# 机器学习

## 04 集成学习

李祎

liyi@dlut.edu.cn

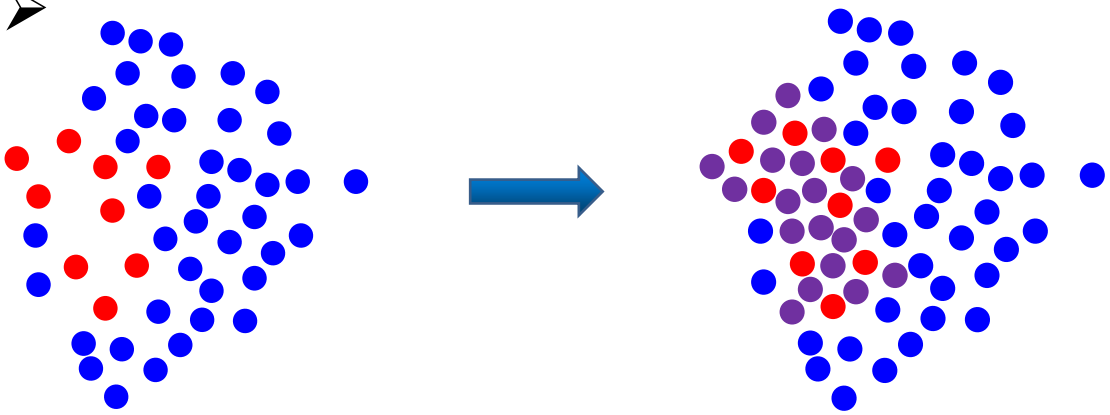
大连理工大学 人工智能学院
School of Artificial Intelligence, Dalian University of Technology

# 目录

# 类别不平衡问题



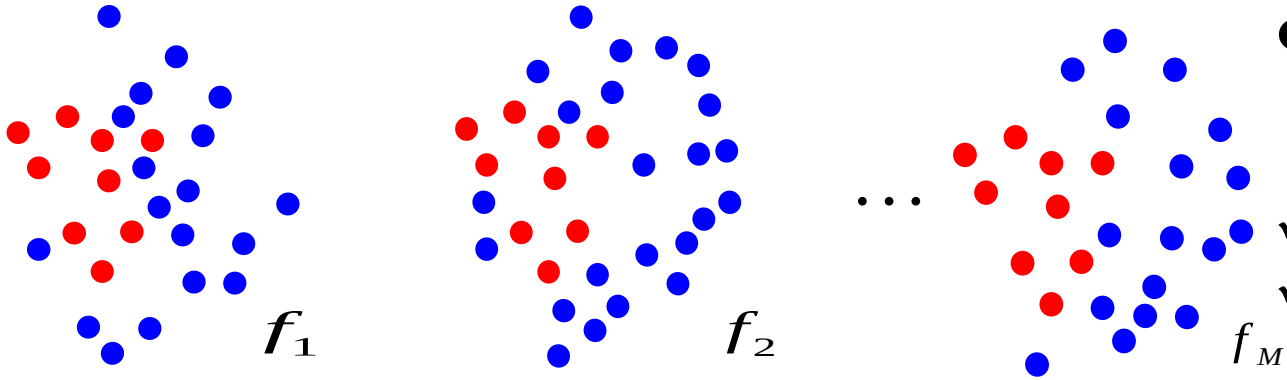
- 过采样 (oversampling):



- ✓ 样本复制
- ✓ 样本插值
- ✓ 样本生成 (GAN)

[1] Chawla N V, Bowyer K W, et al. **SMOTE: Synthetic Minority Over-Sampling Technique**. ***JAIR***, 2002.

- 欠采样 (undersampling)




- 集成学习

$$f = \frac{1}{M}\sum_{m=1}^{M} f_M$$

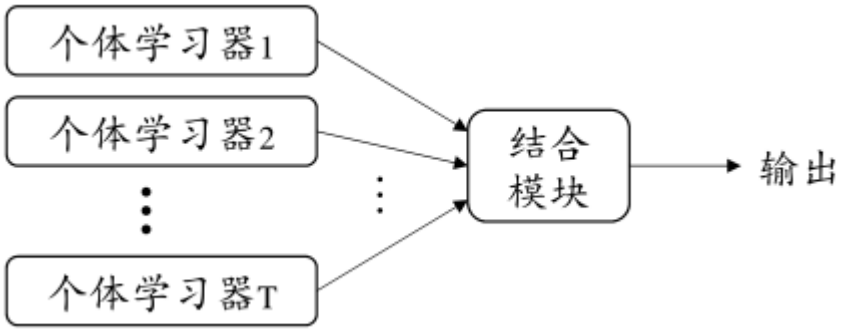
- ✓ EasyEnsemble[2]
- ✓ BalanceCascade[2]

[2] Xu-Ying Liu, Jianxin Wu, Zhi-Hua Zhou. **Exploratory Undersampling for Class-Imbalance Learning**. ***IEEE TSMCB***, 2009.

# 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




都是同一类型学习器：同质　基学习器　　基学习算法

包含不同类型学习器：异质　组件学习器

# 个体与集成



- 例如：一共有500个样本数据，每个子模型只用100个样本数据进行训练。假如每个子模型只有51 %的准确率（至少要比随机猜一个50%要高一点），
- 如果只有一个子模型，整体准确率：51%
- 如果有三个子模型，整体准确率： $0.51^3 + C_3^2 \cdot 0.51^2 \cdot 0.49 = 51.5\%$
- 如果有500个子模型，整体准确率： $\sum_{i=251}^{500} C_{500}^i \cdot 0.51^i \cdot 0.49^{500-i} = 65.6\%$
- 如果子模型的准确率60%，500个子模型集成：

$$\sum_{i=251}^{500} C_{500}^i \cdot 0.6^i \cdot 0.4^{500-i} = 99.999\%$$

# 个体与集成



- 如何获得比单一学习器更好的性能？
- 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

- 集成个体应该：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：

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

- 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



- 上面的分析有一个关键假设：基学习器的误差相互独立
- 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
  - 个体学习器间存在强依赖关系、必须串行生成的序列化方法：Boosting
  - 个体学习器间不存在强依赖关系、可同时生成的并行化方法：Bagging

# Bagging与随机森林

# Bagging



- 并行式集成学习方法最著名的代表
- 基本思想：并行构建 T 个分类器、并行训练，最终将所有分类器的结果进行综合（平均或投票），达到最终的预测结果。
- Bagging: Bootstrap AGGregatING
- 自主采样法 (bootstrap sampling)

# 自助采样法



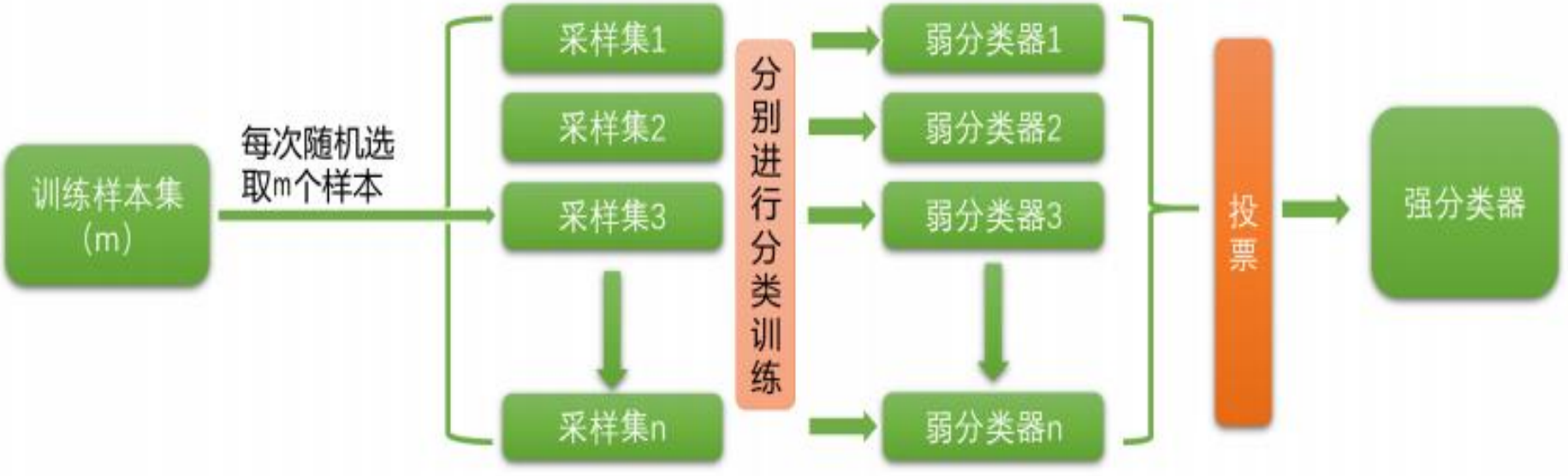
- 给定包含m个样本的数据集D，采样产生数据集D’：
- 每次随机从D中挑选一个样本，将其拷贝放入D’，然后再将该样本放回初始数据集D中，使得该样本在下次采样时仍有可能被采到；
- 这个过程重复执行m次后，我们就得到了包含m个样本的数据集D’。
- D中有一部分样本会在D’中多次出现，而另一部分样本不出现。可以做一个简单的估计，样本在m次采样中始终不被采到的概率是 $(1-\frac{1}{m})^m$ ，取极限得到：

$$\lim_{m\to\infty}(1-\frac{1}{m})^m \to \frac{1}{e} \approx 0.368$$

# Bagging



- 采样出 T 个含 m 个训练样本的采样集
- 基于每个采样集训练出一个基学习器
- 再将这些基学习器进行结合



# 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: $h_t = \mathfrak{L}(D, \mathcal{D}_{bs})$
3: **end for**

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

# 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
- 直接适用于多分类、回归等任务
- 可使用包外估计
  - 由于基学习器只使用了初始训练集中约63.2%的样本，剩下的约36.8%的样本可用作验证集来对泛化性能进行“包外估计”（out-of-bag estimate）。

# Bagging：包外估计



- 需要记录每个基学习器使用的训练样本$D_t$
- $H^{oob}(x)$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

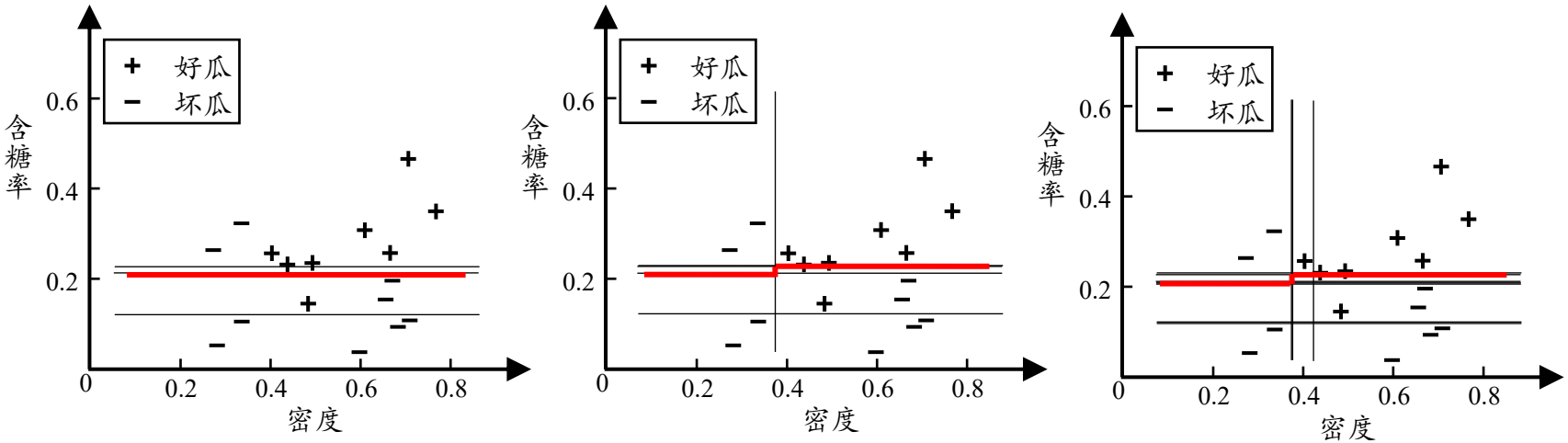
$$H^{oob}(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{I}(h_t(\boldsymbol{x}) = y) \cdot \mathbb{I}(\boldsymbol{x} \notin D_t)$$

- 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实验结果






(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

# 偏差-方差分解



对回归任务，泛化误差可通过“偏差-方差分解”拆解为：

$$E(f;D) = bias^2(\boldsymbol{x}) + var(\boldsymbol{x}) + \varepsilon^2$$

期望输出与真实输出的差别

$$bias^2(\boldsymbol{x}) = \left(\bar{f}(\boldsymbol{x}) - y\right)^2$$

同样大小的训练集的变动，所导致的性能变化

$$var(\boldsymbol{x}) = \mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)^2\right]$$

训练样本的标记与真实标记有区别

表达了当前任务上任何学习算法所能达到的期望泛化误差下界

$$\varepsilon^2 = \mathbb{E}_D\left[\left(y_D - y\right)^2\right]$$

泛化性能是由学习算法的能力、数据的充分性以及学习任务本身的难度共同决定

# 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
  - 传统决策树：

    当前结点的所有属性中选择一个最优属性。
  - RF：

    对基决策树的每个结点，

    先从该结点的属性集合中随机选择一个包含k个属性的子集，

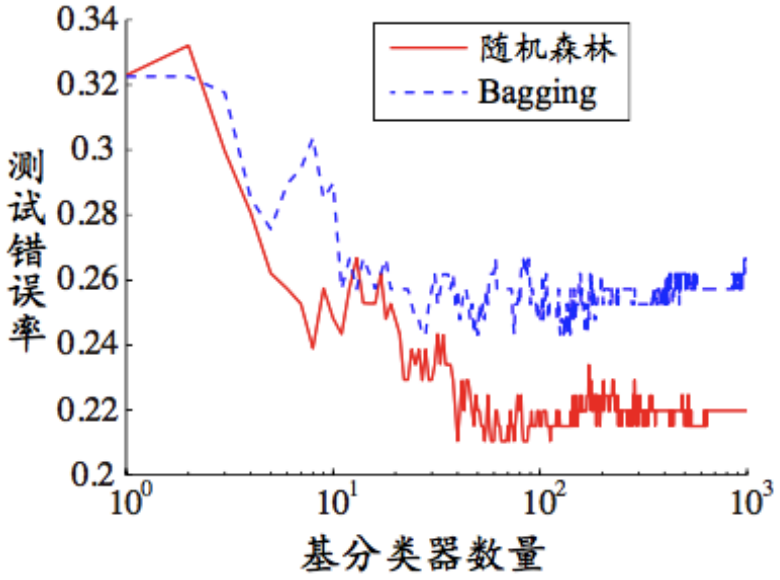
    然后再从这个子集中选择一个最优属性用于划分。

# 随机森林

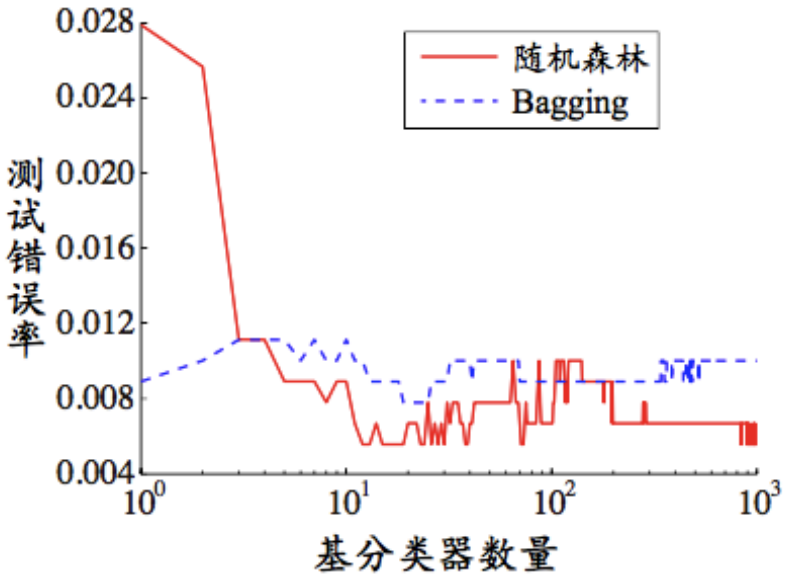


**基学习器的多样性：**

- Bagging 中仅通过样本扰动产生，
- 随机森林中增加了属性扰动，使得最终性能通过个体学习器之间差异度进一步提升。




(a) glass 数据集



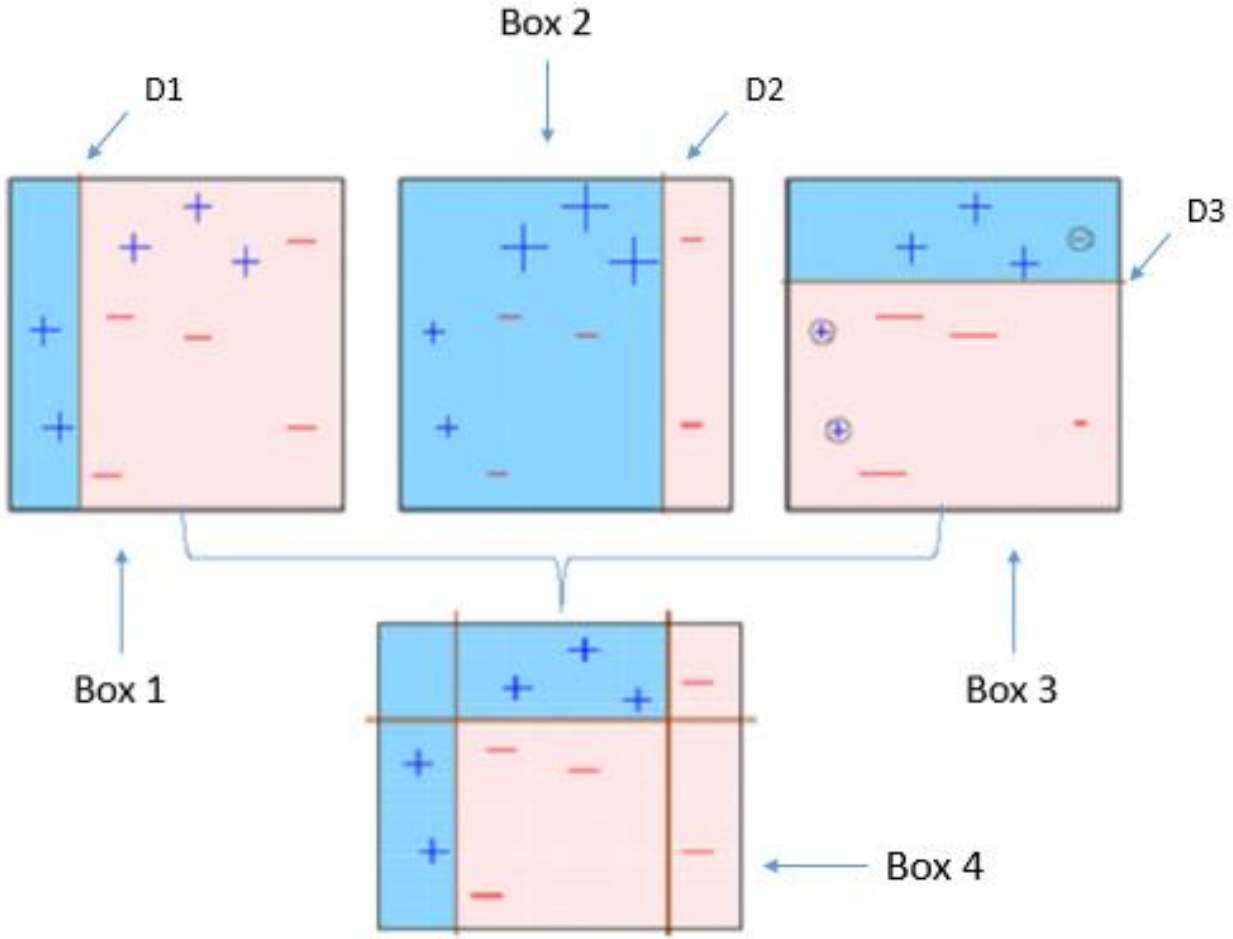

(b) auto-mpg 数据集

# Boosting

# Boosting



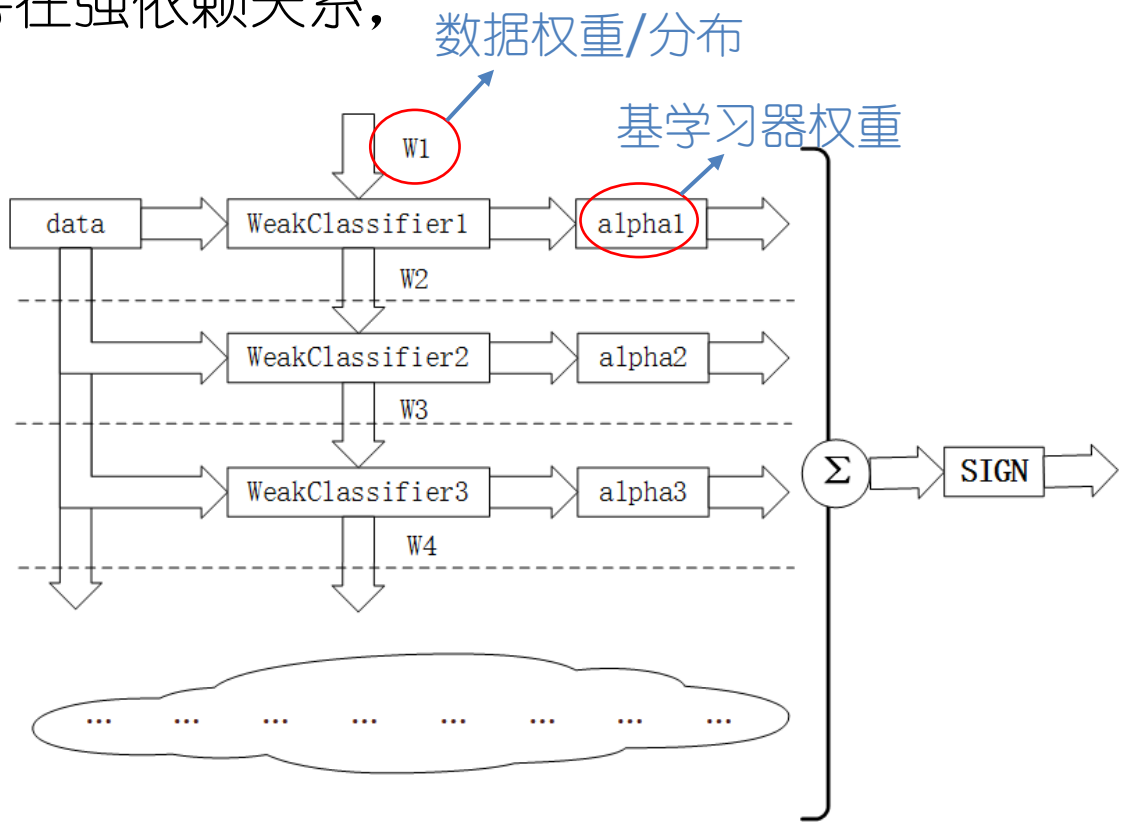
- 个体学习器存在强依赖关系，
- 串行生成
- 每次调整训练数据的样本分布



# Boosting



- 个体学习器存在强依赖关系，
- 串行生成




1. 给定初始训练数据，由此训练出第一个基学习器，并确定其权重；
2. 对样本分布进行调整，在之前学习器做错的样本上投入更多关注；
3. 用调整后的样本，训练下一个基学习器，确定其权重；
4. 重复上述过程 T 次，将 T 个学习器加权结合。

# AdaBoost



- 给定数据集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$

  其中，$y_i \in \{-1, +1\}$

- 模型输出：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：0/1损失的一致性替代损失函数

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

# 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

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

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

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

指数损失函数最小化，则分类错误率也将最小化，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。

# 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

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

- 令指数损失函数的导数为0，即

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

# 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

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

# AdaBoost推导



- 于是，理想的基学习器：

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

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

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

# 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：

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

# AdaBoost推导



- 则理想的基学习器

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

- 最终的样本分布更新公式

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

# 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: $\mathcal{D}_1(\boldsymbol{x}) = 1/m.$ 初始化样本权重

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

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

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

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

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

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

8: **end for**

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

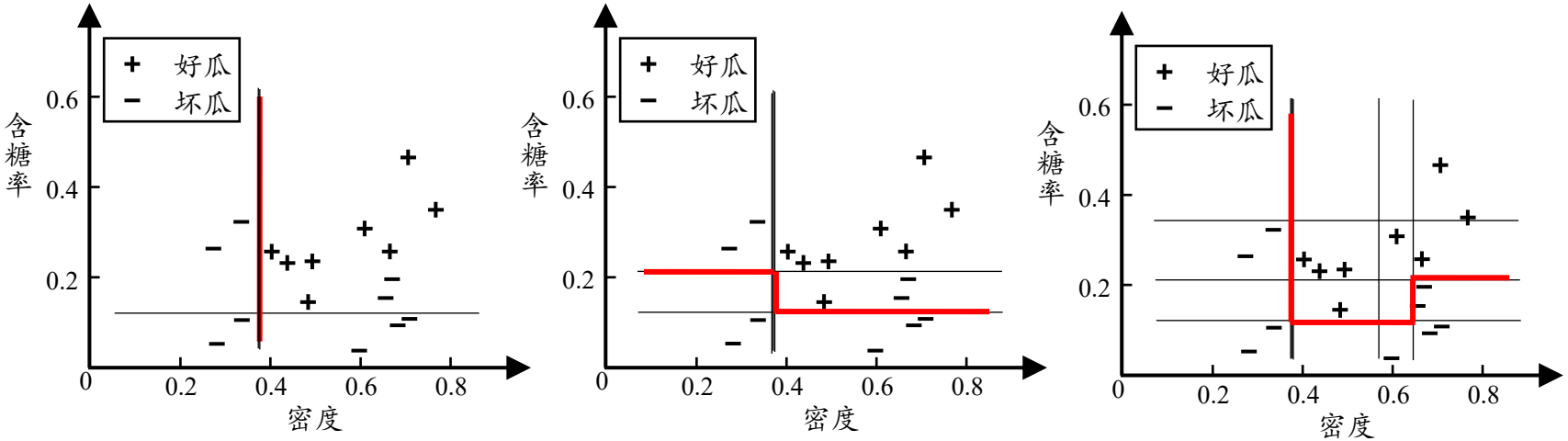
# AdaBoost特点



- 优点
  - 不容易发生过拟合；
  - 由于AdaBoost并没有限制弱学习器的种类，所以可以使用不同的学习算法来构建弱分类器；
  - 相对于bagging算法和Random Forest算法，AdaBoost充分考虑的每个分类器的权重；
  - AdaBoost的参数少，实际应用中不需要调节太多的参数。
- 缺点
  - AdaBoost迭代次数（弱分类器数目）不好设定，可以使用交叉验证来确定；
  - 对异常样本敏感，异常样本在迭代中可能会获得较高的权重，影响最终的强学习器的预测准确性；
  - 训练比较耗时。

# 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：boosting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

# 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

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

# 平均法



- 简单平均法是加权平均法的特例
- 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
  - 不同方式确定基学习器权重
- 加权平均法未必一定优于简单平均法
  - 训练样本不充分或含有噪声，导致学出的权重不完全可靠
- 一般而言，个体学习器性能相差较大：加权平均

  性能相近：简单平均法

# 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）

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

不同类型的$h_i^j(x)$值不能混用

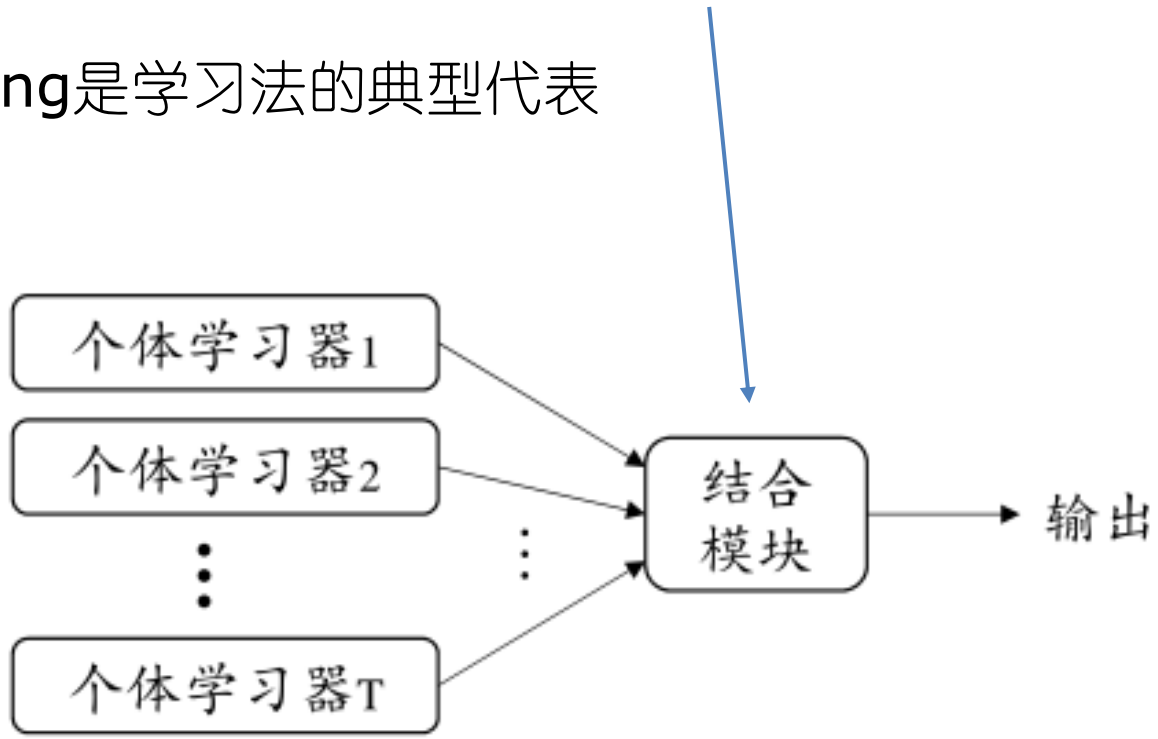
- 加权投票法（weighted voting）

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

# 学习法



- 初级学习器+次级学习器/元学习器
- Stacking是学习法的典型代表



# 多样性

# 理论分析“好而不同”



- 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：

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

# 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

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

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

# 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$x$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



- 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

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

- 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)

$$\kappa = \frac{p_1 - p_2}{1 - p_2} \qquad p_1 = \frac{a + d}{m},$$

$$p_2 = \frac{(a+b)(a+c) + (c+d)(b+d)}{m^2}$$

# 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

- 不同的多样性增强机制可以同时使用
  - 例如，随机森林

# 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

# 总结



- 什么是集成学习
- Bagging与随机森林
- Boosting
  - Adaboost
- 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