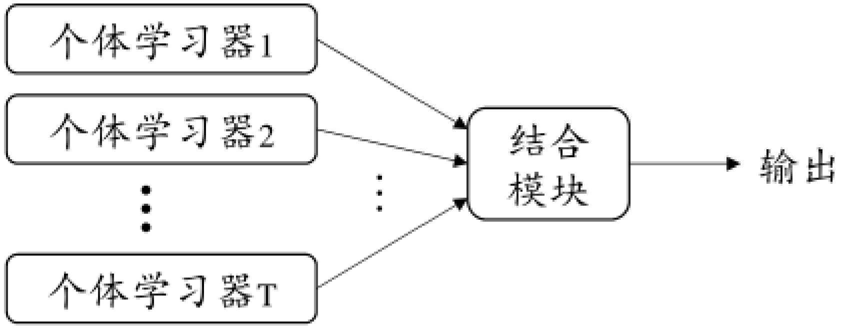
第八章：集成学习

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



- 同质集成 (homogeneous)：基学习器
- 异质集成 (heterogeneous)：组件学习器
- 弱学习器

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
  - 准确性
  - 多样性

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

设 x_i 为每一个分类器分类正确的次数，则 $x_i \sim \mathrm{B}(1, 1-\epsilon)$ $i = 1\ 2\ 3\ \ldots\ \mathrm{T}$,

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

$$P(H(x) \neq f(x)) = P(X \leq \lfloor T/2 \rfloor)$$

$$\leqslant P(X \leq T/2)$$

$$= P\left[X - (1-\epsilon)T \leqslant \frac{T}{2} - (1-\epsilon)T\right]$$

$$= P\left[X - (1-\epsilon)T \leqslant -\frac{T}{2}(1-2\epsilon)\right]$$

$$= P\left[\sum_{i=1}^{\mathrm{T}} x_i - \sum_{i=1}^{\mathrm{T}} \mathbb{E}(x_i) \leqslant -\frac{T}{2}(1-2\epsilon)\right]$$

$$= P\left[\frac{1}{\mathrm{T}}\sum_{i=1}^{\mathrm{T}} x_i - \frac{1}{\mathrm{T}}\sum_{i=1}^{\mathrm{T}} \mathbb{E}(x_i) \leqslant -\frac{1}{2}(1-2\epsilon)\right]$$

Hoeffding 不等式

$$\boxed{P\left(\frac{1}{m}\sum_{i=1}^{m} x_i - \frac{1}{m}\sum_{i=1}^{m} \mathbb{E}(x_i) \leqslant -\delta\right) \leqslant \exp\left(-2m\delta^2\right)}$$

令 $\delta = \frac{(1-2\epsilon)}{2}, m = T$ 得

$$P(H(\boldsymbol{x}) \neq f(\boldsymbol{x})) = \sum_{k=0}^{\lfloor T/2 \rfloor} \binom{T}{k} (1-\epsilon)^k \epsilon^{T-k}$$

$$\leqslant \exp\left(-\frac{1}{2}T(1-2\epsilon)^2\right)$$

个体与集成 – 简单分析

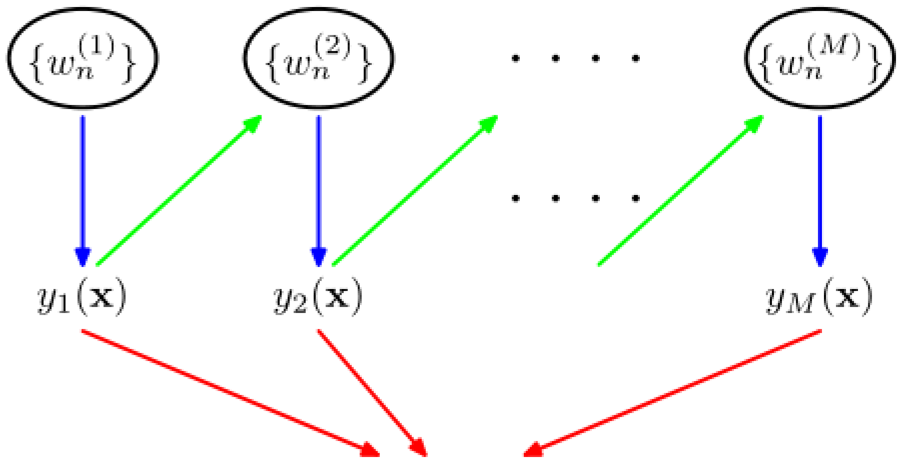
- 上面的分析有一个关键假设：基学习器的误差相互独立
- 现实任务中，个体学习器是为解决同一个问题训练出来的，显然不可能互相独立
- 事实上，个体学习器的“准确性”和“多样性”本身就存在冲突
- 如何产生“好而不同”的个体学习器是集成学习研究的核心
- 集成学习大致可分为两大类
  - Boosting
  - Bagging与随机森林

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

- 纠正弱分类器所犯的错误
- 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



$$Y_M(\mathbf{x}) = \mathrm{sign}\left(\sum_m^M \alpha_m y_m(\mathbf{x})\right)$$

Boosting - Boosting算法

Input: Sample distribution $\mathcal{D}$;
Base learning algorithm $\mathfrak{L}$;
Number of learning rounds T.

Process:

1. $\mathcal{D}_1 = \mathcal{D}$. % Initialize distribution
2. **for** $t = 1, \ldots, T$:
3. $h_t = \mathfrak{L}(\mathcal{D}_t)$; % Train a weak learner from distribution $\mathcal{D}_t$
4. $\epsilon_t = P_{\boldsymbol{x} \sim D_t}(h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}))$; % Evaluate the error of h_t
5. $\mathcal{D}_{t+1} = Adjust\_Distribution(\mathcal{D}_t, \epsilon_t)$
6. **end**

Output: $H(\boldsymbol{x}) = Combine\_Outputs(\{h_1(\boldsymbol{x}), \ldots, h_t(\boldsymbol{x})\})$

- Boosting族算法最著名的代表是AdaBoost

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

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，说明指数损失函数是分类任务原来0/1损失函数的一致的替代函数。

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α_t应使得$\alpha_t h_t$最小化指数损失函数

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

$$\epsilon_t = P_{\boldsymbol{x}\sim\mathcal{D}_t}\left(h_t(\boldsymbol{x})\neq f(\boldsymbol{x})\right)$$

- 令指数损失函数的导数为0，即

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

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

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

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

- 泰勒展开近似为

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

$\boxed{f^2(x)=h^2(x)=1}$

Boosting – AdaBoost推导

- 于是，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$D_t$ 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：

$$h_t(\boldsymbol{x}) = \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\frac{e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]} f(\boldsymbol{x})h(\boldsymbol{x})\right]$$

$$= \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[f(\boldsymbol{x})h(\boldsymbol{x})\right] .$$

- 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

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

- 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 – 例1

- 对下表的训练数据，用AdaBoost算法学习一个强分类器。假设弱分类器由 $x<v$ 或 $x>v$ 产生。

序号	1	2	3	4	5	6	7	8	9	10
x	0	1	2	3	4	5	6	7	8	9
y	1	1	1	-1	-1	-1	1	1	1	-1

初始化数据权值分布

$$D_1 = (w_{11}, w_{12}, \cdots, w_{110})$$

$$w_{1i} = 0.1\ , \quad i = 1, 2, \cdots, 10$$

分类误差率最低基分类器

$$G_1(x) = \begin{cases} 1, & x < 2.5 \\ -1, & x > 2.5 \end{cases}$$

Boosting – 例1

- 对下表的训练数据，用AdaBoost算法学习一个强分类器。假设弱分类器由 $x<v$ 或 $x>v$ 产生。

序号	1	2	3	4	5	6	7	8	9	10
x	0	1	2	3	4	5	6	7	8	9
y	1	1	1	-1	-1	-1	1	1	1	-1

$G_1(x)$ 在训练数据集上的误差率 $e_1 = P(G_1(x_i) \neq y_i) = 0.3$.

计算 $G_1(x)$ 的系数： $\alpha_1 = \frac{1}{2}\log\frac{1-e_1}{e_1} = 0.4236$.

更新训练数据的权值分布：

$$D_2 = (w_{21}, \cdots, w_{2i}, \cdots, w_{210})$$

$$w_{2i} = \frac{w_{1i}}{Z_1}\exp(-\alpha_1 y_i G_1(x_i)), \quad i = 1, 2, \cdots, 10$$

$$D_2 = (0.07143, 0.07143, 0.07143, 0.07143, 0.07143, 0.07143, 0.16667, 0.16667, 0.16667, 0.07143)$$

训练数据集上有 3 个误分类点

$$f_1(x) = 0.4236 G_1(x)$$

Boosting – 例1

- 对下表的训练数据，用AdaBoost算法学习一个强分类器。假设弱分类器由 $x<v$ 或 $x>v$ 产生。

序号	1	2	3	4	5	6	7	8	9	10
x	0	1	2	3	4	5	6	7	8	9
y	1	1	1	-1	-1	-1	1	1	1	-1

训练数据D_2上，分类误差率最低的基本分类器 $G_2(x)=\begin{cases}1, & x<8.5\\ -1, & x>8.5\end{cases}$

$G_2(x)$ 在训练数据集上的误差率 $e_2=0.2143$.

计算 $\alpha_2=0.6496$.

更新训练数据权值分布：

$$D_3=(0.0455, 0.0455, 0.0455, 0.1667, 0.1667, 0.1667, \\ 0.1060, 0.1060, 0.1060, 0.0455)$$

$$f_2(x)=0.4236G_1(x)+0.6496G_2(x)$$

训练数据集上有 3 个误分类点

Boosting – 例1

- 对下表的训练数据，用AdaBoost算法学习一个强分类器。假设弱分类器由 $x<v$ 或 $x>v$ 产生。

序号	1	2	3	4	5	6	7	8	9	10
x	0	1	2	3	4	5	6	7	8	9
y	1	1	1	-1	-1	-1	1	1	1	-1

训练数据D_3上，分类误差率最低的基本分类器 $G_3(x)=\begin{cases}1, & x>5.5\\ -1, & x<5.5\end{cases}$

$G_3(x)$ 在训练样本集上的误差率 $e_3=0.1820$.

计算 $\alpha_3=0.7514$.

$f_3(x)=0.4236G_1(x)+0.6496G_2(x)+0.7514G_3(x)$ 误分类点个数为0

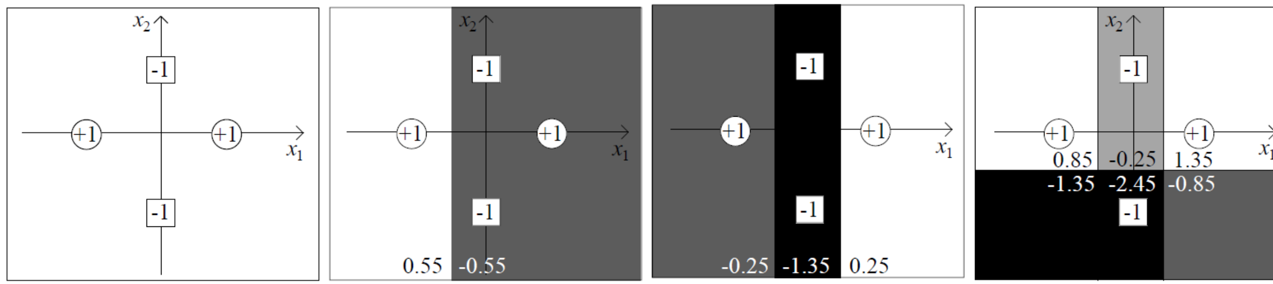
最终分类器为

$$G(x)=\text{sign}[f_3(x)]=\text{sign}[0.4236G_1(x)+0.6496G_2(x)+0.7514G_3(x)]$$

Boosting – 例2

- XOR问题的AdaBoost实现

$$\left\{\begin{array}{l}(\boldsymbol{z}_1=(+1,0), y_1=+1) \\ (\boldsymbol{z}_2=(-1,0), y_2=+1) \\ (\boldsymbol{z}_3=(0,+1), y_3=-1) \\ (\boldsymbol{z}_4=(0,-1), y_4=-1)\end{array}\right\}$$



(a) The XOR data (b) 1st round (c) 2nd round (d) 3rd round

基学习器集：

$$h_1(\boldsymbol{x})=\left\{\begin{array}{ll}+1, & \text{if } (x_1>-0.5) \\ -1, & \text{otherwise}\end{array}\right.$$

$$\boxed{h_2(\boldsymbol{x})=\left\{\begin{array}{ll}-1, & \text{if } (x_1>-0.5) \\ +1, & \text{otherwise}\end{array}\right.}$$

$$\boxed{h_3(\boldsymbol{x})=\left\{\begin{array}{ll}+1, & \text{if } (x_1>+0.5) \\ -1, & \text{otherwise}\end{array}\right.}$$

$$h_4(\boldsymbol{x})=\left\{\begin{array}{ll}-1, & \text{if } (x_1>+0.5) \\ +1, & \text{otherwise}\end{array}\right.$$

$$\boxed{h_5(\boldsymbol{x})=\left\{\begin{array}{ll}+1, & \text{if } (x_2>-0.5) \\ -1, & \text{otherwise}\end{array}\right.}$$

$$h_6(\boldsymbol{x})=\left\{\begin{array}{ll}-1, & \text{if } (x_2>-0.5) \\ +1, & \text{otherwise}\end{array}\right.$$

$$h_7(\boldsymbol{x})=\left\{\begin{array}{ll}+1, & \text{if } (x_2>+0.5) \\ -1, & \text{otherwise}\end{array}\right.$$

$$\boxed{h_8(\boldsymbol{x})=\left\{\begin{array}{ll}-1, & \text{if } (x_2>+0.5) \\ +1, & \text{otherwise}\end{array}\right.}$$

$t=1 \quad h_2: \; 0.5 \ln 3 \approx 0.55$

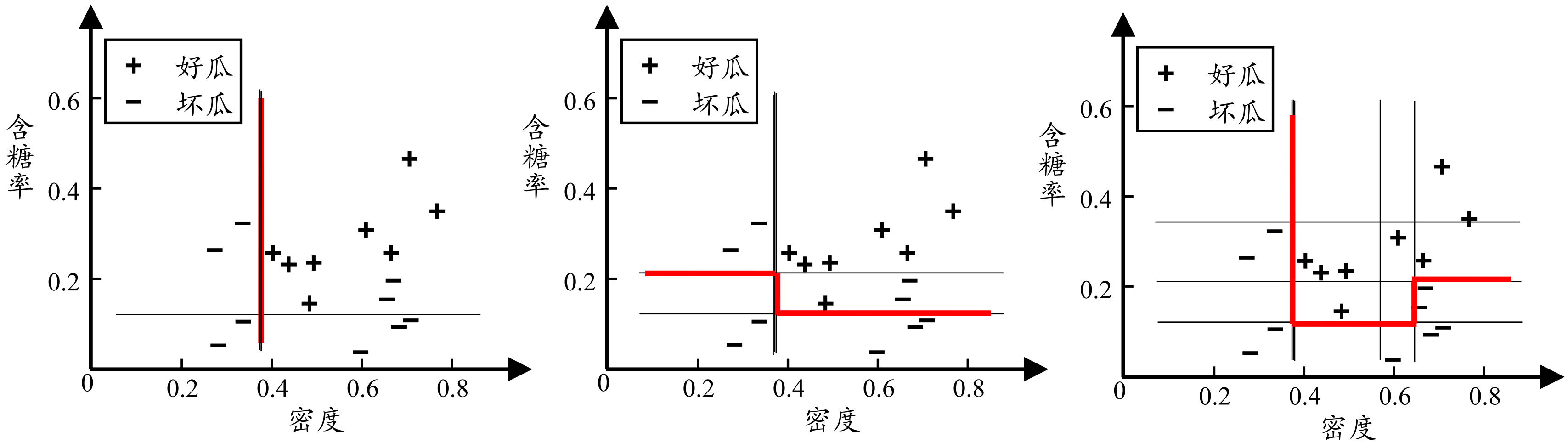
$t=2 \quad h_3: \; 0.8$

$t=3 \quad h_5: \; 1.1$

Boosting – AdaBoost注意事项

- 数据分布的学习
  - 重赋权法
  - 重采样法
- 重启动，避免训练过程过早停止

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

$$\lim_{x\to\infty}\left(1-\frac{1}{m}\right)^m=\frac{1}{e}=0.368$$

Bagging与随机森林 – Bagging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: **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

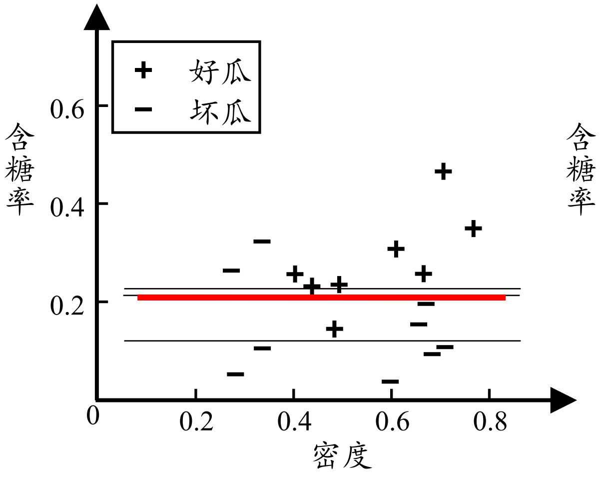
- $H^{oob}(x)$表示对样本x的包外预测，即仅考虑那些未使用样本x训练的基学习器在x上的预测

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

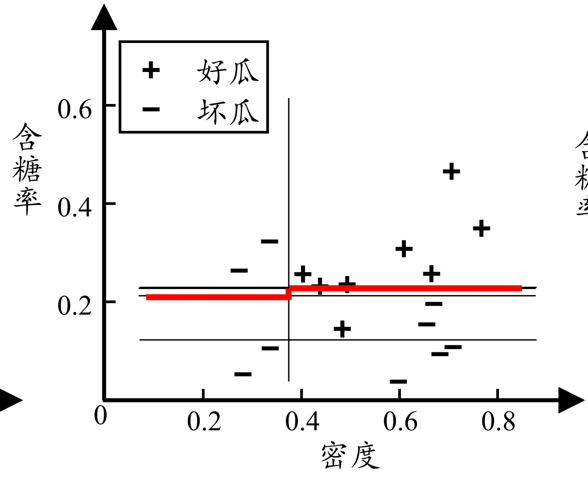
- Bagging泛化误差的包外估计为：

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

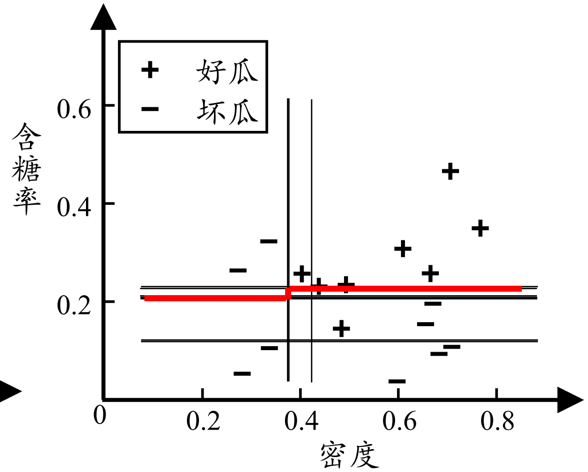
Bagging与随机森林- Bagging实验



(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- Bagging例子

- 现有一组某市房屋价格与房屋位置数据，其中X表示房屋到市中心的直线距离。试用Bagging集成学习方法构造一个包含三个线性回归模型的集成模型，并使用该集成模型预测距离市中心 5.5 千米的房屋价格

序号	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
X(km)	4.2	7.1	6.3	1.1	0.2	4.0	3.5	8	2.3
y(元/m^2)	8600	6100	6700	12000	14200	8500	8900	6200	11200

Bagging与随机森林- Bagging例子

- 通过对样本数据集D进行三次自助采样获得三个训练样本子集 D_1,D_2 , D_3

D_1

序号	1	2	3	4	5	6	7	8	9
$X(km)$	4.2	4.2	4.2	6.3	1.1	0.2	3.5	3.5	2.3
$y(元/m^2)$	8600	8600	8600	6700	12000	14200	8900	8900	11200

D_2

$X(km)$	4.2	4.2	7.1	7.1	1.1	4.0	4.0	3.5	2.3
$y(元/m^2)$	8600	8600	6100	6100	12000	8500	8500	8900	11200

D_3

$X(km)$	4.2	1.1	0.2	4.0	4.0	4.0	4.0	3.5	8
$y(元/m^2)$	8600	12000	14200	8500	8500	8500	8500	8900	6200

Bagging与随机森林- Bagging例子

- 假设线性回归模型为$L(X)=\theta_0 X+\theta_1$，则可分别通过训练集D_1,D_2 ，D_3构造相应的弱学习器L_1,L_2,L_3。使用最小二乘法,不难得到L_1,L_2,L_3的具体表达式:

$$L_1(X)=-1216.488X+13731.8219$$

$$L_2(X)=-984.0959X+12822.6216$$

$$L_3(X)=-1015.2945X+13044.9688$$

使用简单平均法集成L_1,L_2,L_3，得到如下$L(X)$集成模型:

$$L(X)=[L_1(X)+L_2(X)+L_3(X)]/3$$

Bagging与随机森林- Bagging例子

- $X = 5.5$ 时，可得：

$$L(5.5) = \frac{bias(L_1) + bias(L_2) + bias(L_2)}{3} = \mu$$

由此可见，通过Bagging集成学习产生的集成模型$L(X)$并未改善对弱回归器的预测偏差。假设三个弱回归器的预测方差var(L)均为σ^2，则集成模型L的预测方差为：

$$var(L) = var\left[\frac{l_1 + l_2 + l_3}{3}\right] = \frac{\sigma^2}{3}$$

即集成模型的预测方差仅为弱回归器预测方差的1/3。因此通过*Bagging*集成策略可以有效降低模型输出预测的方差。

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

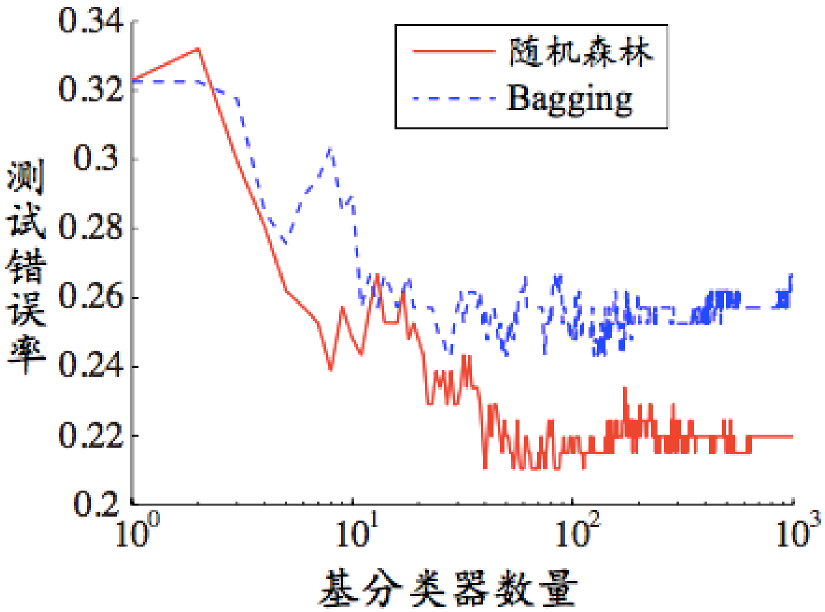
Input: Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
Feature subset size K.

Process:

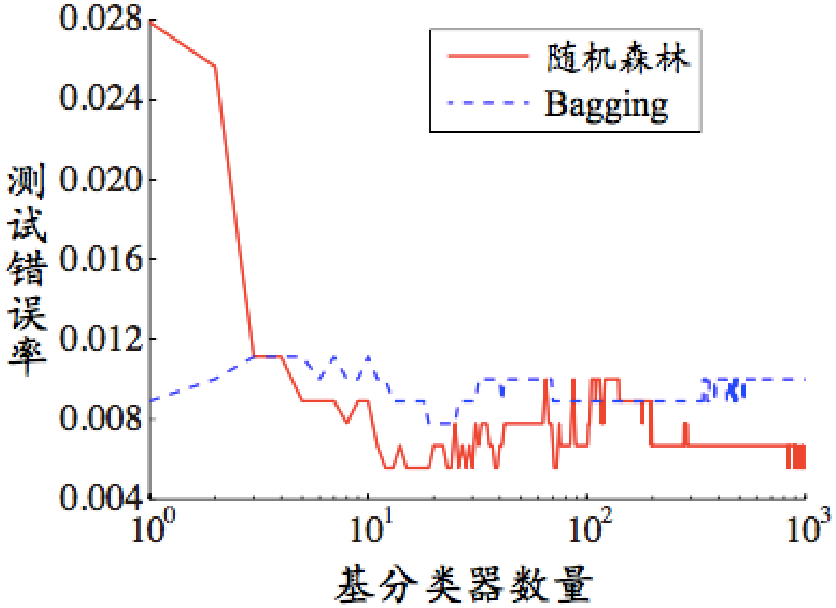
1. $N \leftarrow$ create a tree node based on D;
2. **if** *all instances in the same class* **then return** N
3. $\mathcal{F} \leftarrow$ the set of features that can be split further;
4. **if** $\mathcal{F}$ *is empty* **then return** N
5. $\tilde{\mathcal{F}} \leftarrow$ select K features from $\mathcal{F}$ 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

Bagging与随机森林 － 随机森林例子

- 对如下的感冒诊断样本数据集，试用该数据集构造一棵作为随机森林基学习器的 CART 决策树。确定某结点的划分属性时，若该结点所对应属性集合具有m个特征，则规定从中随机选择$s=\lceil \log_2 m \rceil$个属性计算用于确定划分属性的基尼指数

编号	体温	流鼻涕	肌肉疼	头疼	感冒
1	较高	是	是	否	是
2	非常高	否	否	否	否
3	非常高	是	否	是	是
4	正常	是	是	是	是
5	正常	否	否	是	否
6	较高	是	否	否	是
7	较高	是	否	是	是
8	非常高	是	是	否	是
9	较高	否	是	是	是
10	正常	是	否	否	否
11	正常	是	否	是	是
12	正常	否	是	是	是
13	较高	否	否	否	否
14	非常高	否	是	否	是
15	非常高	否	是	否	是
16	较高	否	否	是	是

Bagging与随机森林 – 随机森林例子

- 数据有 4 个属性，即m=4。故从中随机选择$s=\lceil \log_2 4 \rceil=2$个属性用于计算确定该决策树第一个结点的划分属性。通过随机抽样，选择“流鼻涕”和“肌肉疼”这两个属性进行计算。
- 首先考察“流鼻涕”属性，根据是否流鼻涕可以将数据集划分为：

$$D_1 = \{1,3,4,6,7,8,10,11\};\ D_2 = \{2,5,9,12,13,14,15,16\}$$

- 分别计算D_1和 D_2的基尼值：

$$Gini(D_1) = 1 - \left(\frac{7}{8}\right)^2 - \left(\frac{1}{8}\right)^2 = 0.21875，\ Gini(D_2) = 1 - \left(\frac{5}{8}\right)^2 - \left(\frac{3}{8}\right)^2 = 0.46875$$

- “流鼻涕” 属性的基尼指数为：

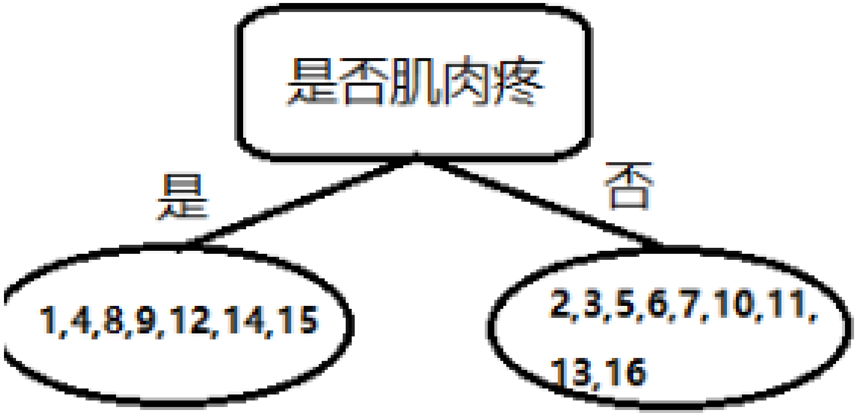
$$Gini(D,\text{流鼻涕}) = \frac{8}{16} \times Gini(D_1) + \frac{8}{16} \times Gini(D_2) = 0.34375$$

Bagging与随机森林 – 随机森林例子

- “肌肉疼” 属性的基尼指数为：

$$Gini(D,\text{肌肉疼}) = \frac{7}{16} \times Gini(D_1) + \frac{9}{16} \times Gini(D_2) = 0.2778$$

- 根据上述计算结果，选择“肌肉疼”作为决策树根节点的划分属性，得到如图所示的初始决策树

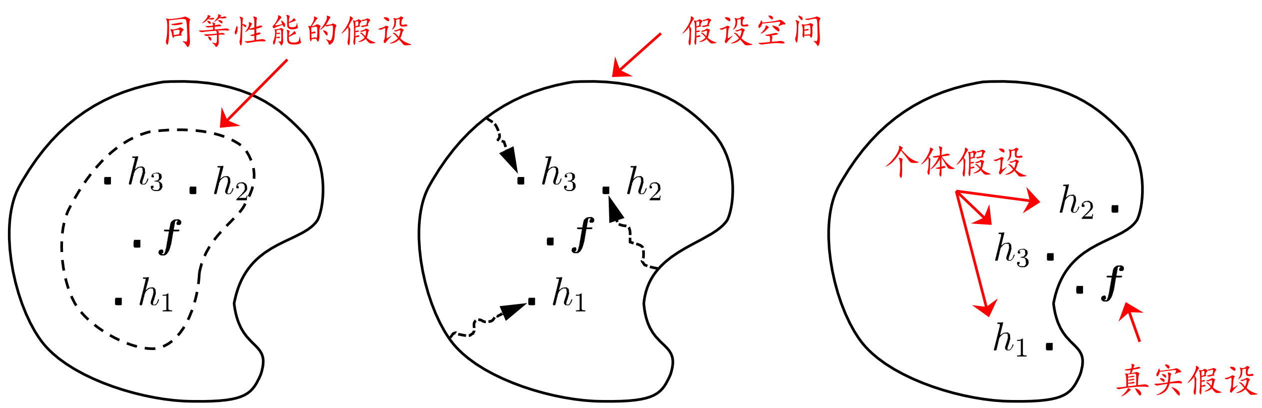


集成学习

- 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

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:

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

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：

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

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

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

- 上式对所有样本x均成立，令$p(x)$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)

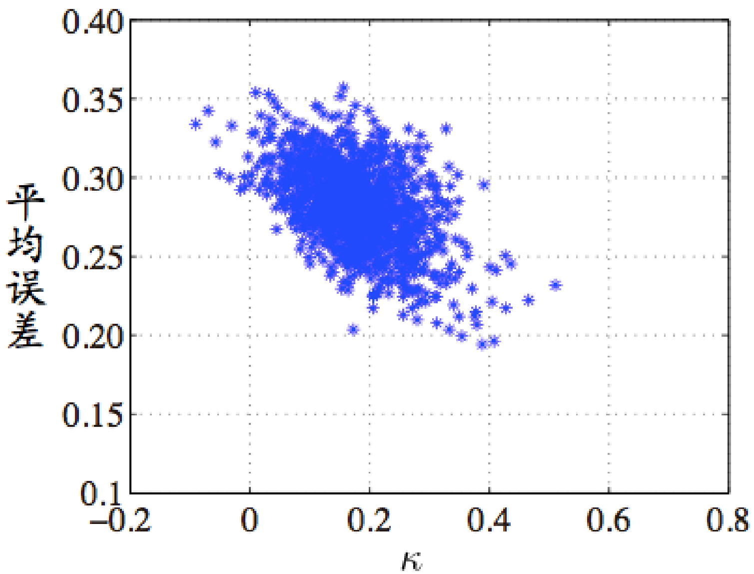
$$Q_{ij} = \frac{ad - bc}{ad + bc} \qquad |Q_{ij}| \leq |\rho_{ij}|$$

- K-统计量(Kappa-Statistic)

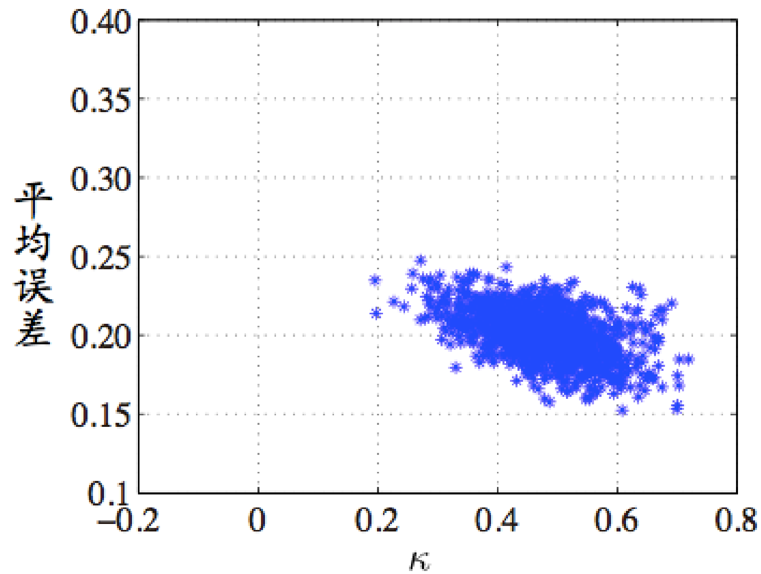
$$\kappa = \frac{p_1 - p_2}{1 - p_2} \qquad \begin{aligned} p_1 &= \frac{a+d}{m}, \\ p_2 &= \frac{(a+b)(a+c) + (c+d)(b+d)}{m^2} \end{aligned}$$

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：

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