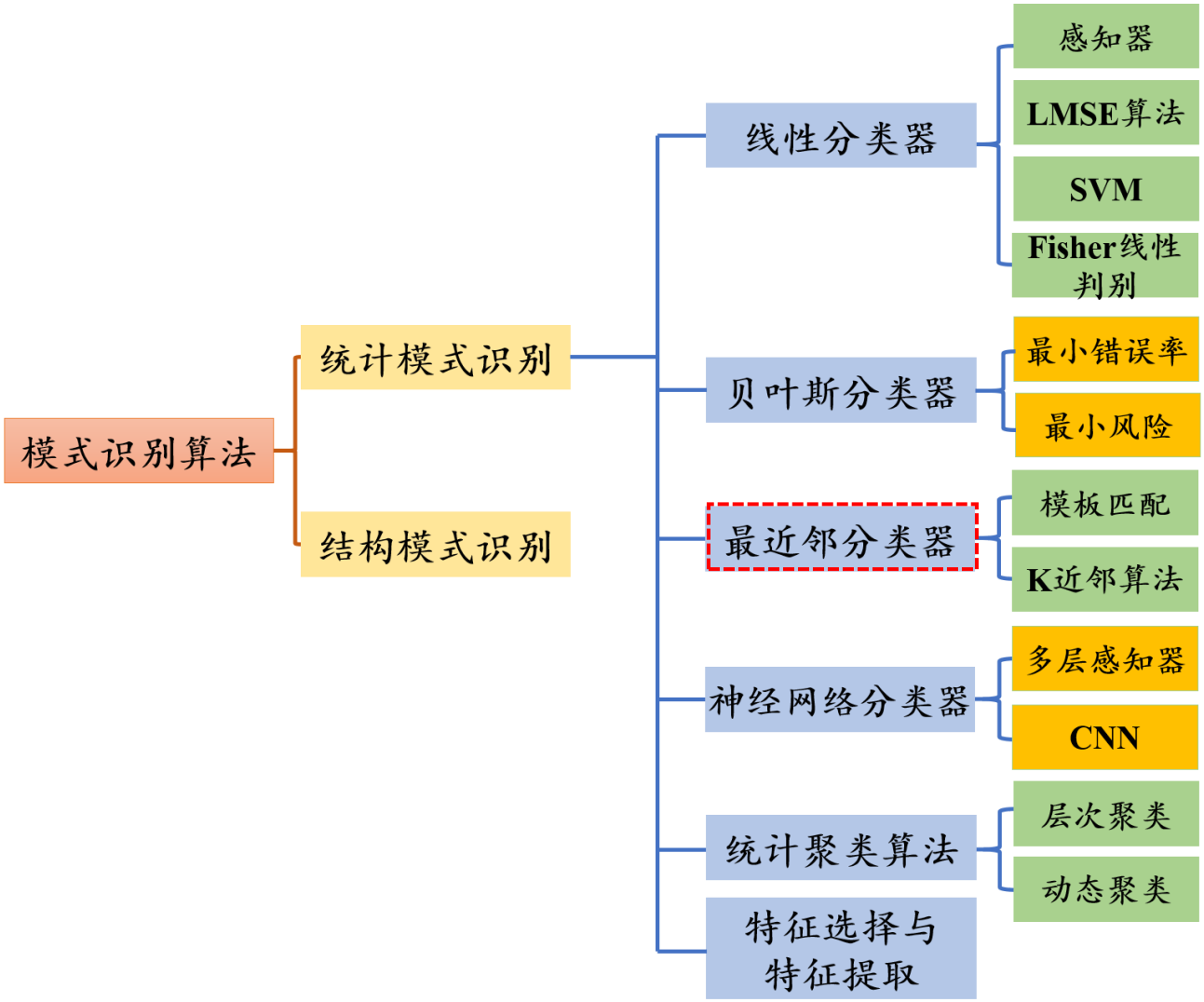
模式识别课程

近邻法

张俊超

中南大学
航空航天学院

模式识别-近邻法

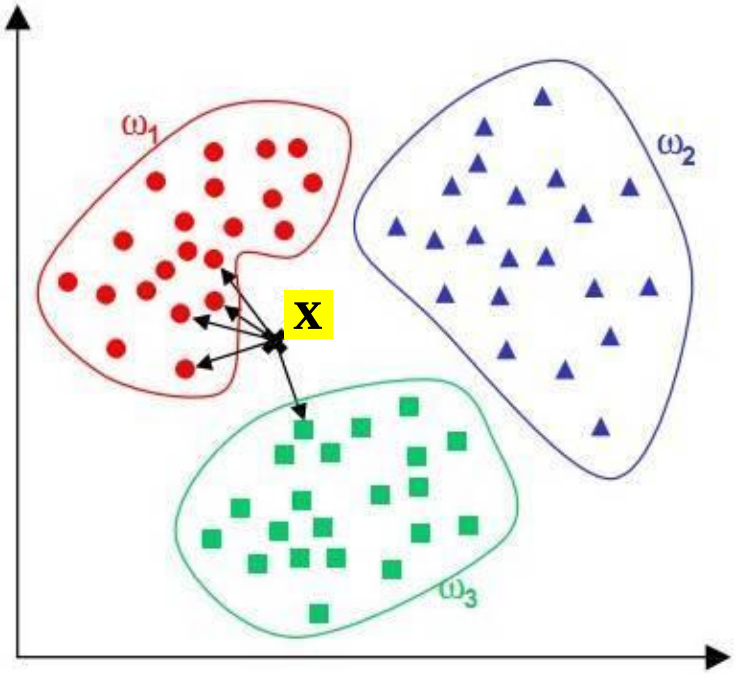

模式识别算法
统计模式识别
结构模式识别
线性分类器
感知器
LMSE算法
SVM
Fisher线性判别
贝叶斯分类器
最小错误率
最小风险
最近邻分类器
模板匹配
K近邻算法
神经网络分类器
多层感知器
CNN
统计聚类算法
层次聚类
动态聚类
特征选择与特征提取

近邻分类器-最近邻法

最近邻决策：

样本间的距离度量：$\delta\left(\mathbf{x}_i,\mathbf{x}_j\right)$

若有$\delta\left(\mathbf{x},\mathbf{x}_i\right)=\min_{j=1,2,\ldots,N}\left\{\delta\left(\mathbf{x},\mathbf{x}_j\right)\right\}$，且$\mathbf{x}_i\in\omega_i$　　则$\mathbf{x}\in\omega_i$



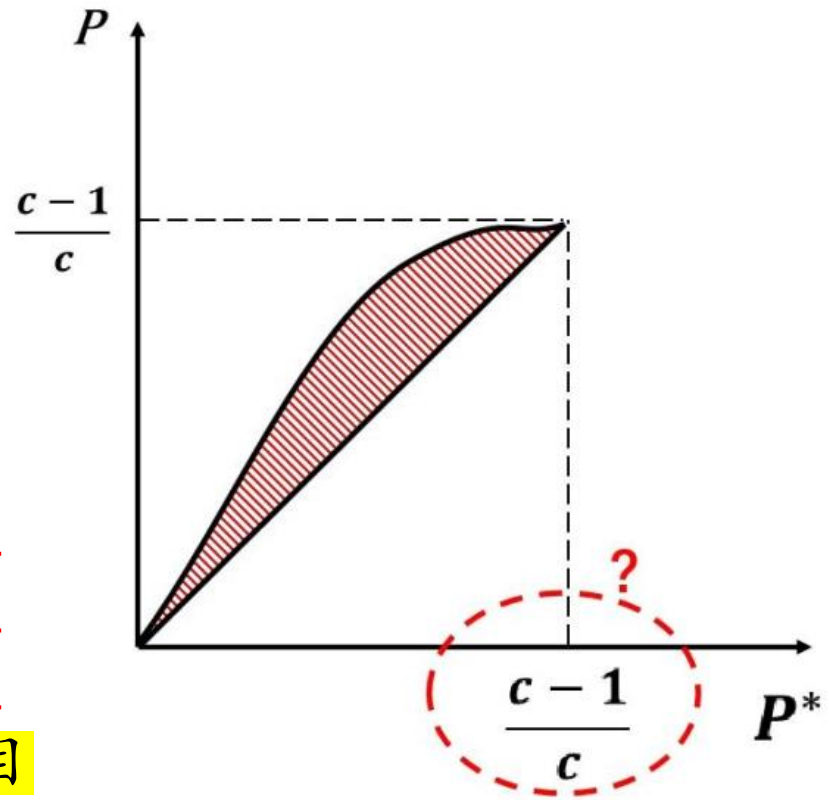
近邻分类器-最近邻法

最近邻法的错误率：

$N \to \infty$,错误类P满足：

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

P^*为贝叶斯错误率



结论告诉我们：

最近邻法的渐进错误率最坏不会超出两倍的贝叶斯错误率，最好则有可能接近或达到贝叶斯错误率。（该结论在样本数目趋于无穷大时成立）

近邻分类器-最近邻法

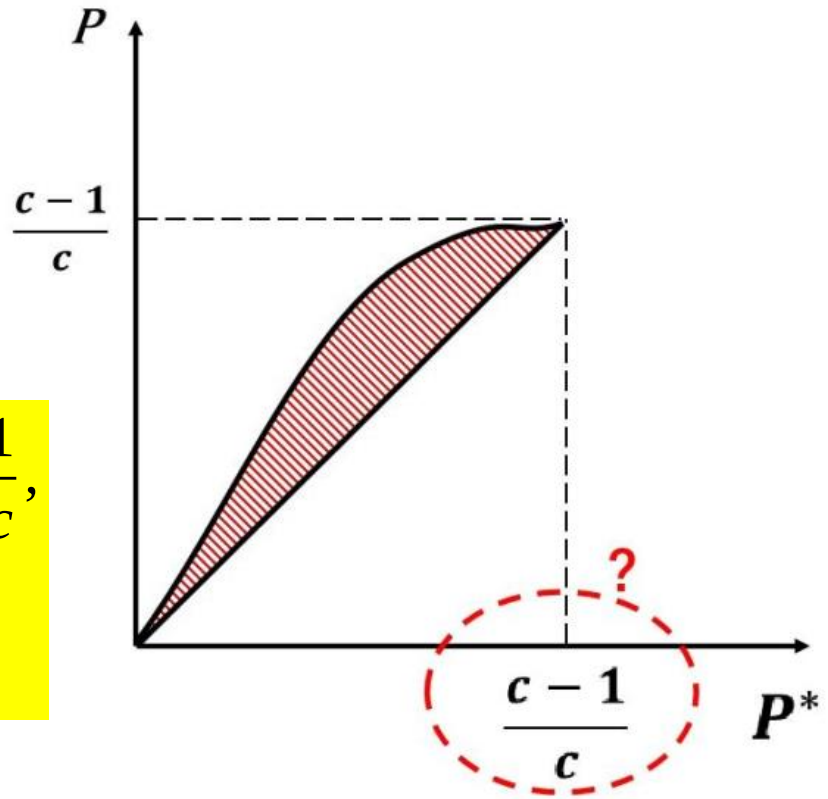
- 为什么错误率的下界是贝叶斯错误率?
- 为什么最小错误率贝叶斯分类器的错误率最大不会超过 (c-1)/c?

$N \to \infty \Rightarrow p(\omega_i \mid \mathbf{x}') \to p(\omega_i \mid \mathbf{x})$

$\mathbf{x}'$是$\mathbf{x}$的最近邻点

最大后验概率$p(\omega_i \mid \mathbf{x})$一定不小于$\frac{1}{c}$,

因为$\sum_{i=1}^{c} p(\omega_i \mid \mathbf{x}) = 1$。 故$P^* \le \frac{c-1}{c}$



近邻分类器-k近邻法

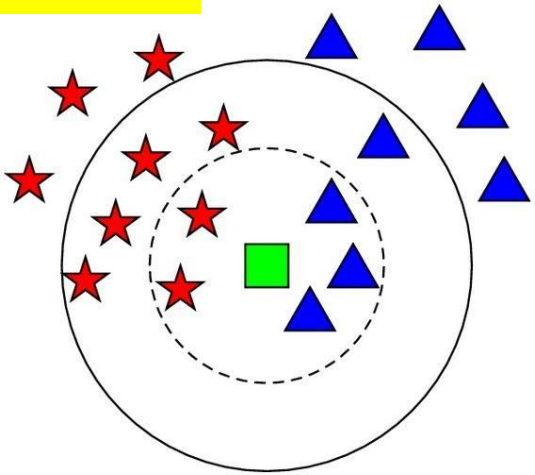
*KNN*决策规则：

$\mathbf{x}$的k个最近邻记作：$\left(\mathbf{x}_1^{'},\mathbf{x}_2^{'},\ldots,\mathbf{x}_k^{'}\right)$，其中属于$\omega_i$的样本数为$n_i$

若有$n_i=\max\limits_{j=1,2,\ldots,c}\left\{n_j\right\}$　　则$\mathbf{x}\in\omega_i$

$N\to\infty$，错误类P满足：

$$P^*\le P\le\sum_{i=0}^{(k-1)/2}C_k^i\left(\left(P^*\right)^{i+1}\left(1-P^*\right)^{k-i}+\left(P^*\right)^{k-i}\left(1-P^*\right)^{i+1}\right)$$



近邻分类器-k近邻法

k-近邻分类器原理简单，无需对样本集进行概率分布统计，实现起来十分方便。但是它有两个缺陷：

1. 对于每一个待分类的样本 x，都必须计算 x 到样本集中所有样本的距离，从而找出 x 的 k 个近邻来完成分类，因此算法的**计算量**随着样本集的增大而增大。

2. 样本集中的所有样本都必须被使用，这给算法带来了巨大的**存储压力**。

Improve

近邻分类器-k近邻法

```
% 生成数据
randn('seed',2020);
mu1 = [2 3];
sigma1 = [0.5 0;
          0 0.2];
data1 = mvnrnd(mu1,sigma1,500);
label1 = ones(500,1);

randn('seed',2021);
mu2 = [4 4];
sigma2 = [0.5 0;
          0 0.8];
data2 = mvnrnd(mu2,sigma2,500);
label2 = ones(500,1) + 1;

train = [data1(1:400,:);data2(1:400,:)];
train_label= [label1(1:400,:);label2(1:400,:)];

test = [data1(401:end,:);data2(401:end,:)];
test_label= [label1(401:end,:);label2(401:end,:)];
```

近邻分类器-k近邻法

```
%% random the order of data
[trainrow,traincol] = size(train);
[testrow,testcol] = size(test);
%
List_train = randperm(trainrow);
train = train(List_train,:);
train_label = train_label(List_train,:);
%
List_test = randperm(testrow);
test = test(List_test,:);
test_label = test_label(List_test,:);
%%
label = unique(train_label);
[labelnum,~] = size(label);
test_predict = zeros(testrow,1);
```

近邻分类器-k近邻法

```
label = unique(train_label);
[labelnum,~] = size(label);
test_predict = zeros(testrow,1);

for i=1:testrow
    knn = zeros(K,2);
    for j=1:K
        distance = norm(train(j,:)-test(i,:));
        knn(j,1) = distance;
        knn(j,2) = train_label(j,1);
    end
    knn = sortrows(knn,1);  % in ascending order
    for j=K+1:trainrow
        distance = norm(train(j,:)-test(i,:));
        if distance<knn(K,1)
            knn(K,1) = distance;
            knn(K,2) = train_label(j,1);
            knn = sortrows(knn,1);
        end
    end
```

近邻分类器-k近邻法

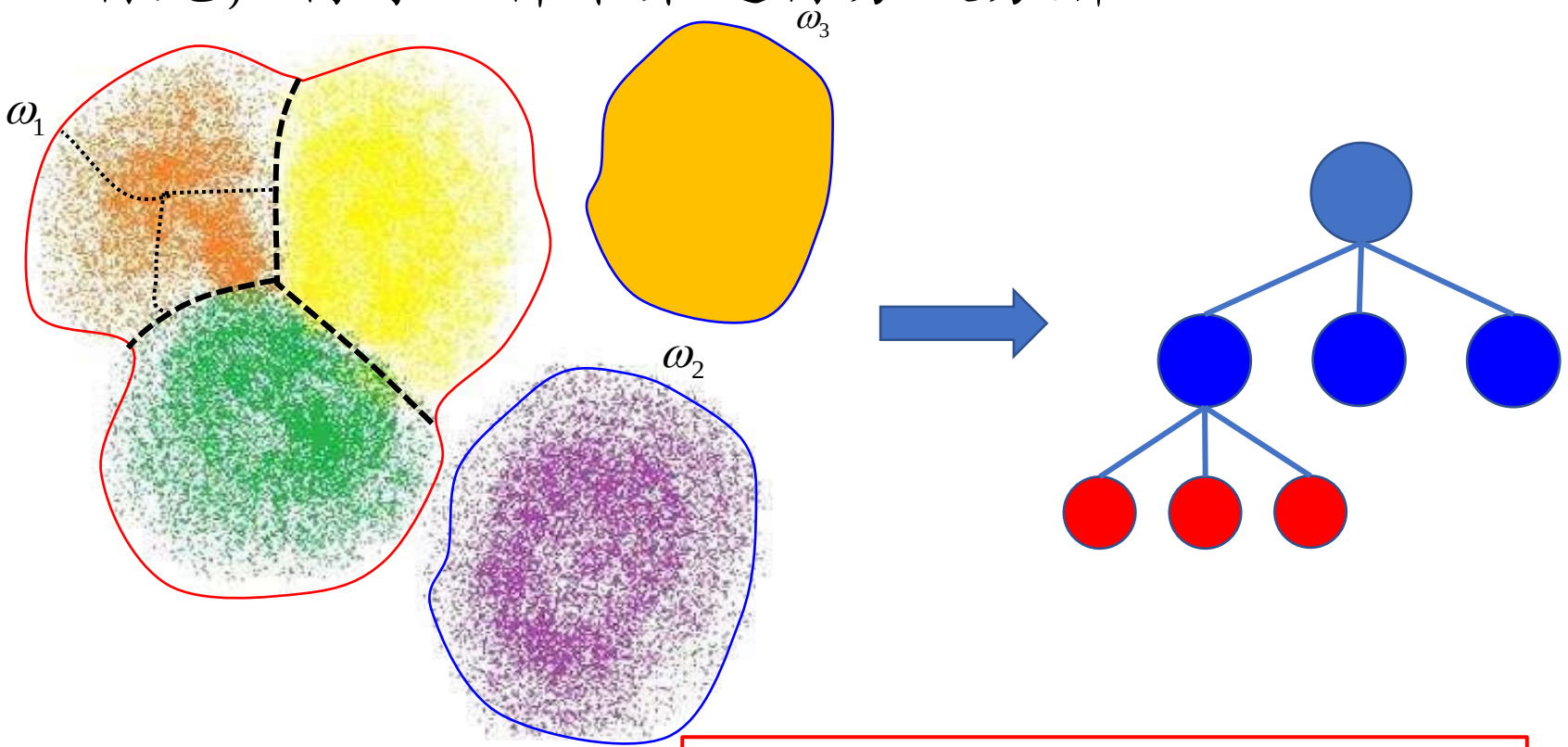
```
    % classification decision
    labelcount = [label zeros(labelnum,1)];
    for n=1:K
        for m=1:labelnum
            if knn(n,2)==labelcount(m,1)
                labelcount(m,2) = labelcount(m,2)+1;
                break
            end
        end
    end
    labelcount = sortrows(labelcount,2);
    test_predict(i,1) = labelcount(labelnum,1);
end

accu = sum(test_predict==test_label)/testrow;
fprintf('The parameter K = %d, the accuracy of KNN algorithm is %f.\n',K,accu);
```

近邻分类器-k近邻法

The parameter K = 1, the accuracy of KNN algorithm is 0.925000.
The parameter K = 3, the accuracy of KNN algorithm is 0.950000.
The parameter K = 5, the accuracy of KNN algorithm is 0.950000.
The parameter K = 20, the accuracy of KNN algorithm is 0.950000.

近邻分类器-快速KNN算法

- 首先，将原始样本集进行分级分解



p：搜索树中的一个节点，代表了一组样本
M_p：p 中所有样本的重心
r_p：p 的半径，即 p 中样本到 M_p 的最大距离

近邻分类器-快速KNN算法

- 搜索：

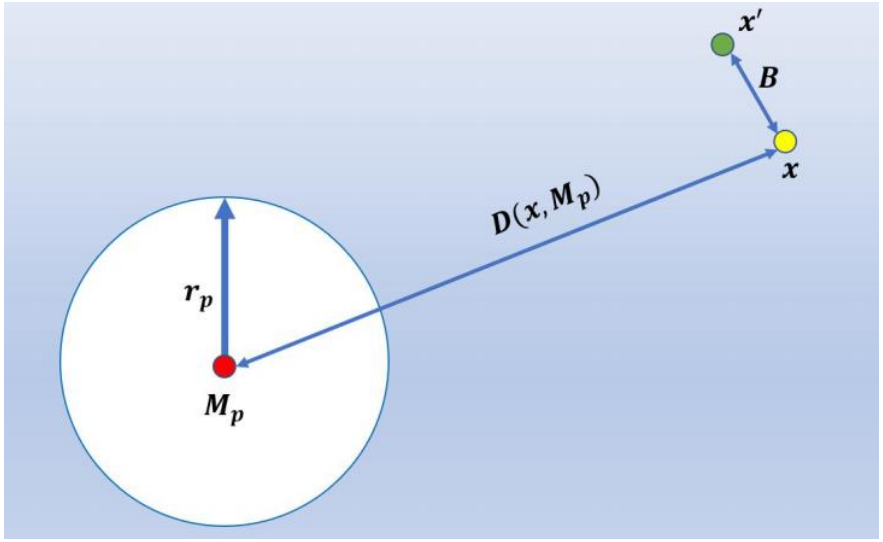
规则一：如果存在$D(\mathbf{x}, M_p) > B + r_p$，则$p$中所有样本均不是$\mathbf{x}$的最近邻点

(B是$\mathbf{x}$到当前最近点$\mathbf{x}'$的距离)

规则二：p中的样本$\mathbf{x}_i$满足$D(\mathbf{x}, M_p) > B + D(\mathbf{x}_i, M_p)$，则$\mathbf{x}_i$不是$\mathbf{x}$的最近邻点

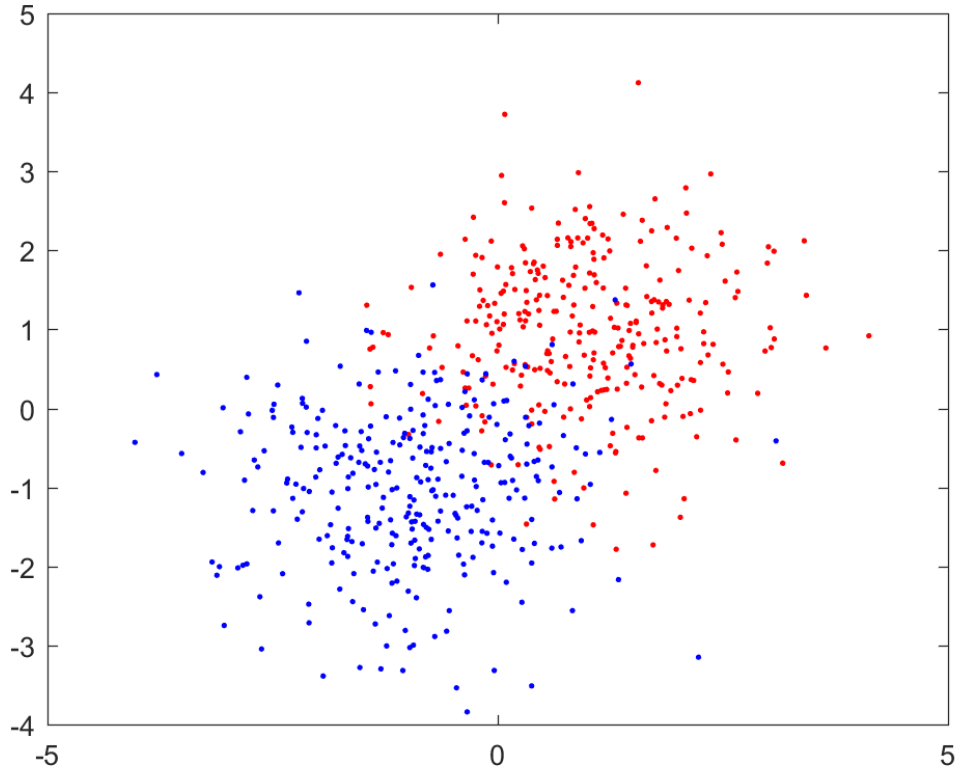
M_p，r_p，$D(\mathbf{x}_i, M_p)$可以事先计算，只计算一次

减少了计算量，没减少存储量



近邻分类器-剪辑近邻法

- 不同类别的样本在分布上有重叠区域
- 错误率主要来自于重叠区域



近邻分类器-剪辑近邻法

- 基本思路：

考察样本是否为误导样本

- 是→剪辑
- 否→保留

为了消除考试集、训练集划分中的偶然性造成的影响，当样本数较多时，人们设计了一种多重剪辑方法 MULTIEDIT[②]：

（1）划分　把样本集随机划分为 s 个子集，$\mathscr{X}_1$，…$\mathscr{X}_s$，$s \geqslant 3$。

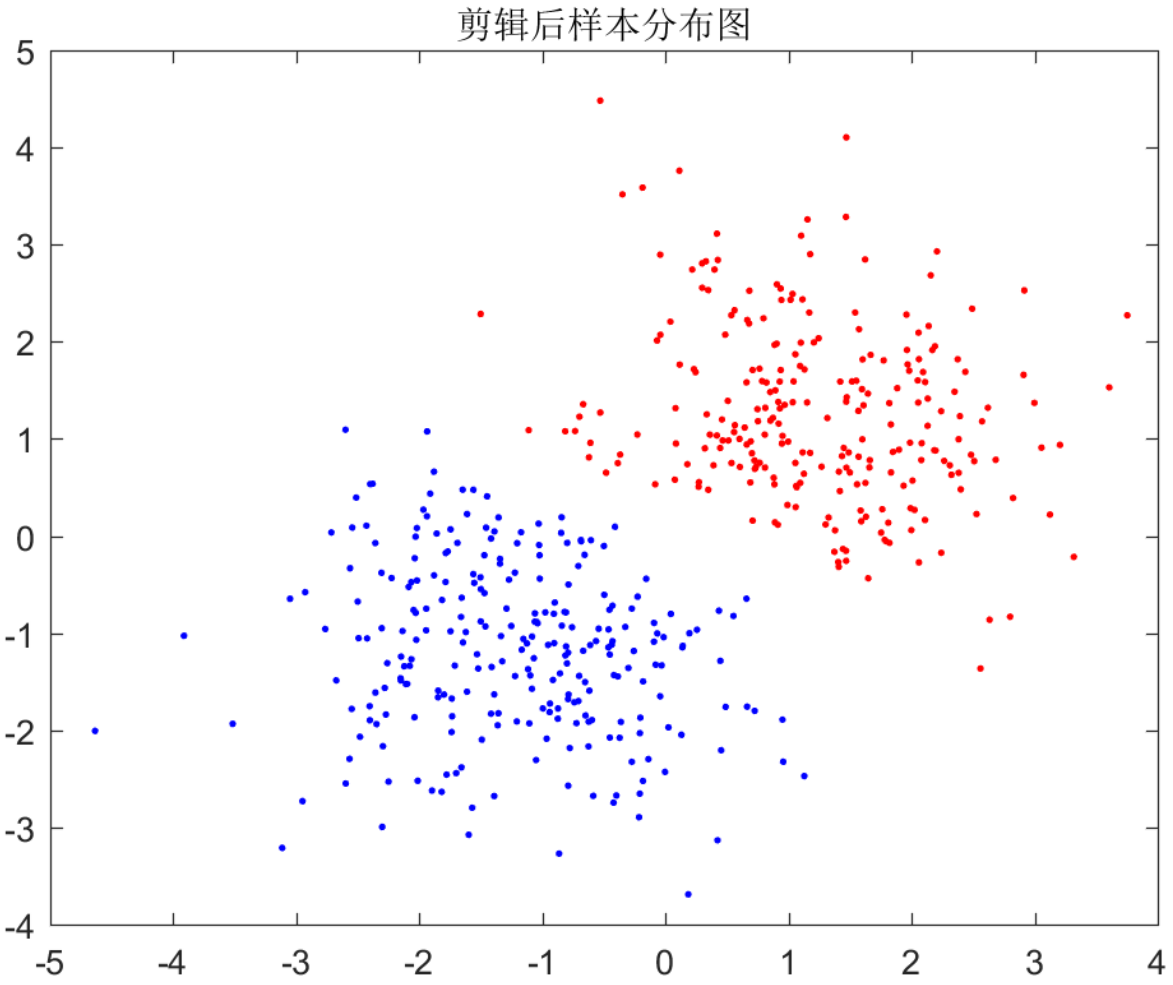
（2）分类　用 $\mathscr{X}_{(i+1) \bmod (s)}$ 对 $\mathscr{X}_i$ 中的样本分类，$i=1,\cdots,s$。比如，如果 $s=3$，则用 $\mathscr{X}_2$ 对 $\mathscr{X}_1$ 分类，用 $\mathscr{X}_3$ 对 $\mathscr{X}_2$ 分类，用 $\mathscr{X}_1$ 对 $\mathscr{X}_3$ 分类。

（3）剪辑　从各个子集中去掉在(2)中被分错的样本。

（4）混合　把剩下的样本合在一起，形成新的样本集 $\mathscr{X}^{NE}$。

（5）迭代　用新的样本集 $\mathscr{X}^{NE}$ 替代原样本集，转(1)。如果在最近的 m 次迭代中都没有样本被剪掉，则终止迭代，用最后的 $\mathscr{X}^{NE}$ 作为剪辑后的样本集。

近邻分类器-剪辑近邻法



近邻分类器-剪辑近邻法

剪辑近邻法好处：

- 减少数据存储量
- 减少分类过程中的计算量
- 改进分类器性能，降低错误率

近邻分类器-压缩近邻法

- 剪辑近邻法对数量的压缩效果有限
- 靠近类别中心的大量样本被保留，这些样本对分类决策结果影响很小

压缩近邻法算法思想：

- 利用训练样本集生成新样本集
- 保留最少量样本，且维持分类正确
- 最大程度保留原训练集的分类特性

近邻分类器-压缩近邻法

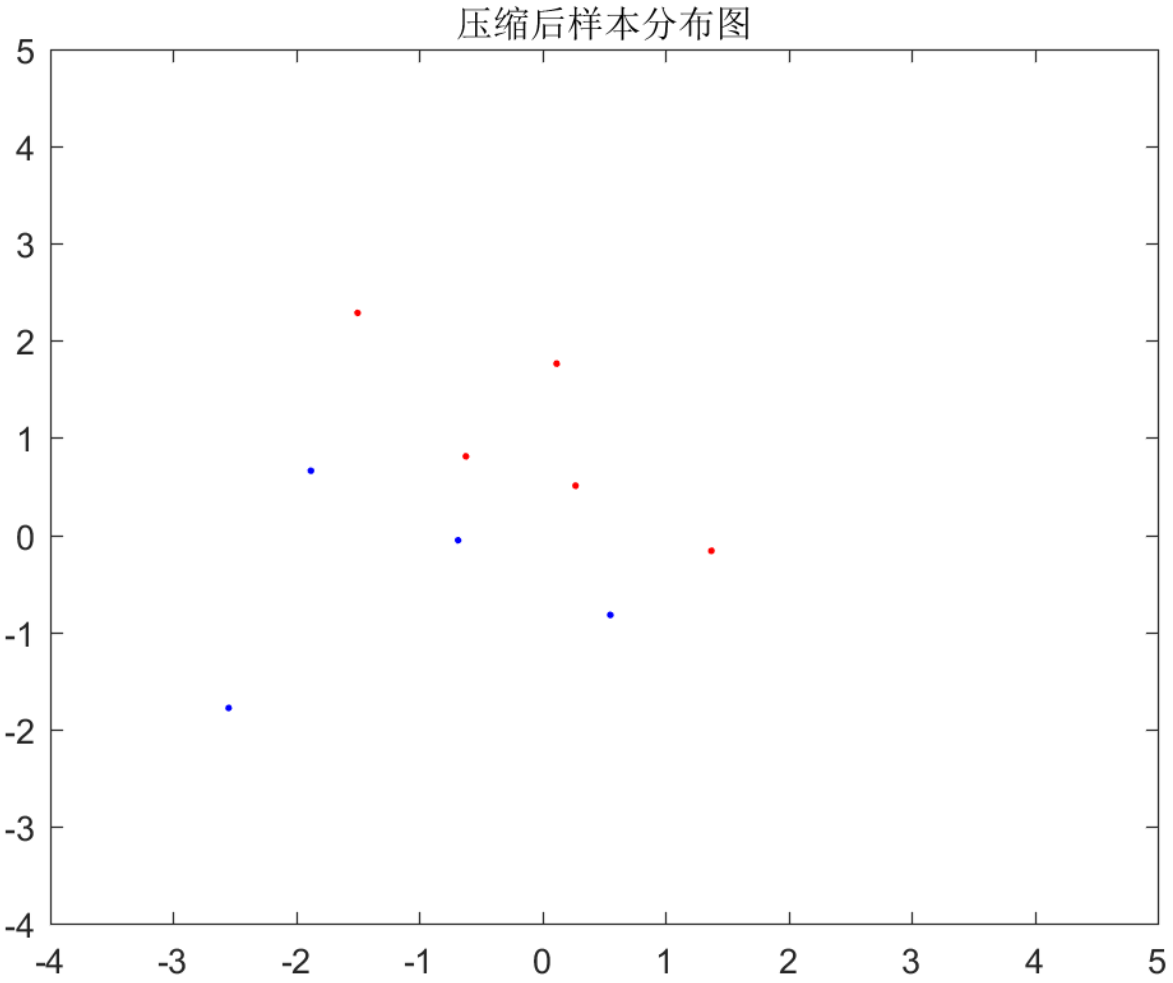
定义两个存储器：Store　　Grabbag

（1）**初始化：Store为空，Grabbag存储训练集**，从Grabbag中任意选择一样本存入Store

（2）**生成样本集：**用最近邻法对Grabbag中每一个样本使用Store中的样本集进行分类，**若分类错误**，则移入Store

（3）**终止条件：**若Grabbag中所有样本都可以用Store中的样本集**分类正确**，或Grabbag成为**空集**，则算法终止，否则重复（2）

近邻分类器-压缩近邻法



近邻分类器-压缩近邻法

剪辑和压缩的Matlab代码

https://blog.csdn.net/yanxiaopan/article/details/53932327?utm_source=blogxgwz1

```
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
```

模式识别-近邻法

模式识别-近邻法