

集成学习

1. 概念

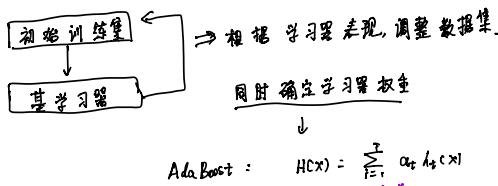
通过结合模块，将不同的个体学习器组合，得到更好的输出

同质集成：同学习器（同 SVM, 同 DT）

异质：保证个体学习器的准确性和多样性。

2. 集成方法

(1) 权值生成：Boosting.



$$\text{AdaBoost: } H(x) = \sum_{t=1}^T \alpha_t h_t(x)$$

只应用二分类，降低偏差

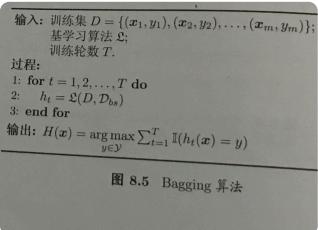
其中重赋权只为每个训练样本赋权重，有些学习器无法使用权重，根据样本分布对训练样本重新采样

(2) 并行生成：Bagging 和 RF

有放回采样十个含 m 个训练样本的采样集。（初始训练集约有 36.8% 未被采样）

训练出 T 个基分类器，再结合

RF: Bagging + 随机属性选择



可用于多分类，降低偏差。

3. 融合策略

(1) 数值型 | 平均：相近使用
加权平均：个体学习器差异大使用，(w 与 E 成反比)

(2) 分类任务 | 绝对多数
相对多数
加权投票

注意：类标记与类概率的转化。

若基学习器类型不同，类概率值不能直接比较。

(3) Stacking 方法

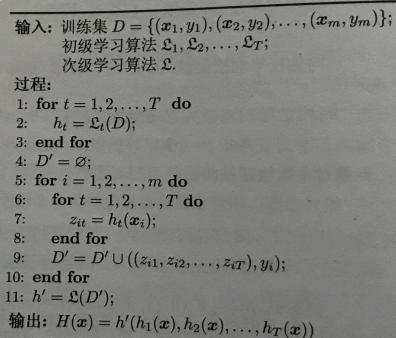
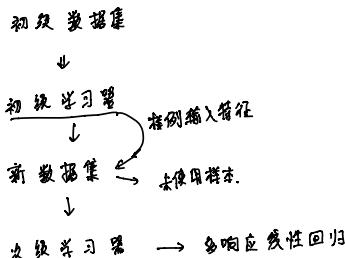


图 8.9 Stacking 算法

4. 多样性

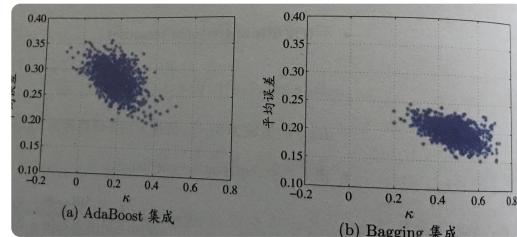
1. 定义： $E = \overline{E} - \overline{A}$ ，个体学习器准确性越高，多样性越大 \Rightarrow 集成越好

2. 如何度量

$h_1 = +1$	$h_2 = -1$
$h_3 = +1$	a
$h_4 = -1$	d

$$\begin{aligned} &\text{不一致} : \frac{b+c}{m}, \text{ 越大, 多样性越大} \\ &\text{相关系数} : \frac{ad-bc}{\sqrt{(a+b)(a+c)(c+d)(b+d)}} \quad \left\{ \begin{array}{l} 0 \quad h_1, h_2 \text{ 无关} \\ + \quad \text{正相关} \\ - \quad \text{负相关} \end{array} \right. \\ &\text{Q统计量} : \frac{ad-bc}{ad+bc} \\ &\text{k-统计量} : \frac{P_1 - P_2}{1 - P_2} \quad \text{其中} \quad \left\{ \begin{array}{l} P_1 = \frac{ad}{m} \\ P_2 = \frac{(a+b)(a+c) + (c+d)(b+d)}{m^2} \end{array} \right. \\ &\left. \begin{array}{l} h_1, h_2 \text{ 完全一致, } k=1 \\ \text{偶然一致, } k=0 \end{array} \right. \end{aligned}$$

3. 并行
 样本扰动：适用于 DT, MVA 不适合学习
 属性扰动：参数 RF
 算法扰动：调参



聚类：内小外大

1. 性能度量

(1) 参考模型：JC, FM, Rand

(2) 内部指标：DB, Dunn

2. 距离计算

非负，同一，对称 三角不等式

$$\begin{aligned} \text{有序属性} : & \text{闵可夫斯基距离} \left(\sum_{i=1}^n |x_{ia} - x_{bi}|^p \right)^{\frac{1}{p}} \\ \{1, 2, 3\} & \left\{ \begin{array}{l} p=2, \text{ 欧氏距离} \\ p=1, \text{ 曼哈顿距离} \end{array} \right. \end{aligned}$$

$$\text{无序属性} : VDM = \sum_{i=1}^n \left| \frac{m_{a,i}}{m_{a,a}} - \frac{m_{b,i}}{m_{b,b}} \right|^p$$

$$\text{混合} \left(\sum_{i=1}^n |x_{ia} - x_{ib}|^p + \sum_{i=n+1}^k VDM_p(x_{ia}, x_{ib}) \right)^{\frac{1}{p}}$$

注：相似度度量不满足三角不等式，视观察自定义距离。

3. 原型聚类：K-means, LVA, 高斯混合

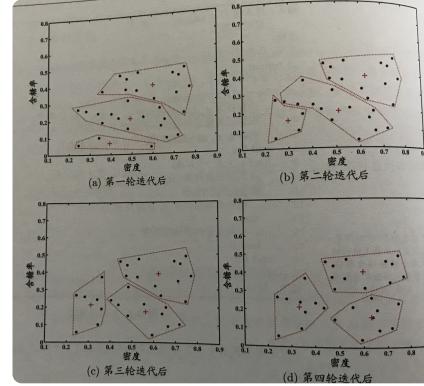
4. 密度聚类，层次聚类。

1.

```

输入: 样本集  $D = \{x_1, x_2, \dots, x_m\}$ ;
聚类簇数  $k$ ;
过程:
1: 从  $D$  中随机选择  $k$  个样本作为初始均值向量  $\{\mu_1, \mu_2, \dots, \mu_k\}$ 
2: repeat
3:   令  $C_i = \emptyset$  ( $1 \leq i \leq k$ )
4:   for  $j = 1, 2, \dots, m$  do
5:     计算样本  $x_j$  与各均值向量  $\mu_i$  ( $1 \leq i \leq k$ ) 的距离:  $d_{ji} = \|x_j - \mu_i\|_2$ ;
6:     根据距离最近的均值向量确定  $x_j$  的簇标记:  $\lambda_j = \arg \min_{i \in \{1, 2, \dots, k\}} d_{ji}$ ;
7:     将样本  $x_j$  划入相应的簇:  $C_{\lambda_j} = C_{\lambda_j} \cup \{x_j\}$ ;
8:   end for
9:   for  $i = 1, 2, \dots, k$  do
10:    计算新均值向量:  $\mu'_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$ ;
11:    if  $\mu'_i \neq \mu_i$  then
12:      将当前均值向量  $\mu_i$  更新为  $\mu'_i$ 
13:    else
14:      保持当前均值向量不变
15:    end if
16:  end for
17: until 当前均值向量均未更新
输出: 簇划分  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$ 

```

图 9.2 k 均值算法

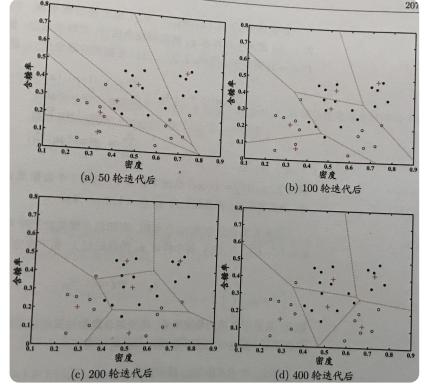
2.

```

输入: 样本集  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$ ;
原型向量个数  $q$ , 各原型向量预设的类别标记  $\{t_1, t_2, \dots, t_q\}$ ;
学习率  $\eta \in (0, 1)$ ;
过程:
1: 初始化一组原型向量  $\{p_1, p_2, \dots, p_q\}$ 
2: repeat
3:   从样本集  $D$  随机选取样本  $(x_j, y_j)$ ;
4:   计算样本  $x_j$  与  $p_i$  ( $1 \leq i \leq q$ ) 的距离:  $d_{ji} = \|x_j - p_i\|_2$ ;
5:   找出与  $x_j$  距离最近的原型向量  $p_{i^*}$ ,  $i^* = \arg \min_{i \in \{1, 2, \dots, q\}} d_{ji}$ ;
6:   if  $y_j = t_{i^*}$  then
7:      $p' = p_{i^*} + \eta \cdot (x_j - p_{i^*})$ 
8:   else
9:      $p' = p_{i^*} - \eta \cdot (x_j - p_{i^*})$ 
10:  end if
11:  将原型向量  $p_{i^*}$  更新为  $p'$ 
12: until 满足停止条件
输出: 原型向量  $\{p_1, p_2, \dots, p_q\}$ 

```

图 9.4 学习向量量化算法



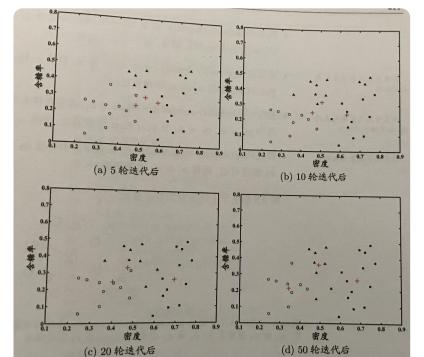
3.

```

输入: 样本集  $D = \{x_1, x_2, \dots, x_m\}$ ;
高斯混合成分个数  $k$ ;
过程:
1: 初始化高斯混合分布的模型参数  $\{(\alpha_i, \mu_i, \Sigma_i) \mid 1 \leq i \leq k\}$ 
2: repeat
3:   for  $j = 1, 2, \dots, m$  do
4:     根据式(9.30)计算  $x_j$  由各混合成分生成的后验概率, 即
 $\gamma_{ji} = p_M(z_j = i | x_j)$  ( $1 \leq i \leq k$ )
5:   end for
6:   for  $i = 1, 2, \dots, k$  do
7:     计算新均值向量:  $\mu'_i = \frac{\sum_{j=1}^m \gamma_{ji} x_j}{\sum_{j=1}^m \gamma_{ji}}$ ;
8:     计算新协方差矩阵:  $\Sigma'_i = \frac{\sum_{j=1}^m \gamma_{ji} (x_j - \mu'_i)(x_j - \mu'_i)^T}{\sum_{j=1}^m \gamma_{ji}}$ ;
9:     计算新混合系数:  $\alpha'_i = \frac{\sum_{j=1}^m \gamma_{ji}}{m}$ ;
10:  end for
11:  将模型参数  $\{(\alpha_i, \mu_i, \Sigma_i) \mid 1 \leq i \leq k\}$  更新为  $\{(\alpha'_i, \mu'_i, \Sigma'_i) \mid 1 \leq i \leq k\}$ 
12: until 满足停止条件
13:  $C_i = \emptyset$  ( $1 \leq i \leq k$ )
14: for  $j = 1, 2, \dots, m$  do
15:   根据式(9.31)确定  $x_j$  的簇标记  $\lambda_j$ ;
16:   将  $x_j$  划入相应的簇:  $C_{\lambda_j} = C_{\lambda_j} \cup \{x_j\}$ 
17: end for
输出: 簇划分  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$ 

```

图 9.6 高斯混合聚类算法

图 9.7 高斯混合聚类($k=3$)在不同轮数迭代后的聚类结果, 其中样本集 C_1 、 C_2 与 C_3 中的样本点分别用“○”、“■”与“▲”表示。各高斯混合成分的均值向量用“+”表示。

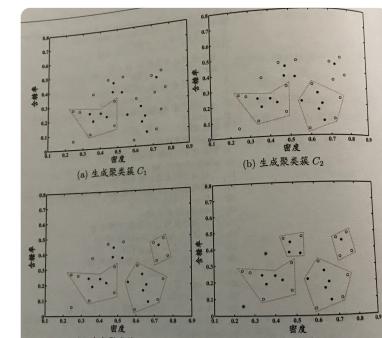
4.

```

输入: 样本集  $D = \{x_1, x_2, \dots, x_m\}$ ;
邻域参数  $(\epsilon, MinPts)$ ;
过程:
1: 初始化核心对象集合:  $\Omega = \emptyset$ 
2: for  $j = 1, 2, \dots, m$  do
3:   随机选取一个样本  $x_j$  的  $\epsilon$ -邻域  $N_\epsilon(x_j)$ ;
4:   if  $|N_\epsilon(x_j)| > MinPts$  then
5:     将样本  $x_j$  加入核心对象集合:  $\Omega \cup \{x_j\}$ 
6:   end if
7: end for
8: 初始化聚类簇数:  $k = 0$ 
9: 初始化未访问样本集合:  $\Gamma = D$ 
10: while  $\Omega \neq \emptyset$  do
11:   记录当前尚未访问样本集合:  $\Gamma_{old} = \Gamma$ ;
12:   随机选取一个核心对象  $o \in \Omega$ , 初始化队列  $Q = <\!o\!>$ ;
13:    $\Gamma = \Gamma \setminus \Gamma_{old}$ 
14:   while  $Q \neq \emptyset$  do
15:     取出队列  $Q$  中的首个样本  $q$ ;
16:     if  $|N_\epsilon(q)| > MinPts$  then
17:       令  $\Delta = N_\epsilon(q) \cap \Gamma$ ;
18:       将  $\Delta$  中的样本加入队列  $Q$ ;
19:        $\Gamma = \Gamma \setminus \Delta$ ;
20:     end if
21:   end while
22:    $k = k + 1$ , 生成聚类簇  $C_k = \Gamma_{old} \setminus \Gamma$ ;
23:    $\Omega = \Omega \setminus C_k$ 
24: end while
输出: 簇划分  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$ 

```

图 9.8 DBSCAN 算法

图 9.9 DBSCAN 算法($\epsilon = 0.11$, $MinPts = 5$)生成聚类簇的先后情况。核心对象, 非核心对象, 噪声样本分别用“+”、“○”与“▲”表示。红色虚线显示出簇划分。

```

输入: 样本集  $D = \{x_1, x_2, \dots, x_m\}$ ;
聚类簇距离度量函数  $d_i$ ;
聚类簇数  $k$ 。
过程:
1: for  $j = 1, 2, \dots, m$  do
2:    $C_j = \{x_j\}$ 
3: end for
4: for  $i = 1, 2, \dots, m$  do
5:   for  $j = i+1, \dots, m$  do
6:      $M(i, j) = d(C_i, C_j)$ ;
7:      $M(j, i) = M(i, j)$ 
8:   end for
9: end for
10: 设置当前聚类簇个数:  $q = m$ 
11: while  $q > k$  do
12:   找出距离最近的两个聚类簇  $C_{j^*}$  和  $C_{j^*}$ ;
13:   合并  $C_{j^*}$  和  $C_{j^*}$ :  $C_{j^*} = C_{j^*} \cup C_{j^*}$ ;
14:   for  $j = j^* + 1, j^* + 2, \dots, q$  do
15:     将聚类簇  $C_j$  重编号为  $C_{j-1}$ 
16:   end for
17:   删除距离矩阵  $M$  的第  $j^*$  行与第  $j^*$  列;
18:   for  $i = 1, 2, \dots, q-1$  do
19:      $M(i, j^*) = d(C_i^*, C_{j^*})$ 
20:   end for
21:    $q = q - 1$ 
22: end while
23: 输出: 聚类簇划分  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$ 

```

图 9.11 AGNES 算法

