



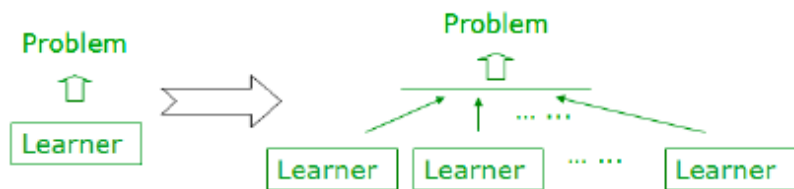
第八讲 集成学习

高级机器学习



集成学习

集成学习 (Ensemble Learning): 利用多个学习器解决问题



在现实任务中展现出极好的性能

- ❑ KDDCup'07: 1st place for "... Decision Forests and ..."
- ❑ KDDCup'08: 1st place of Challenge1 for a method using Bagging; 1st place of Challenge2 for "... Using an Ensemble Method "
- ❑ KDDCup'09: 1st place of Fast Track for "Ensemble ... "; 2nd place of Fast Track for "... bagging ... boosting tree models ...", 1st place of Slow Track for "Boosting ... "; 2nd place of Slow Track for "Stochastic Gradient Boosting"
- ❑ KDDCup'10: 1st place for "... Classifier ensembling"; 2nd place for "... Gradient Boosting machines ... "

- ❑ KDDCup'11: 1st place of Track 1 for "A Linear Ensemble ... "; 2nd place of Track 1 for "Collaborative filtering Ensemble", 1st place of Track 2 for "Ensemble ..."; 2nd place of Track 2 for "Linear combination of ..."
- ❑ KDDCup'12: 1st place of Track 1 for "Combining... Additive Forest..."; 1st place of Track 2 for "A Two-stage Ensemble of..."
- ❑ KDDCup'13: 1st place of Track 1 for "Weighted Average Ensemble"; 2nd place of Track 1 for "Gradient Boosting Machine"; 1st place of Track 2 for "Ensemble the Predictions"
- ❑ KDDCup'14: 1st place for "ensemble of GBM, ExtraTrees, Random Forest..."; 2nd place for "use both R and Python GBMs"; 3rd place for "gradient boosting machines... random forests" and "the weighted average of..."
- ❑ KDDCup'15: 1st place for "Three-Stage Ensemble and Feature Engineering for MOOC Dropout Prediction"
- ❑ KDDCup'16: 1st place for "Gradient Boosting Decision Tree"; 2nd place for "Ensemble of Different Models for Final Prediction"

近十年 KDDCup 获胜者，以及Netflix、Kaggle 等诸多数据分析竞赛的获胜者，几乎无一例外使用了集成学习

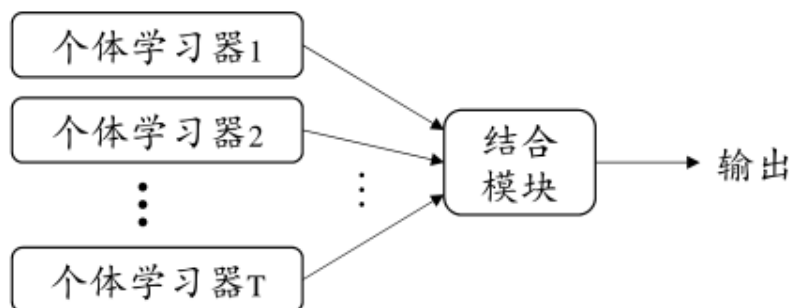
想获胜，用集成

集成学习

- 个体与集成
- Boosting
- Bagging与随机森林
- 结合策略
- 多样性

个体与集成

□ 集成学习(ensemble learning)通过多学习器来提升性能



	测试例1	测试例2	测试例3		测试例1	测试例2	测试例3		测试例1	测试例2	测试例3
h_1	√	√	×	h_1	√	√	×	h_1	√	×	×
h_2	×	√	√	h_2	√	√	×	h_2	×	√	×
h_3	√	×	√	h_3	√	√	×	h_3	×	×	√
集群	√	√	√	集群	√	√	×	集群	×	×	×
(a) 集群提升性能			(b) 集群不起作用			(c) 集群起负作用					

□ 关键：好而不同

个体与集成 - 理想情况分析

- 二分类问题，假设基分类器的错误率为： $P(h_i(\mathbf{x}) \neq f(\mathbf{x})) = \epsilon$
- 投票法结合 T 个分类器，超过半数的基分类器正确则分类就正确

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

- 假设基分类器的错误率相互独立，则由Hoeffding不等式可得：

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

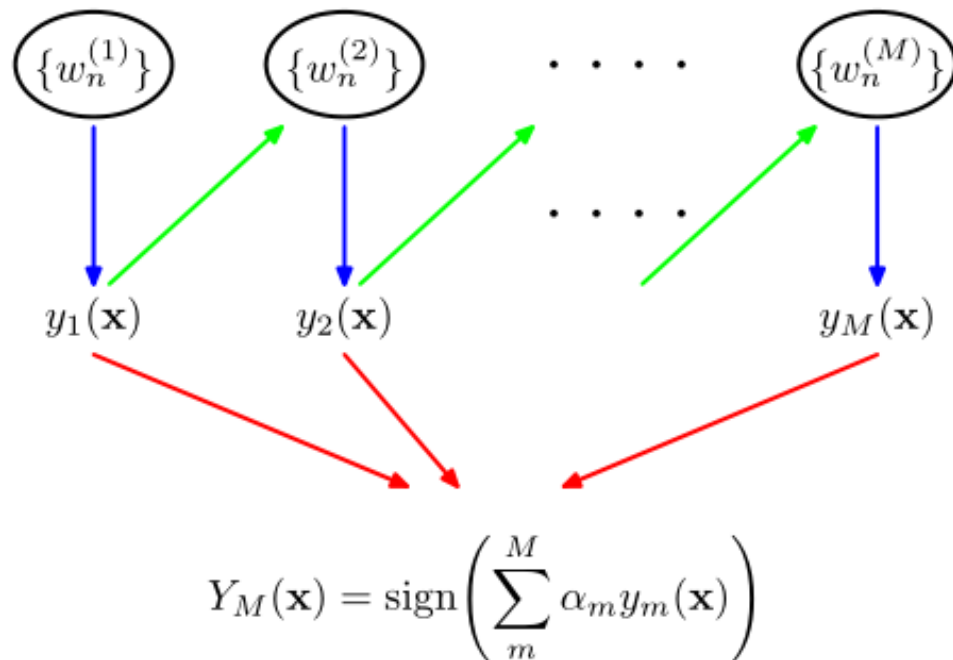
- 上式显示，在一定条件下，随着集成分类器数目的增加，集成的错误率将指数级下降，最终趋向于0

个体与集成 - 现实

- 关键假设：基学习器的误差相互独立
- 现实任务中，个体学习器来自同一个问题，显然不可能完全独立
- 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：串行 vs 并行

Boosting

- 每次调整训练数据的样本分布
- 串行生成
- 个体学习器存在强依赖关系



Boosting - Boosting算法

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

Process:

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

Output: $H(\mathbf{x}) = \text{Combine_Outputs}(\{h_1(\mathbf{x}), \dots, h_t(\mathbf{x})\})$

□ Boosting族算法最著名的代表是AdaBoost

Boosting – AdaBoost推导

□ 基学习器的线性组合

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

□ 最小化指数损失函数

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

□ 令指数损失函数最小化，则上式对的偏导值为0，即

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

$$\text{sign}(H(\mathbf{x})) = \text{sign}\left(\frac{1}{2} \ln \frac{P(f(\mathbf{x}) = 1 | \mathbf{x})}{P(f(\mathbf{x}) = -1 | \mathbf{x})}\right)$$

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

$$= \begin{cases} 1, & P(f(\mathbf{x}) = 1 | \mathbf{x}) > P(f(\mathbf{x}) = -1 | \mathbf{x}) \\ -1, & P(f(\mathbf{x}) = 1 | \mathbf{x}) < P(f(\mathbf{x}) = -1 | \mathbf{x}) \end{cases}$$

$$= \arg \max_{y \in \{-1, 1\}} P(f(\mathbf{x}) = y | \mathbf{x}),$$

$\text{sign}(H(\mathbf{x}))$ 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

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

- 令指数损失函数的导数为0，即

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

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

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

- 泰勒展开近似为

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

Boosting – AdaBoost推导

□ 理想的基学习器更新:

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

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

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

Boosting – AdaBoost推导

□ 根据数学期望，这等价于：

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

□ 由 $f(\mathbf{x}), h(\mathbf{x}) \in \{-1, +1\}$ 有：

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

□ 理想的基学习器

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

Boosting – AdaBoost推导

- 最终的样本分布更新公式

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

Boosting – AdaBoost算法

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

过程:

1: $\mathcal{D}_1(\mathbf{x}) = 1/m$.

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

3: $h_t = \mathcal{L}(D, \mathcal{D}_t)$;

4: $\epsilon_t = P_{\mathbf{x} \sim \mathcal{D}_t}(h_t(\mathbf{x}) \neq f(\mathbf{x}))$;

5: **if** $\epsilon_t > 0.5$ **then break**

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

7:
$$\mathcal{D}_{t+1}(\mathbf{x}) = \frac{\mathcal{D}_t(\mathbf{x})}{Z_t} \times \begin{cases} \exp(-\alpha_t), & \text{if } h_t(\mathbf{x}) = f(\mathbf{x}) \\ \exp(\alpha_t), & \text{if } h_t(\mathbf{x}) \neq f(\mathbf{x}) \end{cases}$$
$$= \frac{\mathcal{D}_t(\mathbf{x}) \exp(-\alpha_t f(\mathbf{x}) h_t(\mathbf{x}))}{Z_t}$$

8: **end for**

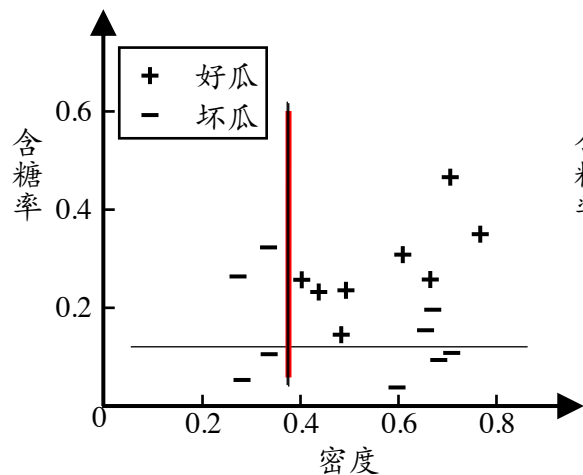
输出: $H(\mathbf{x}) = \text{sign} \left(\sum_{t=1}^T \alpha_t h_t(\mathbf{x}) \right)$

□ 数据分布的学习

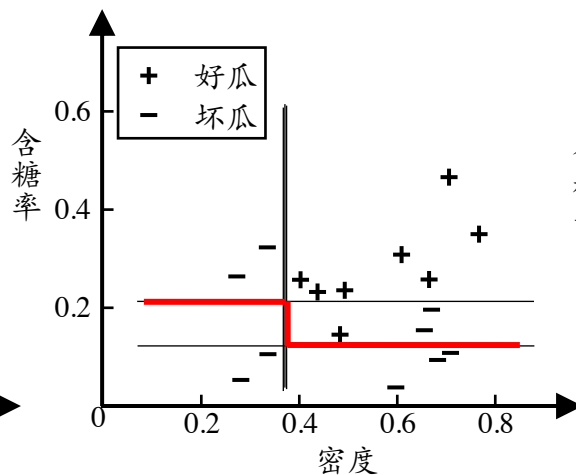
- 重赋权法、重采样法

□ 重启动, 避免训练过程过早停止

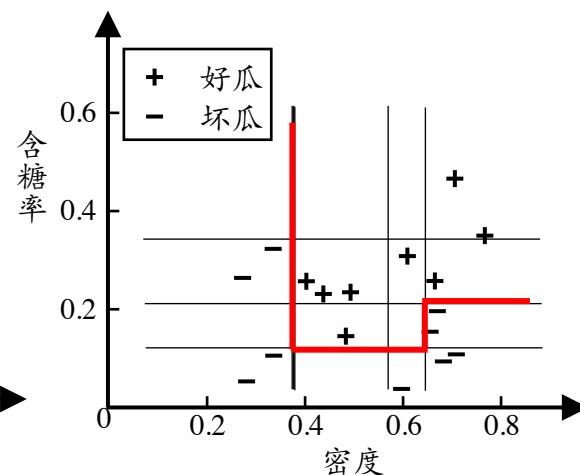
Boosting – AdaBoost实验



(a) 3个基学习器



(b) 5个基学习器



(c) 11个基学习器

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
- Bagging与随机森林
- 结合策略
- 多样性

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

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

过程:

```
1: for  $t = 1, 2, \dots, T$  do  
2:    $h_t = \mathcal{L}(D, \mathcal{D}_{bs})$   
3: end for
```

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

□ 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

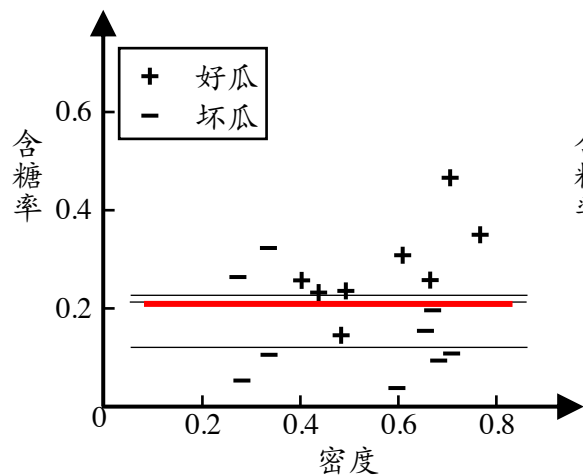
- $H^{oob}(x)$ 表示对样本 x 的包外预测，即仅考虑那些未使用样本 x 训练的基学习器在 x 上的预测

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

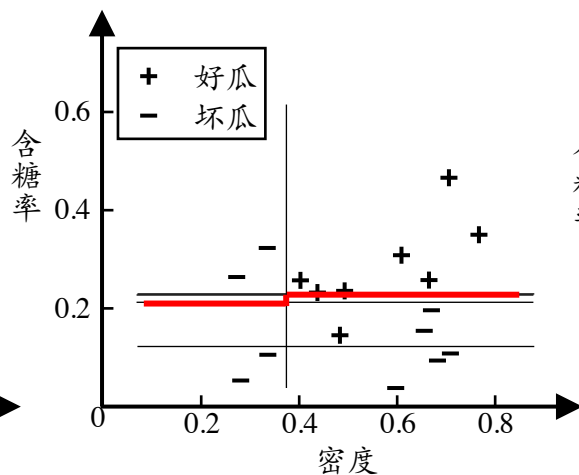
- Bagging泛化误差的包外估计为：

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

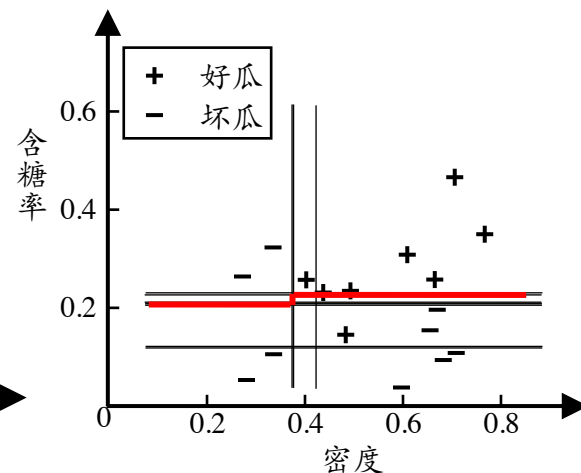
Bagging与随机森林- Bagging实验



(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

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

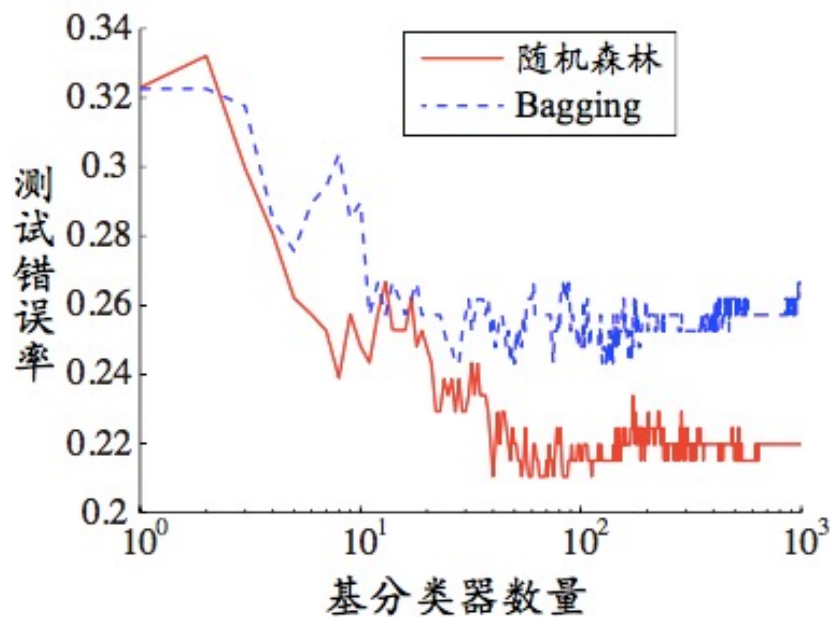
Process:

□ 属性选择的随机性

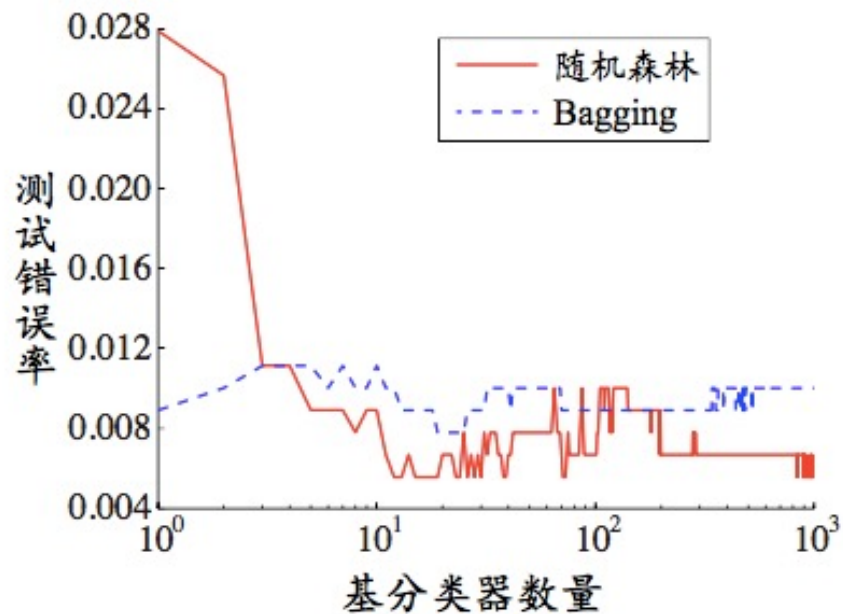
1. $N \leftarrow$ create a tree node based on D ;
2. **if all instances in the same class then return N**
3. $\mathcal{F} \leftarrow$ the set of features that can be split further;
4. **if \mathcal{F} is empty then return N**
5. $\tilde{\mathcal{F}} \leftarrow$ select K features from \mathcal{F} randomly;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



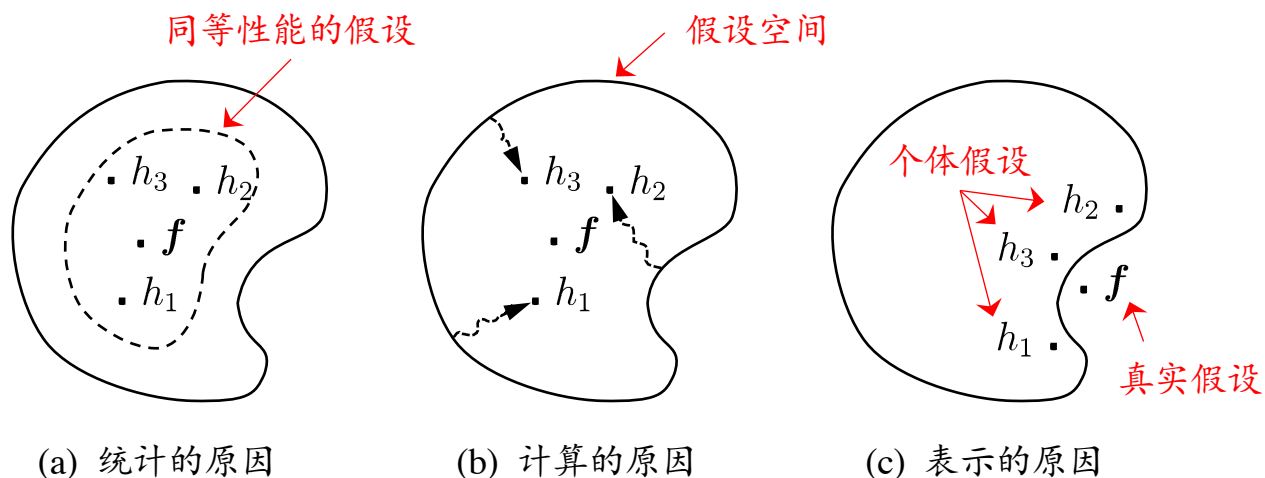
(b) auto-mpg 数据集

集成学习

- 个体与集成
- Boosting
- Bagging与随机森林
- 结合策略
- 多样性

结合策略

□ 学习器的组合可以从三个方面带来好处



□ 简单平均法

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

□ 加权平均法

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

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)

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

- 相对多数投票法 (plurality voting)

$$H(\mathbf{x}) = c_{\arg \max_j \sum_{i=1}^T h_i^j(\mathbf{x})}$$

- 加权投票法 (weighted voting)

$$H(\mathbf{x}) = c_{\arg \max_j \sum_{i=1}^T w_i h_i^j(\mathbf{x})}$$

结合策略 - 学习法

□ Stacking是学习法的典型代表

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

Process:

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

Output: $H(\mathbf{x}) = h'(h_1(\mathbf{x}), \dots, h_T(\mathbf{x}))$

集成学习

- 个体与集成
- Boosting
- Bagging与随机森林
- 结合策略
- 多样性

误差-分歧分解

误差-分歧分解 (error-ambiguity decomposition):

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

Ensemble error *Ave. error of individuals* *Ave. "ambiguity" of individuals* (*"ambiguity" later called "diversity"*)

The more **accurate** and **diverse** the individual learners, the better the ensemble

However,

- the "ambiguity" does not have an operable definition
- The error-ambiguity decomposition is derivable only for regression setting with squared loss

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

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

输入属性扰动

□ 随机子空间算法(random subspace)

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

过程:

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

输出: $H(\mathbf{x}) = \arg \max_{y \in \mathcal{Y}} \sum_{t=1}^T \mathbb{I}(h_t(\text{Map}_{\mathcal{F}_t}(\mathbf{x})) = y)$

输出表示扰动 & 算法参数扰动

输出表示扰动

□ 翻转法(Flipping Output)

- 随机改变输入样本的标记

□ 输出调剂法(Output Smearing)

- 分类输出改为回归输出得到分类器

□ ECOC法

- 多类任务分解为一系列两类任务来求解

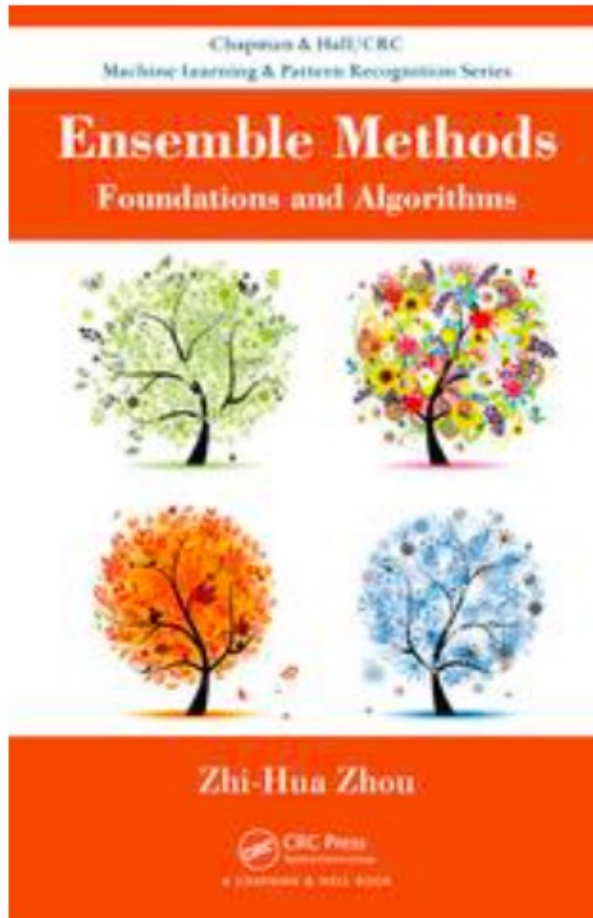
算法参数扰动

□ 负相关法

- 强制要求个体神经网络采用不同的参数

□ 不同的多样性增强机制同时使用

更多集成学习的内容



Z.-H. Zhou.
Ensemble Methods:
Foundations and Algorithms,
Boca Raton, FL: Chapman &
Hall/CRC, Jun. 2012.
(ISBN 978-1-439-830031)

集成学习

- 个体与集成：知道个体分类器的定义和集成学习的定义
- Boosting：Boosting思想和adaboost的实现
- Bagging与随机森林：思想和实现方式
- 结合策略：几种常用策略以及stacking的优缺点
 - 平均法
 - 投票法
 - 学习法
- 多样性：多样性扰动的几种办法
 - 误差-分歧分解
 - 多样性度量
 - 多样性扰动