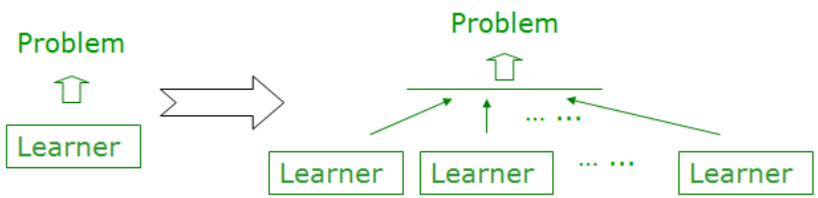
高级机器学习

集成学习

集成学习

Ensemble Learning (集成学习) :

Using multiple learners to solve the problem



Demonstrated great performance in real practice

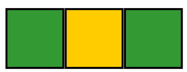
- KDDCup'07: 1^{st} place for "… Decision Forests and …"
- KDDCup'08: 1^{st} place of Challenge1 for a method using Bagging; 1^{st} place of Challenge2 for "… Using an Ensemble Method "
- KDDCup'09: 1^{st} place of Fast Track for "Ensemble … "; 2^{nd} place of Fast Track for "… bagging … boosting tree models …", 1^{st} place of Slow Track for "Boosting … "; 2^{nd} place of Slow Track for "Stochastic Gradient Boosting"
- KDDCup'10: 1^{st} place for "… Classifier ensembling"; 2^{nd} place for "… Gradient Boosting machines … "
- KDDCup'11: 1^{st} place of Track 1 for "A Linear Ensemble … "; 2^{nd} place of Track 1 for "Collaborative filtering Ensemble", 1^{st} place of Track 2 for "Ensemble …"; 2^{nd} place of Track 2 for "Linear combination of …"
- KDDCup'12: 1^{st} place of Track 1 for "Combining… Additive Forest…"; 1^{st} place of Track 2 for "A Two-stage Ensemble of…"
- KDDCup'13: 1^{st} place of Track 1 for "Weighted Average Ensemble" ; 2^{nd} place of Track 1 for "Gradient Boosting Machine"; 1^{st} place of Track 2 for "Ensemble the Predictions"
- KDDCup'14: 1^{st} place for "ensemble of GBM, ExtraTrees, Random Forest…" and "the weighted average" ; 2^{nd} place for "use both R and Python GBMs"; 3^{rd} place for "gradient boosting machines… random forests" and "the weighted average of…"
- KDDCup'15: 1^{st} place for "Three-Stage Ensemble and Feature Engineering for MOOC Dropout Prediction"
- KDDCup'16: 1^{st} place for "Gradient Boosting Decision Tree"; 2^{nd} place for "Ensemble of Different Models for Final Prediction"
- KDDCup'17: 1^{st} and 2^{nd} place of Task 1 for "XGBoost"; 1^{st} place of Task 2 for "XGBoost", 2^{nd} place of Task 2 for "Weighted Average of Multiple Models"
- KDDCup'18: 1^{st} place for "Gradient Boosting"; 2^{nd} place for "Two-stage stacking"; 3^{rd} place for "Weighted Average of Multiple Models"

During the past decade, almost all winners of KDDCup, Netflix competition, Kaggle competitions, etc., utilized ensemble techniques in their solutions

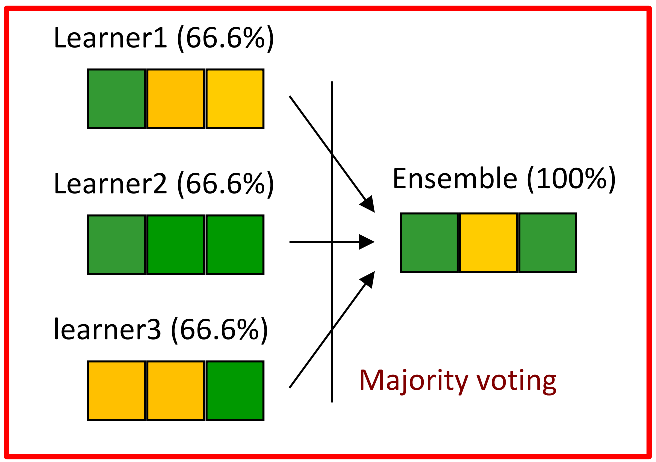
To win? Ensemble !

个体与集成

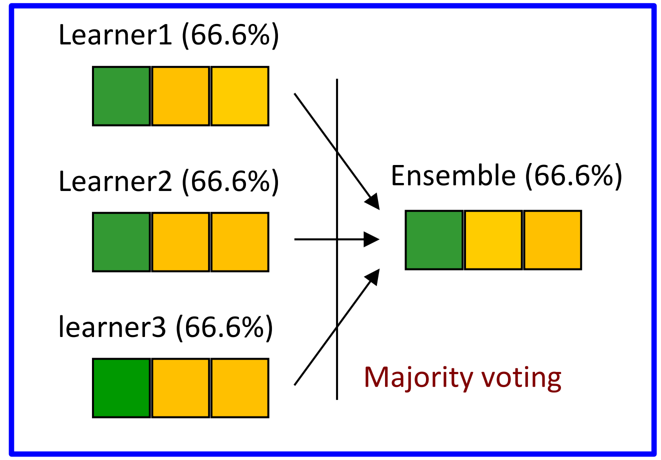
Some intuitions:



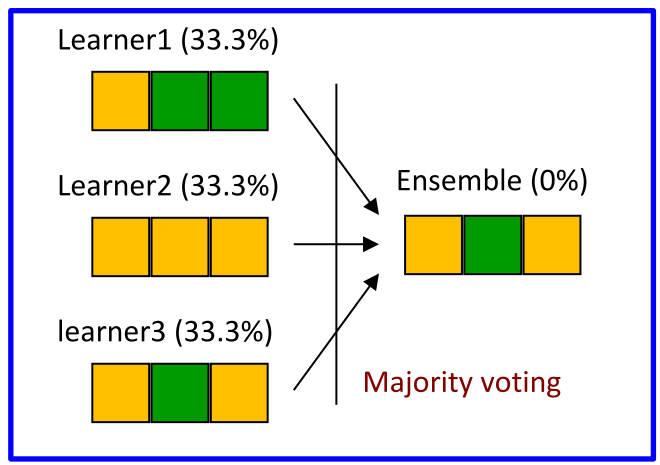
Ground-truth



Ensemble really helps



Individuals must be different



Individuals must be not-bad

令个体学习器“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： $P\left(h_i\left(\boldsymbol{x}\right) \neq f\left(\boldsymbol{x}\right)\right) = \epsilon$
- 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

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

- 假设基分类器的错误率相互独立，则由Hoeffding不等式可得：

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

- 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

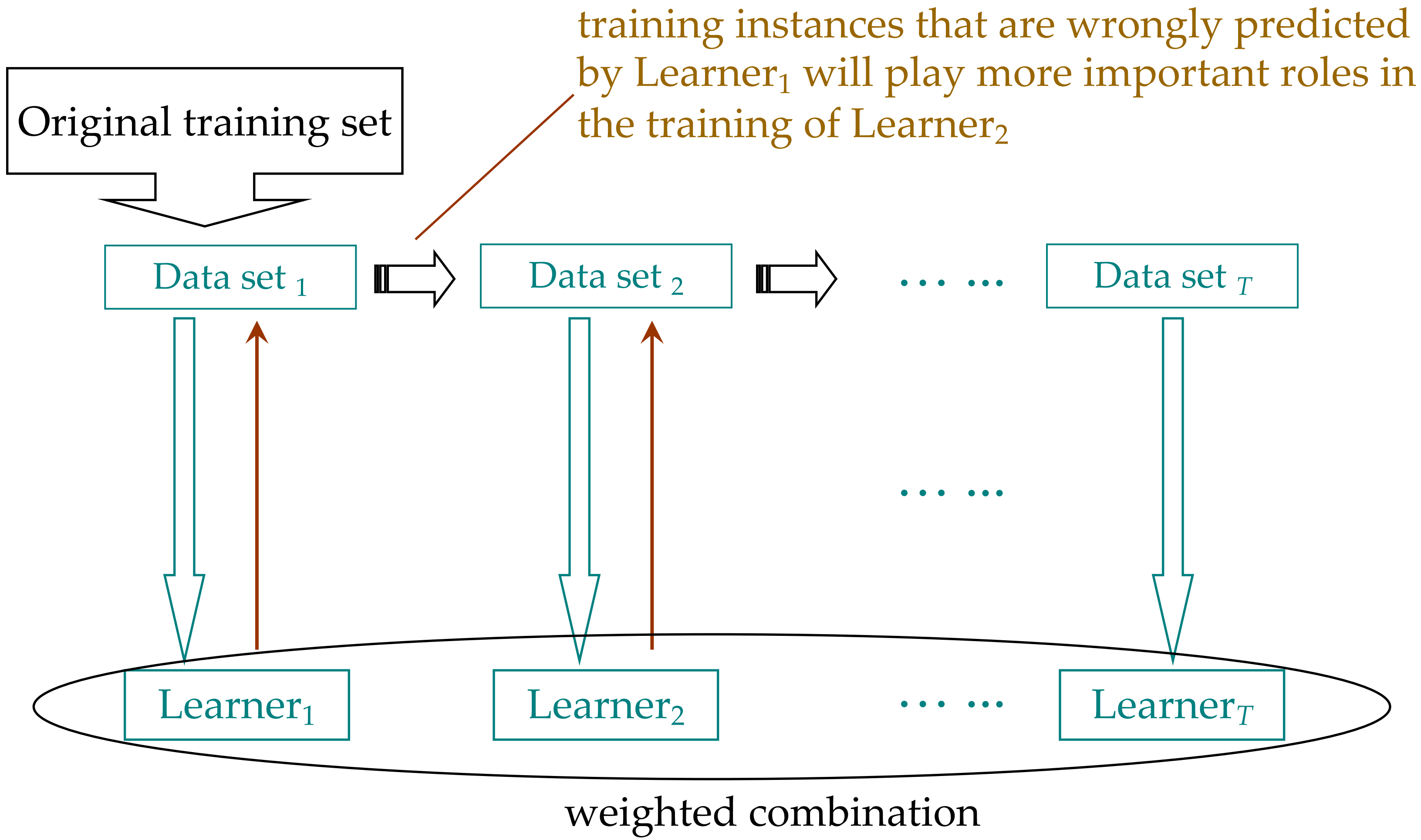
个体与集成-现实

- 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

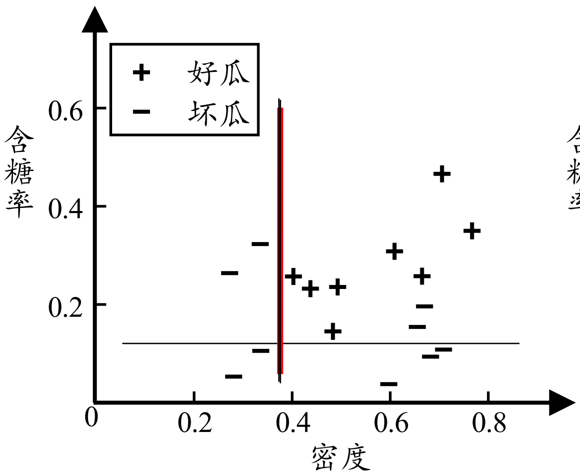
很多成功的集成学习方法

- 串行化方法
  - **AdaBoost** [Freund & Schapire, JCSS97]
  - GradientBoost [Friedman, AnnStat01]
  - LPBoost [Demiriz, Bennett, Shawe-Taylor, MLJ06]
  - …… ……
- 并行化方法
  - **Bagging** [Breiman, MLJ96]
  - Random Forest [Breiman, MLJ01]
  - Random Subspace [Ho, TPAMI98]
  - …… ……

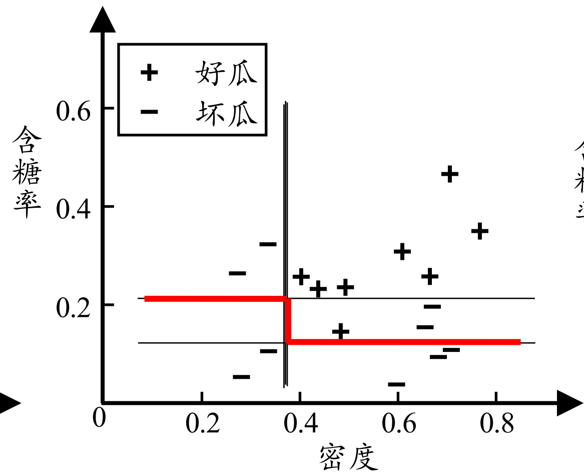
Boosting: A flowchart illustration



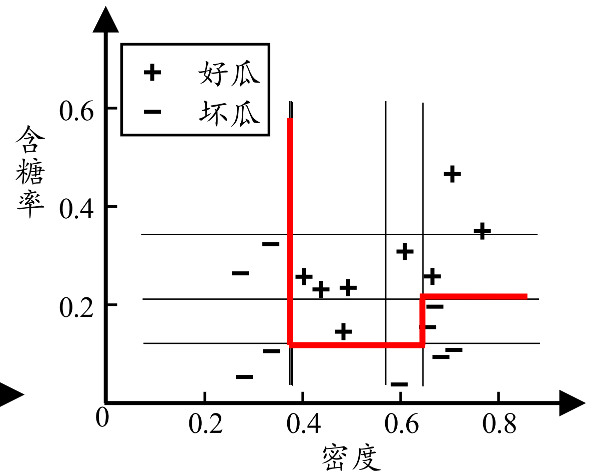
Boosting – AdaBoost实验



(a) 3个基学习器



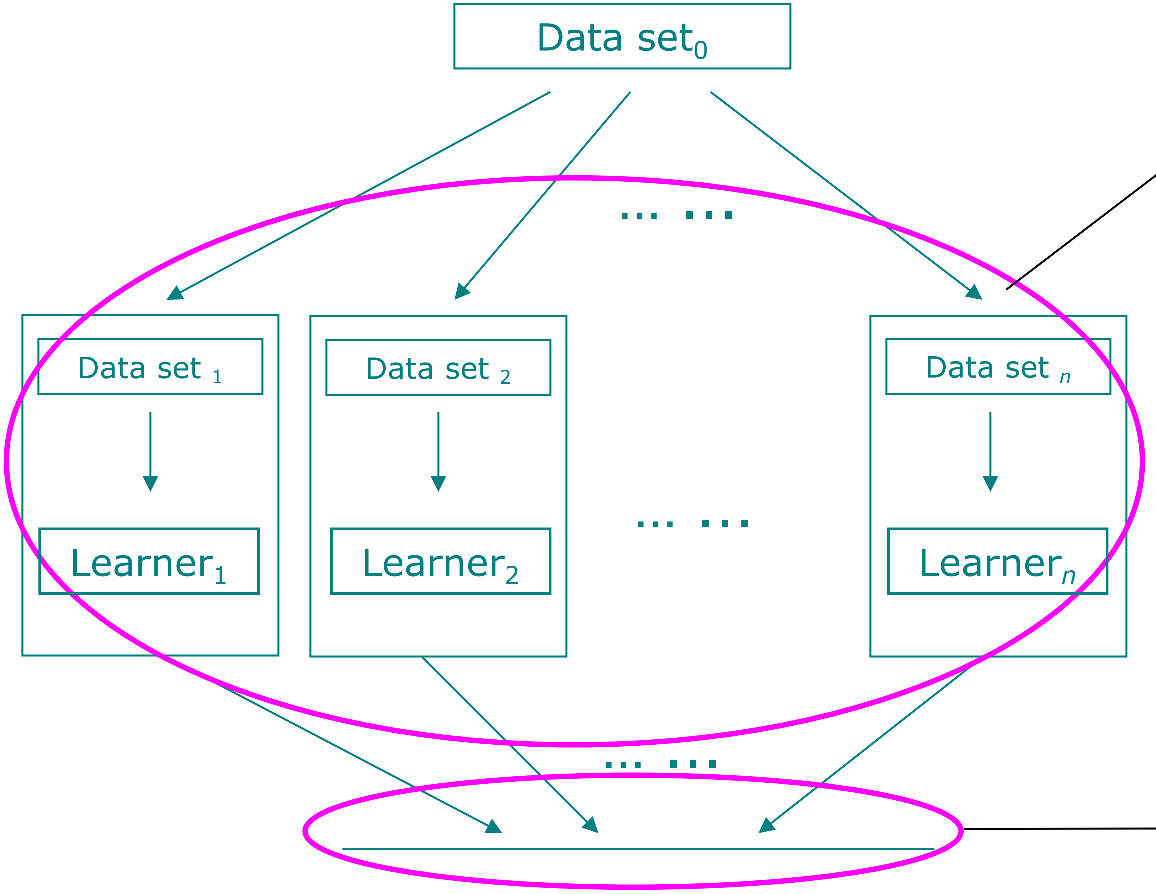
(b) 5个基学习器



(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

Bagging



bootstrap a set of learners

generate many data sets from the original data set through bootstrap sampling (random sampling with replacement), then train an individual learner per data set

voting for classification

the output is the class label receiving the most number of votes

averaging for regression

the output is the average output of the individual learners

Bagging与随机森林

- 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

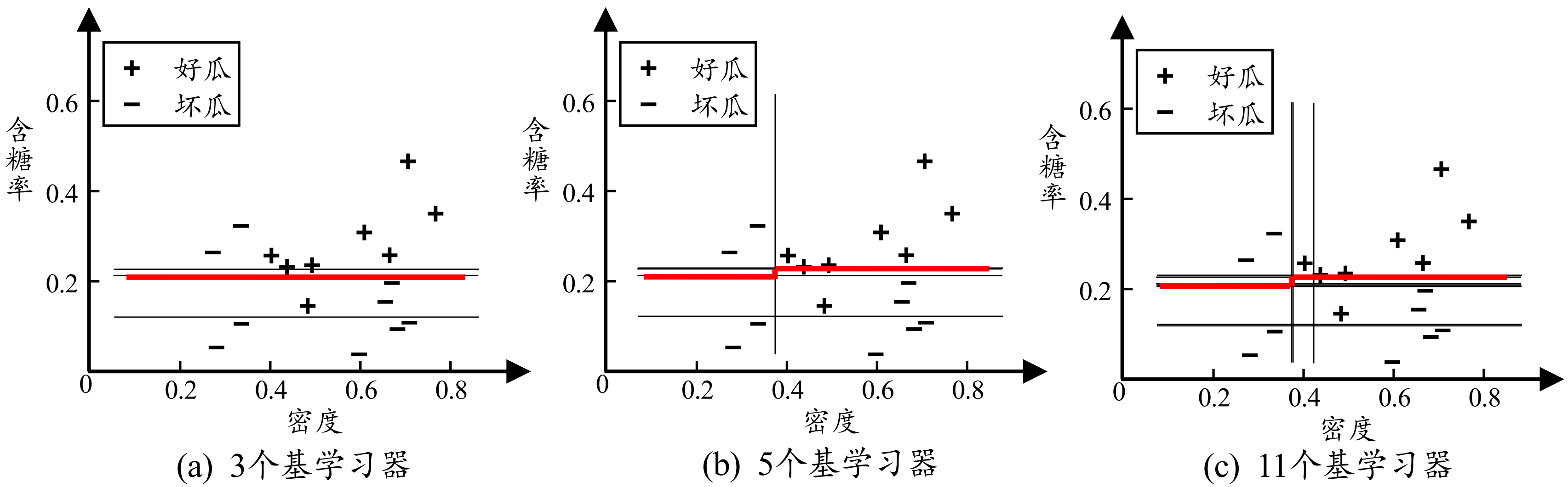
Input: Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
Feature subset size K.

Process:

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

Output: A random decision tree

Bagging实验



- 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

学习器结合

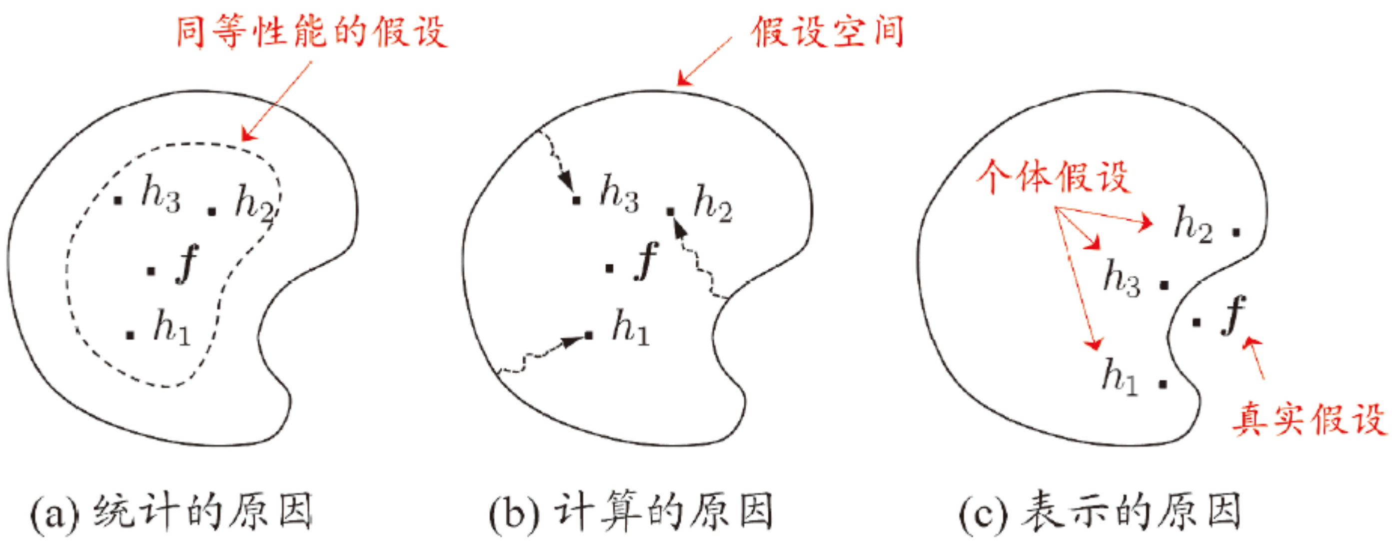


图 8.8 学习器结合可能从三个方面带来好处 [Dietterich, 2000]

常用结合方法：

- 投票法
  - 绝对多数投票法
  - 相对多数投票法
  - 加权投票法
- 平均法
  - 简单平均法
  - 加权平均法
- 学习法

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

集成学习

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

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

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

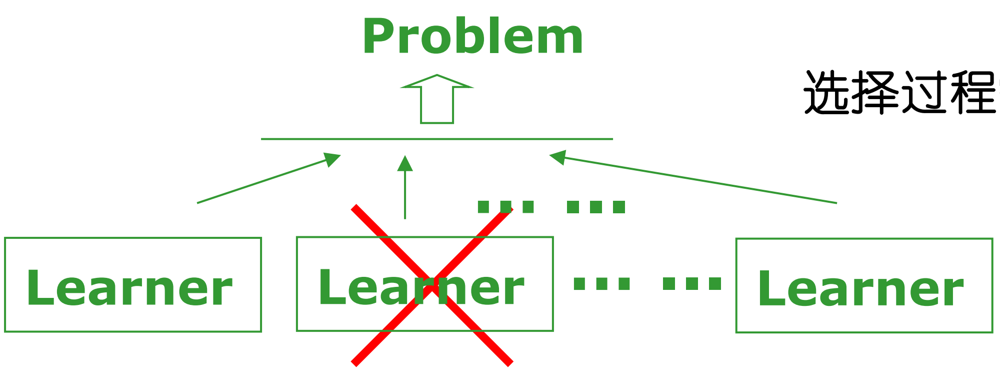
However,

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

“越多越好”？

选择性集成 (selective ensemble):

给定一组个体学习器，从中选择一部分来构建集成，经常会比使用所有个体学习器更好 (更小的存储/时间开销，更强的泛化性能)



选择过程需考虑个体 性能 与 多样性/互补性

仅选出“精度最高的”通常不好！

集成修剪 (ensemble pruning)
[Margineantu & Dietterich, ICML’97]
较早出现，针对序列型集成
减小集成规模、降低泛化性能

选择性集成 [Zhou, et al, AIJ 02] 稍晚，
针对并行型集成，MCBTA (Many could be better than all)定理
减小集成规模、增强泛化性能

目前“集成修剪”与“选择性集成”基本被视为同义词

多样性

"多样性" (diversity) 是集成学习的关键

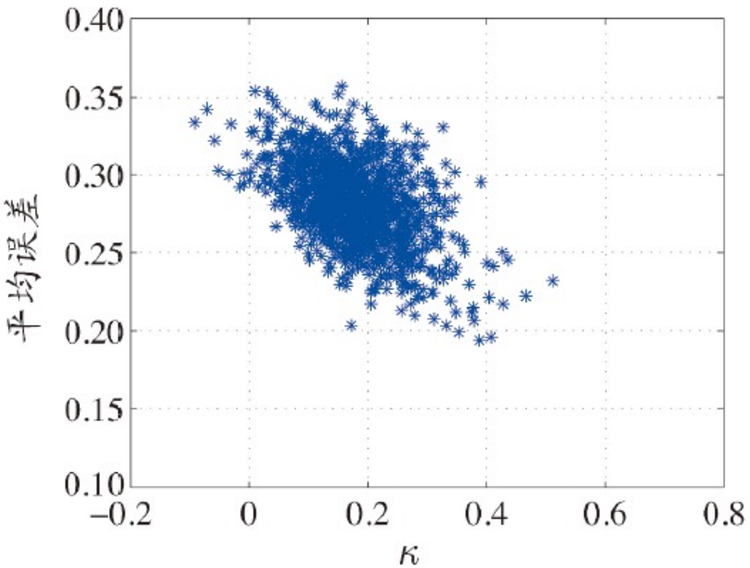
多样性度量

一般通过两分类器的预测结果列联表定义

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

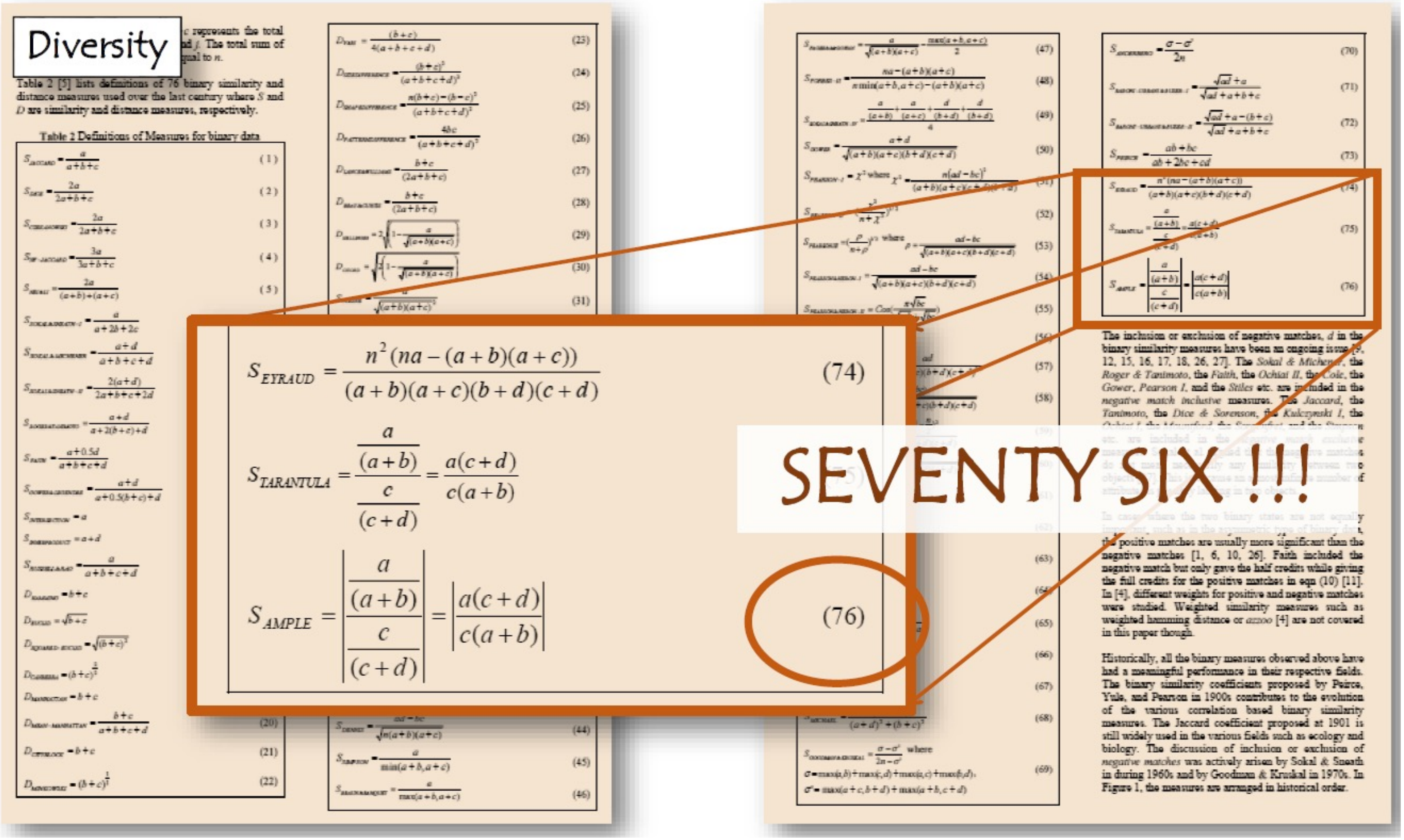
- 不合度量 (disagreement measure)
- 相关系数 (correlation coefficient)
- Q-统计量 (Q-statistic)
- κ-统计量（κ-statistic）
- … …

κ-误差图



每一对分类器作为图中的一个点

研究者提出了很多 Diversity measure



From [L. Kuncheva, ICPRAM’16 keynote]

However, …

- [Kuncheva & Whitaker, MLJ 2003]: Empirical study shows that there seems no clear relation between many diversity measures and the ensemble performance

- [Tang, Suganthan, Yao, MLJ 2006]: Exploiting many diversity measures explicitly is ineffective in constructing consistently stronger ensembles

There is no well-accepted definition/formulation of diversity

“What is diversity” remains the holy grail problem of ensemble learning

多样性增强常用策略

- 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 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_m, y_m)\}$;
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: $\quad \mathcal{F}_t = \mathrm{RS}(D, d')$
3: $\quad D_t = \mathrm{Map}_{\mathcal{F}_t}(D)$
4: $\quad h_t = \mathfrak{L}(D_t)$
5: **end for**

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

输出表示扰动

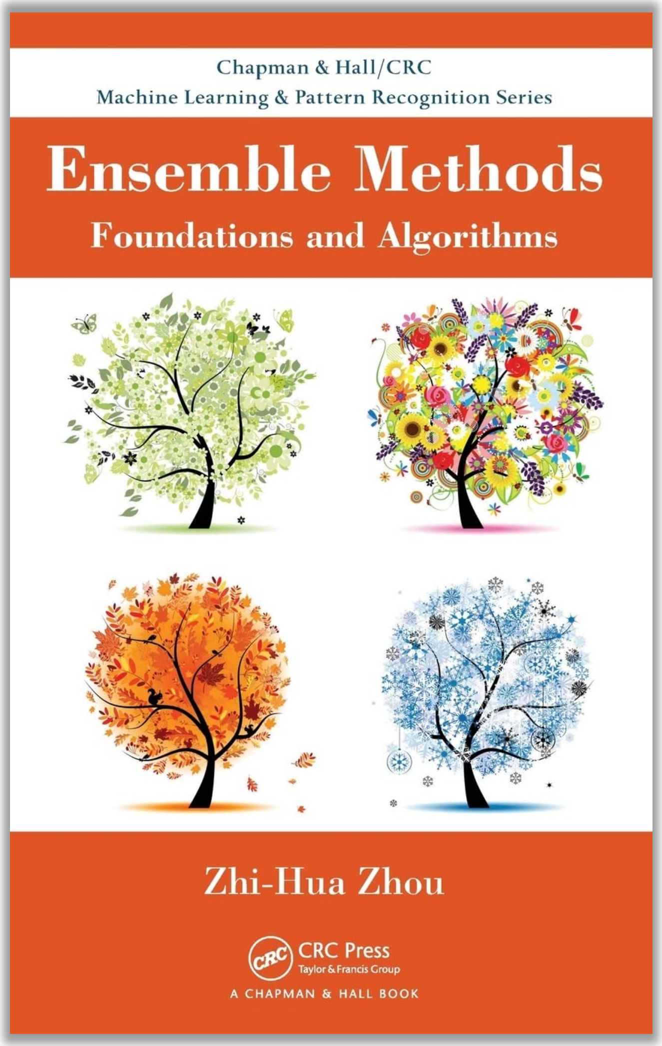
- 翻转法(Flipping Output)
  - 随机改变输入样本的标记
- 输出调剂法(Output Smearing)
  - 分类输出改为回归输出得到分类器
- ECOC法
  - 多类任务分解为一系列两类任务来求解

算法参数扰动

- 负相关法
  - 强制要求个体模型采用不同的参数

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, 2012. (ISBN 978-1-439-830031)

小结

- 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