第10章　集成学习

刘家锋

哈尔滨工业大学

第10章　集成学习

10.1 分类器性能评价

性能评价指标

- **分类错误率**
  - 分类器在样本集$D = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\}$上的错误率：

$$E(g; D) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(g(\mathbf{x}_i) \neq y_i)$$

其中，$\mathbb{I}(\alpha)$为指示函数：

$$\mathbb{I}(\alpha) = \begin{cases} 1, & \alpha = \text{true} \\ 0, & \alpha = \text{false} \end{cases}$$

性能评价指标

- **查准率和查全率**
  - 检索任务可以看作是两分类问题，相关内容称为正例，无关内容称为反例；
  - 分类的结果可以用混淆矩阵表示：
    - 真正例(TP)：真实的正例被分类为正例的数量；
    - 假反例(FN)：真实的正例被分类为反例的数量；
    - 假正例(FP)：真实的反例被分类为正例的数量；
    - 真反例(TN)：真实的反例被分类为反例的数量；

真实类别	预测结果	
	正例	反例
正例	TP(真正例)	FN(假反例)
反例	FP(假正例)	TN(真反例)

性能评价指标

- **检索任务的性能评价指标**
  - 查准率：检索出来的结果中，真实正例所占的比例

$$P = \frac{TP}{TP + FP}$$

  - 查全率：所有真实正例中，被检索出来的比例，也被称为召回率

$$R = \frac{TP}{TP + FN}$$

  - 查全率和查准率是相互矛盾的，一般与检出的总数有关：
    - 检索结果数量多，查全率高，查准率低；
    - 检索结果数量少，查全率低，查准率低；

性能评价指标

- **检索任务的性能指标**
  - F_1度量：综合考虑查全率和查准率，取两者的几何平均

$$F_1 = \frac{2 \times P \times R}{P + R} = \frac{2 \times TP}{n + TP - TN}$$

  - F_β度量：通过参数β调节对查全率和查准率的关心程度

$$F_\beta = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) \times R}$$

$\beta > 1$更关心查全率，$\beta < 1$更关心查准率；

性能评价的方法

- **训练集和测试集**
  - 评价最好的方法是使用训练集S学习模型，使用另外的测试集T评估模型的性能；
  - 数据集D既用于训练，又用于测试，需要适当的划分：

$$D = S \cup T, \qquad S \cap T = \Phi$$

- **常用的方法**
  - 留出法(hold-out)；
  - 交叉验证法(cross validation)
  - 自助法(bootstrap)

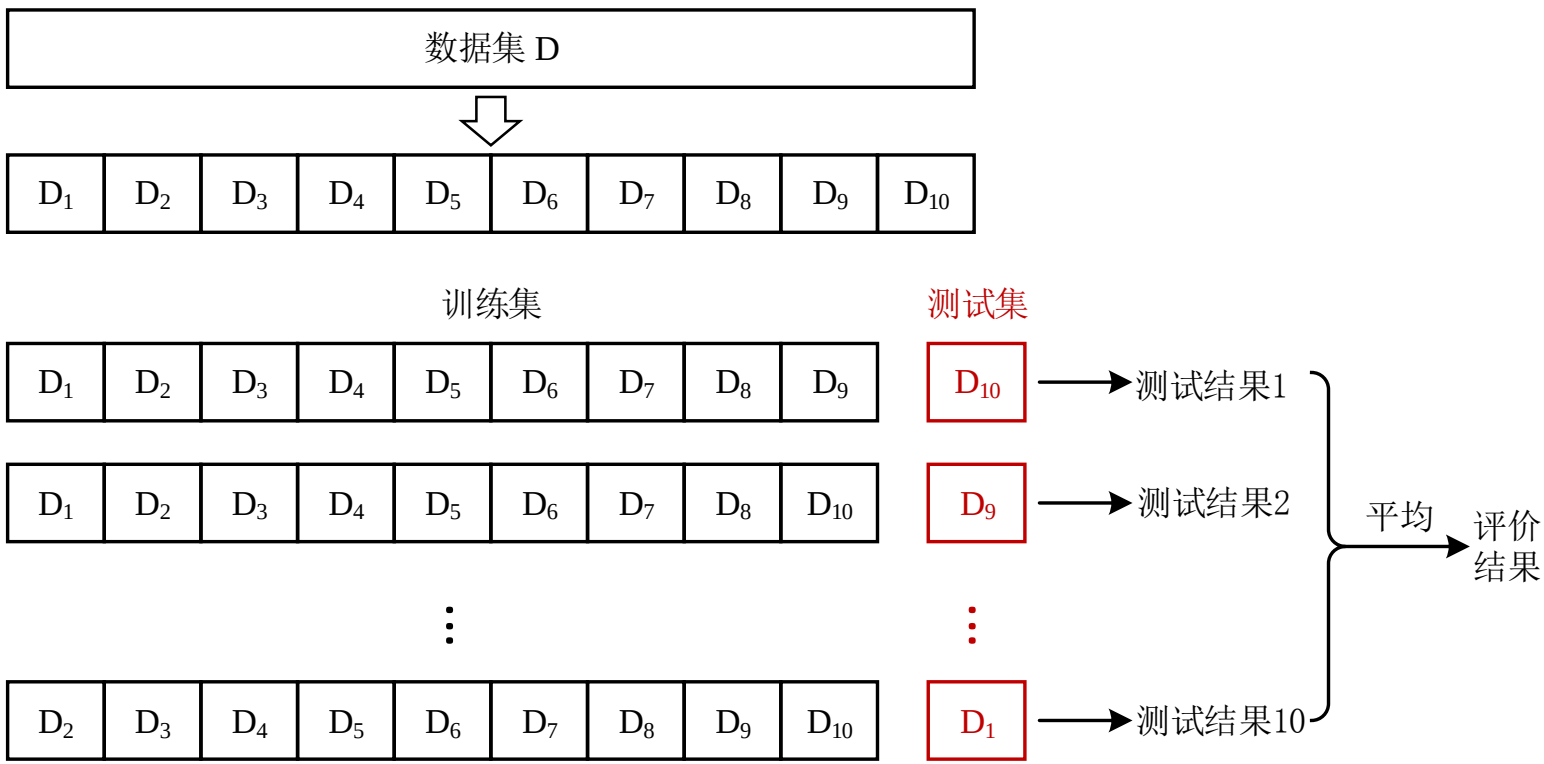
留出法

- **数据集的划分原则**
  - 保持数据分布一致性，不同类别样本的比例应该是一致的；
  - 测试集不宜太大或太小，一般选择样本总数的20% ~ 30%；

- **评价过程**
  - 按比例将数据集D随机划分为S和T；
  - 训练集S学习模型，T测试模型的性能；
  - 重复划分和测试若干次，计算平均性能；

交叉验证法

- **k-折交叉验证法**
  - 数据集D随机划分为k个子集；
  - $k-1$个子集用于训练，1个用于测试；



自助法(Bootstrap)

- **自助法的过程**
  - 从数据集D中有放回地随机抽样n个样本，构成训练集S；
  - 从数据集D中有放回地随机抽样n个样本，构成测试集T；
  - 重复若干次，计算平均的评估结果；
- **自助法的优点**
  - 数据集D中约有30%的样本没有出现在训练集S中：

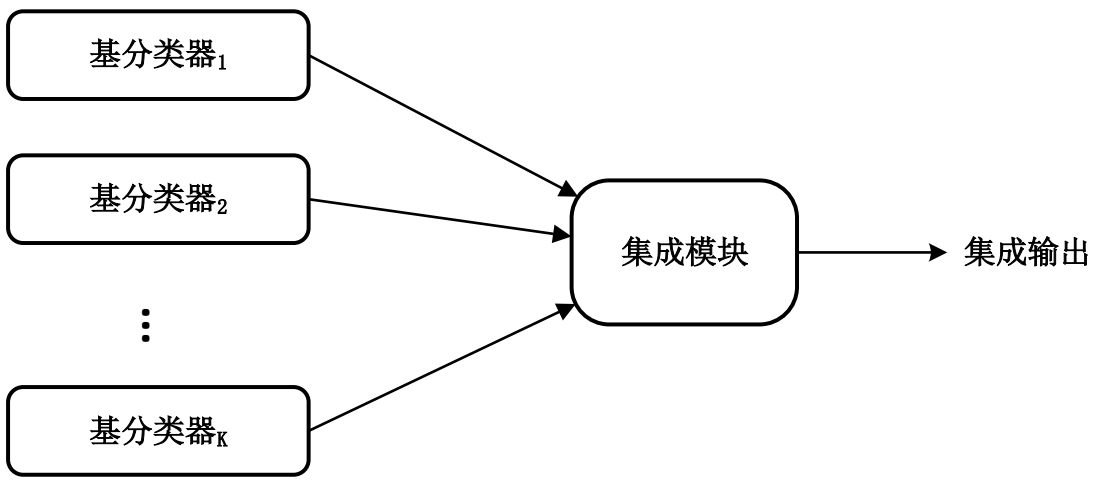
$$P(\mathbf{x} \notin S) \approx \lim_{n\to\infty} \left(1 - \frac{1}{n}\right)^n = \frac{1}{e} \approx 0.368, \quad \forall \mathbf{x} \in D$$

  - S和T的样本数与D相同，测试可以进行任意多次，评估结果更接近最终模型的性能；
  - 自助法更适用于样本数n较小时，样本数多时留出法和交叉验证更常用；

10.2 分类器集成

多分类器集成

- 个体与集成
  - 个体分类器：由一个现有的学习算法从训练数据产生，也称为“基分类器”
  - 集成学习：组合多个个体分类器，取得比个体分类器更好的性能，也称为“多分类器系统”等



集成分类器的性能

- **集成学习如何能够获得更好的性能？**
  - 要求基分类器有一定的“准确性”和“多样性”；
  - 例如：集成三个不同的分类器g_1, g_2, g_3，在三个不同测试样本$\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$上的分类结果

	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
g_1	✓	✓	×
g_2	×	✓	✓
g_3	✓	×	✓
集成	✓	✓	✓

(a) 性能提升

	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
g_1	✓	✓	×
g_2	✓	✓	×
g_3	✓	✓	×
集成	✓	✓	×

(b) 性能不变

	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$
g_1	✓	×	×
g_2	×	✓	×
g_3	×	×	✓
集成	×	×	×

(c) 性能下降

基分类器的生成

- **多样性学习算法**
  - 相同的训练样本集；
  - 不同的学习算法，包括学习算法的不同超参数设置；
- **多样性训练集**
  - 相同的学习算法，不同的训练样本集；
  - 随机划分训练集，或者Bootstrap抽样训练集；

10.3 Bagging

Bagging算法

Algorithm 1 Bagging算法

Input: 训练集$D = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\}$，基分类器学习算法$\mathcal{L}$，基分类器数量$K$

Output: 集成分类器$G(\mathbf{x})$

procedure BAGGING
  for $k = 1, \cdots, K$ **do**
    $D_{bs} = bootstrap(D)$ ▷ bootstrap抽样
    $g_k = \mathcal{L}(D_{bs})$ ▷ 训练基学习器
  end for
  return $G(\mathbf{x}) = \arg\max_{y \in \mathcal{Y}} \sum_{k=1}^{K} \mathbb{I}(g_k(\mathbf{x}) = y)$ ▷ 集成预测
end procedure

随机森林

- **随机森林(Random Forrest)**
  - 以决策树作为基分类器，采用bagging算法集成学习；
  - 为了增加基分类器的多样性，除了训练集上的随机性之外，增加了划分特征的随机性；
  - 对于每个节点，先从特征集合中随机选择包含k个元素的子集，在子集中选择最优特征用于划分；
  - 决策树学习过程中，一般不做剪枝处理；

10.4 Boosting

序列化集成学习方法

- **Boosting**
  - Bagging算法属于并行化的集成学习方法，基分类器之间互无关联；
  - Boosting算法是序列化的集成学习，下一个基分类器的学习需要根据之前的学习结果来调整；
  - 这其中最有代表性的是AdaBoost算法；
- **AdaBoost集成分类器**
  - AdaBoost一般用于二分类问题，类别标记$y_i \in \{-1, +1\}$；
  - 学习K个基分类器$\{h_t(\mathbf{x})\}$，加权线性组合：

$$H(\mathbf{x}) = \sum_{k=1}^{K} \alpha_k h_k(\mathbf{x})$$

  - 基分类器的权重α_k是算法学习得到的参数；

AdaBoost算法

- **样本集的加权重采样**
  - AdaBoost算法中基分类器的训练集也是由样本集D重采样得到的；
  - 样本是按照不断变化的分布，而不是均匀分布来抽样的；
  - 简单地说，为每个样本赋予一个权重w_i，以w_i为概率决定是否抽样相应样本；
  - 学习得到一个新的基分类器$h_k(\mathbf{x})$之后，对权重调整，被正确分类的样本权重降低，错误分类的权重增加：

$$w_i \leftarrow \frac{w_i}{z_k} \times \begin{cases} e^{-\alpha_k}, & h_k\text{正确分类}\mathbf{x}_i \\ e^{\alpha_k}, & h_k\text{错误分类}\mathbf{x}_i \end{cases}$$

其中，z_k为归一化因子，保证$\sum_{i=1}^{n} w_i = 1$；

AdaBoost算法

- **基分类器的权重**
  - 基分类器$h_k(\mathbf{x})$的权重α_k体现分类性能，与错误率ϵ_k有关；
  - 错误率ϵ_k同样是在一个加权重采样样本集上统计的，而不是初始训练集D；
  - 令依据权重$\{w_i\}$抽样的数据集为D_ϵ，错误率为：

$$\epsilon_k = \frac{1}{n} \sum_{\mathbf{x} \in D_\epsilon} \mathbb{I}(h_k(\mathbf{x}) \neq y)$$

  - 基分类器$h_k(\mathbf{x})$的权重：

$$\alpha_k = \frac{1}{2} \ln \left(\frac{1 - \epsilon_k}{\epsilon_k} \right)$$

AdaBoost算法

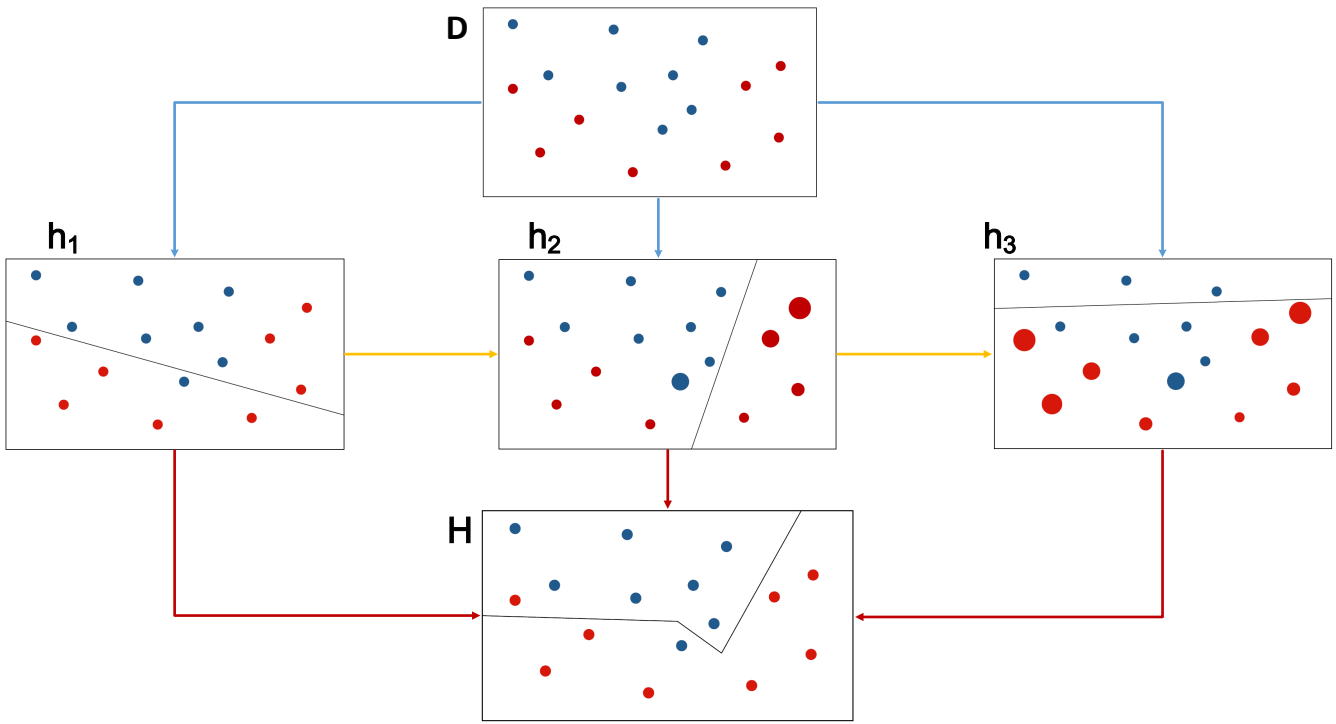
Algorithm 2 AdaBoost算法

Input: 训练集$D = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\}$，基分类算法$\mathcal{L}$，基分类器数量$K$
Output: 集成分类器$H(\mathbf{x})$

procedure AdaBoost
  $w_i = 1/n, i = 1, \cdots, n$ ▷ 初始化样本权重
  for $k = 1, \cdots, K$ **do**
    $D_w = bootstrap(D, \{w_i\})$ ▷ 加权抽样
    $D_\epsilon = bootstrap(D, \{w_i\})$
    $h_k = \mathcal{L}(D_w)$ ▷ 训练基分类器
    $\epsilon_k = \frac{1}{n} \sum_{\mathbf{x} \in D_\epsilon} \mathbb{I}(h_k(\mathbf{x}) \neq y)$ ▷ 计算分类误差
    if $\epsilon_k > 0.5$ **then break**
    $\alpha_k = \frac{1}{2} \ln\left(\frac{1-\epsilon_k}{\epsilon_k}\right), w_i \leftarrow \frac{w_i}{z_k} \times \begin{cases} e^{-\alpha_k}, & h_k\text{正确分类}\mathbf{x}_i \\ e^{\alpha_k}, & h_k\text{错误分类}\mathbf{x}_i \end{cases}$
  end for
  return $H(\mathbf{x}) = \text{sign}\left(\sum_{k=1}^{K} \alpha_k h_k(\mathbf{x})\right)$ ▷ 集成预测
end procedure

AdaBoost算法

- **AdaBoost示例**
  - AdaBoost算法学习3个基分类器的集成，基分类器采用线性分类器；



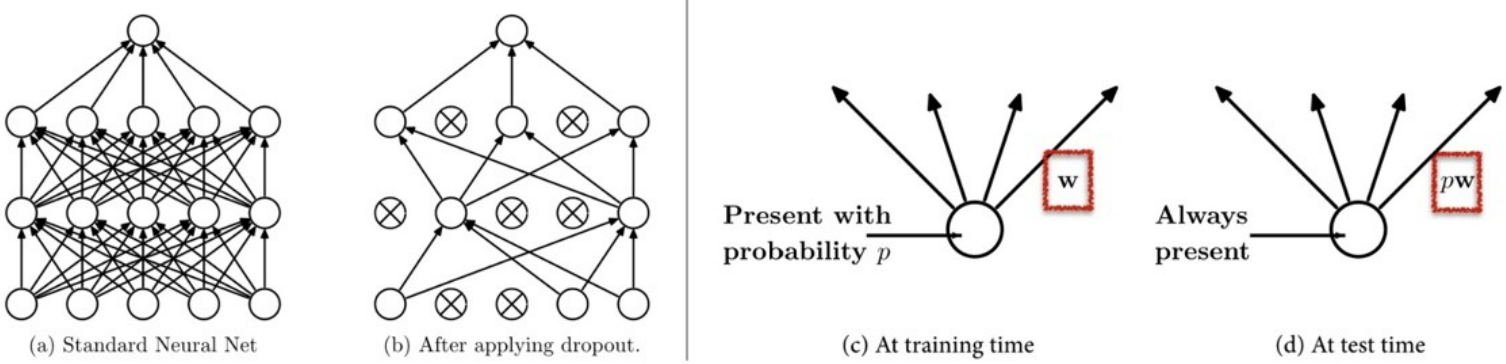
10.5 神经网络的集成

神经网络的集成方法

- **一般的集成方法**
  - Bagging算法生成多个神经网络，集成网络的分类结果；
  - 设置不同的网络层数、不同的隐含层神经元数量，可以学习得到不同结构的网络；
  - 相同结构的网络，设置不同的参数初始值，不同的学习率、收敛条件，可以得到不同参数的网络；
  - 随机梯度学习算法每次学习，产生不同的网络学习结果；
- **直接集成的缺点**
  - 计算效率低，神经网络的训练和预测耗时长，难以应用于大数据集训练的深度神经网络；
  - 集成神经网络的数量不会很多；

Drop Out方法

- 学习时
  - 每一轮迭代，以一定的概率p（如0.5）随机地保留网络中的一部分神经元，屏蔽其它神经元；
  - 被屏蔽的神经元，在前馈和反馈中都不起作用；
- 分类时
  - 每个神经元的输出乘以p；



(a) Standard Neural Net (b) After applying dropout. (c) At training time (d) At test time

Drop Out的解释

- **学习时**
  - 相当于随机抽取了一个网络进行学习；
  - 每一次抽取的网络权值是共享的，但连接结构是随机的；
- **分类时**
  - 可以想象为，按照同样的方式随机地抽取一个网络，计算神经网络的输出；
  - 重复随机抽取无穷多个网络，所有网络的输出结果以几何平均的方式集成；