	Iris-setosa
43	4.4	3.2	1.3	0.2	Iris-setosa
44	5.0	3.5	1.6	0.6	Iris-setosa
45	5.1	3.8	1.9	0.4	Iris-setosa
46	4.8	3.0	1.4	0.3	Iris-setosa
47	5.1	3.8	1.6	0.2	Iris-setosa
48	4.6	3.2	1.4	0.2	Iris-setosa
49	5.3	3.7	1.5	0.2	Iris-setosa
50	5.0	3.3	1.4	0.2	Iris-setosa
51	7.0	3.2	4.7	1.4	Iris-versicolor
52	6.4	3.2	4.5	1.5	Iris-versicolor
53	6.9	3.1	4.9	1.5	Iris-versicolor
54	5.5	2.3	4.0	1.3	Iris-versicolor
55	6.5	2.8	4.6	1.5	Iris-versicolor
56	5.7	2.8	4.5	1.3	Iris-versicolor
57	6.3	3.3	4.7	1.6	Iris-versicolor
58	4.9	2.4	3.3	1.0	Iris-versicolor
59	6.6	2.9	4.6	1.3	Iris-versicolor
60	5.2	2.7	3.9	1.4	Iris-versicolor
61	5.0	2.0	3.5	1.0	Iris-versicolor
62	5.9	3.0	4.2	1.5	Iris-versicolor
63	6.0	2.2	4.0	1.0	Iris-versicolor
64	6.1	2.9	4.7	1.4	Iris-versicolor
65	5.6	2.9	3.6	1.3	Iris-versicolor
66	6.7	3.1	4.4	1.4	Iris-versicolor
67	5.6	3.0	4.5	1.5	Iris-versicolor
68	5.8	2.7	4.1	1.0	Iris-versicolor
69	6.2	2.2	4.5	1.5	Iris-versicolor
70	5.6	2.5	3.9	1.1	Iris-versicolor
71	5.9	3.2	4.8	1.8	Iris-versicolor
72	6.1	2.8	4.0	1.3	Iris-versicolor
73	6.3	2.5	4.9	1.5	Iris-versicolor
74	6.1	2.8	4.7	1.2	Iris-versicolor
75	6.4	2.9	4.3	1.3	Iris-versicolor
76	6.6	3.0	4.4	1.4	Iris-versicolor
77	6.8	2.8	4.8	1.4	Iris-versicolor
78	6.7	3.0	5.0	1.7	Iris-versicolor
79	6.0	2.9	4.5	1.5	Iris-versicolor
80	5.7	2.6	3.5	1.0	Iris-versicolor
81	5.5	2.4	3.8	1.1	Iris-versicolor
82	5.5	2.4	3.7	1.0	Iris-versicolor
83	5.8	2.7	3.9	1.2	Iris-versicolor
84	6.0	2.7	5.1	1.6	Iris-versicolor
85	5.4	3.0	4.5	1.5	Iris-versicolor
86	6.0	3.4	4.5	1.6	Iris-versicolor
87	6.7	3.1	4.7	1.5	Iris-versicolor
88	6.3	2.3	4.4	1.3	Iris-versicolor
89	5.6	3.0	4.1	1.3	Iris-versicolor
90	5.5	2.5	4.0	1.3	Iris-versicolor
91	5.5	2.6	4.4	1.2	Iris-versicolor
92	6.1	3.0	4.6	1.4	Iris-versicolor
93	5.8	2.6	4.0	1.2	Iris-versicolor
94	5.0	2.3	3.3	1.0	Iris-versicolor
95	5.6	2.7	4.2	1.3	Iris-versicolor
96	5.7	3.0	4.2	1.2	Iris-versicolor
97	5.7	2.9	4.2	1.3	Iris-versicolor
98	6.2	2.9	4.3	1.3	Iris-versicolor
99	5.1	2.5	3.0	1.1	Iris-versicolor
100	5.7	2.8	4.1	1.3	Iris-versicolor
101	6.3	3.3	6.0	2.5	Iris-virginica
102	5.8	2.7	5.1	1.9	Iris-virginica
103	7.1	3.0	5.9	2.1	Iris-virginica
104	6.3	2.9	5.6	1.8	Iris-virginica
105	6.5	3.0	5.8	2.2	Iris-virginica
106	7.6	3.0	6.6	2.1	Iris-virginica
107	4.9	2.5	4.5	1.7	Iris-virginica
108	7.3	2.9	6.3	1.8	Iris-virginica
109	6.7	2.5	5.8	1.8	Iris-virginica
110	7.2	3.6	6.1	2.5	Iris-virginica
111	6.5	3.2	5.1	2.0	Iris-virginica
112	6.4	2.7	5.3	1.9	Iris-virginica
113	6.8	3.0	5.5	2.1	Iris-virginica
114	5.7	2.5	5.0	2.0	Iris-virginica
115	5.8	2.8	5.1	2.4	Iris-virginica
116	6.4	3.2	5.3	2.3	Iris-virginica
117	6.5	3.0	5.5	1.8	Iris-virginica
118	7.7	3.8	6.7	2.2	Iris-virginica
119	7.7	2.6	6.9	2.3	Iris-virginica
120	6.0	2.2	5.0	1.5	Iris-virginica
121	6.9	3.2	5.7	2.3	Iris-virginica
122	5.6	2.8	4.9	2.0	Iris-virginica
123	7.7	2.8	6.7	2.0	Iris-virginica
124	6.3	2.7	4.9	1.8	Iris-virginica
125	6.7	3.3	5.7	2.1	Iris-virginica
126	7.2	3.2	6.0	1.8	Iris-virginica
127	6.2	2.8	4.8	1.8	Iris-virginica
128	6.1	3.0	4.9	1.8	Iris-virginica
129	6.4	2.8	5.6	2.1	Iris-virginica
130	7.2	3.0	5.8	1.6	Iris-virginica
131	7.4	2.8	6.1	1.9	Iris-virginica
132	7.9	3.8	6.4	2.0	Iris-virginica
133	6.4	2.8	5.6	2.2	Iris-virginica
134	6.3	2.8	5.1	1.5	Iris-virginica
135	6.1	2.6	5.6	1.4	Iris-virginica
136	7.7	3.0	6.1	2.3	Iris-virginica
137	6.3	3.4	5.6	2.4	Iris-virginica
138	6.4	3.1	5.5	1.8	Iris-virginica
139	6.0	3.0	4.8	1.8	Iris-virginica
140	6.9	3.1	5.4	2.1	Iris-virginica
141	6.7	3.1	5.6	2.4	Iris-virginica
142	6.9	3.1	5.1	2.3	Iris-virginica
143	5.8	2.7	5.1	1.9	Iris-virginica
144	6.8	3.2	5.9	2.3	Iris-virginica
145	6.7	3.3	5.7	2.5	Iris-virginica
146	6.7	3.0	5.2	2.3	Iris-virginica
147	6.3	2.5	5.0	1.9	Iris-virginica
148	6.5	3.0	5.2	2.0	Iris-virginica
149	6.2	3.4	5.4	2.3	Iris-virginica
150	5.9	3.0	5.1	1.8	Iris-virginica

7.1交叉验证法(Cross Validation)

交叉验证法是目前常用的实验验证方法。对于训练集中N个样本，将其伪随机地划分为K个大小相同的子集，且每个子集中先验概率相同。

取其中的一个子集为测试集，另外$(K-1)$作为训练集。



记录每个测试集上被错分的样本个数ε_i。这样顺次将K个子集都测试一遍，得到ε_i（$i=1,2,\dots,K$）。总的分类错误率计算为

$$err\% = \frac{\sum_{i=1}^{K} \varepsilon_i}{N}$$

以上为1次CV-K测试。

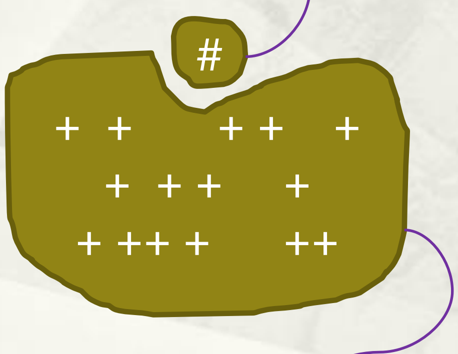
取n次CV-K测试的平均值作为分类器性能的错误率评测依据。

7.2留一法(Let One Out)

对于N个样本，每次取出1个样本作为测试样本，其它$(N-1)$个供训练用。N次计算后得到错分样本总数ε与N的比值作为评判依据。即，

$$err\% = \frac{\varepsilon}{N}$$

测试集（1个样本）



训练集($N-1$个样本)