# 第八章 集成学习

王博：自动化（人工智能）学院
wangbo@hdu.edu.cn

# 集成学习

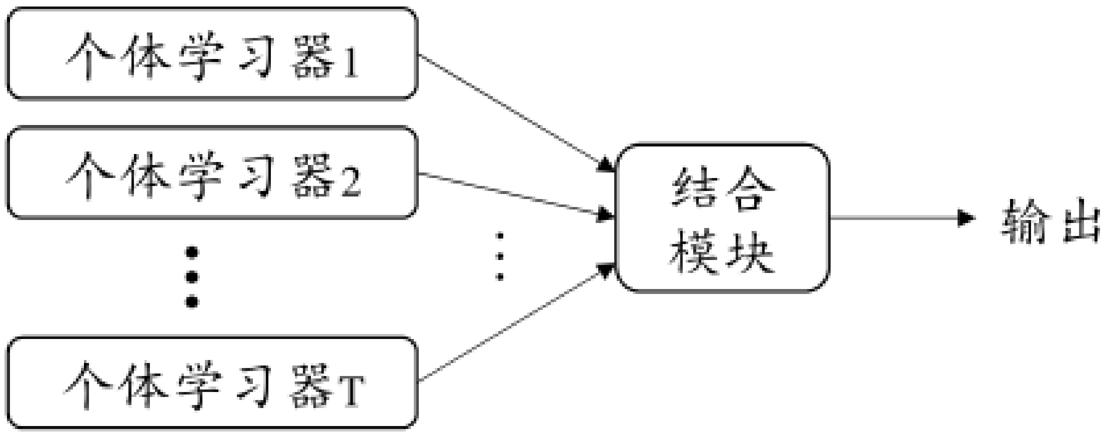
- 个体与集成
- Boosting
  - Adaboost
- Bagging与随机森林
- 结合策略
  - 平均法
  - 投票法
  - 学习法
- 多样性
  - 误差-分歧分解
  - 多样性度量
  - 多样性扰动

# 集成学习

- 个体与集成
- Boosting
  - Adaboost
- Bagging与随机森林
- 结合策略
  - 平均法
  - 投票法
  - 学习法
- 多样性
  - 误差-分歧分解
  - 多样性度量
  - 多样性扰动

# 个体与集成

- 集成学习(ensemble learning)通过构建并结合多个学习器来提升性能



# 个体与集成

- 考虑一个简单的例子，在二分类问题中，假定3个分类器在三个样本中的表现如下图所示，其中√ 表示分类正确，X 号表示分类错误，集成的结果通过投票产生。

| | 测试例1 | 测试例2 | 测试例3 |
|---|---|---|---|
| $h_1$ | √ | √ | × |
| $h_2$ | × | √ | √ |
| $h_3$ | √ | × | √ |
| 集群 | √ | √ | √ |

(a) 集群提升性能

| | 测试例1 | 测试例2 | 测试例3 |
|---|---|---|---|
| $h_1$ | √ | √ | × |
| $h_2$ | √ | √ | × |
| $h_3$ | √ | √ | × |
| 集群 | √ | √ | × |

(b) 集群不起作用

| | 测试例1 | 测试例2 | 测试例3 |
|---|---|---|---|
| $h_1$ | √ | × | × |
| $h_2$ | × | √ | × |
| $h_3$ | × | × | √ |
| 集群 | × | × | × |

(c) 集群起负作用

- 集成个体应：好而不同

# 个体与集成 – 简单分析

- 考虑二分类问题，假设基分类器的错误率为：

$$P\left(h_i\left(\boldsymbol{x}\right) \neq f\left(\boldsymbol{x}\right)\right) = \epsilon$$

- 假设集成通过简单投票法结合$T$个分类器，若有超过半数的基分类器正确则分类就正确

$$H(\boldsymbol{x}) = \operatorname{sign}\left(\sum_{i=1}^{T} h_i\left(\boldsymbol{x}\right)\right)$$

# 个体与集成 － 简单分析

- 假设基分类器的错误率相互独立，则由Hoeffding不等式可得集成的错误率为：

$$
\begin{aligned}
P(H(\boldsymbol{x}) \neq f(\boldsymbol{x})) &= \sum_{k=0}^{\lfloor T/2 \rfloor} \binom{T}{k} (1-\epsilon)^k \epsilon^{T-k} \\
&\leq \exp\left(-\frac{1}{2} T (1-2\epsilon)^2\right)
\end{aligned}
$$

- 上式显示，在一定条件下，随着集成分类器数目的增加，集成的错误率将指数级下降，最终趋向于0

# 个体与集成 － 简单分析

➢ 推导过程

$$x_i \sim \mathcal{B}(1, 1-\epsilon)$$

$$\mathrm{X} = \sum_{i=1}^{\mathrm{T}} x_i$$

$$\mathbb{E}(X) = \sum_{i=1}^{\mathrm{T}} \mathbb{E}(x_i) = (1-\epsilon)T$$

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \le -t\right) \le \exp\left(-\frac{2nt^2}{(b-a)^2}\right)$$

$$\begin{aligned}
P(H(x) \neq f(x)) =& P(X \le \lfloor T/2 \rfloor) \\
&\leqslant P(X \le T/2) \\
&= P\left[X - (1-\epsilon)T \leqslant \frac{T}{2} - (1-\epsilon)T\right] \\
&= P\left[X - (1-\epsilon)T \leqslant -\frac{T}{2}(1-2\epsilon)\right] \\
&= P\left[\sum_{i=1}^{\mathrm{T}} x_i - \sum_{i=1}^{\mathrm{T}} \mathbb{E}(x_i) \leqslant -\frac{T}{2}(1-2\epsilon)\right] \\
&= P\left[\frac{1}{\mathrm{T}}\sum_{i=1}^{\mathrm{T}} x_i - \frac{1}{\mathrm{T}}\sum_{i=1}^{\mathrm{T}} \mathbb{E}(x_i) \leqslant -\frac{1}{2}(1-2\epsilon)\right]
\end{aligned}$$

# 个体与集成 – 简单分析

- 上述结论关键假设：基学习器的误差相互独立
- 实际个体学习器是为解决同一个问题训练出来的，显然不可能互相独立

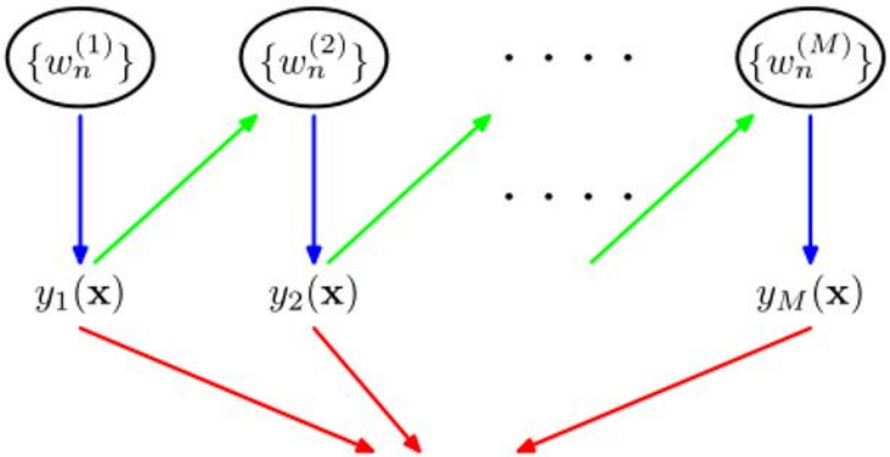
- 如何产生“好而不同”的个体学习器是集成学习研究的核心
- 集成学习大致可分为两大类
  - 个体学习存在强依赖关系，必须串行生成的序列化方法Boosting算法
  - 不存在强依赖关系，可同时生成的并行化方法Bagging和Random Forest

# 集成学习

- 个体与集成
- **Boosting**
  - Adaboost
- Bagging与随机森林
- 结合策略
  - 平均法
  - 投票法
  - 学习法
- 多样性
  - 误差-分歧分解
  - 多样性度量
  - 多样性扰动

# Boosting

- 个体学习器存在强依赖关系，
- 串行生成
- 每次调整训练数据的样本分布




$$Y_M(x) = sign(\sum_{m=1}^{M} \alpha_m y_m(x))$$

# Boosting – AdaBoost算法

**输入:** 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
基学习算法 $\mathfrak{L}$;
训练轮数 $T$.

**过程:**

1: $\mathcal{D}_1(\boldsymbol{x}) = 1/m$.
2: **for** $t = 1, 2, \ldots, T$ **do**
3: $h_t = \mathfrak{L}(D, \mathcal{D}_t)$;
4: $\epsilon_t = P_{\boldsymbol{x} \sim \mathcal{D}_t}(h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}))$;
5: **if** $\epsilon_t > 0.5$ **then break**
6: $\alpha_t = \frac{1}{2} \ln \left( \frac{1-\epsilon_t}{\epsilon_t} \right)$;
7: $$\begin{aligned} \mathcal{D}_{t+1}(\boldsymbol{x}) &= \frac{\mathcal{D}_t(\boldsymbol{x})}{Z_t} \times \begin{cases} \exp(-\alpha_t), & \text{if } h_t(\boldsymbol{x}) = f(\boldsymbol{x}) \\ \exp(\alpha_t), & \text{if } h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}) \end{cases} \\ &= \frac{\mathcal{D}_t(\boldsymbol{x})\exp(-\alpha_t f(\boldsymbol{x}) h_t(\boldsymbol{x}))}{Z_t} \end{aligned}$$
8: **end for**

**输出:** $F(x) = \text{sign}\left( \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \right)$

# Boosting – AdaBoost推导

- 基学习器的线性组合

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

- 最小化指数损失函数

$$\ell_{\exp}(H \mid \mathcal{D}) = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}}[e^{-f(\boldsymbol{x})H(\boldsymbol{x})}]$$

# Boosting – AdaBoost推导

- 若 $H(x)$ 能令指数损失函数最小化，则上式对 $H(x)$ 的偏导值为0，即

$$\frac{\partial \ell_{\exp}(H \mid \mathcal{D})}{\partial H(\boldsymbol{x})} = -e^{-H(\boldsymbol{x})} P(f(\boldsymbol{x}) = 1 \mid \boldsymbol{x}) + e^{H(\boldsymbol{x})} P(f(\boldsymbol{x}) = -1 \mid \boldsymbol{x})$$

$$H(\boldsymbol{x}) = \frac{1}{2} \ln \frac{P(f(x) = 1 \mid \boldsymbol{x})}{P(f(x) = -1 \mid \boldsymbol{x})}$$

$$\begin{aligned}
\operatorname{sign}(H(\boldsymbol{x})) &= \operatorname{sign}\left(\frac{1}{2} \ln \frac{P(f(x) = 1 \mid \boldsymbol{x})}{P(f(x) = -1 \mid \boldsymbol{x})}\right) \\
&= \begin{cases} 1, & P(f(x) = 1 \mid \boldsymbol{x}) > P(f(x) = -1 \mid \boldsymbol{x}) \\ -1, & P(f(x) = 1 \mid \boldsymbol{x}) < P(f(x) = -1 \mid \boldsymbol{x}) \end{cases} \\
&= \underset{y \in \{-1,1\}}{\arg\max}\, P(f(x) = y \mid \boldsymbol{x}),
\end{aligned}$$

$sign(H(x))$达到了贝叶斯最优错误率。

# Boosting – AdaBoost推导

- 当基分类器$h_t$基于分布$D_t$产生后，该基分类器的权重$\alpha_t$应使得$\alpha_t h_t$最小化指数损失函数

$$
\begin{aligned}
\ell_{\exp}\left(\alpha_t h_t \mid \mathcal{D}_t\right) &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}\right] \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[e^{-\alpha_t}\mathbb{I}\left(f\left(\boldsymbol{x}\right)=h_t\left(\boldsymbol{x}\right)\right)+e^{\alpha_t}\mathbb{I}\left(f\left(\boldsymbol{x}\right)\neq h_t\left(\boldsymbol{x}\right)\right)\right] \\
&= e^{-\alpha_t}P_{\boldsymbol{x}\sim\mathcal{D}_t}\left(f\left(\boldsymbol{x}\right)=h_t\left(\boldsymbol{x}\right)\right)+e^{\alpha_t}P_{\boldsymbol{x}\sim\mathcal{D}_t}\left(f\left(\boldsymbol{x}\right)\neq h_t\left(\boldsymbol{x}\right)\right) \\
&= e^{-\alpha_t}\left(1-\epsilon_t\right)+e^{\alpha_t}\epsilon_t
\end{aligned}
\qquad \epsilon_t = P_{\boldsymbol{x}\sim\mathcal{D}_t}\left(h_t(\boldsymbol{x})\neq f(\boldsymbol{x})\right)
$$

- 令指数损失函数的导数为0，即

$$
\frac{\partial \ell_{\exp}(\alpha_t h_t \mid \mathcal{D}_t)}{\partial \alpha_t} = -e^{-\alpha_t}(1-\epsilon_t)+e^{\alpha_t}\epsilon_t \qquad \alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)
$$

# Boosting – AdaBoost推导

- 在获得$H_{t-1}$之后的样本分布进行调整，使得下一轮的基学习器$h_t$能纠正$H_{t-1}$的一些错误，理想的$h_t$能纠正全部错误

$$\ell_{\exp}(H_{t-1}+h_t \mid \mathcal{D}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})(H_{t-1}(\boldsymbol{x})+h_t(\boldsymbol{x}))}]$$
$$= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}e^{-f(\boldsymbol{x})h_t(\boldsymbol{x})}]$$

- 0处泰勒展开近似为

$$\ell_{\exp}(H_{t-1}+h_t \mid \mathcal{D}) \simeq \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\left(1-f(\boldsymbol{x})h_t(\boldsymbol{x})+\frac{f^2(\boldsymbol{x})h_t^2(\boldsymbol{x})}{2}\right)\right]$$
$$= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\left(1-f(\boldsymbol{x})h_t(\boldsymbol{x})+\frac{1}{2}\right)\right]$$

# Boosting – AdaBoost推导

➤ 理想的基学习器：

$$
\begin{aligned}
h_t(\boldsymbol{x}) &= \arg\min_h \ell_{\exp}(H_{t-1}+h \mid \mathcal{D}) \\
&= \arg\min_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\left(1-f(\boldsymbol{x})h(\boldsymbol{x})+\frac{1}{2}\right)\right] \\
&= \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}f(\boldsymbol{x})h(\boldsymbol{x})\right] \\
&= \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\frac{e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]}f(\boldsymbol{x})h(\boldsymbol{x})\right],
\end{aligned}
$$

➤ 注意到 $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]$ 是一个常数，令Dt表示一个分布：

$$
\mathcal{D}_t(\boldsymbol{x}) = \frac{\mathcal{D}(\boldsymbol{x})e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]}
$$

# Boosting – AdaBoost推导

- 根据数学期望的定义，这等价于令:

$$
\begin{aligned}
h_t(\boldsymbol{x}) &= \underset{h}{\arg\max}\, \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\frac{e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]} f(\boldsymbol{x})h(\boldsymbol{x})\right] \\
&= \underset{h}{\arg\max}\, \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[f(\boldsymbol{x})h(\boldsymbol{x})\right] \ .
\end{aligned}
$$

- 由$f(x), h(x) \in \{-1, +1\}$有:

$$
f(\boldsymbol{x})h(\boldsymbol{x}) = 1 - 2\,\mathbb{I}\big(f(\boldsymbol{x}) \neq h(\boldsymbol{x})\big)
$$

# Boosting – AdaBoost推导

- 则理想的基学习器

$$h_t(\boldsymbol{x}) = \arg\min_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[\mathbb{I}\big(f(\boldsymbol{x}) \neq h(\boldsymbol{x})\big)\right]$$

- 最终的样本分布更新公式

$$\begin{aligned}
\mathcal{D}_{t+1}(\boldsymbol{x}) &= \frac{\mathcal{D}(\boldsymbol{x})\, e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}\right]} \\
&= \frac{\mathcal{D}(\boldsymbol{x})\, e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})} e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}\right]} \\
&= \mathcal{D}_t(\boldsymbol{x}) \cdot e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})} \frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\right]}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}\right]}
\end{aligned}$$

# Boosting – AdaBoost说明

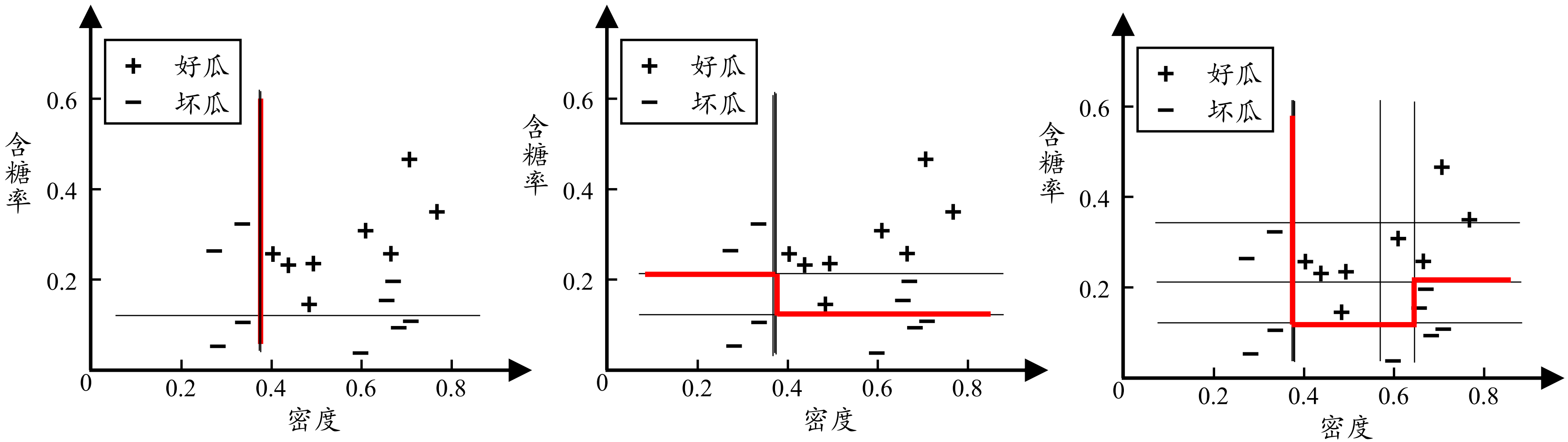
错误率与数据分布的关系？？？

- 数据分布的学习
  - 重赋权法

权重与对应的分类器

$$\alpha_m h_m(x) \qquad \alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

# Boosting – AdaBoost实验




(a) 3个基学习器 (b) 5个基学习器 (c) 11个基学习器

➢ 从偏差-方差的角度：降低偏差，可对泛化性能相当弱的学习器构造出很强的集成

# 集成学习

- 个体与集成
- **Boosting**
  - Adaboost
- **Bagging与随机森林**
- 结合策略
  - 平均法
  - 投票法
  - 学习法
- 多样性
  - 误差-分歧分解
  - 多样性度量
  - 多样性扰动

# Bagging与随机森林

- 个体学习器不存在强依赖关系
- 并行化生成
- 自助采样法

# Bagging与随机森林 － Bagging算法

➢ 自助采样法

**输入:** 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
基学习算法 $\mathfrak{L}$;
训练轮数 $T$.

**过程:**

1: **for** $t = 1, 2, \ldots, T$ **do**

2: $\quad h_t = \mathfrak{L}(D, \mathcal{D}_{bs})$

3: **end for**

**输出:** $H(\boldsymbol{x}) = \underset{y \in \mathcal{Y}}{\arg\max} \sum_{t=1}^{T} \mathbb{I}(h_t(\boldsymbol{x}) = y)$

# Bagging与随机森林 – Bagging算法特点

- 时间复杂度低
  - 假定基学习器的计算复杂度为O(m)，采样与投票/平均过程的复杂度为O(s)，则bagging的复杂度大致为T(O(m)+O(s))
  - 由于O(s)很小且T是一个不大的常数
  - 因此训练一个bagging集成与直接使用基学习器的复杂度同阶
- 可使用包外（没有选择到的数据）估计

# Bagging与随机森林 – 包外估计

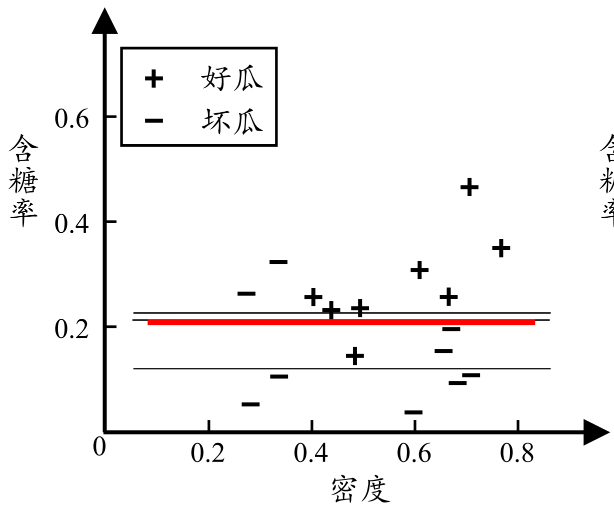
- $H^{oob}$(x)表示对样本x的包外预测，即仅考虑那些未使用样本x训练的基学习器在x上的预测

$$H^{oob}(\boldsymbol{x}) = \underset{y \in \mathcal{Y}}{\arg\max} \sum_{t=1}^{T} \mathbb{I}(h_t(\boldsymbol{x}) = y) \cdot \mathbb{I}(\boldsymbol{x} \notin D_t)$$

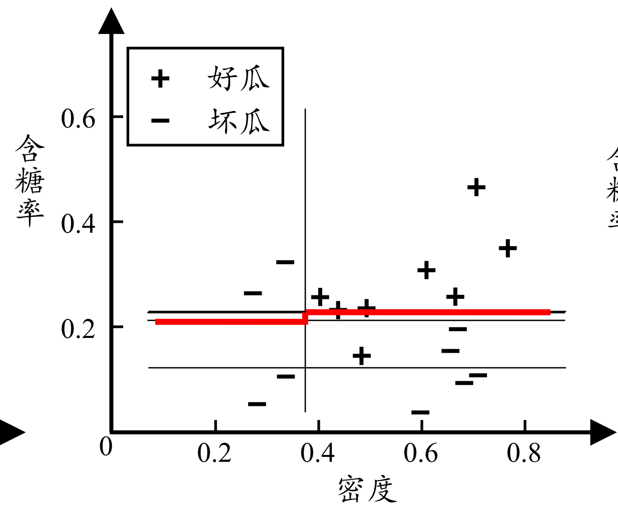
- Bagging泛化误差的包外估计为：

$$\epsilon^{oob} = \frac{1}{|D|} \sum_{(\boldsymbol{x},y) \in D} \mathbb{I}(H^{oob}(\boldsymbol{x}) \neq y)$$

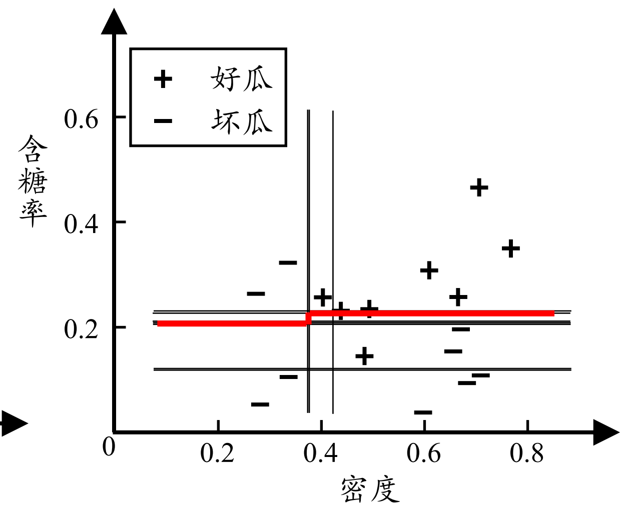
# Bagging与随机森林- Bagging实验



(a) 3个基学习器



(b) 5个基学习器



(c) 11个基学习器

- 从偏差-方差的角度：降低方差，在不剪枝的决策树、神经网络等易受样本影响的学习器上效果更好

# Bagging与随机森林-随机森林

- 随机森林(Random Forest，简称RF)是bagging的一个扩展变种
- 采样的随机性
- 属性选择的随机性

# Bagging与随机森林 – 随机森林算法

## ➢ 随机森林算法

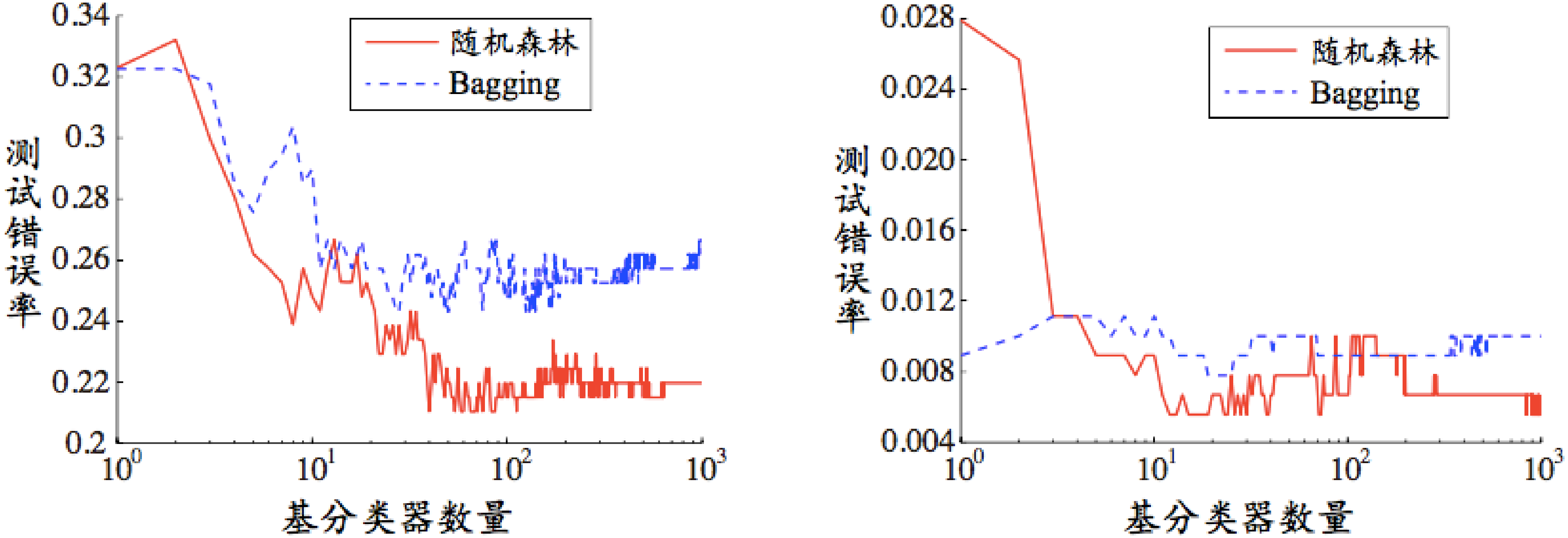
**Input:** Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
Feature subset size $K$.

**Process:**

1. $N \leftarrow$ create a tree node based on $D$;
2. **if** *all instances in the same class* **then return** $N$
3. $\mathcal{F} \leftarrow$ the set of features that can be split further;
4. **if** $\mathcal{F}$ *is empty* **then return** $N$
5. $\tilde{\mathcal{F}} \leftarrow$ select $K$ features from $\mathcal{F}$ randomly;
6. $N.f \leftarrow$ the feature which has the best split point in $\tilde{\mathcal{F}}$;
7. $N.p \leftarrow$ the best split point on $N.f$;
8. $D_l \leftarrow$ subset of $D$ with values on $N.f$ smaller than $N.p$ ;
9. $D_r \leftarrow$ subset of $D$ with values on $N.f$ no smaller than $N.p$ ;
10. $N_l \leftarrow$ call the process with parameters $(D_l, K)$;
11. $N_r \leftarrow$ call the process with parameters $(D_r, K)$;
12. **return** $N$

**Output:** A random decision tree

# Bagging与随机森林 – 随机森林实验



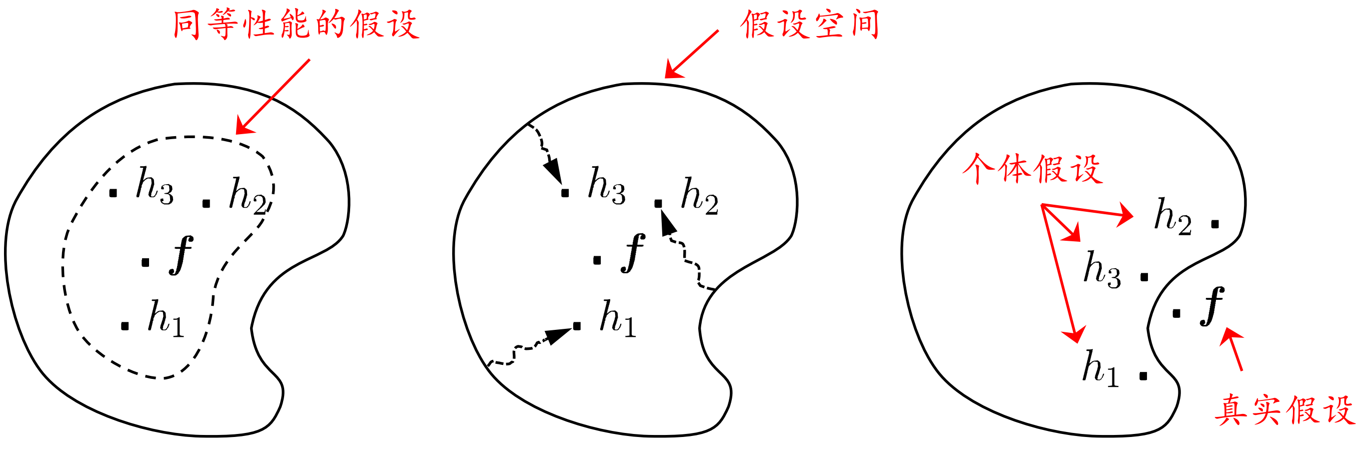

(a) glass 数据集　　(b) auto-mpg 数据集

# 集成学习

- 个体与集成
- Boosting
  - Adaboost
- Bagging与随机森林
- 结合策略
  - 平均法
  - 投票法
  - 学习法
- 多样性
  - 误差-分歧分解
  - 多样性度量
  - 多样性扰动

# 结合策略

- 学习器的组合可以从三个方面带来好处




(a) 统计的原因　(b) 计算的原因　(c) 表示的原因

统计方面，多个假设在训练集上达到同等性能；

计算方面，陷入局部最优

表示方面，学习任务的真实假设不在当前算法的假设空间

# 结合策略 – 平均法

- 简单平均法

$$H(\boldsymbol{x}) = \frac{1}{T}\sum_{i=1}^{T} h_i(\boldsymbol{x}).$$

- 加权平均法

$$H(\boldsymbol{x}) = \sum_{i=1}^{T} w_i h_i(\boldsymbol{x}) , \qquad w_i \geq 0 \ \text{ and } \ \sum_{i=1}^{T} w_i = 1 .$$

# 结合策略－平均法

- 简单平均法是加权平均法的特例
- 加权平均法在二十世纪五十年代被广泛使用
- 集成学习中的各种结合方法都可以看成是加权平均法的变种或特例
- 加权平均法可认为是集成学习研究的基本出发点
- 加权平均法未必一定优于简单平均法

# 结合策略 - 投票法

- 绝对多数投票法（majority voting）

$$H(\boldsymbol{x}) = \begin{cases} c_j & \text{if } \sum_{i=1}^{T} h_i^j(\boldsymbol{x}) > \frac{1}{2} \sum_{k=1}^{l} \sum_{i=1}^{T} h_i^k(\boldsymbol{x}) \\ \text{rejection} & \text{otherwise}. \end{cases}$$

- 相对多数投票法（plurality voting）

$$H(\boldsymbol{x}) = c_{\underset{j}{\arg\max} \sum_{i=1}^{T} h_i^j(\boldsymbol{x})}$$

- 加权投票法（weighted voting）

$$H(\boldsymbol{x}) = c_{\underset{j}{\arg\max} \sum_{i=1}^{T} w_i h_i^j(\boldsymbol{x})}$$

# 结合策略－学习法

- Stacking是学习法的典型代表

**Input:** Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
First-level learning algorithms $\mathfrak{L}_1, \ldots, \mathfrak{L}_T$;
Second-level learning algorithm $\mathfrak{L}$.

**Process:**

1. **for** $t = 1, \ldots, T$: % Train a first-level learner by applying the
2. $h_t = \mathfrak{L}_t(D)$; % first-level learning algorithm $\mathfrak{L}_t$
3. **end**
4. $D' = \emptyset$; % Generate a new data set
5. **for** $i = 1, \ldots, m$:
6. **for** $t = 1, \ldots, T$:
7. $z_{it} = h_t(\boldsymbol{x}_i)$;
8. **end**
9. $D' = D' \cup ((z_{i1}, \ldots, z_{iT}), y_i)$;
10. **end**
11. $h' = \mathfrak{L}(D')$; % Train the second-level learner $h'$ by applying % the second-level learning algorithm $\mathfrak{L}$ to the % new data set $D'$.

**Output:** $H(\boldsymbol{x}) = h'(h_1(\boldsymbol{x}), \ldots, h_T(\boldsymbol{x}))$

多响应线性回归(MLR)作为次级学习器的学习算法 效果较好

- 贝叶斯模型平均(BMA)

# 集成学习

- 个体与集成
- Boosting
  - Adaboost
- Bagging与随机森林
- 结合策略
  - 平均法
  - 投票法
  - 学习法
- 多样性
  - 误差-分歧分解
  - 多样性度量
  - 多样性扰动

# 多样性－误差-分歧分解

- 定义学习器$h_i$的分歧(ambiguity)：

$$A(h_i \mid \boldsymbol{x}) = \left(h_i(\boldsymbol{x}) - H(\boldsymbol{x})\right)^2$$

- 集成的分歧：

$$\begin{aligned}\overline{A}(h \mid \boldsymbol{x}) &= \sum\nolimits_{i=1}^{T} w_i A(h_i \mid \boldsymbol{x}) \\ &= \sum\nolimits_{i=1}^{T} w_i \left(h_i(\boldsymbol{x}) - H(\boldsymbol{x})\right)^2\end{aligned}$$

# 多样性 – 误差-分歧分解

- 分歧项代表了个体学习器在样本$x$上的不一致性，即在一定程度上反映了个体学习器的多样性，个体学习器$h_i$和集成$H$的平方误差分别为

$$E(h_i \mid \boldsymbol{x}) = \left(f(\boldsymbol{x}) - h_i(\boldsymbol{x})\right)^2$$

$$E(H \mid \boldsymbol{x}) = \left(f(\boldsymbol{x}) - H(\boldsymbol{x})\right)^2$$

# 多样性 – 误差-分歧分解

- 令 $\overline{E}(h \mid \boldsymbol{x}) = \sum_{i=1}^{T} w_i \cdot E(h_i \mid \boldsymbol{x})$ 表示个体学习器误差的加权均值，有

$$\begin{aligned}\overline{A}(h \mid \boldsymbol{x}) &= \sum_{i=1}^{T} w_i E(h_i \mid \boldsymbol{x}) - E(H \mid \boldsymbol{x}) \\ &= \overline{E}(h \mid \boldsymbol{x}) - E(H \mid \boldsymbol{x}) .\end{aligned}$$

- 上式对所有样本$\boldsymbol{x}$均成立，令$\boldsymbol{p}(\boldsymbol{x})$表示样本的概率密度，则在全样本上有

$$\sum_{i=1}^{T} w_i \int A(h_i \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x} = \sum_{i=1}^{T} w_i \int E(h_i \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x} - \int E(H \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}$$

# 多样性 – 误差-分歧分解

- 个体学习器$h_i$在全样本上的泛化误差和分歧项分别为：

$$E_i = \int E(h_i \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}$$

$$A_i = \int A(h_i \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}$$

- 集成的泛化误差为：

$$E = \int E(H \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}$$

- 令 $\overline{E} = \sum_{i=1}^{T} w_i E_i$表示个体学习器泛化误差的加权均值，$\overline{A} = \sum_{i=1}^{T} w_i A_i$ 表示个体学习器的加权分歧值，有

$$E = \overline{E} - \overline{A}$$

# 多样性 – 误差-分歧分解

- 这个漂亮的式子显示:个体学习器精确性越高、多样性越大，则集成效果越好。称为误差-分歧分解

- 为什么不能直接把$\bar{E}-\bar{A}$作为优化目标来求解?
  - 现实任务中很难直接对$\bar{E}-\bar{A}$进行优化，
    - 它们定义在整个样本空间上
    - $\bar{A}$不是一个可直接操作的多样性度量
    - 上面的推导过程只适用于回归学习，难以直接推广到分类学习任务上去

# 多样性 – 多样性度量

- 多样性度量(diversity measure)用于度量集成中个体学习器的多样性
- 对于二分类问题，分类器$h_i$与$h_j$的预测结果联立表(contingency table)为

| | $h_i=+1$ | $h_i=-1$ |
|---|---|---|
| $h_j=+1$ | $a$ | $c$ |
| $h_j=-1$ | $b$ | $d$ |

$$a+b+c+d=m$$

# 多样性 – 多样性度量

## ➢ 常见的多样性度量

- 不合度量(Disagreement Measure)

$$dis_{ij} = \frac{b+c}{m}$$

- 相关系数(Correlation Coefficient)

$$\rho_{ij} = \frac{ad-bc}{\sqrt{(a+b)(a+c)(c+d)(b+d)}}$$

# 多样性 – 多样性度量

## ➢ 常见的多样性度量

- Q-统计量(Q-Statistic)

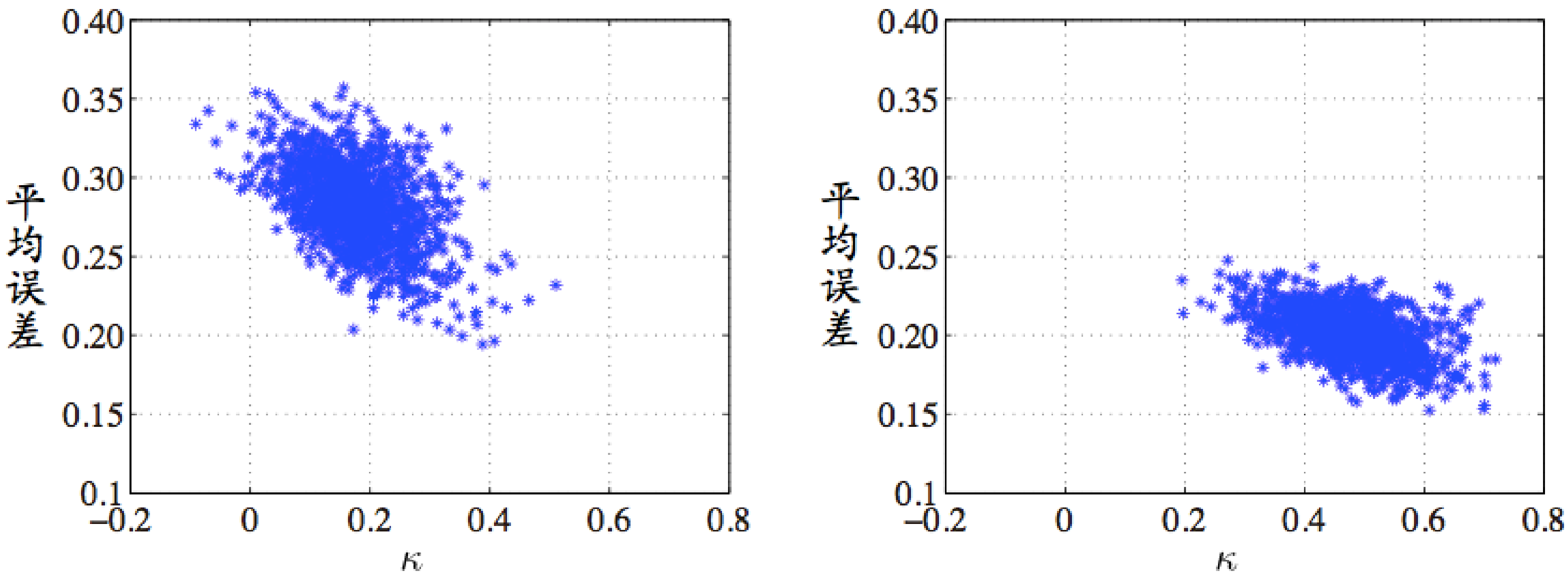
$$Q_{ij} = \frac{ad - bc}{ad + bc} \qquad |Q_{ij}| \le |\rho_{ij}|$$

- K-统计量(Kappa-Statistic)

$$\kappa = \frac{p_1 - p_2}{1 - p_2} \qquad \begin{aligned} p_1 &= \frac{a+d}{m}, \\ p_2 &= \frac{(a+b)(a+c)+(c+d)(b+d)}{m^2} \end{aligned}$$

# 多样性 – 多样性度量

- $k$ –误差图




(a) AdaBoost 集成　　(b) Bagging 集成

# 多样性 － 多样性增强

- 常见的增强个体学习器的多样性的方法
  - 数据样本扰动
  - 输入属性扰动
  - 输出表示扰动
  - 算法参数扰动

# 多样性 – 多样性增强 – 数据样本扰动

- 数据样本扰动通常是基于采样法
  - Bagging中的自助采样法
  - Adaboost中的序列采样

数据样本扰动对“不稳定基学习器”很有效

- 对数据样本的扰动敏感的基学习器(不稳定基学习器)
  - 决策树，神经网络等
- 对数据样本的扰动不敏感的基学习器(稳定基学习器)
  - 线性学习器，支持向量机，朴素贝叶斯，k近邻等

# 多样性 – 多样性增强 – 输入属性扰动

- 随机子空间算法(random subspace)

---

**输入:** 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_m, y_m)\}$;
基学习算法 $\mathfrak{L}$;
基学习器数 $T$;
子空间属性数 $d'$.

**过程:**

1: **for** $t = 1, 2, \ldots, T$ **do**
2: $\mathcal{F}_t = \mathrm{RS}(D, d')$
3: $D_t = \mathrm{Map}_{\mathcal{F}_t}(D)$
4: $h_t = \mathfrak{L}(D_t)$
5: **end for**

**输出:** $H(\boldsymbol{x}) = \underset{y \in \mathcal{Y}}{\arg\max} \sum_{t=1}^{T} \mathbb{I}\left(h_t\left(\mathrm{Map}_{\mathcal{F}_t}(\boldsymbol{x})\right) = y\right)$

---

# 多样性 – 多样性增强 – 输出表示扰动

- 翻转法(Flipping Output)
- 输出调剂法(Output Smearing)
- ECOC法

# 多样性 － 多样性增强 － 算法参数扰动

- 负相关法

- 不同的多样性增强机制同时使用

# 阅读材料

- 集成学习方面的主要推荐读物是[Zhou,2012]，本章提及的所有内容在该书中都有更深入的详细介绍。[Kuncheva,2004;Rockach,2010b]可供参考，[Schapire and Freund,2012]则是专门关于Boosting的著作，集成学习方面有一些专门性的会议MCS(International Workshop on Multiple Classifier System).

- Boosting源于[Schapire,1990]对[Kearns and Valiant,1989]提出的”弱分类器是否等价于强学习”这个重要理论问题的构造性证明。最初的Boosting算法仅有理论意义，经数年努力后[Freund and Schapire,1997]提出Adaboost，并因此或得理论计算机科学方面的重要奖项—哥德尔奖。关于Boosting和Bagging已有的很多理论研究结果课参阅[Zhou,2012]第2-3章。