机器学习导论

第7章 集成学习

谢茂强

南开大学软件学院

Ensemble Learning 集成学习

- 多分类器系统(Multi-Classifier System)
- 基于委员会的学习(Committee-based Learning)

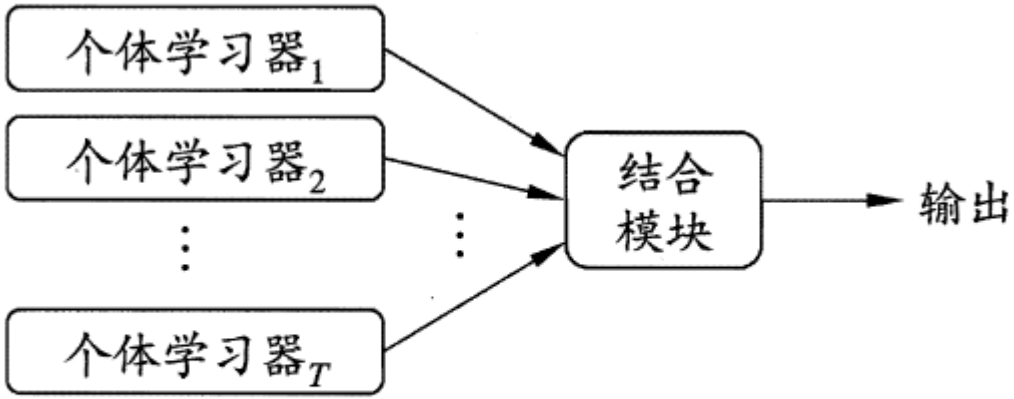


图 8.1 集成学习示意图

目录

个体与集成

- 个体和集体的优缺点
- 集体的一种分类
  - 同质集成学习（Homogeneous EL）：比如神经网络
  - 异质集成学习（Heterogeneous EL）
- 个体学习器（基学习器、弱学习器、分类器）

集成策略的好坏

- 个体学习器要有一定的准确性、并且要有一定的多样性（Diversity，或差异性）即“好且不同”

	测试例1	测试例2	测试例3
h_1	√	√	×
h_2	×	√	√
h_3	√	×	√
集成	√	√	√

(a) 集成提升性能

	测试例1	测试例2	测试例3
h_1	√	√	×
h_2	√	√	×
h_3	√	√	×
集成	√	√	×

(b) 集成不起作用

	测试例1	测试例2	测试例3
h_1	√	×	×
h_2	×	√	×
h_3	×	×	√
集成	×	×	×

(c) 集成起负作用

图 8.2 集成个体应“好而不同”（h_i 表示第 i 个分类器）

集成分类器错误率上限

- 在基分类器的错误率相互独立的情况下，根据Hoeffding不等式，集成的错误率为：

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

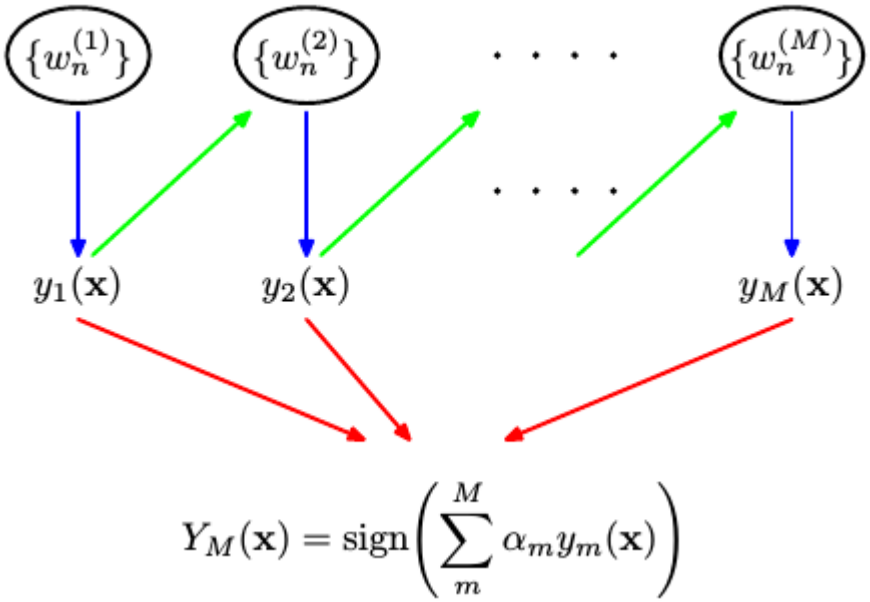
$$\leqslant\exp\left(-\frac{1}{2}T\left(1-2\epsilon\right)^{2}\right)\ . \qquad (8.3)$$

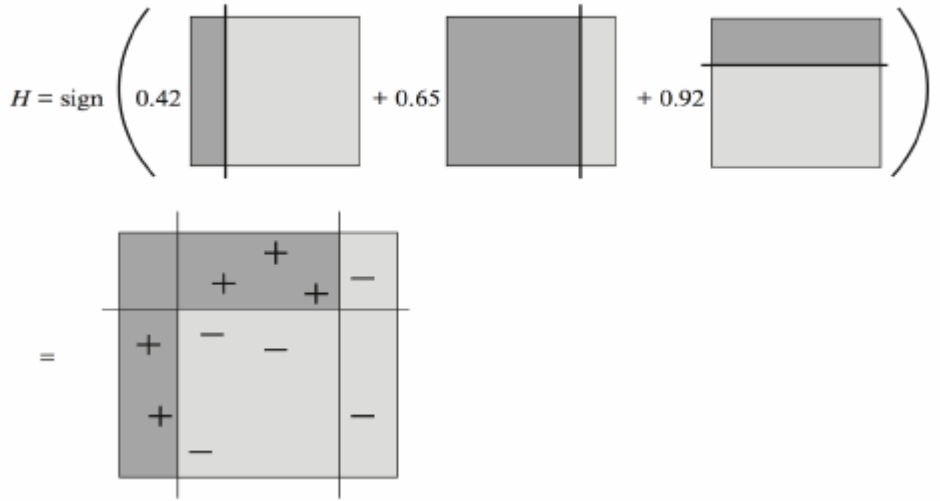
- 随着T的增大，错误率将指数级下降，并趋近于0
- 个体分类器目标相同、数据集相同，独立性很难成立
- 准确率与差异性的矛盾

2. Boosting历史

- Valiant和 Kearns于1984年首次提出了 PAC学习模型中弱学习算法和强学习算法的等价性问题,即任意给定仅比随机猜测略好的弱学习算法 ,是否可以将其提升为强学习算法？如果二者等价 ,那么只需找到一个比随机猜测略好的弱学习算法就可以将其提升为强学习算法 ,而不必寻找很难获得的强学习算法。
- 1995年 , Freund和 Schapire改进了Boosting算法 ,提出了 AdaBoost (Adaptive Boosting)算法
- 2000年 , Friedman和Hastie等人从统计角度重新解释AdaBoost

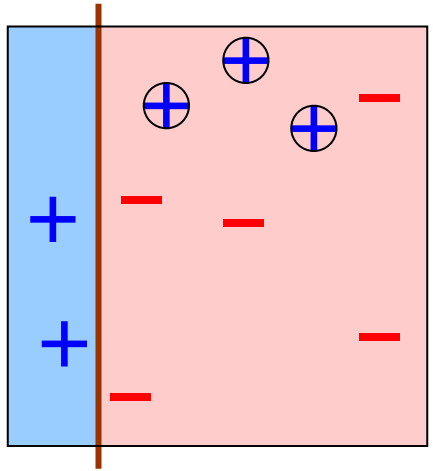
2.1 AdaBoost算法示意





AdaBoost算法示意

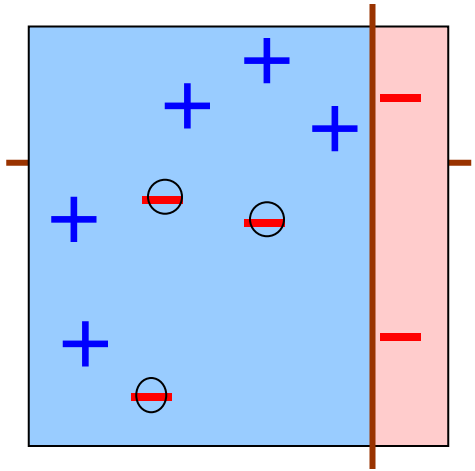
弱分类器：横向或纵向的分类面（比如，1层决策树）



第 1 次迭代

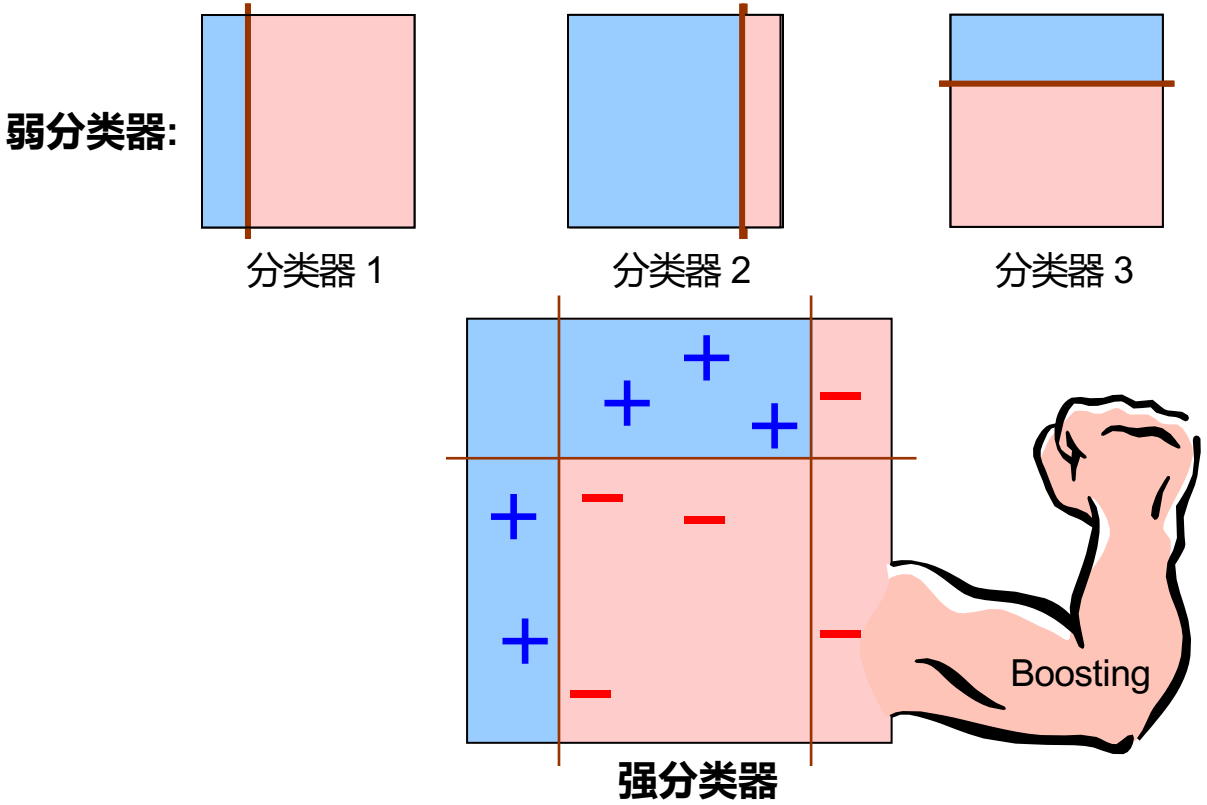
AdaBoost算法示意

弱分类器：横向或纵向的分类面



第 3 次迭代

AdaBoost算法示意



D：训练数据集

$\mathcal{D}_t$：第t轮，训练数据集各样本的权重

初始化样本权值分布.

基于分布 $\mathcal{D}_t$ 从数据集 D 中训练出分类器 h_t.

估计 h_t 的误差.

确定分类器 h_t 的权重.

更新样本分布，其中 Z_t 是规范化因子，以确保 $\mathcal{D}_{t+1}$ 是一个分布.

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

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

图 8.3 AdaBoost算法

AdaBoost的判决函数

- AdaBoost可以看作是加权(Re-weighting)的投票法：

$$H(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \tag{8.4}$$

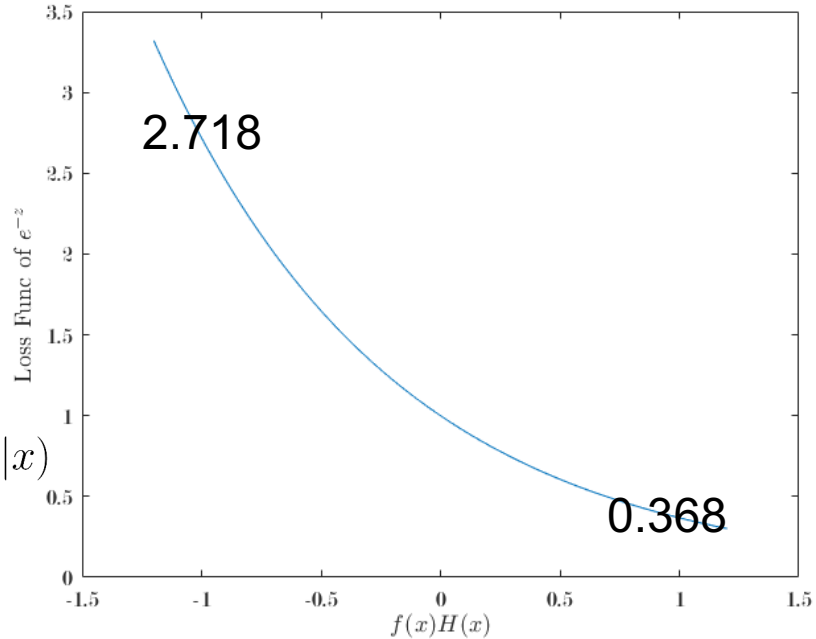
2.2 AdaBoost的损失函数

- 使用指数函数(Exponential Loss Function)度量经验误差

$f(x)$:代表样本x的标注类别，在二分类任务中，取值为1或-1。

$$\mathbb{E}\left[e^{-f(x)H(x)}|x\right]$$

$$= e^{-1\cdot H(x)}P(f(x)=1|x) + e^{-(-1)\cdot H(x)}P(f(x)=-1|x)$$



J. Friedman , T. Hastie , R. Tibshirani. Additive logistic Regression: A Statistical View of Boosting. The Annals of Statistics. 28(2):337-407. 2000

2.2 从损失函数极值角度来看集成学习模型H(x)

$$\frac{\partial \mathbb{E}\left[e^{-f(x)H(x)}|x\right]}{\partial H(x)} = -e^{-H(x)}P(f(x)=1|x) + e^{H(x)}P(f(x)=-1|x)$$

上式的导数为0时

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

意味着：当H(x)取上式时，指数损失函数会取到最小值。

2.3.1 算法：AdaBoost弱分类器h_t的权重优化

- 根据样本的错分情况，调整 t 步后的基分类器 h_t 的权重，进而通过后续加入的基分类器降低样本的错分率
- 加入权重 α_t 后，分类器 h_t 的损失函数为：

$$
\begin{aligned}
\ell_{\exp}\left(\alpha_t h_t \mid \mathcal{D}_t\right) &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}\right] \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[e^{-\alpha_t}\mathbb{I}\left(f\left(\boldsymbol{x}\right)=h_t\left(\boldsymbol{x}\right)\right)+e^{\alpha_t}\mathbb{I}\left(f\left(\boldsymbol{x}\right)\neq h_t\left(\boldsymbol{x}\right)\right)\right] \\
&= e^{-\alpha_t}P_{\boldsymbol{x}\sim\mathcal{D}_t}\left(f\left(\boldsymbol{x}\right)=h_t\left(\boldsymbol{x}\right)\right)+e^{\alpha_t}P_{\boldsymbol{x}\sim\mathcal{D}_t}\left(f\left(\boldsymbol{x}\right)\neq h_t\left(\boldsymbol{x}\right)\right) \\
&= e^{-\alpha_t}\left(1-\epsilon_t\right)+e^{\alpha_t}\epsilon_t\,, \qquad (8.9)
\end{aligned}
$$

2.3.1 如何计算h_t的权重α_t

- 最小化带权重的h_t的损失函数：

其中 $\epsilon_t = P_{\boldsymbol{x}\sim\mathcal{D}_t}\big(h_t(\boldsymbol{x}) \neq f(\boldsymbol{x})\big)$. 考虑指数损失函数的导数

$$\frac{\partial \ell_{\exp}(\alpha_t h_t \mid \mathcal{D}_t)}{\partial \alpha_t} = -e^{-\alpha_t}(1-\epsilon_t) + e^{\alpha_t}\epsilon_t \,, \tag{8.10}$$

令式(8.10)为零可解得

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \,, \tag{8.11}$$

2.3.2 调整下一轮样本权重

- 【主要思路】：通过调整样本权重，使得加入新的基分类器后，尽可能纠正之前的全部错误。
- 最小化加入新基分类器后的损失函数：

$$\begin{aligned}\ell_{\exp}(H_{t-1}+h_t \mid \mathcal{D}) &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})(H_{t-1}(\boldsymbol{x})+h_t(\boldsymbol{x}))}] \\ &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}e^{-f(\boldsymbol{x})h_t(\boldsymbol{x})}] \ . \end{aligned} \tag{8.12}$$

2.3.2 使用泰勒展开近似损失函数

- 泰勒展开式

$$f(x)=\frac{f(x_0)}{0!}+\frac{f'(x_0)}{1!}(x-x_0)+\frac{f''(x_0)}{2!}(x-x_0)^2+\ldots+\frac{f^{(n)}(x_0)}{n!}(x-x_0)^n+R_n(x)$$

$$e^x \simeq 1+x+\frac{x^2}{2!}+\frac{x^3}{3!}+\ldots$$

- 泰勒展开式近似为：

注意到 $f^2(\boldsymbol{x})=h_t^2(\boldsymbol{x})=1$, 式(8.12)可使用 $e^{-f(\boldsymbol{x})h_t(\boldsymbol{x})}$ 的泰勒展式近似为

$$\begin{aligned}\ell_{\exp}(H_{t-1}+h_t \mid \mathcal{D}) &\simeq \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\left(1-f(\boldsymbol{x})h_t(\boldsymbol{x})+\frac{f^2(\boldsymbol{x})h_t^2(\boldsymbol{x})}{2}\right)\right] \\ &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\left(1-f(\boldsymbol{x})h_t(\boldsymbol{x})+\frac{1}{2}\right)\right]. \qquad (8.13)\end{aligned}$$

2.3.2 根据上一轮样本权重训练ht的候选分类器，并进行选择

$$
\begin{aligned}
h_t(\boldsymbol{x}) &= \arg\min \ell_{\exp}(H_{t-1}+h \mid \mathcal{D}) \\
&= \arg\min_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\left(1-f(\boldsymbol{x})h(\boldsymbol{x})+\frac{1}{2}\right)\right] \\
&= \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}f(\boldsymbol{x})h(\boldsymbol{x})\right] \\
&= \arg\max_h \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\frac{e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\right]}f(\boldsymbol{x})h(\boldsymbol{x})\right],
\end{aligned}
\tag{8.14}
$$

其中，$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\right]$ 是一个常数.

2.3.2 根据现有$H_t(x)$调整下一轮样本权重 $\mathcal{D}_{t+1}$

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 $\mathcal{D}_t$ 表示一个分布

$$
\begin{aligned}
\mathcal{D}_{t+1}(\boldsymbol{x}) &= \frac{\mathcal{D}(\boldsymbol{x})\, e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}\right]} \\
&= \frac{\mathcal{D}(\boldsymbol{x})\, e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})} e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}\right]} \\
&= \mathcal{D}_t(\boldsymbol{x}) \cdot e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})} \frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\right]}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}\right]}\,, \qquad (8.19)
\end{aligned}
$$

2.3.3 弱分类器h_t-一层决策树

- 从训练数据集D的n个属性中选择最值得划分的属性。
- 度量属性a分类能力的度量“信息增益”(Information Gain)

$$\mathrm{Gain}(D,a)=\mathrm{Ent}(D)-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\mathrm{Ent}(D^v)\ . \tag{4.2}$$

“信息熵”(information entropy)是度量样本集合纯度最常用的一种指标. 假定当前样本集合 D 中第 k 类样本所占的比例为 p_k $(k=1,2,\ldots,|\mathcal{Y}|)$, 则 D 的信息熵定义为

$$\mathrm{Ent}(D)=-\sum_{k=1}^{|\mathcal{Y}|}p_k\log_2 p_k\ . \tag{4.1}$$

$\mathrm{Ent}(D)$ 的值越小, 则 D 的纯度越高.

D：训练数据集

$\mathcal{D}_t$：第t轮，训练数据集各样本的权重

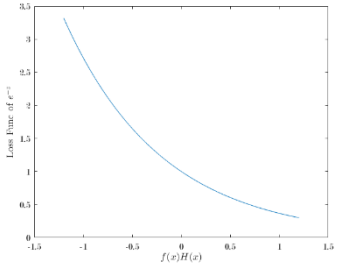
初始化样本权值分布.

基于分布 $\mathcal{D}_t$ 从数据集 D 中训练出分类器 h_t.

估计 h_t 的误差.

确定分类器 h_t 的权重.

更新样本分布，其中 Z_t 是规范化因子，以确保 $\mathcal{D}_{t+1}$ 是一个分布.



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

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

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

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

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

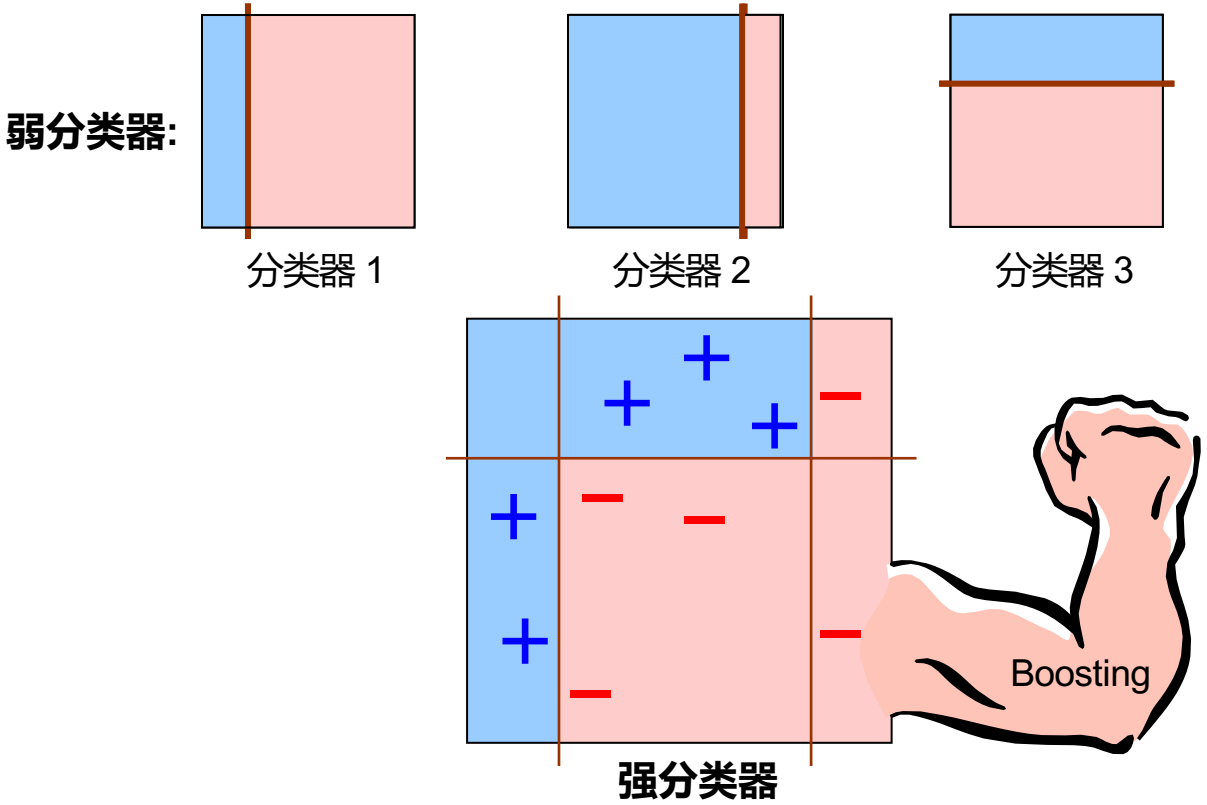
$$= \frac{\mathcal{D}_t(\boldsymbol{x}) \exp(-\alpha_t f(\boldsymbol{x}) h_t(\boldsymbol{x}))}{Z_t}$$

8: **end for**

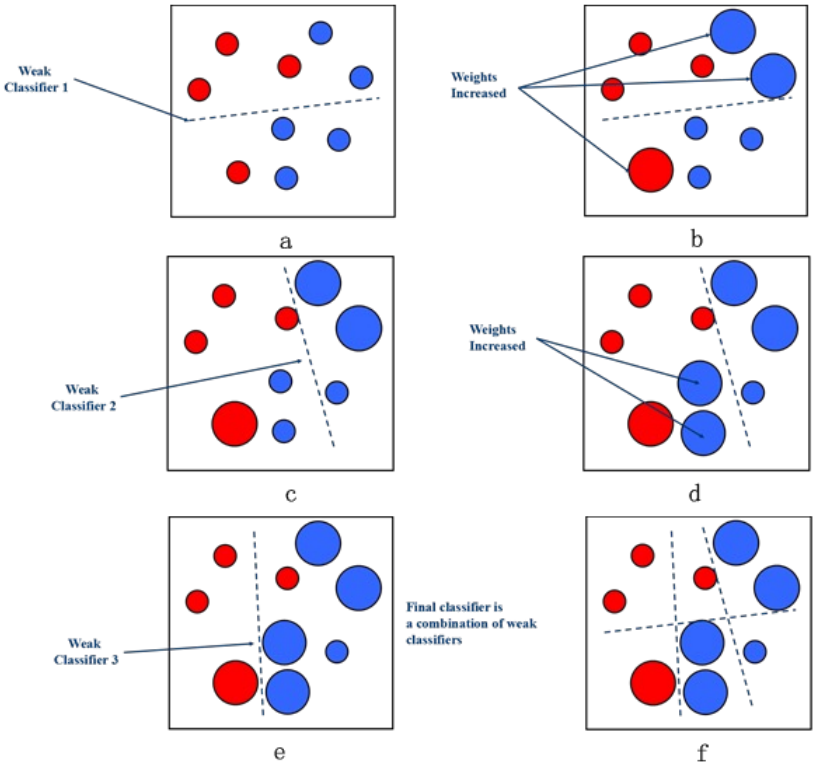
输出： $H(\boldsymbol{x}) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \right)$

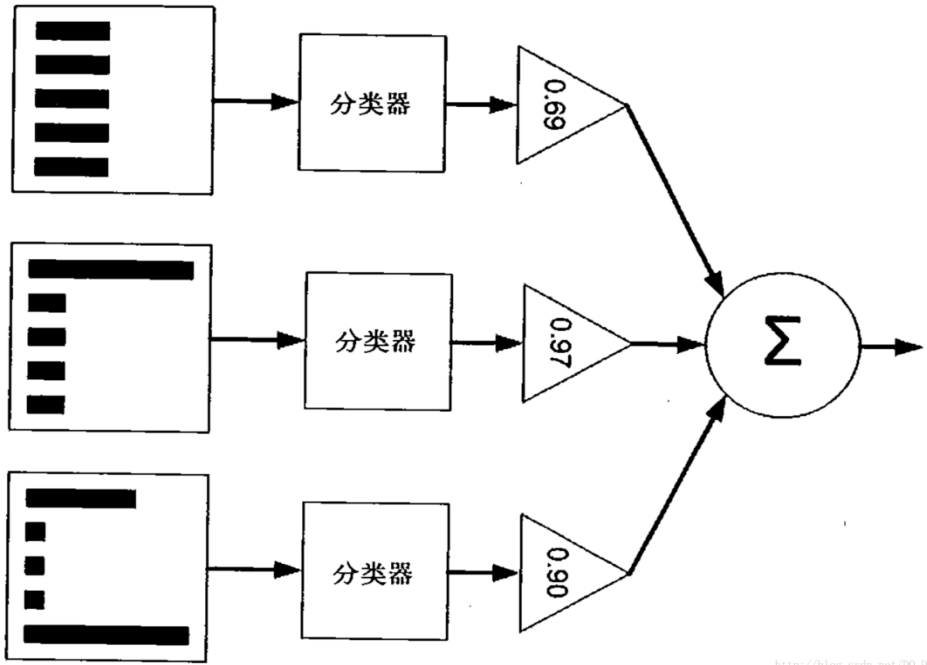
图 8.3 AdaBoost算法

AdaBoost示意



示意2：通过Reweighting获得子分类器

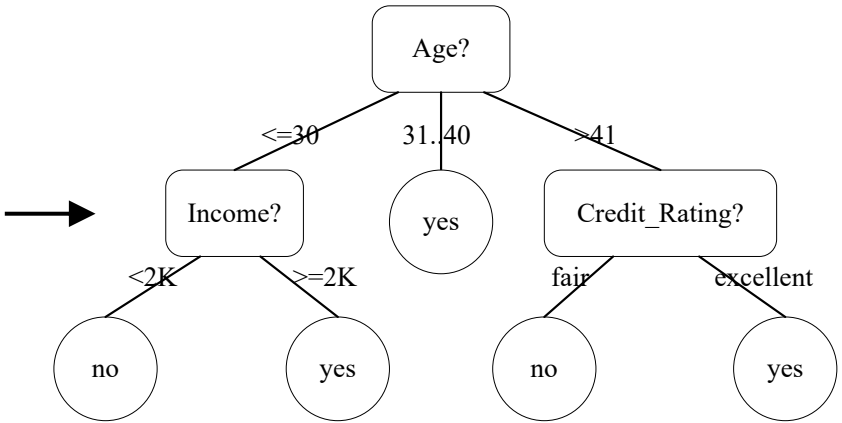




延伸：决策树分类器

- 从属性角度来递归分割训练样本的方式来获得决策树分类器
- 而分割属性则选择当前数据集中具有最大信息增益的那个属性

Age	Income	Student	Credit_rating	Buys _computer
<=30	<2K	No	excellent	no
<=30	<2K	No	Fair	no
<=30	<2K	Yes	Fair	no
<=30	>=2K	No	excellent	yes
<=30	>=2K	Yes	fair	yes
31..40	>=2K	No	fair	yes
31..40	<2K	Yes	excellent	yes
31..40	<2K	Yes	excellent	yes
31..40	<2K	No	excellent	yes
31..40	>=2K	Yes	excellent	yes
>40	<2K	Yes	excellent	yes
>40	<2K	Yes	excellent	yes
>40	>=2K	No	fair	no
>40	<2K	Yes	fair	no

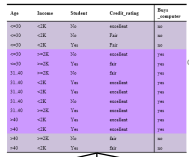


Quinlan. Induction of Decision Tree. 1986

数据：AllElectronic：韩家炜,数据挖掘：概念与技术(第1版), 第7章, 表7.1.

通过划分来降熵

- 以AllElectronic数据训练决策树的根节点为例。



D

以age为划分属性

Income	Student	Credit_rating	Buys _computer
<2K	No	excellent	no
<2K	No	Fair	no
<2K	Yes	Fair	no
>=2K	No	excellent	yes
>=2K	Yes	fair	yes

D^1

Income	Student	Credit_rating	Buys _computer
>=2K	No	fair	yes
<2K	Yes	excellent	yes
<2K	Yes	excellent	yes
<2K	No	excellent	yes
>=2K	Yes	excellent	yes

D^2

Income	Student	Credit_rating	Buys _computer
<2K	Yes	excellent	yes
<2K	Yes	excellent	yes
>=2K	No	fair	no
<2K	Yes	fair	no

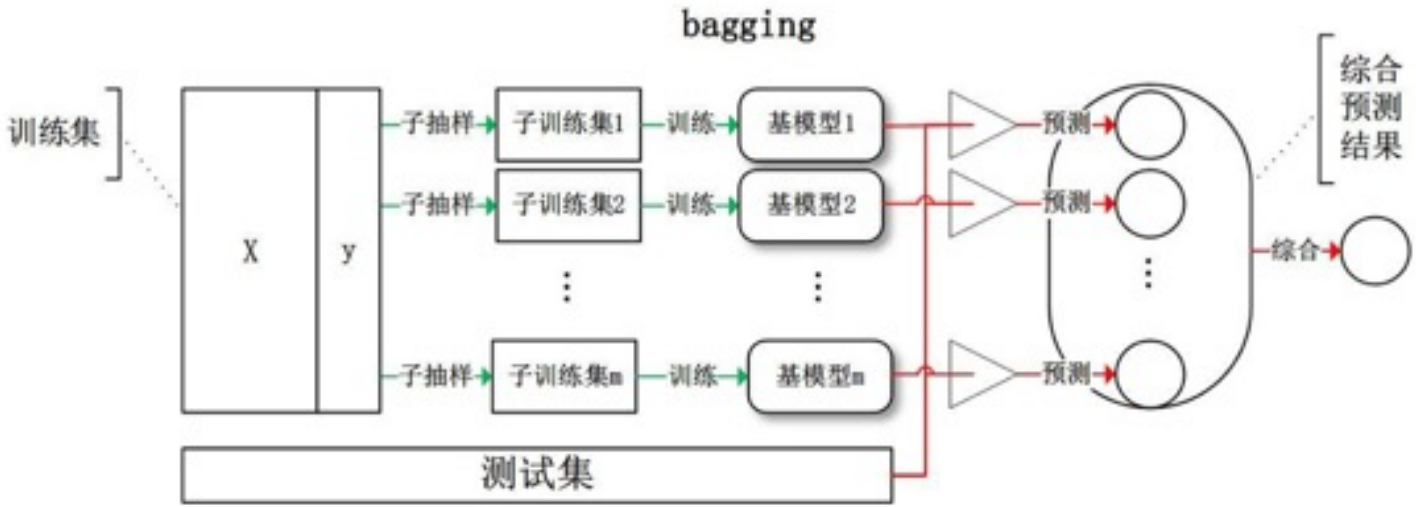
D^3

假定离散属性 a 有 V 个可能的取值 $\{a^1, a^2, \ldots, a^V\}$, 若使用 a 来对样本集 D 进行划分, 则会产生 V 个分支结点, 其中第 v 个分支结点包含了 D 中所有在属性 a 上取值为 a^v 的样本, 记为 D^v.

目录

Bagging

- Bagging[Breiman, 1996a]: Bootstrap AGGregatING(自举汇聚法)
- 靠抽样构建具有差异度的基模型，相比之下，属于级联算法



Breiman, L. Bagging Predictors. Machine Learning, 24(2):123-140. 1996

Bagging算法

$\mathcal{D}_{bs}$ 是自助采样产生的样本分布.

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

图 8.5 Bagging 算法

Bootstrap Sampling(自助采样法)

- 给定包含m个样本的数据集D, 我们对它进行m次放回采样，得到数据集D’。改变了数据集分布，引入了估计偏差。
- 样本在m次采样始终不被采到的概率是$(1-1/m)^m$,其极限为：

$$\lim_{m \mapsto \infty} \left(1 - \frac{1}{m}\right)^m \mapsto \frac{1}{e} \approx 0.368 , \qquad (2.1)$$

- 对于样本规模小，难以进行交叉检验时经常使用。

B. Efron and R. Tibshirani. An Introduction to Bootstrap. 1993.

Bagging特点

- 是一种策略，适于多分类、回归等任务。
- 抽样带来剩余样本，可以用作验证集，对泛化性能进行“包外估计”(out-of-bag estimation)
  - 决策树的剪枝
  - 降低样本对神经网络对的扰动等

随机森林(Random Forest)

- 以决策树为基学习器构建Bagging集成分类器；
- 在决策树的训练过程中引入了随机属性选择
  - 先从d个属性中随机选择k个属性
  - 再从这k个属性中选择一个最有属性
  - 推荐 $k=log_2 d$

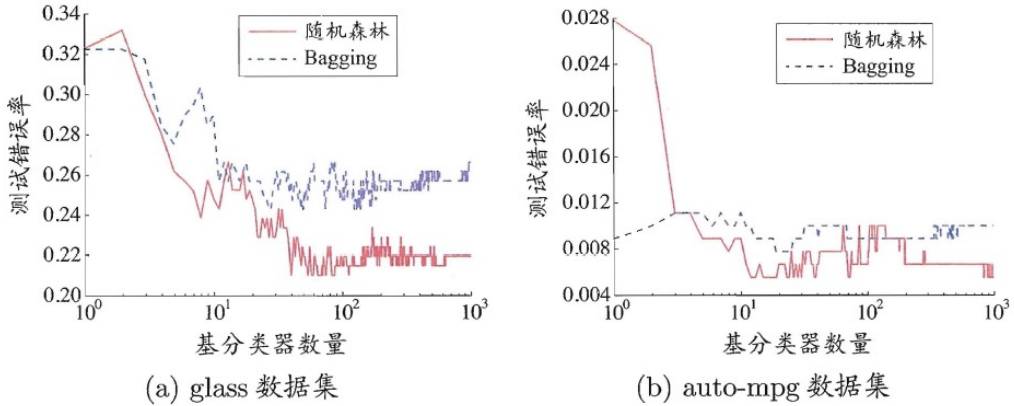
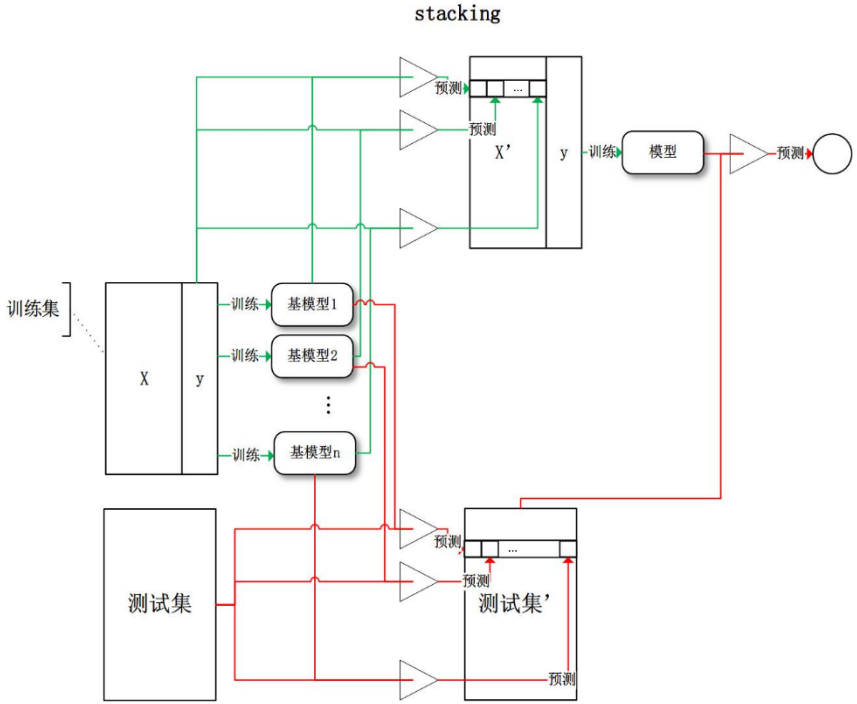


图 8.7 在两个 UCI 数据上，集成规模对随机森林与 Bagging 的影响

目录

Stacking:合成器使用机器学习

- 使用机器学习技术根据基模型（子模型）的输出来预测最后的结果。



Stacking算法

输入： 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
初级学习算法 $\mathfrak{L}_1, \mathfrak{L}_2, \ldots, \mathfrak{L}_T$;
次级学习算法 $\mathfrak{L}$.

过程：

1: **for** $t = 1, 2, \ldots, T$ **do**
2: $\quad h_t = \mathfrak{L}_t(D)$;　使用初级学习算法 $\mathfrak{L}_t$ 产生初级学习器 h_t.
3: **end for**
4: $D' = \varnothing$;
5: **for** $i = 1, 2, \ldots, m$ **do**
6: $\quad$ **for** $t = 1, 2, \ldots, T$ **do**
7: $\quad\quad z_{it} = h_t(\boldsymbol{x}_i)$;
8: $\quad$ **end for**
9: $\quad D' = D' \cup ((z_{i1}, z_{i2}, \ldots, z_{iT}), y_i)$;　生成次级训练集.
10: **end for**
11: $h' = \mathfrak{L}(D')$;　在 D' 上用次级学习算法 $\mathfrak{L}$ 产生次级学习器 h'.

输出： $H(\boldsymbol{x}) = h'(h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_T(\boldsymbol{x}))$

图 8.9 Stacking 算法

总结

- 集成学习（个体与集成）
- Boosting (Re-weighting)与AdaBoost
- Bagging 与 Bootstrap抽样 (Re-sampling)
- Stacking