MACHINE LEARNING

机器学习

Ensemble Learning

集成学习

参考：
《机器学习》

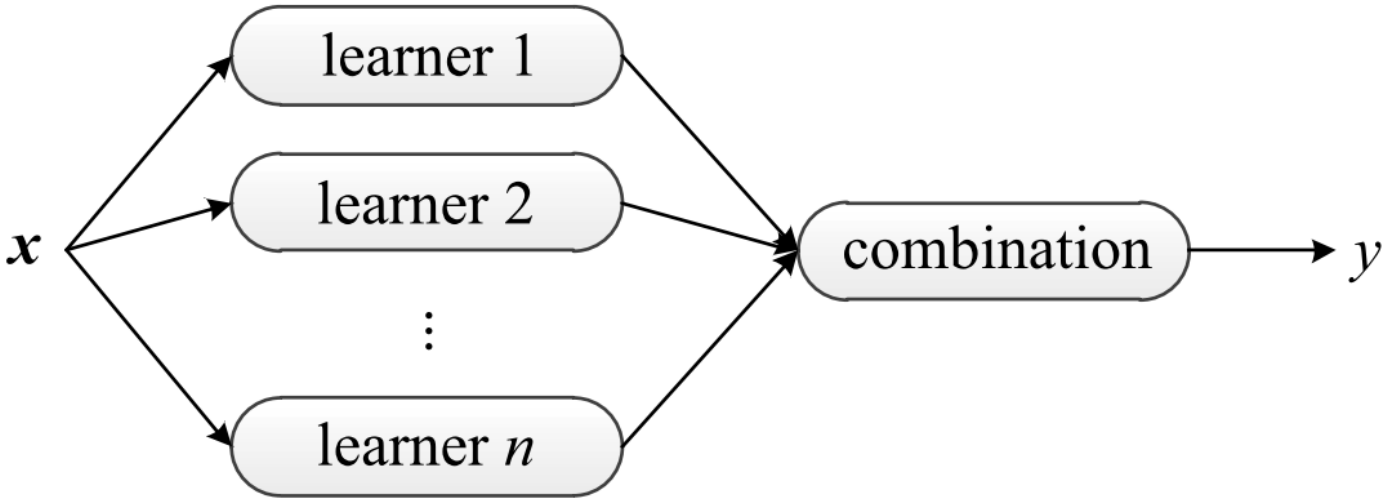
Machine Learning Course

Copyright belongs to Wenting Tu.

集成学习

- 定义

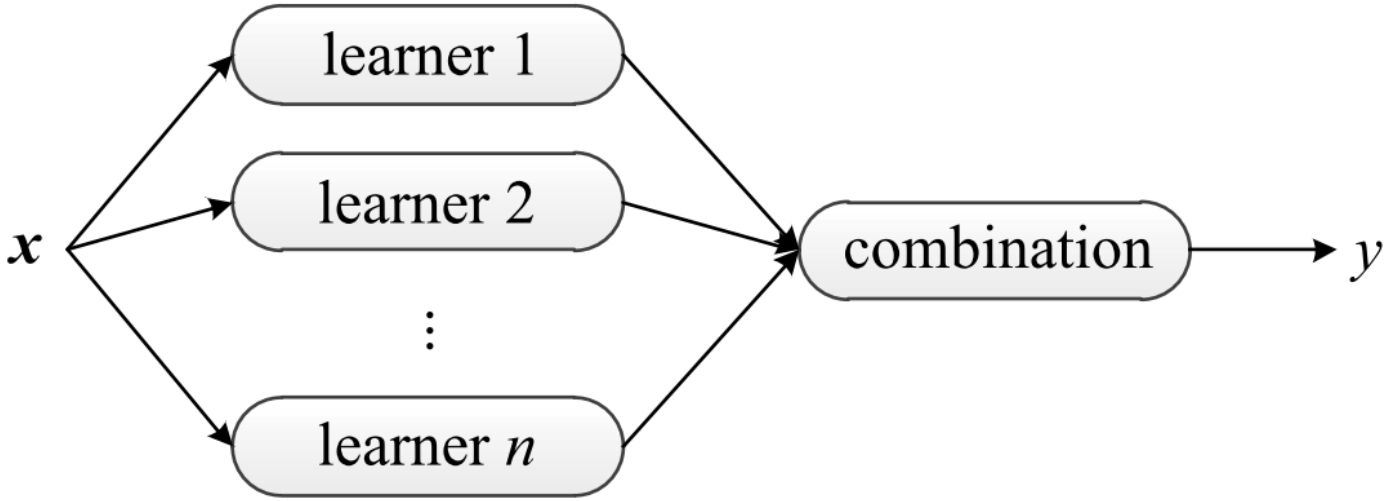
集成学习(ensemble learning)通过构建并结合多个学习器来完成学习任务，有时也被称为多分类器系统(multi-classifier system) 、 基于委员会的学习(committee-based learning) 等。



集成学习

- 一般结构

先产生一组“个体学习器” (individual learner), 再用某种策略将它们结合起来.。



- 类型

“同质”的(homogeneous)：集成中只包含同种类型的个体学习器
“异质”的(homogeneous)：个体学习器由不同的学习算法生成

“平行”的(parallel)：基模型的学习是独立的，可以平行完成的
“序贯”的(parallel)：基模型的学习不是独立的，需要按顺序完成的

集成学习

- 准则

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

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

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

学习器不能太坏，并且要有“多样性”(diversity)， 即学习器间具有差异.

集成学习

- 误差 - 分歧分解 *error - ambiguity decomposition*

$$y_{COM}(\boldsymbol{x}) = \frac{1}{M}\sum_{m=1}^{M} y_m(\boldsymbol{x})$$

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

$\hat{E}$ 为集成模型的错误率

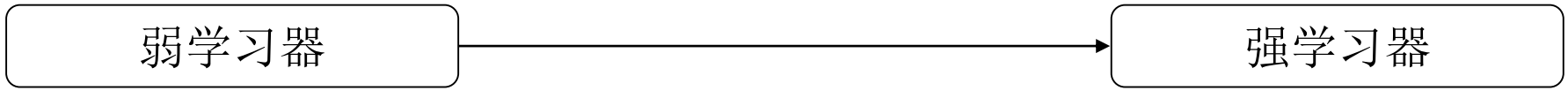
$\bar{E}$ 为基模型的平均错误率

$\bar{A}$ 为基模型的多样性

为了获得表现优异的集成学习模型，我们应该尽量地整合独立、多样并且具有一定效果（起码好于随机猜测）的基模型

Boosting

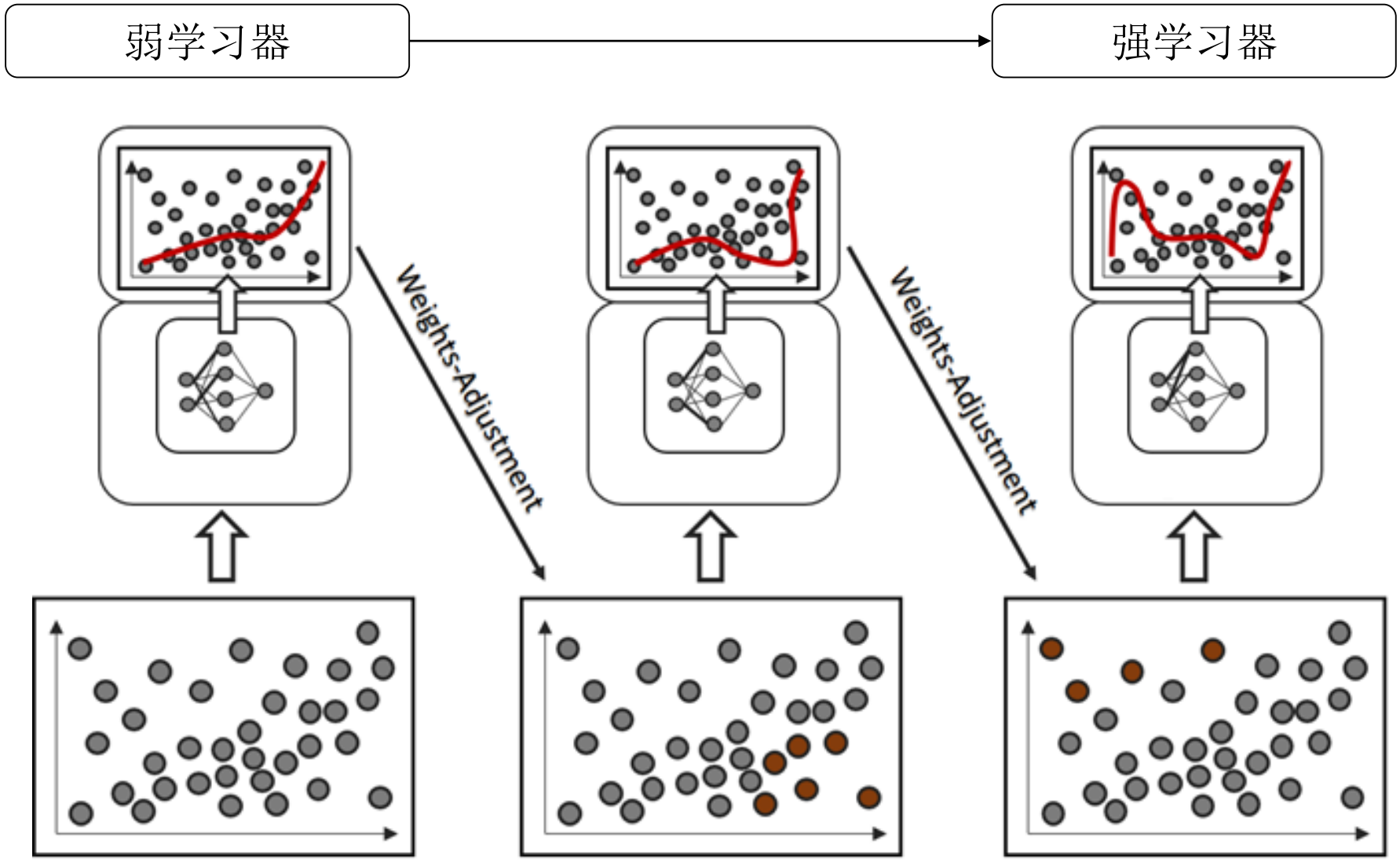
- 定义



先从初始训练集训练出一个基学习器，
再根据基学习器的表现对训练样本分布进行调整，
使得先前基学习器做错的训练样本在后续受到更多关注，
然后基于调整后的样本分布来训练下一个基学习器；
如此重复进行，直至基学习器数目达到事先指定的值T，
最终将这T个基学习器进行加权结合.

Boosting

- 定义



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

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

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

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

Boosting

- Adaboost

> 针对二分类

$y_i \in \{-1, +1\}$

> 指数损失函数

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

- Gradient Boosting

> 支持分类与回归

> 许多可导损失函数

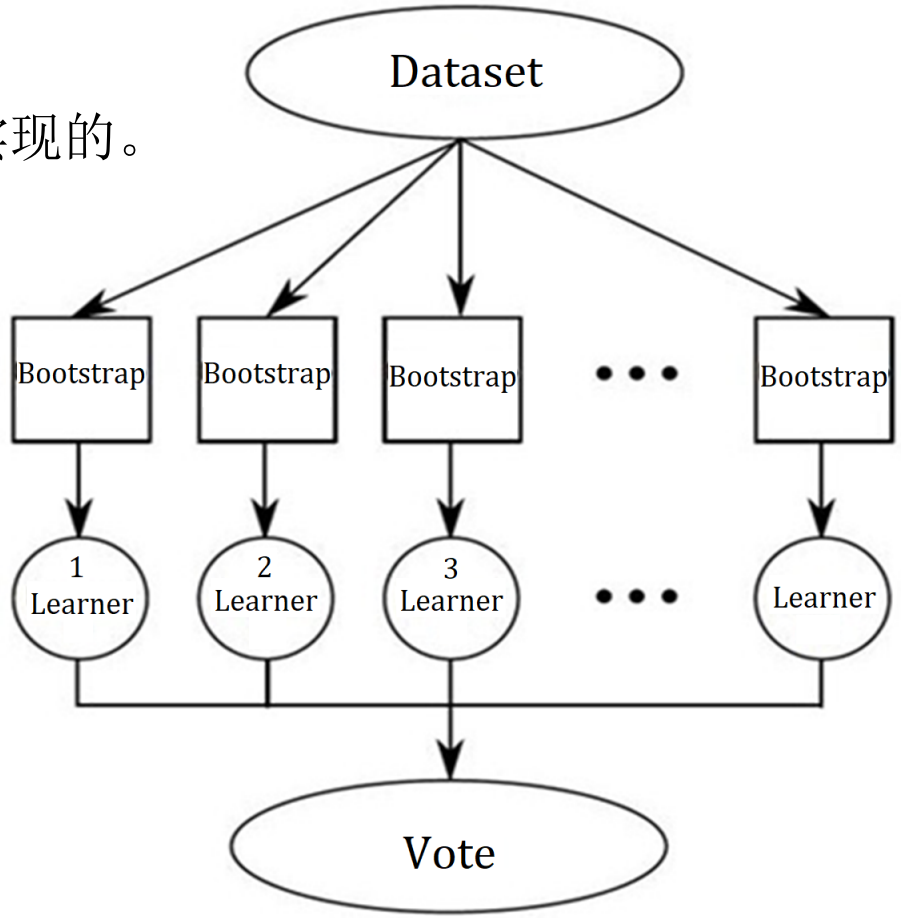
Bagging

- 定义

Bagging 是并行式集成学习方法最著名的代表。
它得名于它是基于自助(bootstrapping)采样法实现的。

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.

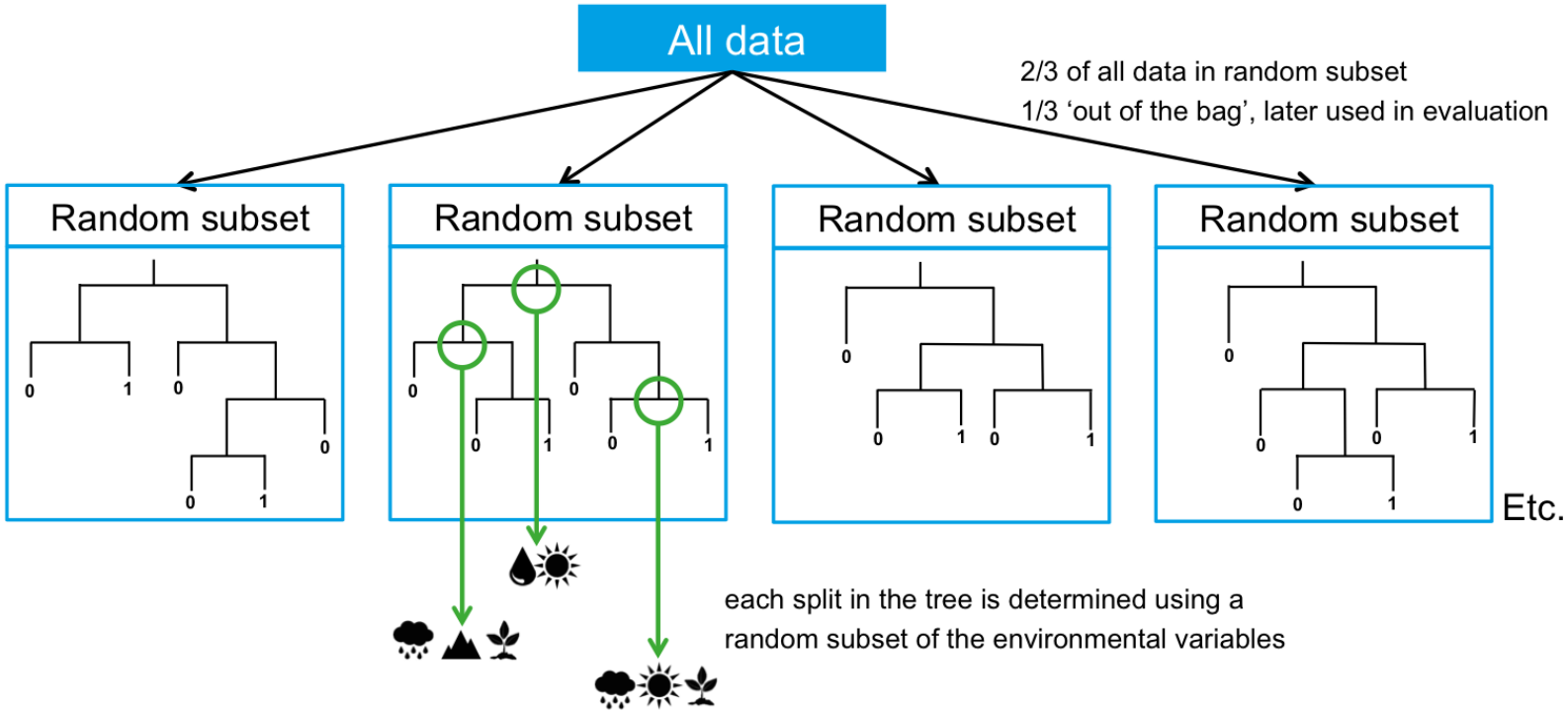
过程:
1: for $t = 1, 2, \ldots, T$ do
2: $h_t = \mathfrak{L}(D, \mathcal{D}_{bs})$
3: end for
输出: $H(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{I}(h_t(\boldsymbol{x}) = y)$



Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

Bagging与随机森林

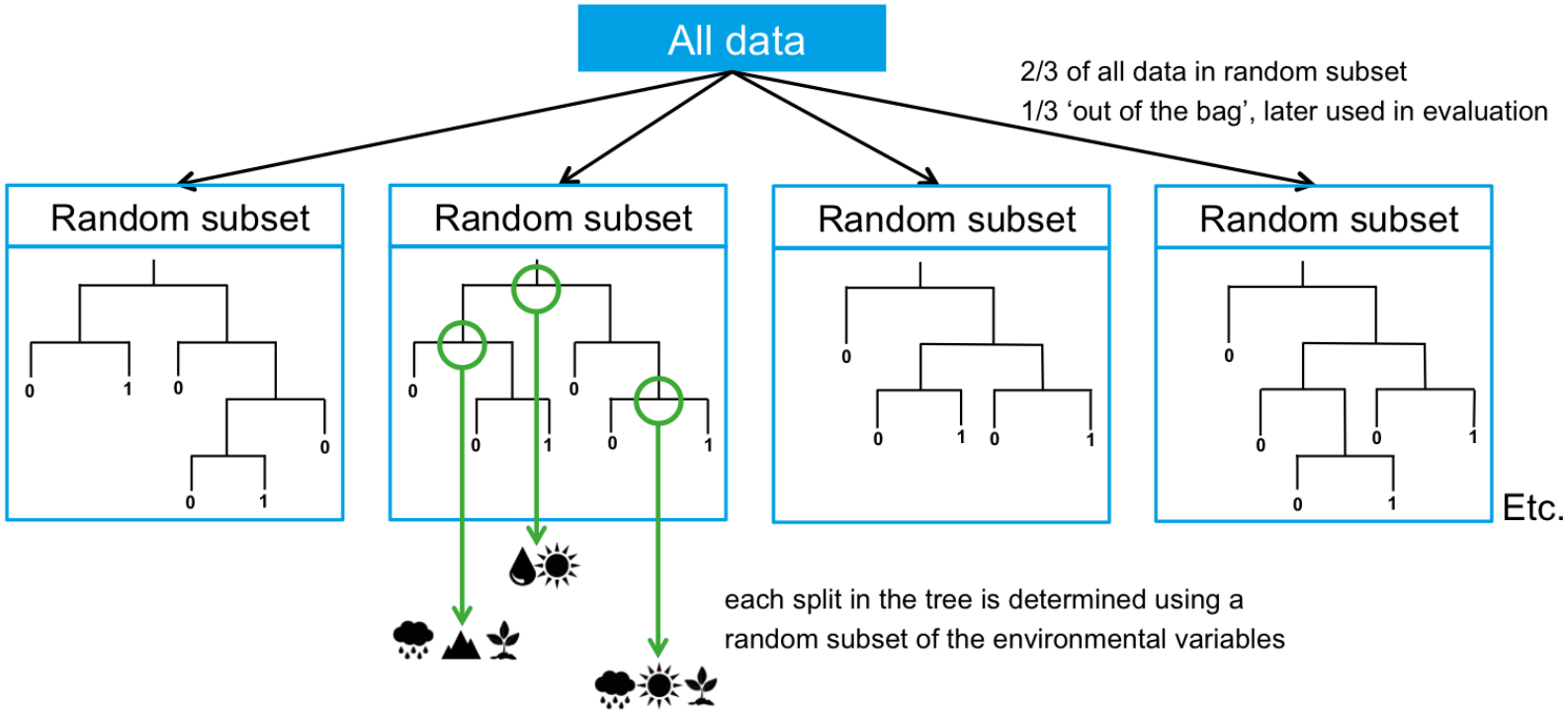
- 随机森林(Random Forest,简称 RF)



随机森林$(Random\ Forest$, 简称 $RF)$是 $Bagging$ 的一个扩展变体. RF 在以决策树为基学习器构建 $Bagging$ 集成的基础上，进一步在决策树的训练过程中引入了随机属性选择.

Bagging与随机森林

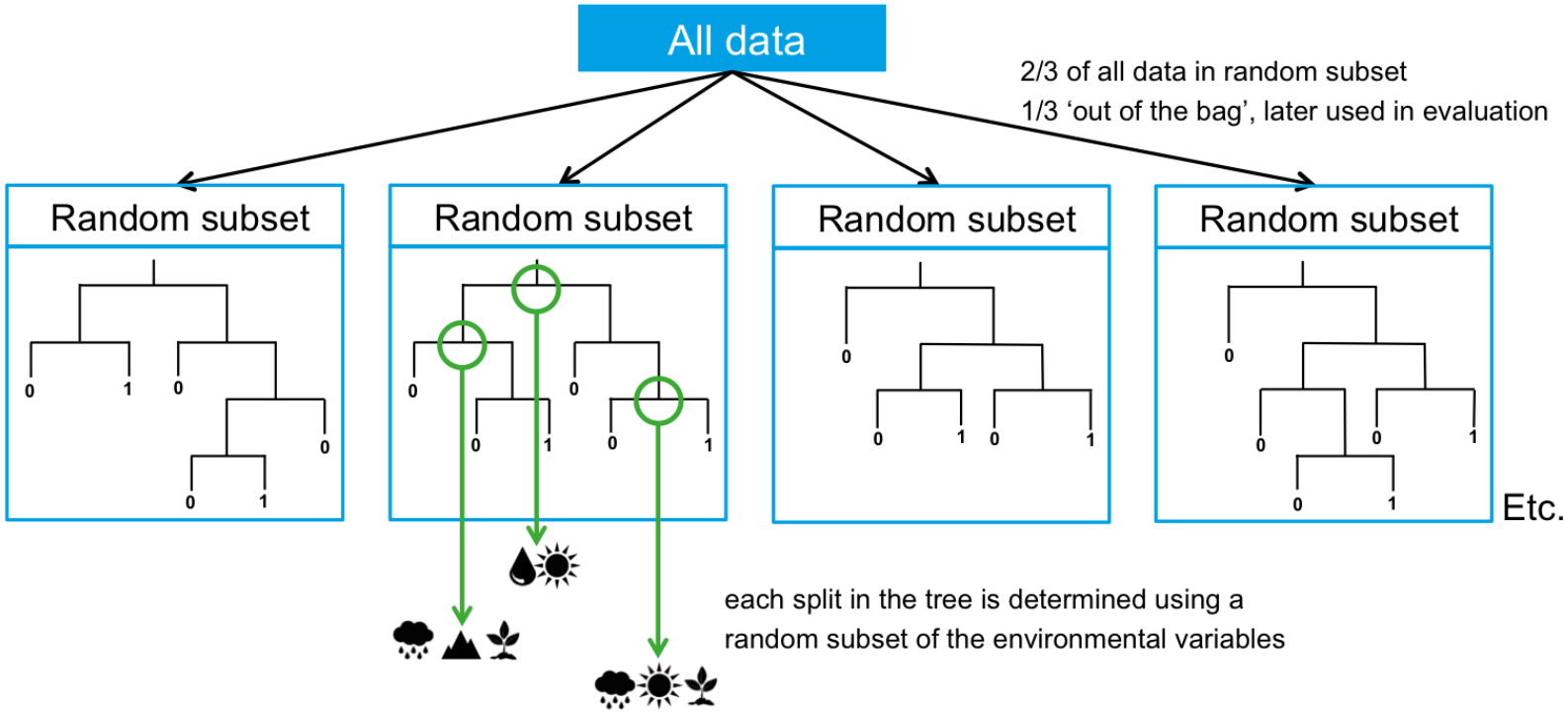
- 随机森林(Random Forest,简称 RF)



k控制了随机性的引入程度在RF中，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.

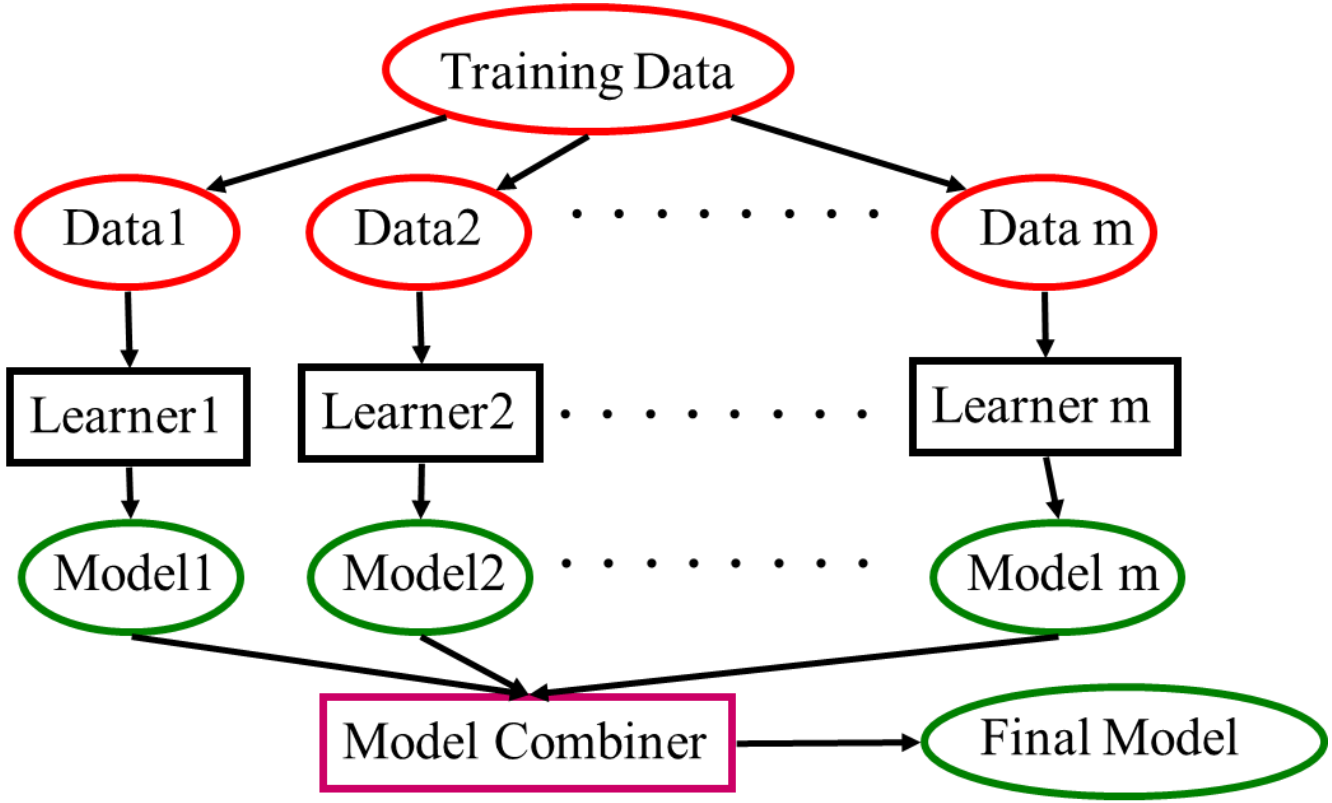
Bagging与随机森林

- 随机森林(Random Forest,简称 RF)



与 *Bagging* 中基学习器的"多样性"仅通过样本扰动(通过对初始训练集采样)而来不同，随机森林中基学习器的多样性不仅来自样本扰动，还来自属性扰动，这就使得最终集成的泛化性能可通过个体学习器之间差异度的增加而进一步提升.

结合策略学习法

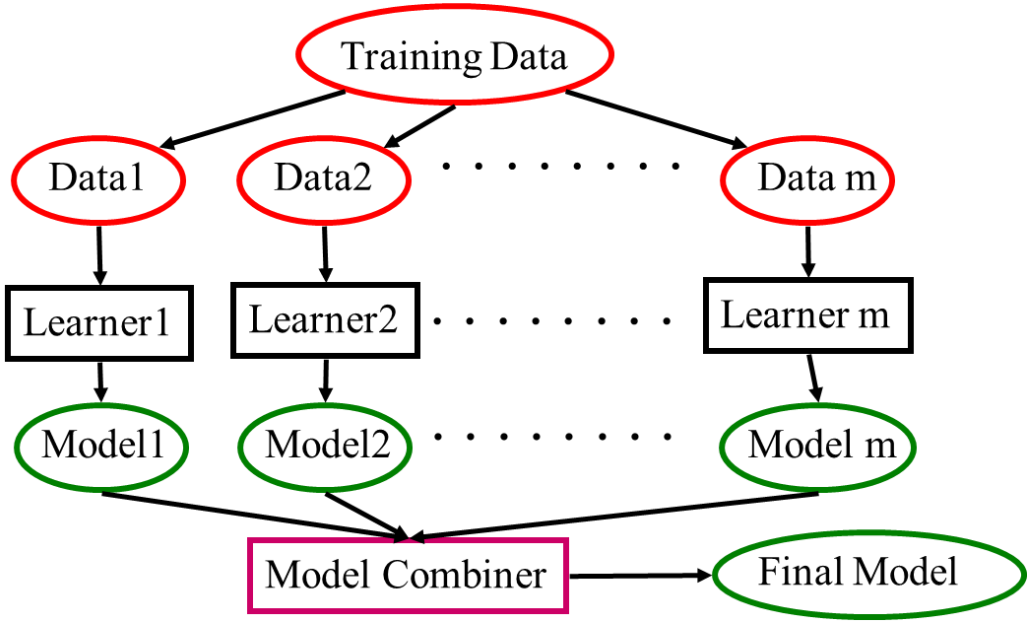


假定集成包含T个基学习器$\{h_1, h_2, \ldots, h_T\}$,其中$h_i$在示例x上的输出为$h_i(x)$.
对h_i运用合适的结合策略合可能会从多个方面带来好处

结合策略学习法

- Stacking

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: $\quad h_t = \mathfrak{L}_t(D)$
3: end for
4: $D' = \varnothing$
5: for $i = 1, 2, \ldots, m$ do
6: $\quad$ for $t = 1, 2, \ldots, T$ do
7:
$z_{it} = h_t(\boldsymbol{x}_i)$
8: end for
9 : $\quad D' = D' \cup ((z_{i1}, z_{i2}, \ldots, z_{iT}), y_i)$
10: end for
11: $h' = \mathfrak{L}(D')$
输出: $H(\boldsymbol{x}) = h'(h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_T(\boldsymbol{x}))$



结合策略学习法

- Stacking

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: $\quad h_t = \mathfrak{L}_t(D)$
3: end for
4: $D' = \varnothing$
5: for $i = 1, 2, \ldots, m$ do
6: $\quad$ for $t = 1, 2, \ldots, T$ do
7:
$z_{it} = h_t(\boldsymbol{x}_i)$
8: end for
9 : $\quad D' = D' \cup ((z_{i1}, z_{i2}, \ldots, z_{iT}), y_i)$
10: end for
11: $h' = \mathfrak{L}(D')$
输出: $H(\boldsymbol{x}) = h'(h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_T(\boldsymbol{x}))$

过拟合风险：
学习基学习器的数据与学习融合器的数据发生了重合

结合策略学习法

- Stacking

输入: 训练集
$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$;
初级学习算法 $\mathfrak{L}_1, \mathfrak{L}_2, \ldots, \mathfrak{L}_T$;
次级学习算法 L. 过程:
1 : $D \longrightarrow D_1, D_2, \ldots, D_k$
2: for $j = 1, 2, \ldots, k$ do
3 : $\bar{D}_j = D \backslash D_j$
4 : $h_t^{(j)} = \mathfrak{L}_t(\bar{D}_j)$
5 : for $x_i \in D_j$ do
6: for $t = 1, 2, \ldots, T$ do
7: $z_{it} = h_t^{(j)}(\boldsymbol{x}_i)$
8: end for
9 : $D' = D' \cup ((z_{i1}, z_{i2}, \ldots, z_{iT}), y_i)$
10 : end for
11: end for
12: $h' = \mathfrak{L}(D')$
输出: $H(x) = h'(h_1(x), h_2(x), \ldots, h_T(x))$

用学习基模型未使用的样本来
形成学习融合器的训练样本