

集成学习

高 阳，李文斌

<http://cs.nju.edu.cn/rl>

2023年03月16日

三个臭皮匠能不能顶个诸葛亮？



大纲

集成学习原理

Bagging和随机森林

Boosting和GBDT、XGboost

Adaboost

其他

大纲

集成学习原理

Bagging和随机森林

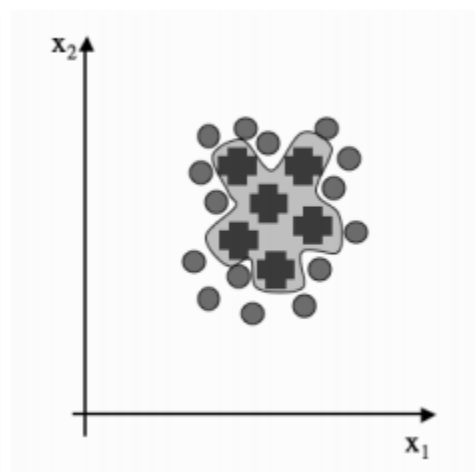
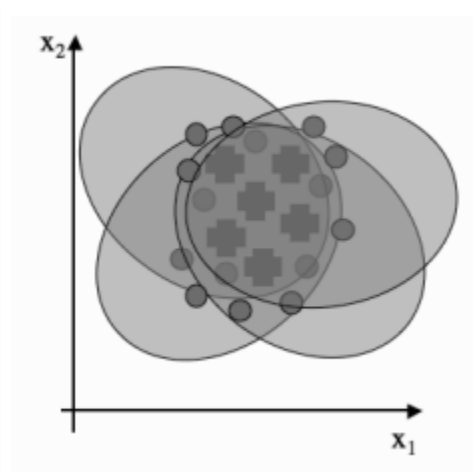
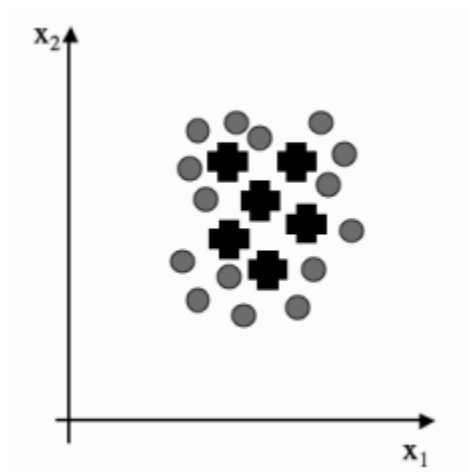
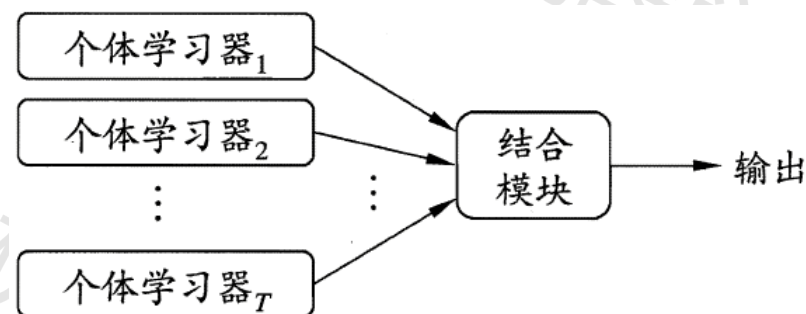
Boosting和GBDT、XGboost

Adaboost

其他

原理

- ✓ 三个臭皮匠，抵上诸葛亮
- ✓ 是一个预测模型的元方法



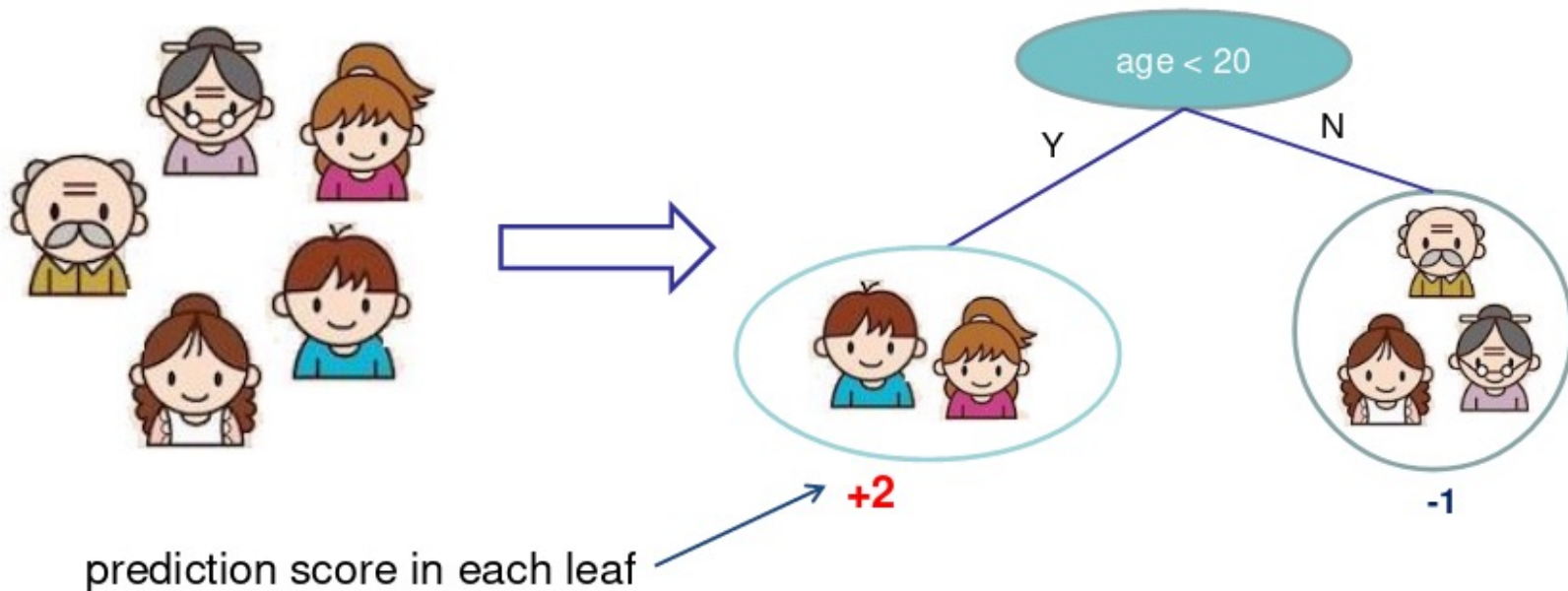
例子

□ 回归树

- 决策规则跟决策树一样
- 每个叶子节点包含一个分数

Input: age, gender, occupation, ...

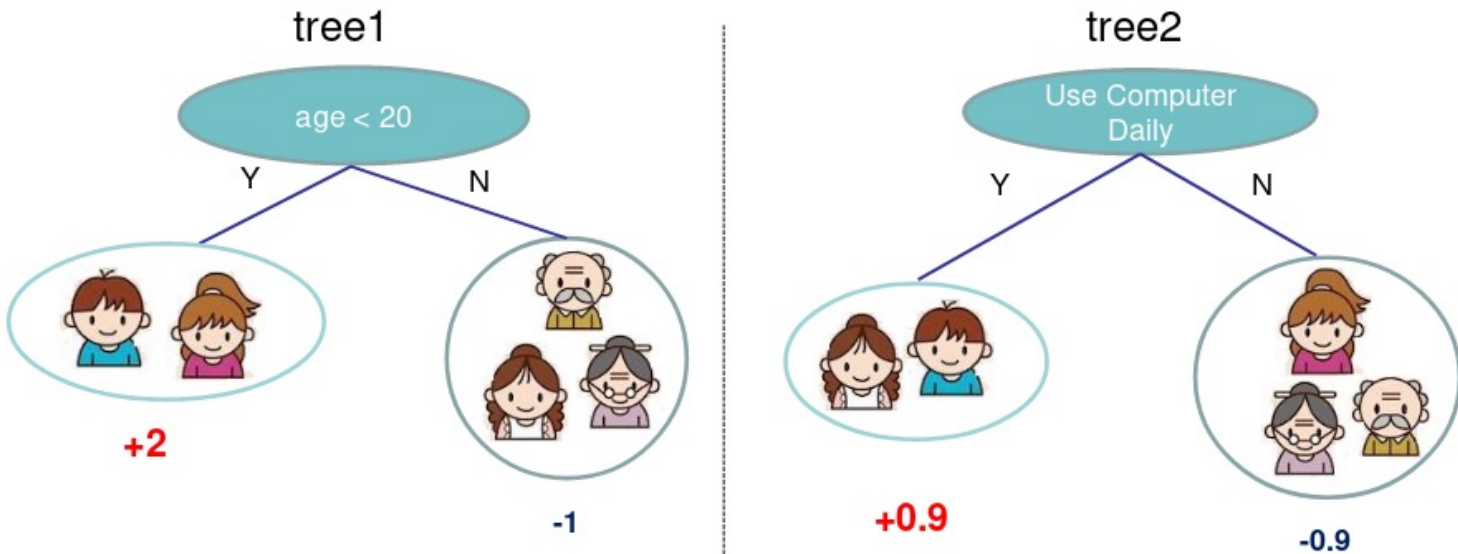
Like the computer game X



例子

□ 回归树集成

- 多棵树一起决策



$$f(\text{boy}) = 2 + 0.9 = 2.9 \quad f(\text{old man}) = -1 - 0.9 = -1.9$$

Prediction of is sum of scores predicted by each of the tree

特点（分类）

- ✓ 多个分类器集成在一起，以提高分类准确率
- ✓ 由训练数据构建基分类器，然后根据预测结果进行投票
- ✓ 集成学习本身不是一种分类器，而是分类器结合方法
- ✓ 通常集成分类器性能会好于单个分类器

准确度较高，差异度较高

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

准确度较高，差异度较低

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

准确度较低，差异度较高

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

举例分析

- ✓ 例：以多数投票法为结合方法
- ✓ 每个二分类器的分类精度为 p
- ✓ 则集成 T 个分类器的分类精度为

(假设基学习器的精度相互独立)

$$\sum_{k=\frac{T}{2}+1}^T \binom{T}{k} p^k (1-p)^{T-k}$$

$p > 0.5$ 且 $T \rightarrow \infty$, 上式 $\rightarrow 1$

准确性，多样性之间的矛盾

Bias-Variance tradeoff

□ Bias (学习结果的期望与真实规律的差距)

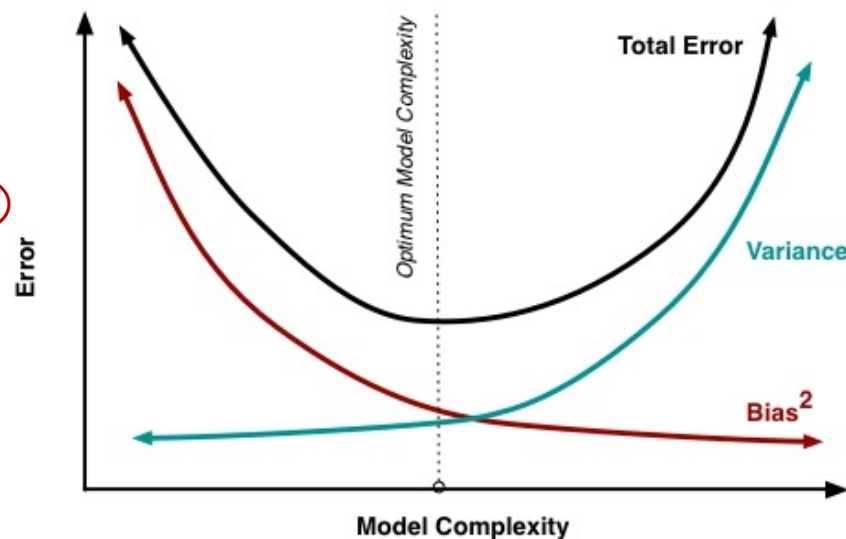
$$\text{Bias} = E[\hat{f}(x)] - f(x)$$

□ Variance (学习结果自身的不稳定性)

$$\text{Variance} = E[(\hat{f}(x) - E[\hat{f}(x)])^2]$$

□ Total Error (以均方误差为例)

$$\text{Err}(x) = \text{Bias}^2 + \text{Variance} + \text{Random Error}$$



核心问题

□ 序列集成(基学习器)法

- ✓ 利用基学习器之间的依赖关系，依次生成
- ✓ 减小偏差bias

□ 并行集成(基学习器)法

- ✓ 利用基学习器之间的独立关系，并行生成
- ✓ 减小方差variance

Q1：如何训练每个学习器；

Q2：如何结合每个学习器。

结合策略

□ 平均法(回归问题)

- ✓ 简单平均
- ✓ 加权平均

□ 投票法(分类问题)

- ✓ 绝对多数
- ✓ 相对多数
- ✓ 加权投票

□ 学习法(Stacking): 本讲最后讨论

多样性策略(学习基学习器)

□ 数据层面

- ✓ 输入样本的扰动, 构建基学习器
- ✓ 输出样本的扰动, 构建基学习器

□ 属性层面

- ✓ 随机选择部分属性, 构建基学习器

□ 参数层面

- ✓ 算法模型参数的扰动, 构建基学习器

大纲

集成学习原理

Bagging和随机森林

Boosting和GBDT、XGboost

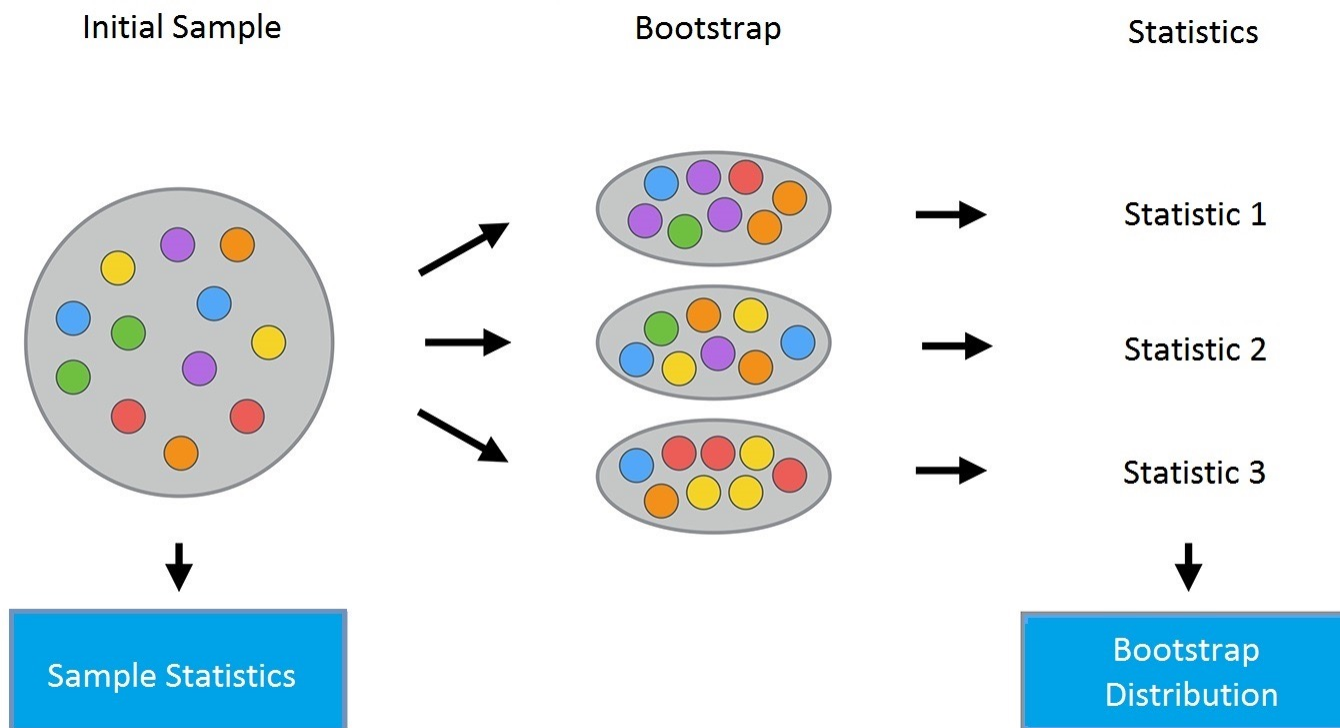
Adaboost

其他

Bagging

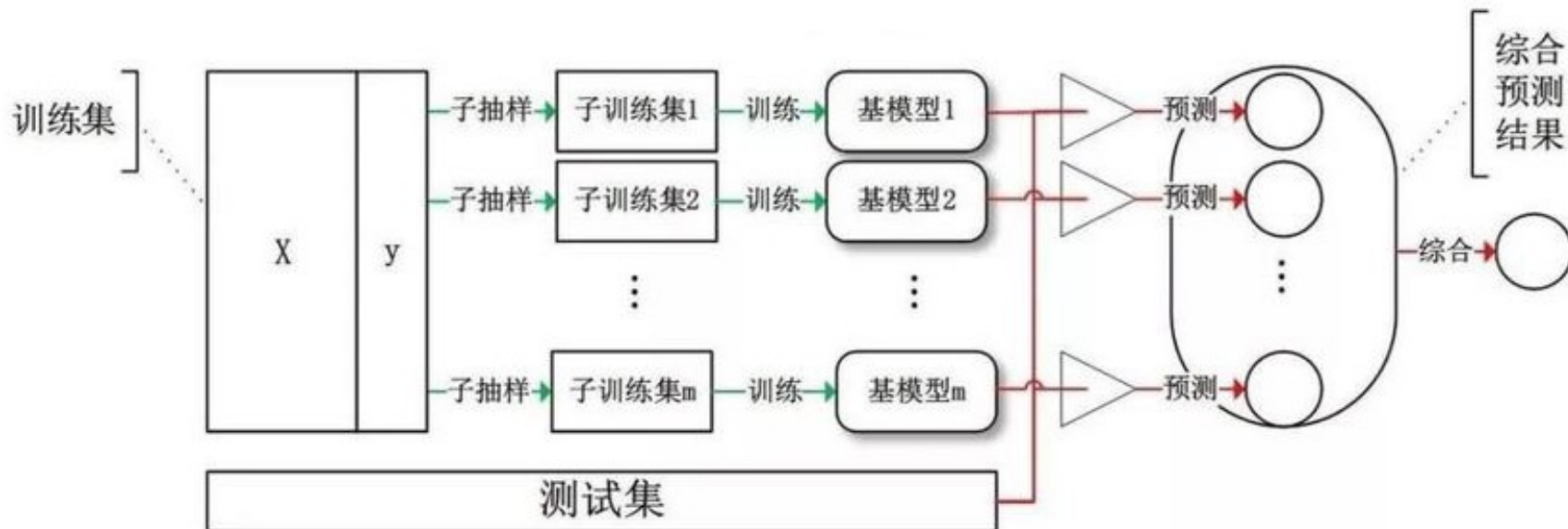
□ Bagging (Bootstrap aggregating)基本原理

- ✓ 有放回采样方法。统计上的目的是得到统计量分布以及置信区间。



Bagging

□ Bagging集成学习框架



Bagging集成学习优点

□ 降低分类器方差，改善泛化

- ✓ 其性能依赖于基分类器的稳定性。如果基分类器稳定，则其误差主要由基分类器的bias决定；
- ✓ 由于采样概率相同，bagging方法并不侧重于任何特定实例；
- ✓ 代表性算法：随机森林（Random Forest）

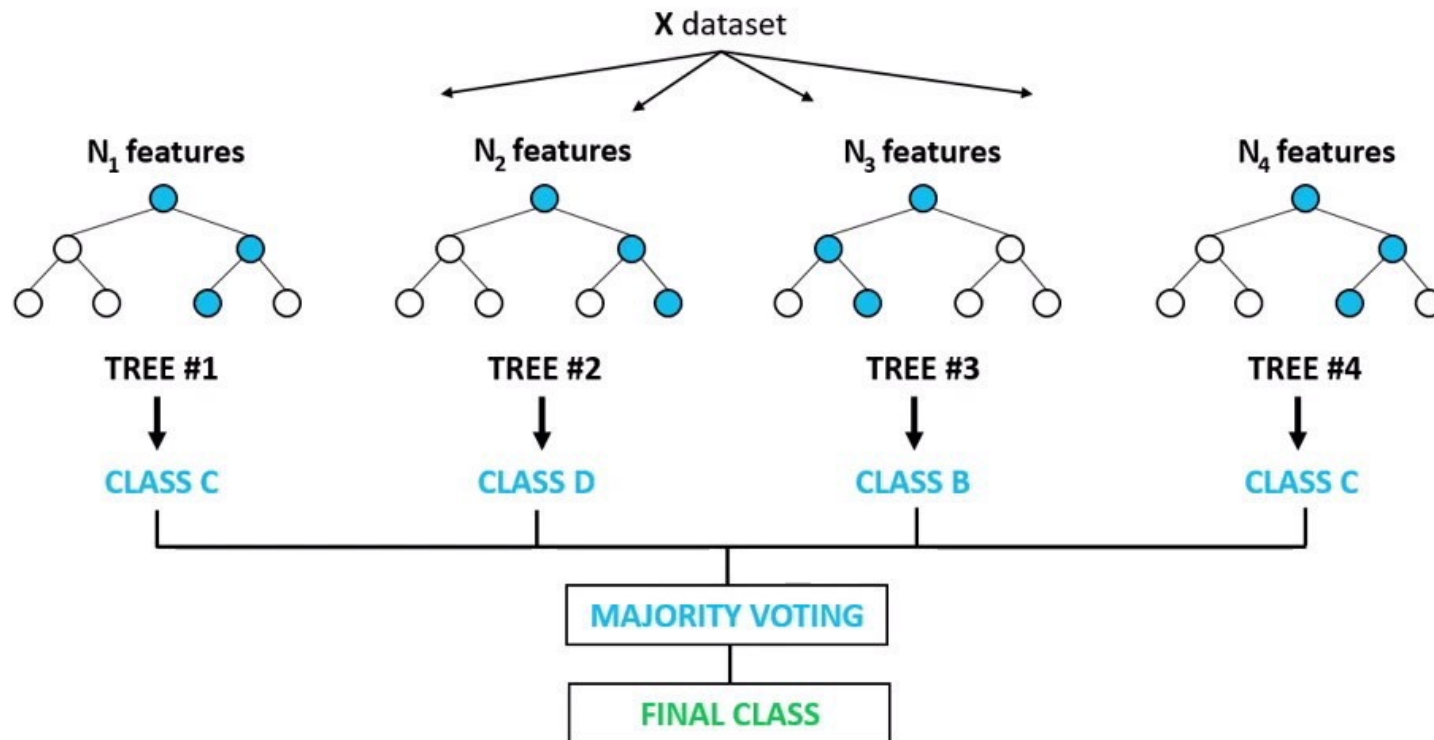
代表性算法：随机森林RF

- ✓ 用 N 来表示训练用例（样本）的个数， M 表示特征数目；输入特征数目 m ，用于确定决策树上一个节点的决策结果；其中 m 应远小于 M 。
-

- ✓ 从 N 个训练用例(样本)中以有放回抽样的方式，取样 N 次，形成一个训练集（即bagging取样），并用未抽到的用例(样本)作预测，评估其误差。
- ✓ 对于每一个节点，随机选择 m 个特征(通常为 M 的均方根 $\log_2 M$)，根据这 m 个特征，计算其最佳的分裂方式。
- ✓ 每棵树都会完整成长而不会剪枝，这有可能在建完一棵正常树状分类器后会被采用。
- ✓ 以上过程做充分多次，以产生足够多的随机树。

代表性算法：随机森林RF

Random Forest Classifier



大纲

集成学习原理

Bagging和随机森林

Boosting和GBDT、XGboost

Adaboost

其他

提升 (Boosting)

□ PAC学习理论

✓ 能否从假设空间 H 中学习到一个好的假设 h

✓ 近似正确：泛化误差 $E(h)$ 足够小

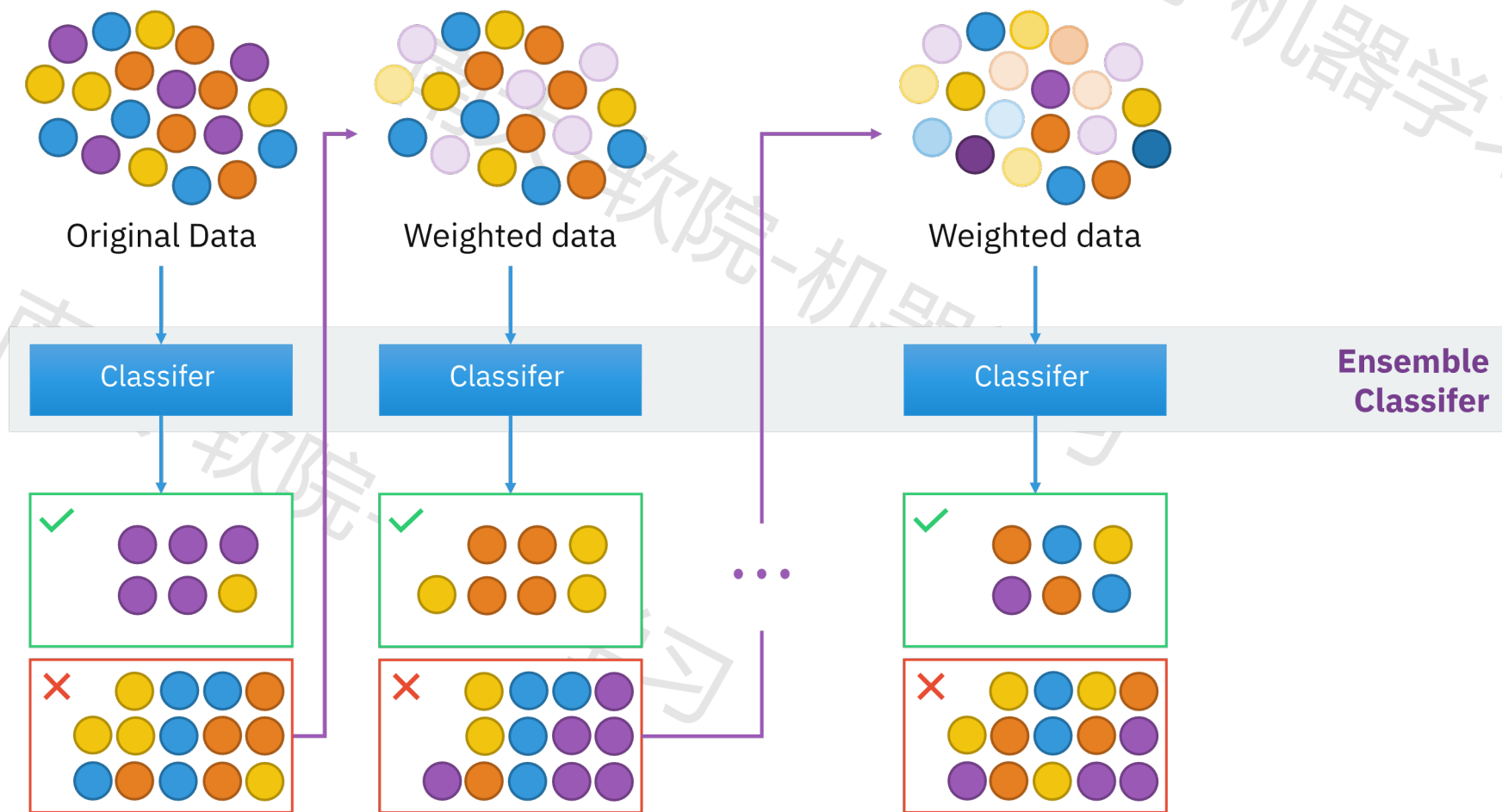
✓ 可能正确：以多大概率达到近似正确

□ 强学习器&弱学习器(Kearns&Valiant, 1984)

✓ 强学习器与弱学习器（略好于随机猜测）等价

✓ 可通过提升方法(Boosting)将弱学习器转为强学习器

Boosting集成学习框架



前向分步加法模型

□ 加法模型

$$f(x) = \sum_{m=1}^M \beta_m b(x; \gamma_m)$$

基函数（共M个）： $b(x; \gamma_m)$

基函数系数： β_m

前向分步加法模型

□ 加法模型的目标函数

$$\min_{(\beta_m, \gamma_m)} \sum_{i=1}^N L(y_i, \sum_{m=1}^M \beta_m b(x_i; \gamma_m))$$

优化复杂！

前向分步算法 (forward stagewise algorithm)：学习目标函数为加法模型，如果能够从前向后，每一步只学习一个基函数及其系数，逐步逼近要优化的总目标函数，就可以简化优化的复杂度。

前向分步加法模型

□ 前向分步算法

输入：训练数据集 $T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ ；损失函数 $L(Y, f(x))$ ；基函数集 $\{b(X; \gamma)\}$ ；

输出：加法模型 $f(x)$ 。

(1) 初始化 $f_0(x) = 0$

(2) 对 $m = 1, 2, \dots, M$

(a) 极小化损失函数： $(\beta_m, \gamma_m) = \operatorname{argmin}_{\beta, \gamma} \sum_{i=1}^N L(y_i, f_{m-1}(x_i) + \beta b(x_i; \gamma))$
得到参数 β_m, γ_m

(b) 更新： $f_m(x) = f_{m-1}(x) + \beta_m b(x; \gamma_m)$

(3) 得到加法模型： $f(x) = f_M(x) = \sum_{m=1}^M \beta_m b(x; \gamma_m)$

提升树（Boosting Tree）

- 提升（Boosting）方法主要采用加法模型，即基函数的线性组合，与前向分步算法。
- 以决策树为基函数的提升方法称为提升树（boosting tree）

$$f_M(x) = \sum_{m=1}^M h_m(x)$$

其中 $h_m(x)$ 表示决策树， M 为决策树的个数

提升树 (Boosting Tree)

□ 回归问题

□ 初始化 $f_0(x) = 0$

□ 对 $m = 1, 2, \dots, M$

✓ 计算 $\operatorname{argmin} \sum_{i=1}^N L(y_i, f_{m-1}(x_i) + h_m(x_i))$

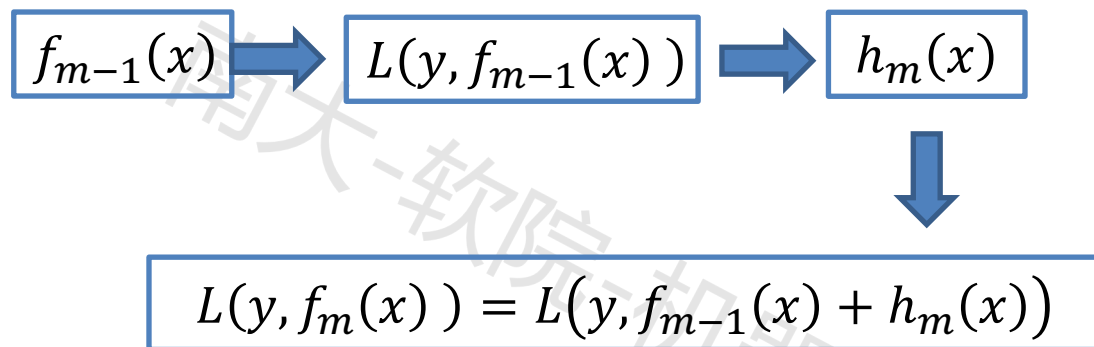
✓ 得到 $h_m(x)$

✓ 更新 $f_m(x) = f_{m-1}(x) + h_m(x)$

□ 得到回归问题提升树

✓ $f_M(x) = \sum_{m=1}^M h_m(x)$

残差



当采用平方误差损失函数

$$L(y, f_{m-1}(x) + h_m(x)) = (y - f_{m-1}(x) - h_m(x))^2 = (r - h_m(x))^2$$

推广到一般损失：第 m 轮第 i 个样本的负梯度

$$-\left[\frac{\partial L(y, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$$

提升树 (Boosting Tree)

- 回归问题

- 初始化 $f_0(x) = 0$

- 对 $m = 1, 2, \dots, M$

- ✓ 计算残差 $r_{m_i} = y_i - f_{m-1}(x), i = 1, 2, \dots, N$

- ✓ 拟合残差 r_{m_i} 学习一个回归树, 得到 $h_m(x)$

- ✓ 更新 $f_m(x) = f_{m-1}(x) + h_m(x)$

- 得到回归问题提升树

- ✓ $f_M(x) = \sum_{m=1}^M h_m(x)$

梯度提升树GBDT

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg \min_{\gamma} \sum_{i=1}^N L(y_i, \gamma)$.

2. For $m = 1$ to M :

(a) For $i = 1, 2, \dots, N$ compute

$$r_{im} = - \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f=f_{m-1}}.$$

(b) Fit a regression tree to the targets r_{im} giving terminal regions R_{jm} , $j = 1, 2, \dots, J_m$.

(c) For $j = 1, 2, \dots, J_m$ compute

$$\gamma_{jm} = \arg \min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma).$$

(d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.

3. Output $\hat{f}(x) = f_M(x)$.

梯度提升树GBDT

□ Gradient Boosting Decision Tree算法流程

- ✓ 初始化弱分类器

- ✓ 循环

 - ✓ 对每个样本计算负梯度

 - ✓ 构建新的样本集合

 - ✓ 根据对树的约束，构建CART树

 - ✓ 计算叶子区域最佳拟合值

 - ✓ 更新得到强学习器

- ✓ 得到最终的强学习器

Decision Tree, Boosting, Gradient Boosting

以CART为基学习器

GBDT例子

□输入样本

- ✓ <样本0: 年龄5岁, 体重20KG, 身高(预测标签值)1.1米>
- ✓ <样本1: 年龄7岁, 体重30KG, 身高(预测标签值)1.3米>
- ✓ <样本2: 年龄21岁, 体重70KG, 身高(预测标签值)1.7米>
- ✓ <样本3: 年龄30岁, 体重60KG, 身高(预测标签值)1.8米>

□预测样本

- ✓ <样本4: 年龄52岁, 体重65KG, 身高(预测标签值)? 米>

□学习参数

- ✓ 学习率0.1, 迭代次数5, 树深度=3

GBDT例子

□ 第一步：初始化弱分类器

$$\sum_{i=0}^N \frac{\partial \left(\frac{1}{2} (y_i - c)^2 \right)}{\partial c} = \sum_{i=0}^N (c - y_i)$$

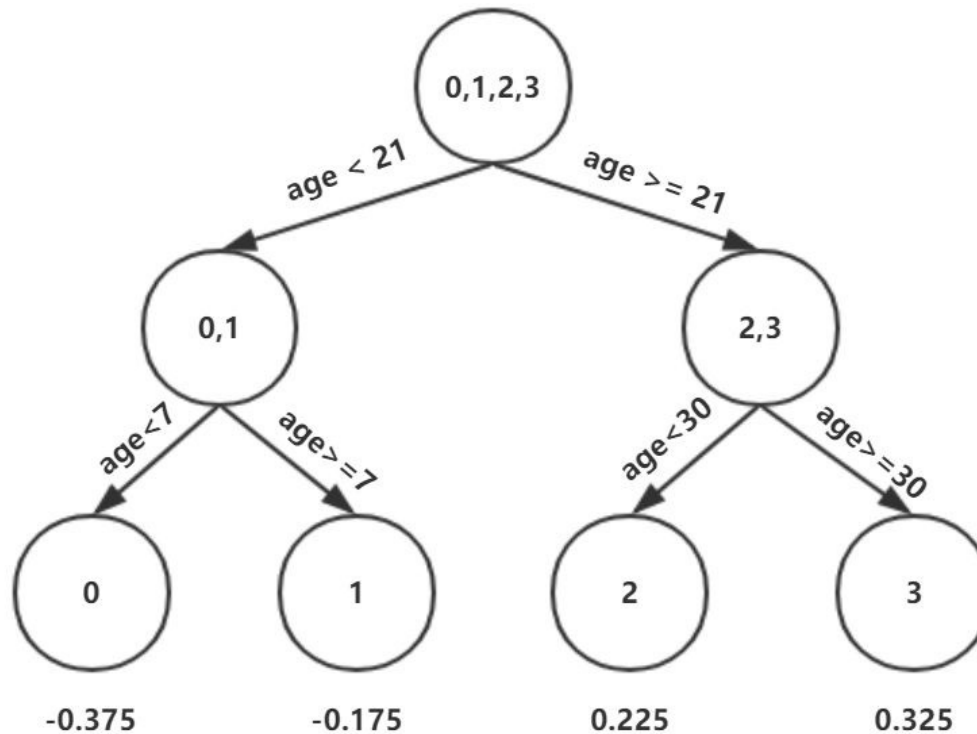
最小化，即是样本均值 $c = 1.475$

□ 第二步：根据残差构造新的样本集合

- ✓ <样本0'：年龄5岁，体重20KG，身高(预测标签值)-0.375米>
- ✓ <样本1'：年龄7岁，体重30KG，身高(预测标签值)-0.175米>
- ✓ <样本2'：年龄21岁，体重70KG，身高(预测标签值)0.225米>
- ✓ <样本3'：年龄30岁，体重60KG，身高(预测标签值)0.325米>

GBDT例子

□ 第三步：构建CART树



拟合残差

$$\underbrace{\gamma_{j1}}_{\gamma} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_i \in R_{j1}} L(y_i, f_0(x_i) + \gamma)$$

GBDT例子

□ 第四步：更新强学习器

$$f_1(x) = f_0(x) + \alpha \sum_{j=1}^4 \gamma_{j1} I(x \in R_{j1})$$

□ 最后一步：

$$f_5(x) = f_0(x) + \sum_{m=1}^5 \sum_{j=1}^4 \gamma_{jm} I(x \in R_{jm})$$

XGBoost

▣ Extreme Gradient Boosting(XGBoost)

- ✓ 是GBDT的一种高效实现
- ✓ 目标函数通过二阶泰勒展开式做近似
- ✓ 定义了树的复杂度，并应用到目标函数中
- ✓ 分裂节点处通过结构打分和分割损失动态生长
- ✓ 分裂节点的候选集合通过一种分布式Quantile Sketch得到
- ✓ 可以处理稀疏、缺失数据
- ✓ 可以通过特征的列采样防止过拟合

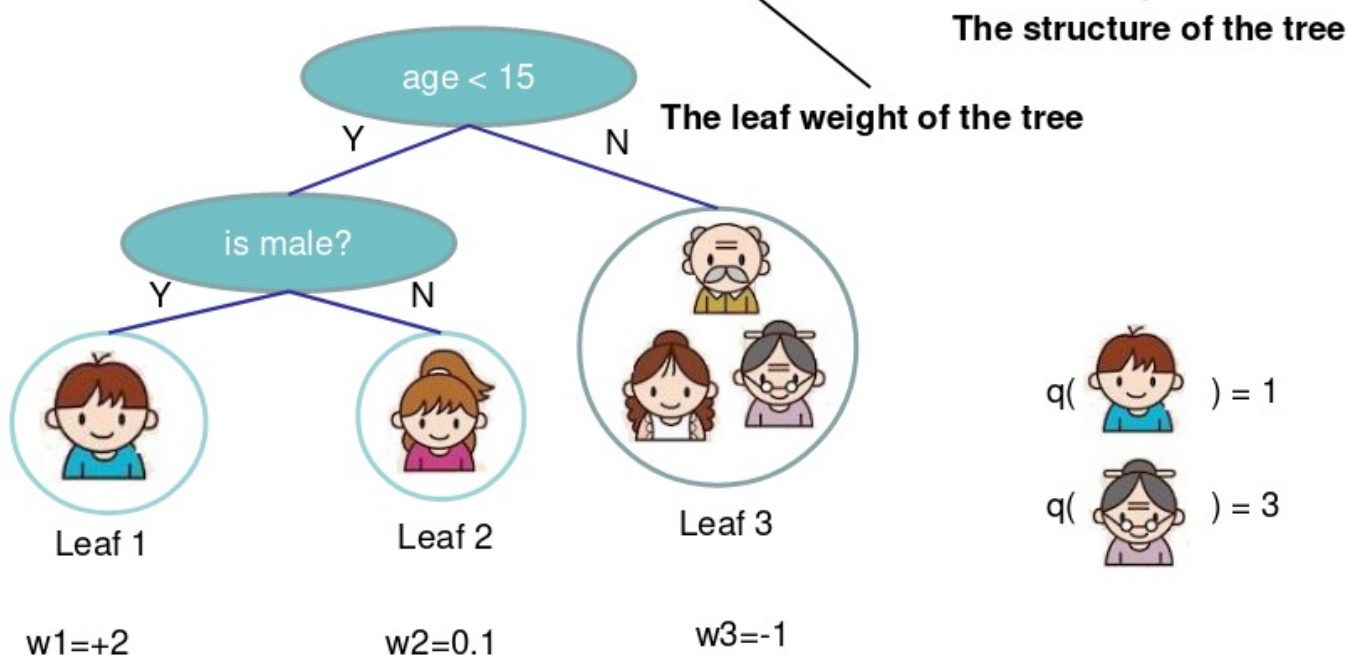
<https://xgboost.readthedocs.io/en/stable/tutorials/model.html>

树的复杂度

□ 优化树的定义

- $w \in \mathbb{R}^T$ 为叶子节点分数构成的向量，其中 T 为叶子的个数
- q 将每个数据样本点指定到相应的叶子节点

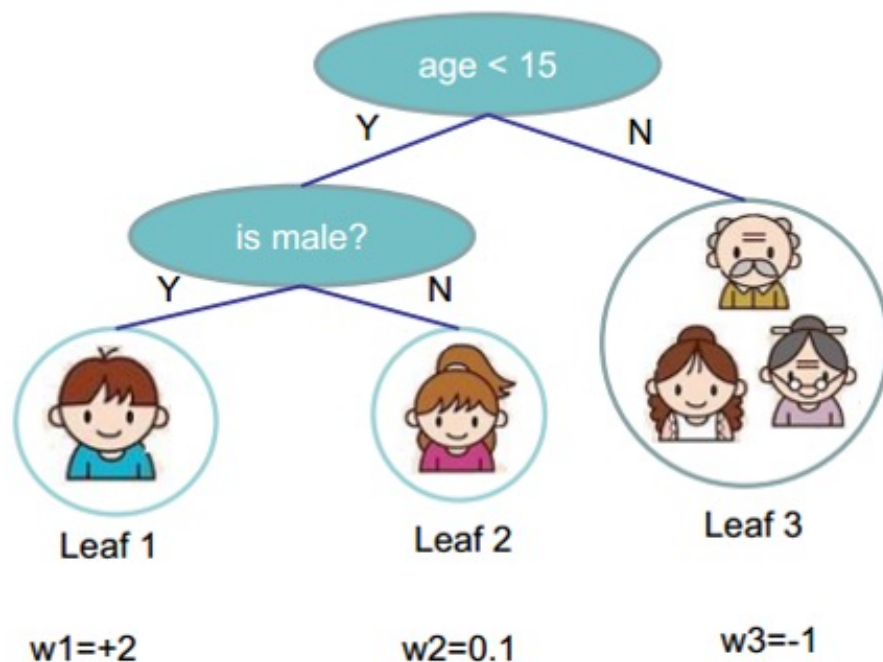
$$f_t(x) = w_{q(x)}, \quad w \in \mathbb{R}^T, q: \mathbb{R}^d \rightarrow \{1, 2, \dots, T\}$$



树的复杂度

$$\Omega(f_t) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2$$

Number of leaves L2 norm of leaf scores



$$\Omega = \gamma 3 + \frac{1}{2} \lambda (4 + 0.01 + 1)$$

目标函数

正则化项

$$Obj^{(t)} = \sum_{i=1}^n l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) + constant$$

泰勒一阶项、二阶项展开






$$Obj^{(t)} \simeq \sum_{i=1}^n \left[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) + constant$$

$$\begin{aligned} Obj^{(t)} &\simeq \sum_{i=1}^n \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \\ &= \sum_{i=1}^n \left[g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2 \right] + \gamma T + \lambda \frac{1}{2} \sum_{j=1}^T w_j^2 \\ &= \sum_{j=1}^T \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T \end{aligned}$$

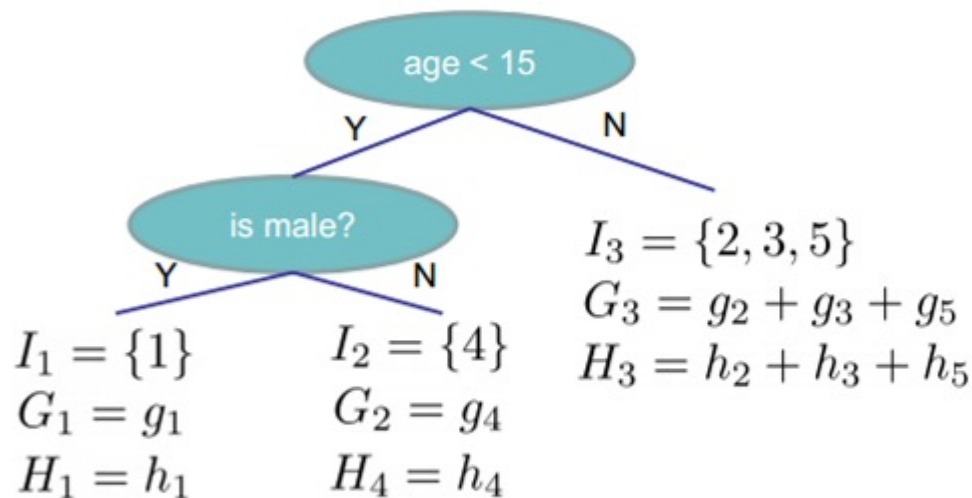
求导

$$w_j^* = -\frac{G_j}{H_j + \lambda} \quad Obj = -\frac{1}{2} \sum_{j=1}^T \frac{G_j^2}{H_j + \lambda} + \gamma T$$

树的打分和分裂

- 1  g1, h1
- 2  g2, h2
- 3  g3, h3
- 4  g4, h4
- 5  g5, h5

一阶、二阶梯度值



$$Obj = -\sum_j \frac{G_j^2}{H_j + \lambda} + 3\gamma$$

当前树的打分

节点分裂:

$$Gain = \underbrace{\frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} \right]}_{\text{左子树分数}} + \underbrace{\frac{G_R^2}{H_R + \lambda}}_{\text{右子树分数}} - \underbrace{\frac{(G_L + G_R)^2}{H_L + H_R + \lambda}}_{\text{不分割我们可以拿到的分数}} - \gamma$$

加入新叶子节点引入的复杂度代价

大纲

集成学习原理

Bagging和随机森林

Boosting和GBDT、XGboost

Adaboost

其他

Boosting

□ 概率近似正确(Probably approximately correct, PAC)学习理论

(Kearns&Valiant, 1984):

- ✓ **强可学习 (strongly learnable)**: 在PAC框架中, 一个概念 (类), 如果存在一个多项式的学习算法能够学习它, 并且正确率很高, 那么就称这个概念是强可学习的;
- ✓ **弱可学习 (weakly learnable)**: 在PAC框架中, 一个概念 (类), 如果存在一个多项式的学习算法能够学习它, 学习的正确率仅比随机猜测略好, 那么就称这个概念是弱可学习的。

Boosting

□ PAC学习理论(Kearns&Valiant, 1984):

- ✓ 强学习器与弱学习器是等价的 (Robert E. Schapire, 1990)
- ✓ 一个概念是强可学习的充分必要条件是这个概念是弱可学习的
- ✓ 可通过提升方法(Boosting)将弱学习器转为强学习器。

AdaBoost

□ Adaptive Boost

思想：从弱学习算法出发，通过改变训练数据的概率分布（权值分布），反复学习，得到一系列弱分类器，然后进行组合，构成一个强分类器。

策略：

- ✓ **权值分布：**提高那些被前一轮弱分类器错误分类样本的权值，降低那些被正确分类样本的权值；
- ✓ **弱分类器组合：**采用加权多数表决策略；增大分类误差率小的弱分类器的权重，减小分类误差率大的弱分类器的权重；

AdaBoost

□ Adaptive Boost

- ✓ 如何计算分类误差率？
- ✓ 如何更新样本权重？
- ✓ 如何得到弱学习器权重？
- ✓ 使用何种结合策略？

训练样本集合

$$T = \{(x_1, y_1), (x_1, y_2), \dots, (x_m, y_m)\}$$

第 k 个弱分类器的输出样本权重

$$D_k = (w_{k,1}, w_{k,2}, \dots, w_{k,m}) \quad w_{1,i} = \frac{1}{m} \quad i = 1, \dots, m$$

AdaBoost

□ 以分类问题为例：分类器的误差率和权重系数

第 k 个弱分类器 $G_k(x)$ 在训练集上的加权分类误差率为

$$e_k = \sum_{i=1}^m w_{k,i} I(G_k(x_i) \neq y_i)$$

第 k 个弱分类器 $G_k(x)$ 的权重系数为

$$\alpha_k = \frac{1}{2} \log \frac{1 - e_k}{e_k}$$

分类误差率越大，权重系数越小

AdaBoost

□ 以分类问题为例：样本的权重和集合策略

第 $k + 1$ 个弱分类器 $G_{k+1}(x)$ 的样本权重为

$$w_{k+1,i} = \frac{w_{k,i}}{Z_K} \exp(-\alpha_k y_i G_k(x_i))$$

规范化因子

更新训练数据集的权值分布

$$D_{k+1} = (w_{k+1,1}, w_{k+1,2}, \dots, w_{k+1,m})$$

$$Z_K = \sum_{i=1}^m w_{k,i} \exp(-\alpha_k y_i G_k(x_i))$$

AdaBoost

□ 以分类问题为例：样本的权重和集合策略

构建基分类器的线性组合

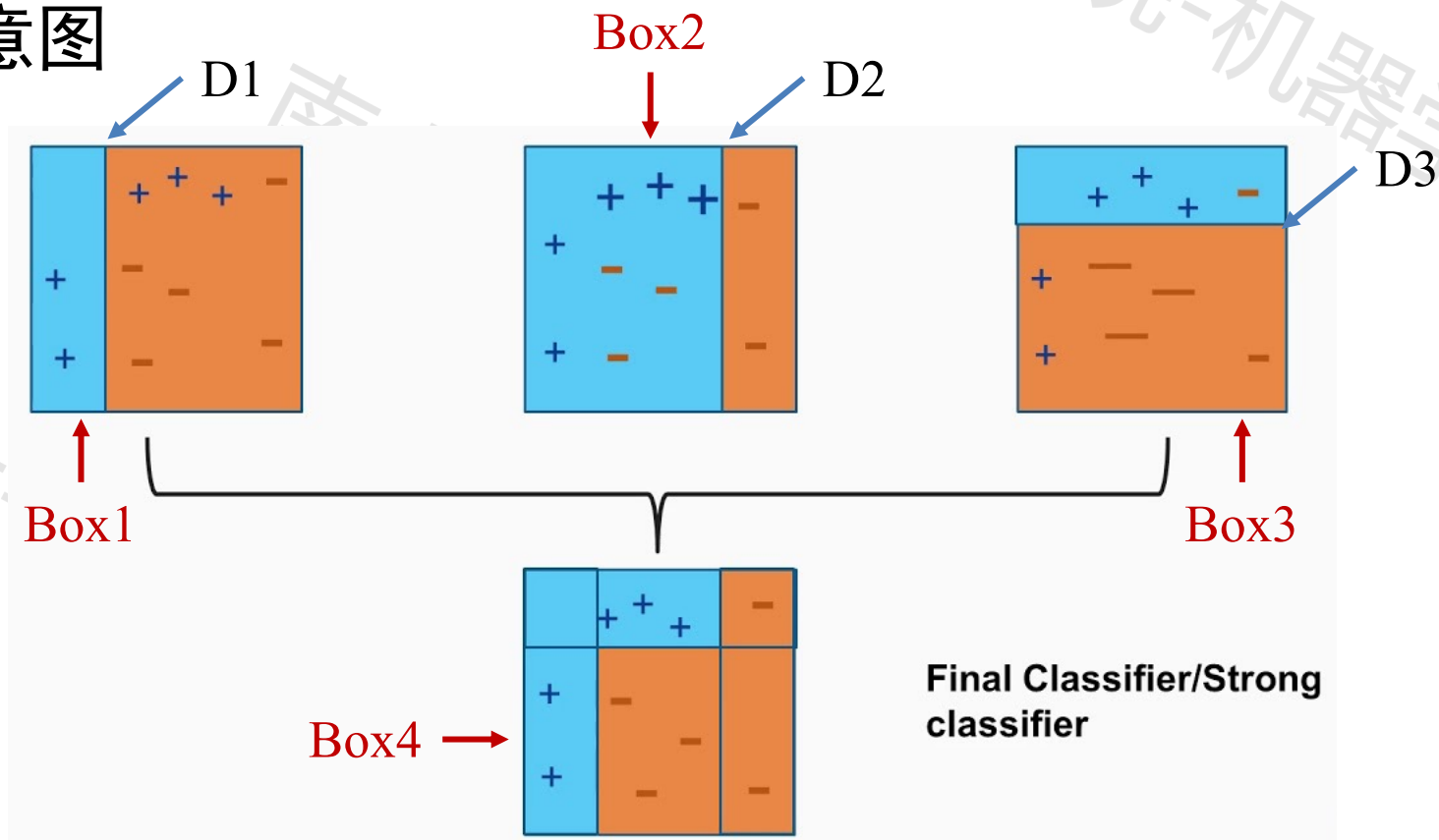
$$f(x) = \sum_{k=1}^K \alpha_k G_k(x_i)$$

Adaboost采用加权表决

$$f(x_i) = \text{sign} \left(\sum_{k=1}^K \alpha_k G_k(x_i) \right)$$

AdaBoost

示意图



Box1~Box3为按顺序执行的3个Learner，D1~D3分别为各个Box的决策边界，“+ -”表示正负样本，Box4综合了基学习器的决策边界

大纲

集成学习原理

Bagging和随机森林

Boosting和GBDT、XGboost

Adaboost

其他

其他

□学习法：Stacking算法

Stacking先从初始数据集训练出初始学习器，然后“生成”一个新数据集用于训练次级学习器（元学习器）。在这个新的数据集中，初始学习器的输出被当作样例输入特征，而初始样本的标记仍被当作样例标记。

其他

□学习法：Stacking算法原理

- ✓ 对于Model_1，将训练集D分为k份，对于每一份，用剩余数据集训练模型，然后预测出这一份的结果
 - ✓ 重复上面步骤，直到每一份都预测出来。得到次级模型的训练集
 - ✓ 得到k份测试集，平均后得到次级模型的测试集
-
- ✓ 对于Model_2、Model_3.....重复以上情况，得到M维数据
 - ✓ 选定次级模型，进行训练预测，一般这最后一层用Logistic Regression

Stacking

□ 算法流程

| Algorithm | Stacking |
|-----------|----------|
|-----------|----------|

1: Input: training data $D = \{x_i, y_i\}_{i=1}^m$

2: Output: ensemble classifier H

3: *Step 1: learn base-level classifiers*

4: for $t = 1$ to T do

5: learn h_t based on D

6: end for

7: *Step 2: construct new data set of predictions*

8: for $i = 1$ to m do

9: $D_h = \{x'_i, y_i\}$, where $x'_i = \{h_1(x_i), \dots, h_T(x_i)\}$

10: end for

11: *Step 3: learn a meta-classifier*

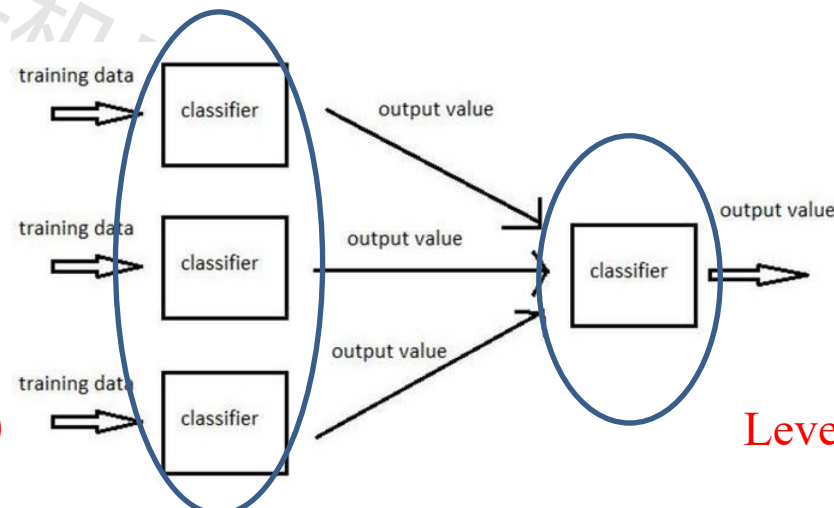
12: learn H based on D_h

13: return H

Stacking

Stacking或者Stacked Generalization堆栈泛化模型的特点

- ✓ 集成学习中异源集成（Heterogenous ensembles）的典型代表
- ✓ 通过使用第一阶段(level 0)的预测作为下一层预测的特征，比起相互独立的预测模型能够有更强的非线性表述能力，降低泛化误差。
- ✓ 它的目标是同时降低机器学习模型的Bias-Variance。



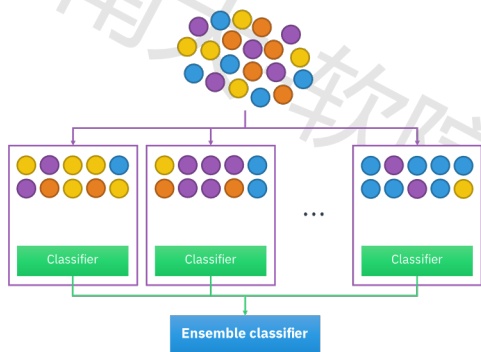
Stacking中的元学习器

- 统计方法类：投票、平均
- 易解释的学习方法类：LR, C45/CART
- 非线性学习算法类：XGBoost, NN, K-NN, RF
- 加权一次/二次线性模型类：Weighted Linear/Quadratic
- 其他复杂类：在线学习、深度学习、群体智能

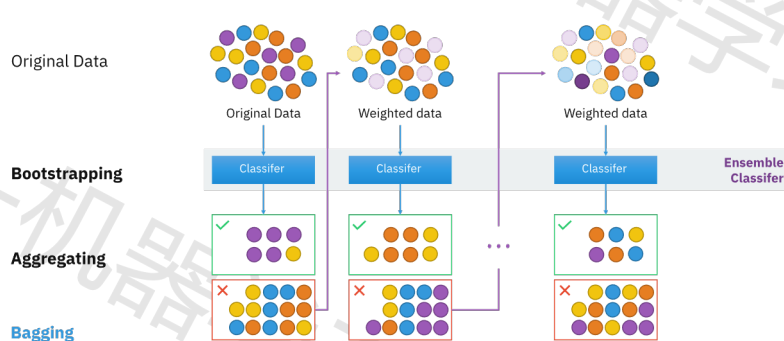
小结

三种集成方法的对比

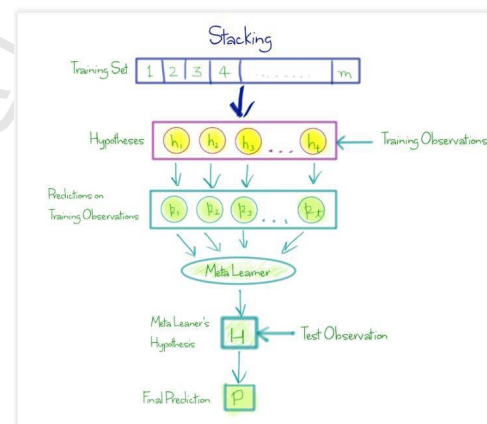
Bagging



Boosting



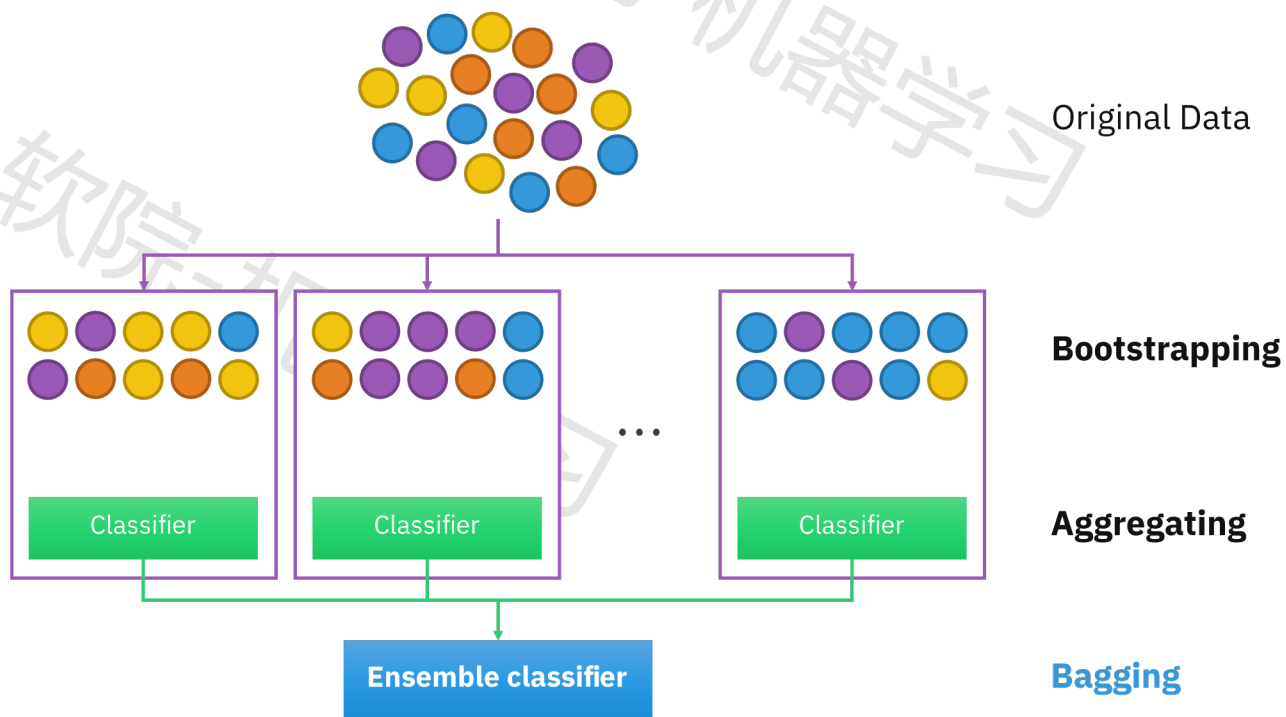
Stacking



小结

三种集成方法的对比

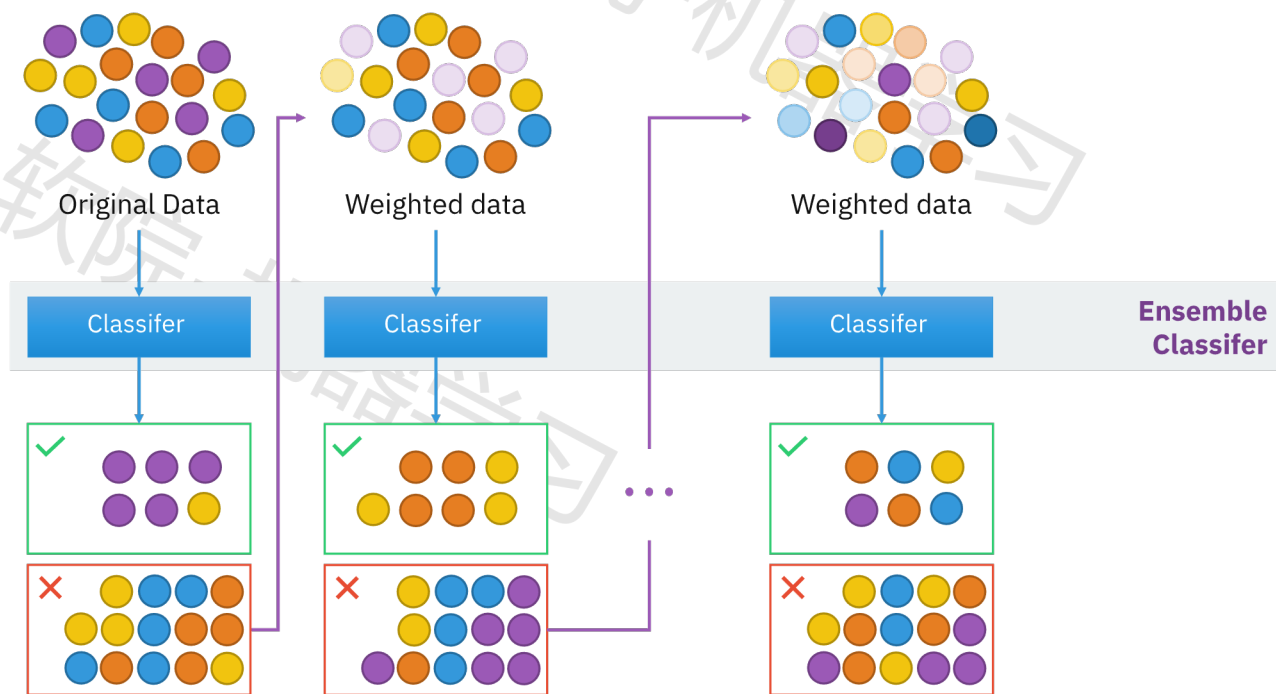
Bagging



小结

三种集成方法的对比

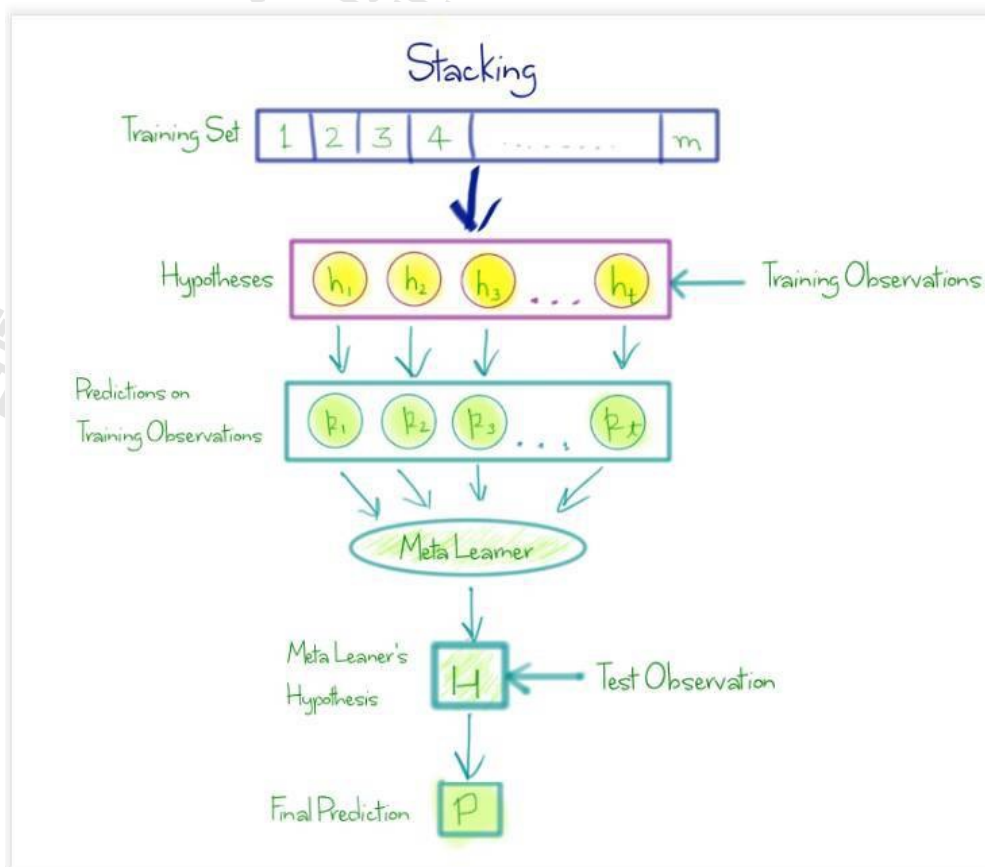
Boosting



小结

三种集成方法的对比

Stacking



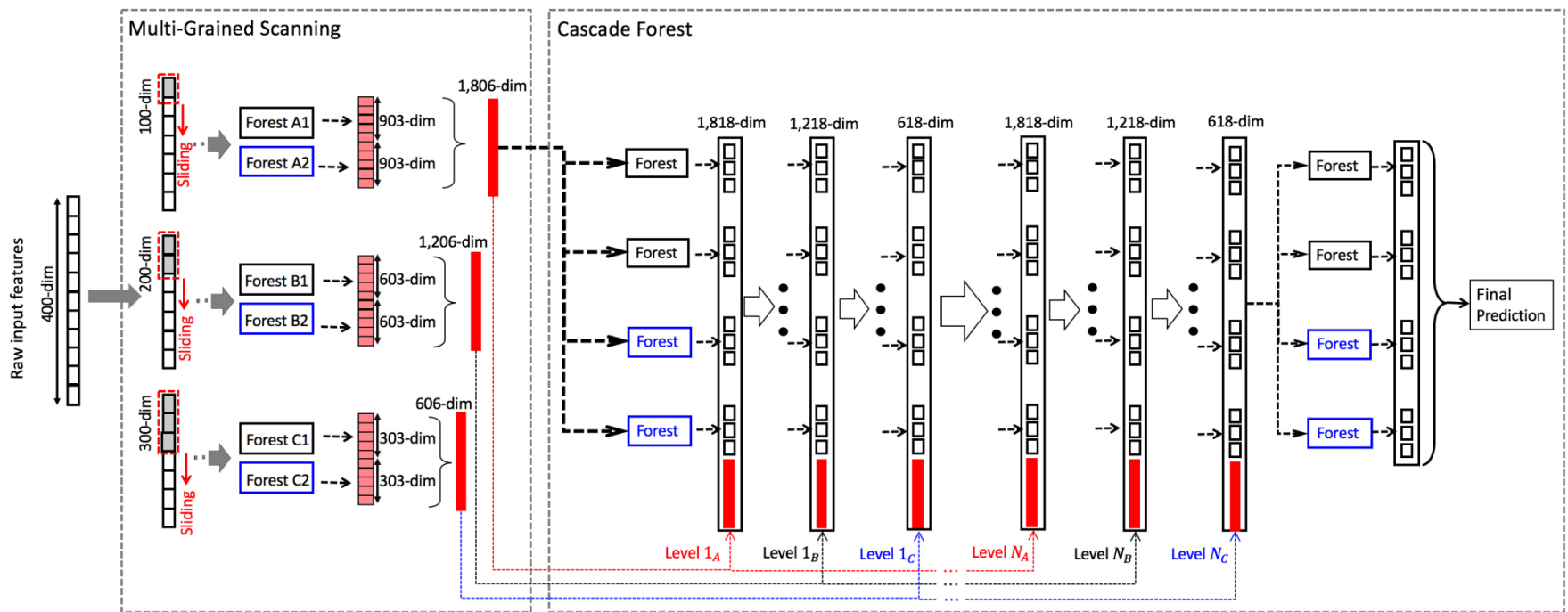
小结

三种集成方法的对比

| | Target | Data | Classifier | Aggregation |
|----------|------------------------|---|--|---------------------------|
| Bagging | mainly variance | replacement resampling (bootstrap) based randomization (parallel) | homogeneous uncorrelated (sub)-learner | voting for classification |
| | | | | averaging for regression |
| Boosting | mainly bias | stepwise misclassification based decorrelation (sequential) | homogeneous weak learner | weighted majority voting |
| Stacking | both bias and variance | various | heterogeneous strong learner | meta-learner |

集成学习的延伸

- 深度学习时代的随机森林？



Deep Forest: Towards an Alternative to Deep Neural Networks. IJCAI 2017

扩展阅读

Zhi-Hua Zhou. [Ensemble Methods: Foundations and Algorithms](#), 2012.

XGBoost: <https://xgboost.readthedocs.io/en/stable/tutorials/model.html>

周志华: Boosting学习理论的探索 —— 一个跨越30年的故事:

<https://mp.weixin.qq.com/s/Jnh7yIOmzbTvWk77zh2-lA>

谢谢！