第八章　集成学习

Castor Ye

1　个体与集成

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-classfier system）、基于委员会的学习（committee-based learning）等。

图 1 显示出集成学习的一般结构：先产生一组“个体学习器”（in 地 vi learner），再用某种策略将它们结合起来。

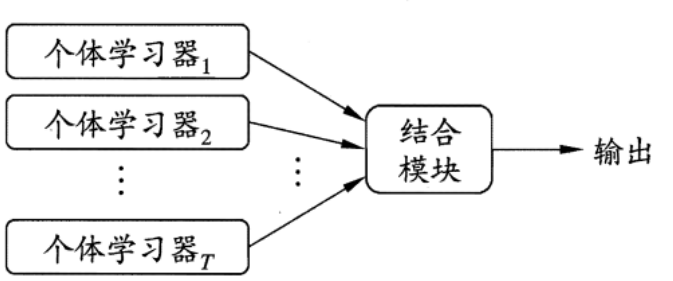
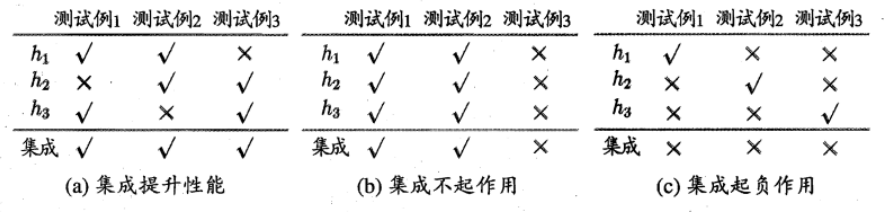


图 1: 集成学习示意图

在集成模型中，若个体学习器都属于同一类别，例如都是决策树或神经网络，则称该集成为“同质”的（homogenneous）。同质集成中的个体学习器亦称“基学习器”（base learner），相应的学习算法称为“基学习算法”（base learning algorithm）。若个体学习器为不同类型的，则称该集成为“异质”的（heterogenous）。异质集成中的个体学习器由不同的学习算法生成，此时不再有基学习器，而称“组件学习器”（component learner）或直接称“个体学习器”。

上面我们已经提到要让集成起来的泛化性能比单个学习器好，但也存在短板效应，所以我们引入两个重要概念：准确性和多样性（diversity）。准确性指的是个体学习器不能太差，要有一定的准确度；多样性则是个体学习器之间的输出要具有差异性。



	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 2: 集成个体应“好而不同”

我们来做个简单的分析，考虑二分类问题 $y \in \{-1, +1\}$ 和真实函数 f，假定基分类器的错误率为 ϵ，即对每个基分类器 h_i 有：

$$P(h_i(x) \neq f(x)) = \epsilon$$

假设集成通过简单投票法结合 T 个基分类器，若有超过半数的基分类器正确，则集成分类就正确：

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

假设基分类器的错误率相互独立，则由 Hoeffding 不等式可知，集成的错误率为：

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

事实上，个体学习器的“准确性”和“多样性”本身存在冲突，如何产生并结合“好而不同”的个体学习器，就是集成学习的核心。

根据个体学习器的生成方式，目前的集成学习方法大致可以分为两大类：

i. Boosting：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。

ii. Bagging 与随机森林（Random Forest）：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。

2 Boosting

Boosting 是一族可将弱学习器提升为强学习器的算法，这族算法的工作机制类似：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养你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，直到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 族算法最著名的代表是 AdaBoost。AdaBoost 使用的是**指数损失函数**，因此 AdaBoost 的权值与样本分布的更新都是围绕着最小化指数损失函数进行的。

定义基学习器的集成为加权结合，则有：

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

AdaBoost 算法的指数损失函数定义为：

$$loss_{\exp}(h) = \mathbb{E}_{x \sim D}[e^{-f(x)H(x)}]$$

具体来说，AdaBoost 算法分 3 步：

i. 初始化训练数据的权值分布，如果有 m 个样本，则每个训练样本最开始时都被赋予相同的权值：$\frac{1}{m}$。

ii. 训练弱分类器，具体训练过程中，如果某个样本点已经被准确地分类，那么在构造下一个训练集中，它的权值就被降低；相反，如果某个样本点没有被准确分类，那么它的权值就得到提高。然后，权值更新过的样本集被用于训练下一个分类器，整个训练过程如此迭代地进行下去。

iii. 将各个训练得到的弱分类器组合成强分类器，各个弱分类器的训练过程结束后，加大分类误差率小的弱分类器的权重，使其在最终的分类函数中起着较大的决定作用，而降低分类误差率大的弱分类器的权重，使其在最终的分类函数中起着较小的决定作用。

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

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

图 3: AdaBoost 算法

可以看出：AdaBoost 的核心步骤就是计算基学习器权重和样本权重分布，那为何是上述的计算公式呢？这就涉及到了我们之前为什么说大部分带参机器学习算法只是改变了损失函数，就是因为大部分模型的参数都是通过最优化损失函数（可能还加个规则项）而计算（梯度下降，坐标下降等）得到，这里正是通过最优化指数损失函数从而得到这两个参数的计算公式，具体的推导过程此处不进行展开。

Boosting 算法要求基学习器能对特定分布的数据进行学习，即每次都更新样本分布权重，这里书上提到了两种方法：“重赋权法”（re-weighting）和“重采样法”（re-sampling），书上的解释有些晦涩，这里进行展开一下：

i. 重赋权法：对每个样本附加一个权重，这时涉及到样本属性与标签的计算，都需要乘上一个权值。

ii. 重采样法：对于一些无法接受带权样本的基学习算法，在每一轮学习中，根据样本分布对训练集重新进行采样，再用重采样得到的样本对基学习器进行训练。大致过程为：根据各个样本的权重，对训练数据进行重采样，初始时样本权重一样，每个样本被采样到的概率一致，每次从 m 个原始的训练样本中按照权重有放回采样 m 个样本作为训练集，然后计算训练集错误率，然后调整权重，重复采样，集成多个基学习器。

从偏差-方差分解来看：Boosting 算法主要关注于降低偏差，每轮的迭代都关注于训练过程中预测错误的样本，将弱学习提升为强学习器。从 AdaBoost 的

算法流程来看，标准的 AdaBoost 只适用于二分类问题。在此，当选为数据挖掘十大算法之一的 AdaBoost 介绍到这里，能够当选正是说明这个算法十分婀娜多姿，背后的数学证明和推导充分证明了这一点，限于篇幅不再继续展开。

3 Bagging 与随机森林

上面已经提到产生“好而不同”的个体学习器是集成学习研究的核心，即在保证基学习器准确性的同时增加基学习器之间的多样性。而这两种算法的基本思（tao）想（lu）都是通过“自助采样”的方法来增加多样性。

3.1 Bagging

Bagging 是并行式集成学习方法，即基学习器的训练之间没有前后顺序，可以同时进行。Bagging 使用“有放回”采样的方式选取训练集，对于包含 m 个样本的训练集，进行 m 次有放回的随即采样操作，从而得到 m 个样本的采样集，这样训练集中有接近 36.8% 的样本没有被采到。按照相同的方式重复进行，我们就可以采集到 T 个包含 m 个样本的数据集，从而训练出 T 个基学习器，最终对这 T 个基学习器的输出进行结合。

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

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

图 4: Bagging 算法

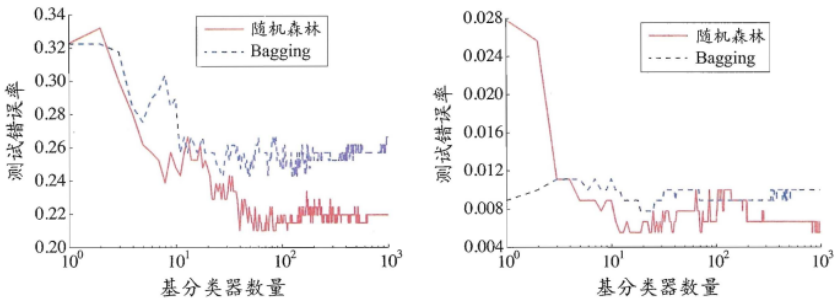
可以看出 Bagging 主要通过样本的扰动来增加基学习器之间的多样性，因此 Bagging 的基学习器应为那些对训练集十分敏感的不稳定学习算法，例如：神经网络与决策树等。从偏差-方差分解来看，Bagging 算法主要关注于降低方差，即通过多次重复训练提高稳定性。不同于 AdaBoost 的是，Bagging 可以十分简单地

移植到多分类、回归等问题。总的说起来则是：AdaBoost 关注于降低偏差，而 Bagging 关注于降低方差。

3.2 随机森林

随机森林（Random Forest）是 Bagging 的一个拓展体，它的基学习器固定为决策树，多棵树也就组成了森林，而“随机”则在于选择划分属性的随机，随机森林在训练基学习器时，也采用有放回采样的方式添加样本扰动，同时它还引入了一种属性扰动，即在基决策树的训练过程中，在选择划分属性时，RF 先从候选属性集中随机挑选出一个包含 K 个属性的子集，再从这个子集中选择最优划分属性，一般推荐 $K = \log_2 d$。

这样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，从而进一步提升了基学习器之间的差异度。相比决策树的 Bagging 集成，随机森林的起始性能较差（由于属性扰动，基决策树的准确度有所下降），但随着基学习器数目的增多，随机森林往往会收敛到更低的泛化误差。同时不同于 Bagging 中决策树从所有属性集中选择最优划分属性，随机森林只在属性集的一个子集中选择划分属性，因此训练效率更高。



(a) glass 数据集　　(b) auto-mpg 数据集

图 5: 在两个 UCI 数据上，集成规模对随机森林与 Bagging 的影响

4 结合策略

结合策略指的是在训练好基学习器后，如何将这些基学习器的输出结合起来产生集成模型的最终输出。

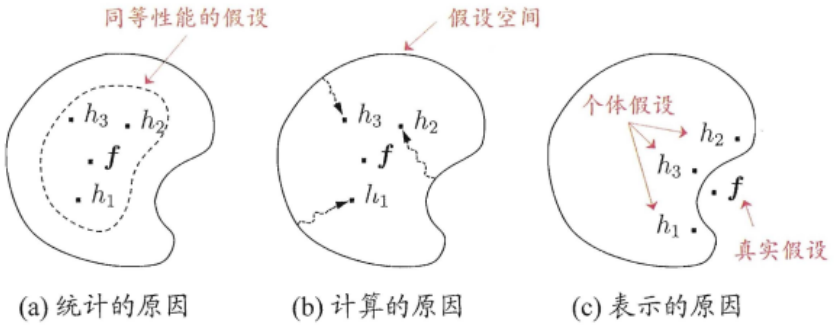


图 6: 学习器结合可能从三个方面带来好处

假定集成包含 T 个基学习器 $\{h_1, h_2, \cdots, h_T\}$，其中 h_i 在示例 x 上的输出为 $h_i(x)$。

4.1 平均法（回归问题）

对数值型输出 $h_i(x) \in \mathbb{R}$，最常见的结合策略是使用平均法（averaging）。

简单平均法（simple averaging）：

$$H(x) = \frac{1}{T} \sum_{i=1}^{T} h_i(x)$$

加权平均法（weighted averaging）：

$$H(x) = \sum_{i=1}^{T} w_i h_i(x)$$

其中 w_i 是个体学习器 h_i 的权重，通常要求 $w_i \geq 0, \sum_{i=1}^{T} w_i = 1$。

易知简单平均法是加权平均法的一种特例，加权平均法可以认为是集成学习研究的基本出发点。由于各个基学习器的权值在训练中得出，一般而言，在个体学习器性能相差较大时宜使用加权平均法，在个体学习器性能相差较小时宜使用简单平均法。

4.2 投票法（分类问题）