Weka 软件中分类算法的实验比较与分析*

柳明†
School of CEI/AI, Nanjing Normal University

摘要

在利用 *Weka* 做分类算法的比较和分析过程中，我们选取了 *Bayes* 分类器中的 *Naïve Bayes*、*Functions* 分类器中的 *Multilayer Perceptron*、*Lazy* 分类器中的 *IBk*、*Meta* 分类器中的 *AdaBoost.M1* 和 *Trees* 分类器中的 *J48*，即从这 *5* 种分类器中各选取 *1* 个，同时选择 *Rules* 分类器中的 *ZeroR* 方法作为此次实验的 *baseline*。数据集有 *5* 个，分别为模式识别课程学习过程中使用的学生身高/体重数据集，*Weka* 环境自带的 *glass* 和 *segment* 数据集，*UCI* 机器学习数据集库中的 *Tic-Tac-Toe* 和 *Occupancy Detection* 数据集。实验中采取的评估方式有 *3* 种：训练集/测试集（数据集已提供时）、交叉检验（*5-fold*）和留出法（*hold-out*）即按比例将整个数据集划分为训练子集和测试子集。在 *5* 个数据集的任务上，多数情况下 *5* 个分类器都有较良好的准确率表现。经过具体分析和对算法参数的微调，我们也能一定程度上让分类器在任务上取得更好的表现。

1. 所用算法的基本原理

Weka [8]中将分类器算法划分为 Bayes、Trees、Functions、Lazy 和 Meta 等类别。如 Bayes 类别中为贝叶斯网络和朴素贝叶斯分类器，Lazy 中为基于惰性学习（lazy learning）的分类方法例如 k 近邻方法等。本小节结合相关资料 [11, 12, 13, 14]就实验中所使用的若干分类器的算法基本原理做出整理和简单的陈述。

朴素贝叶斯. 朴素贝叶斯（Naïve Bayes）法是基于贝叶斯定理（Bayesian theorem）与特征条件独立假设的分类方法。对于给定的训练集，首先基于特征条件独立假设学习输入输出的联合概率分布；然后基于此模型，对给定的输入 x，利用贝叶斯定理求出后验概率最大的输出 y。朴素贝叶斯法实现简单，学习和预测的效率都很高，是一种常用的方法。设输入空间 $\mathcal{X} \subseteq \mathrm{R}^n$ 为 n 维向量的集合，输出空间为类标记集合 $\mathcal{Y} = \{c_1, c_2, \cdots, c_K\}$。输入为特征向量 $x \in \mathcal{X}$，输出为类标记（class label）$y \in \mathcal{Y}$。X 是定义在输入空间 $\mathcal{X}$ 上的随机向量，Y 是定义在输出空间 $\mathcal{Y}$ 上的随机变量。$P(X,Y)$ 是 X 和 Y 的联合概率分布。训练数据集

$$T = \{(x_1, y_1,), (x_2, y_2,), \cdots, (x_N, y_N,)\}$$

由 $P(X,Y)$ 独立同分布产生。朴素贝叶斯法通过训练数据集学习联合概率分布 $P(X,Y)$。具体地，学习以下先验概率分布以及条件概率分布。先验概率分布为

$$P(Y = c_k), \quad k = 1, 2, \cdots, K$$

*本文的编写基于 CVPR'11 论文 [3]的 LaTeX 模板。

†E-mail: lium15@njnu.edu.cn or lium15@alu.hit.edu.cn

条件概率分布为

$$P(X=x \mid Y=c_k)=P(X^{(1)}=x^{(1)},\cdots,X^{(n)}=x^{(n)} \mid Y=c_k),\quad k=1,2,\cdots,K$$

于是学习到联合概率分布 $P(X,Y)$ 。

朴素贝叶斯法对条件概率作了条件独立性的假设，由于这是一个较强的假设，朴素贝叶斯法也由此得名。具体地，条件独立性假设是

$$P(X=x \mid Y=c_k)=P(X^{(1)}=x^{(1)},\cdots,X^{(n)}=x^{(n)} \mid Y=c_k)=\prod_{j=1}^{n} P(X^{(j)}=x^{(j)} \mid Y=c_k) \tag{1}$$

朴素贝叶斯法分类时，对给定的输入 x ，通过学习到的模型计算后验概率 $P(Y=c_k \mid X=x)$ ，将后验概率最大的类作为 x 的类输出。后验概率计算根据贝叶斯定理进行：

$$P(Y=c_k \mid X=x)=\frac{P(X=x \mid Y=c_k)P(Y=c_k)}{\sum_k P(X=x \mid Y=c_k)P(Y=c_k)} \tag{2}$$

将式1代入式2，有

$$P(Y=c_k \mid X=x)=\frac{P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k)}{\sum_k P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k)},\quad k=1,2,\cdots,K$$

这是朴素贝叶斯分类法的基本公式，于是朴素贝叶斯分类器可表示为

$$y=f(x)=\arg\max_{c_k}\frac{P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k)}{\sum_k P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k)} \tag{3}$$

注意到，在式3中分母对所有 c_k 都是相同的，所以，

$$y=\arg\max_{c_k} P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k)$$

在 Weka 中，Bayes 分类器中的 NaiveBayes 即为提供给用户使用的朴素贝叶斯分类器，通过添加参数 “-K” 可以将 useKernelEstimator 属性激活为 True，在 Weka 提供的 Manual [1]中介绍这个参数会使得分类器在处理数值属性时运用核密度估计（kernel density estimation）方法，多数情况下可以提升朴素贝叶斯分类器在具体任务中的表现。

多层感知器. 多层感知器（multi-layer perceptron, MLP）模型是比较经典的人工神经网络方法，它具有从训练数据中学习任意复杂的非线性映射的能力，也包括实现复杂的非线性分类判别函数。在认识到单个感知器的局限后，Rosenblatt 提出了这种多层的学习模型的原型：前一层神经元的输出是后一层神经元的输入，最后一层只有一个神经元，它接收来自前一层的输入，给出作为决策的一个输出。我们可以将许多全连接层堆叠在一起，每一层都输出给上面的层，直到生成最后的输出。这样可以将前 $L-1$ 层看作表示，把最后一层看作线性预测器。图1即为一个多层感知器。

这个多层感知器有 4 个输入，3 个输出，其隐藏层包含 5 个隐藏单元。输入层不涉及任何计算，因此使用此网络产生输出只需要实现隐藏层和输出层的计算。注意这两个层都是全连接的，每个输入都会影响隐藏层的每个神经元，而隐藏层的每个神经元又会影响输出层的每个神经元。约定用矩阵 $\mathbf{X}\in\mathbb{R}^{n\times d}$ 来表示 n 个样本的

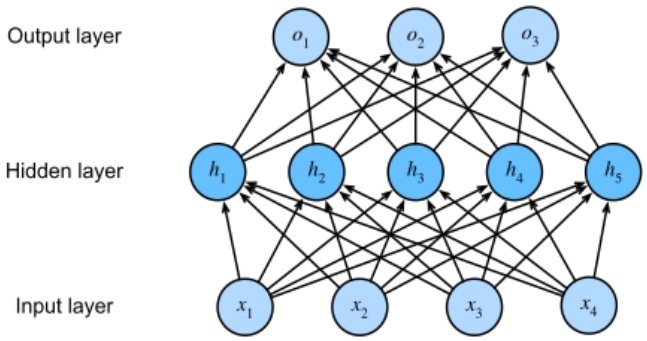

图 1. 一个具有一层隐藏层的多层感知器。

小批量，其中每个样本具有 d 个输入 (特征)。对于具有 h 个隐藏单元的单隐藏层多层感知机，用 $\mathbf{H} \in \mathbb{R}^{n \times h}$ 表示隐藏层的输出，称为隐藏表示。因为隐藏层和输出层都是全连接的，所以我们具有隐藏层权重 $\mathbf{W}^{(1)} \in \mathbb{R}^{d \times h}$ 和隐藏层偏置 $\mathbf{b}^{(1)} \in \mathbb{R}^{1 \times h}$ 以及输出层权重 $\mathbf{W}^{(2)} \in \mathbb{R}^{h \times q}$ 和输出层偏置 $\mathbf{b}^{(2)} \in \mathbb{R}^{1 \times q}$ 。形式上，我们按如下方式计算单隐藏层多层感知机的输出 $\mathbf{O} \in \mathbb{R}^{n \times q}$：

$$\mathbf{H} = \mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)}$$
$$\mathbf{O} = \mathbf{H}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}$$

可以看出，对仿射函数进行仿射变换之后得到的依然是仿射函数，容易证明这一等价性：

$$\mathbf{O} = (\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)})\mathbf{W}^{(2)} + \mathbf{b}^{(2)} = \mathbf{X}\mathbf{W}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(2)} = \mathbf{X}\mathbf{W} + \mathbf{b}$$

为了发挥多层结构的潜力，还需要一个关键要素：在仿射变换之后对每个隐藏单元应用非线性的激活函数（activation function）σ 。一般来说，有了激活函数之后就不可能再将多层感知器退化为线性模型。

$$\mathbf{H} = \sigma(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)})\mathbf{O} = \mathbf{H}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}$$

为了构建更通用的多层感知机，我们可以继续堆叠这样的隐藏层，例如，$\mathbf{H}^{(1)} = \sigma_1(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)})$ 和 $\mathbf{H}^{(2)} = \sigma_2(\mathbf{H}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(2)})$，一层叠一层，从而产生更有表达能力的模型。

多层感知器可以通过隐藏神经元捕捉到我们输入之间复杂的相互作用，这些神经元依赖于每个输入的值。我们可以很容易地设计隐藏节点来执行任意计算。例如，在一对输入上进行基本逻辑操作。多层感知器是通用近似器。即使是网络只有一个隐藏层，给定足够的神经元（可能非常多）和正确的权重，我们可以对任意函数建模，尽管实际中学习该函数是很困难的。

在 Weka 中，Functions 分类器中的 MultilayerPerceptron 默认是只具有一层隐藏层的多层感知器，隐藏层神经元个数默认为数据的属性数和类别数的平均值，激活函数默认为 Sigmoid 函数。使用时可以手动调整批量大小（batchSize）、隐藏层层数及每层的神经元个数（hiddenLayers）、学习率（learningRate）、动量法参数（momentum）和训练 Epoch 数（trainingTime）等参数，尝试让分类器表现得更好。

IBk（即 k 近邻）． Weka 里 Lazy 分类器中的 IBk 是基于 k 近邻法的分类器。k 近邻法（k-nearest neighbor, k-NN）是一种基本分类与回归方法。k 近邻法的输入为实例的特征向量，对应于特征空间的点；输出为实例的类别，可以取多类。k 近邻法假设给定一个训练数据集，其中的实例类别已定。分类时，对新的实例，根据其 k 个最近邻的训练实例的类别，通过多数表决等方式进行预测。由于没有显式的学习过程，k 近邻法是

懒惰学习的代表，在 Weka 中也属于 Lazy 分类器。k 近邻法实际上利用训练数据集对特征向量空间进行划分，并将其作为分类的“模型”。距离的度量、k 值的选择和分类决策规则是 k 近邻法的三个基本要素。k 近邻模型的特征空间一般是 n 维实数空间 R^n。使用的距离是欧氏距离，但也可以是其他距离，比如更一般的 L_p 距离或 Minkowski 距离。k 值的选择会很大程度上影响分类的结果，如果 k 值较小，则整体模型变得复杂，容易过拟合；k 值较大，意味着整体的模型变得简单。k 近邻法中的分类决策规则往往采用多数表决，即由输入实例的 k 个邻近的训练实例中的多数类决定输入实例的类。

AdaBoost.M1 算法. Boosting 方法是一种常用的集成学习（ensemble learning）方法，在分类问题中，它通过改变训练样本的权重，学习多个分类器，并将这些分类器进行组合，提高分类的性能。AdaBoost 是由 Freund 和 Schapire 首先提出来的一种用于二分类问题的集成方法，紧接着出现的 AdaBoost.M1 将二分类扩展到多分类问题。

对 Boosting 方法来说，有两个问题需要回答：一是在每一轮如何改变训练数据的权值或概率分布；二是如何将弱分类器（比较粗糙的分类规则）组合成一个强分类器（精确的分类规则）。关于前一个问题，AdaBoost 的做法是，提高那些被前一轮弱分类器错误分类的样本的权值，而降低那些被正确分类样本的权值。这样一来，那些没有得到正确分类的数据，由于其权值的增加会受到后一轮的弱分类器的更大关注。于是，分类问题被一系列的弱分类器“分而治之”。至于后一个问题，AdaBoost 采用加权多数表决的方法。具体地，加大分类误差率小的弱分类器的权值，使其在表决中起较大作用；减小分类误差率大的弱分类器的权值，使其在表决中起较小的作用。假设给定一个二类分类的训练数据集

$$T = \{(x_1, y_1,), (x_2, y_2,), \cdots, (x_N, y_N,)\}$$

其中，每个样本点由实例与标记组成。实例 $x_i \in \mathcal{X} \subseteq \mathrm{R}^n$，标记 $y_i \in \mathcal{Y} = \{-1, +1\}$，$\mathcal{X}$ 是实例空间，$\mathcal{Y}$ 是标记集合。AdaBoost 利用以下算法，从训练数据中学习一系列弱分类器或基本分类器，并将这些弱分类器线性组合成为一个强分类器。AdaBoost 算法叙述见算法 1。

Algorithm 1 AdaBoost 算法

输入: 训练数据集 $T = \{(x_1, y_1,), (x_2, y_2,), \cdots, (x_N, y_N,)\}$，其中 $x_i \in \mathcal{X} \subseteq \mathrm{R}^n$，$y_i \in \mathcal{Y} = \{-1, +1\}$；弱学习器

输出: 最终分类器 $G(x)$

1. 初始化训练数据的权值分布；

$$D_1 = (w_{11}, \cdots, w_{1i}, \cdots, w_{1N}), \quad w_{1i} = \frac{1}{N} \quad i = 1, 2, \cdots, N$$

2. 对 $m = 1, 2, \cdots, M$

(a). 使用具有权值分布 D_m 的训练数据集学习，得到基本分类器

$$G_m(x) : \mathcal{X} \to \{-1, +1\}$$

(b). 计算 $G_m(x)$ 在训练数据集上的分类误差率

$$e_m = \sum_{i=1}^{N} P(G_m(x_i) \neq y_i) = \sum_{i=1}^{N} w_{mi} I(G_m(x_i) \neq y_i)$$

(c). 计算 $G_m(x)$ 的系数

$$\alpha_m = \frac{1}{2}\ln\frac{1-e_m}{e_m}$$

(d). 更新训练数据集的权值分布

$$D_{m+1} = (w_{m+1,1}, \cdots, w_{m+1,i}, \cdots, w_{m+1,N})$$

$$w_{m+1,i} = \frac{w_{mi}}{Z_m}\exp(-\alpha_m y_i G_m(x_i)), \quad i = 1, 2, \cdots, N$$

其中，Z_m 是规范化因子

$$Z_m = \sum_{i=1}^{N} w_{mi}\exp(-\alpha_m y_i G_m(x_i))$$

即做了一个归一化，使 D_{m+1} 成为一个概率分布。

3. 构建基本分类器的线性组合

$$f(x) = \sum_{m=1}^{M}\alpha_m G_m(x)$$

得到最终分类器

$$G(x) = \text{sign}(f(x)) = \text{sign}\left(\sum_{m=1}^{M}\alpha_m G_m(x)\right)$$

J48 决策树. 决策树（decision tree）是一种基本的分类与回归方法。决策树模型呈树形结构，在分类问题中，表示基于特征对实例进行分类的过程。决策树学习通常包括 3 个步骤：特征选择、决策树生成和决策树的剪枝。这些思想主要来源于 Quinlan 提出的 ID3 算法和 C4.5 算法 [9]。Weka 中的 J48 是 Waikato 大学的开发者们用 Java 实现的 C4.5 决策树（C4.5 的最后一个开源版本是 C4.5 Release 8, 由此命名为 J48） [10]。特征选择是生成决策树的重要步骤，通常特征选择的准则是信息增益（ID3）或信息增益比（C4.5）。而信息增益又由熵与条件熵定义。在信息论中，熵（entropy）用来表示随机变量不确定性的度量。设 X 是一个取有限值的离散随机变量，概率分布为 $P(X = x_i) = p_i, \quad i = 1, 2, \cdots, n$ 。则随机变量 X 的熵定义为

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i \tag{4}$$

通常，式4中的对数以 2 为底或以自然对数 e 为底。设有随机变量 (X, Y)，其联合概率分布为

$$P(X = x_i, Y = y_j) = p_{ij}, \quad i = 1, 2, \cdots, n; \quad j = 1, 2, \cdots, m$$

条件熵 $H(Y|X)$ 表示在已知随机变量 X 的条件下随机变量 Y 的不确定性。随机变量 X 给定条件下随机变量 Y 的条件熵（conditional entropy）$H(Y|X)$，定义为 X 给定条件下 Y 的条件概率分布的熵对 X 的数学期望

$$H(Y|X) = \sum_{i=1}^{n} p_i H(Y \mid X = x_i)$$

其中，$p_i = P(X = x_i), i = 1, 2, \cdots, n$ 。当熵和条件熵中的概率是由数据估计（如极大似然估计）得到的时候，所对应的熵与条件熵分别称为经验熵（empirical entropy）和经验条件熵（empirical conditional entropy）。

如出现零概率，约定 $0\log 0 = 0$。信息增益（information gain）表示得知特征 X 的信息而使得类 Y 的信息的不确定性减少的程度。特征 A 对训练数据集 D 的信息增益 $g(D, A)$，定义为集合 D 的经验熵 $H(D)$ 与特征 A 给定条件下 D 的经验条件熵 $H(D|A)$ 之差，即

$$g(D, A) = H(D) - H(D|A) \tag{5}$$

类似地，特征 A 对训练数据集 D 的信息增益比 $g_R(D, A)$，定义为其信息增益 $g(D, A)$ 与训练数据集 D 关于特征 A 的值的熵 $H_A(D)$ 之比，即

$$g_R(D, A) = \frac{g(D, A)}{H_A(D)} \tag{6}$$

其中，

$$H_A(D) = \sum_{i=1}^{n} -\frac{|D_i|}{|D|} \log_2 \frac{|D_i|}{|D|}$$

n 是特征 A 取值的个数。C4.5 决策树的生成叙述见算法 2。

Algorithm 2 C4.5 的生成算法

输入： 训练数据集 D，特征集 A 阈值 ε

输出： 决策树 T

1. 如果 D 中所有实例属于同一类 C_k，则置 T 为单结点树，并将 C_k 作为该结点的类，返回 T
2. 如果 $A = \varnothing$，则置 T 为单结点树，并将 D 中实例数最大的类 C_k 作为该结点的类，返回 T
3. 否则，按式6计算 A 中各特征对 D 的信息增益比，选择信息增益比最大的特征 A_g
4. 如果 A_g 的信息增益比小于阈值 ε，则置 T 为单结点树，并将 D 中实例数最大的类 C_k 作为该结点的类，返回 T
5. 否则，对 A_g 的每一可能值 a_i，依 $A_g = a_i$ 将 D 分割为若干非空子集 D_i，将 D_i 中实例数最大的类作为标记，构建子结点，由结点及其子结点构成树 T，返回 T
6. 对结点 i，以 D_i 为训练集，以 $A - \{A_g\}$ 为特征集，递归地调用步 1 - 步 5，得到子树 T_i，返回 T_i

2. 实验过程与结果分析

第1节中简要叙述了本次实验所选择使用的 5 个算法的基本原理，同时也为实验准备了 5 个数据集：学生身高/体重数据集，2 个 Weka 文件目录中自带的数据集，2 个从 UCI 机器学习数据集库 [4]中挑选的数据集。在实验中用到的评估方法 [11]有 3 种：(a) 若数据集直接提供了训练集（training set）与测试集（testing set），则直接利用之；(b) 留出法（hold-out）的基本思想是直接将数据集 D 划分为两个互斥的集合，其中一个作为训练集 S，另一个作为测试集 T，即 $D = S \cup T$，$S \cap T = \varnothing$，在 S 上训练出模型之后，用 T 来评估其误差；(c) 交叉验证法（cross validation）的思路则是将数据集 D 划分为 k 个大小相似的互斥子集，即 $D = D_1 \cup D2 \cup \cdots \cup D_k, D_i \cup D_j = \varnothing (i \neq j)$，其中每个子集 D_i 都尽可能保持数据分布的一致性，即从 D 中通过分层采样得到，然后每次用 $k-1$ 个子集的并集作为训练集，余下的那个子集作为测试集，这样就可获得 k 组训练/测试集，可进行 k 次训练与测试，所以又把交叉验证法称为 k 折交叉验证（k-fold cross validation），常用的 k 取值有 10、5、20 等，本次实验我们选择 $k = 5$。

为了让实验结果更具有可比性，我们选择 Weka 提供的 ZeroR 分类器的分类准确率（accuracy）作为此次实验的 baseline。若 5 个分类器表现出了过低的预测准确率，我们再尝试通过调整参数的方式优化之。ZeroR 分类器是一种最简单的分类器，这种方法仅仅根据历史数据统计规律，而选择一种概率最大的类别作为未知

样本的分类结果，相当于 k 值等于样本容量时的 k 近邻方法。对于任意一个未知样本，分类结果都是一样的。ZeroR 分类器简单的以多数类的类别（连续型数据使用简单均值）作为预测值，尽管它没有任何的预测能力，但是它可以作为一种与其他分类器的对比分类器。

学生身高/体重数据集. 身高/体重数据集为一个二分类任务数据集，属性有身高、体重、性别，训练任务是根据学生的身高和体重预测该生的性别。身高/体重数据集提供了训练集和测试集各 1 份，训练集和测试集各有 100 个数据点，男生和女生各占 50%。由于已经提供了训练集和测试集，我们直接使用训练集/测试集评估方式进行实验。

方法	ZeroR	Naive Bayes	MLP	IBk	AdaBoost.M1	J48
准确率（%）	50	92	90	91	89	91

表 1. 各方法处理身高体重数据集的结果（默认参数）。

结果如表1。可以看出 5 种方法均有较好的泛化能力，都在测试集上取得了 90% 左右的准确率。在 Naive Bayes 分类器中将 useKernelEstimator 属性激活为 True，即用核密度估计（KDE）直接估计数据的概率分布而不是事先假定的正态分布，官方手册 [1]中介绍使用核密度估计可以在多数情况下可以提升朴素贝叶斯分类器的表现。采用核密度估计之后朴素贝叶斯分类器在测试集上的准确率为 91%，没有进一步提高。可以认为是由于数据集容量有限，使得估计的精度受限，甚至较朴素贝叶斯方法多出了的几个数据点的误分，并且这个数据集的任务本身（即仅通过身高和体重预测性别）也决定了分类的准确率不可能达到非常高的水平，数据中一定存在着噪声。

glass 数据集. glass 数据集是 Weka 提供的一个多分类任务数据集，也被 UCI 机器学习数据集库 [4]所收录。数据集含有 214 条样本信息，每条样本有 10 种属性，前 9 个属性对应 9 种元素的含量，最后 1 个属性是玻璃的类型（共 7 类）。数据集根据玻璃的化学成分来判断玻璃的类型，目标是确定玻璃的用途。玻璃的用途包括建筑房间用玻璃、车辆上的玻璃、玻璃容器等。确定玻璃的用途类型是为了鉴证。例如在一个车祸或犯罪现场，会有玻璃的碎片，确定这些玻璃碎片的用途和来源，有助于确定谁是过错方或者谁是罪犯。评估方式采用 5 折交叉检验。

方法	ZeroR	Naive Bayes	MLP	IBk	AdaBoost.M1	J48
准确率（%）	35.51	50	68.69	69.62	44.86	65.42

表 2. 各方法处理 glass 数据集的结果（默认参数）。

结果如表2。显然朴素贝叶斯法在这个任务上表现很差，经过调参也并未提高其准确率，我认为是不均衡数据（imbalanced data）导致了朴素贝叶斯分类器出现这样相当低的准确率，而且在面对不均衡数据时，高准确率和低误差变得没有那么重要。我们可以注意一下分类器测试的其他结果，即通过由混淆矩阵（confusion matrix）得出的精确率（precision）、召回率（recall）及 $F1$ 值，其中有 4 类的精确率超过了 50%，3 类的召回率在 0.7 以上，3 类的 $F1$ 值超过了 0.6，所以朴素贝叶斯法虽然准确率低，但是在其他方面依然有一定的表现。

Weka 的 IBk 方法中默认 k 为 1 即最近邻方法，在尝试之后发现令 k 为 3 时，IBk 方法的准确率达到了 70.09%，在此基础上我们将算法中使用的距离度量换成曼哈顿距离（Manhattan Distance），准确率进一步提高

到了 71.50%。尽量如此，在所有算法中表现最好的 IBk 方法也无法很好地解决数据不均衡问题。

用 J48 决策树的效果较 IBk 和多层感知器相差不多，将方法中的 minObjNum（每个叶结点的最少实例数）改为 1 或 7 后准确率会有一定提升，分别为 67.29% 和 66.36%。我认为是由于当数据很不均衡时，很难获得较大的信息增益，导致分类树模型很难生长，效果也自然不好。

Weka 中的 AdaBoost.M1 默认的基分类器是决策树桩（decision stump），决策树桩是仅有一层划分的决策树，所以仅仅只是对每一个属性进行一次判断。从输出的日志记录中可以看出，决策树桩仅根据镁的含量做出划分，只会将数据点预测到固定的几个类别中。将基分类器设定为 J48 决策树，准确率达到了 75.23%，与单独使用 J48 相比有了较大提升，体现了集成学习对于模型优化的帮助。另外，如果以随机森林（random forest）作为 AdaBoost.M1 的基分类器，那么我们的模型在 5 折交叉检验中达到了 80.84% 的准确率，不过应当注意到用随机森林单独做预测的准确率也已经维持在了一个较高的水平（79.43%）。

segment 数据集. Weka 中的 segment 数据集来源于 MIT 的 Vision Group。每个实例都是一块大小为 3×3 的区域。用于训练的实例源于从包含有 7 个户外图像的数据库中随机采样的结果，这些图像中的每个像素都已经被手工分类标注，从而让我们可以创建一个分类器由 19 个属性对图像的不同区域的类别（7 类）做出预测，图像区域类别有：brickface（砖面），sky（天空），foliage（树叶），cement（水泥），window（窗户），path（小路），grass（草）。数据集本身已经提供了训练集（1500 个数据点）和测试集（810 个数据点），所以我们直接采取训练集/测试集评估方式。值得注意的是 segment 的训练集是一个相当均衡的数据集，7 个类别的实例数分布范围为 204∼236，可以期望朴素贝叶斯分类器有比之前良好的表现。

方法	ZeroR	Naive Bayes	MLP	IBk	AdaBoost.M1	J48
准确率（%）	11.60	77.04	93.95	95.80	25.19	96.17

表 3. 各方法处理 segment 数据集的结果（默认参数）。

结果如表3。显然 AdaBoost.M1 的极低准确率是默认的基分类器所导致的，基分类器指定为 J48 决策树之后，准确率达到了比其他几个分类器都要更高的 98.40%。J48 的优异表现也可以归因于信息增益大，有利于分类树模型生长。对于朴素贝叶斯分类器，注意到训练集有 1500 个数据点，较之前的数据集更大，我们激活 useKernelEstimator 属性，采用核密度估计之后朴素贝叶斯法的准确率提升到了 81.60%。

Tic-Tac-Toe 数据集. UCI 机器学习数据集库 [4]中的井字棋（Tic-Tac-Toe）终局数据集由 Johns Hopkins University 在 1991 年贡献，含有 958 种（不考虑对称性）井字棋的终局棋盘盘面。9 个属性直接表示 3×3 棋盘的 9 个格子，约定执“×”一方先行，属性值为“×”表示先行一方下在了该格子里，属性值为“○”表示后行一方下在了该格子里，属性值为“b”表示该格子为空格（blank），没有棋子。正类为执“×”者获胜的终局。评估方式采用 80%：20% 的训练集/测试集划分。

方法	ZeroR	Naive Bayes	MLP	IBk	AdaBoost.M1	J48
准确率（%）	70.31	70.31	97.40	98.43	71.88	83.85

表 4. 各方法处理 Tic-Tac-Toe 数据集的结果（默认参数）。

结果如表4。可以看到朴素贝叶斯的表现和 baseline 一样，并不能很好地处理这个主要基于规则的任务。

多层感知器和 IBk 训练出了极高的预测准确率，单独的 J48 只达到了 83.85%，以 J48 作为基分类器的 AdaBoost.M1 的准确率提升到了 94.79%。

Occupancy Detection 数据集. UCI 机器学习数据集库 [4]中的 Occupancy Detection 数据集的任务是用时间、温度、相对湿度、光照、二氧化碳浓度等指标预测办公室的占用情况，各种环境指标的数据和房间的占用状态通过办公室内安装的传感器和摄像头每分钟采集一次 [2]。数据集本身已经提供了训练集和测试集，所以选择采取训练集/测试集评估方式。

方法	ZeroR	Naive Bayes	MLP	IBk	AdaBoost.M1	J48
准确率（%）	63.53	?	67.96	90.06	97.86	97.49

表 5. 各方法处理 Occupancy Detection 数据集的结果（默认参数）。

结果如表5。因为含有日期类型的数据，所以 Weka 中的朴素贝叶斯分类器不能被用来处理此任务，这一栏的数据缺失。多层感知器的在测试集上的分类准确率为 67.96%，2013 年 TAMU 的研究团队用人工神经网络 [5, 6]处理此类任务，达到了 67% ∼ 69% 与 70.4% ∼ 72.37% 的准确率，但也应当注意到 TAMU 的研究者们所处理的数据集更加复杂，数据的特征也和 UCI 数据集库中的 Occupancy Detection 数据集不尽相同，我们的数据集容量更小，也做了一定程度的简化。我们将 MLP 方法中的 decay 选项设为 True，即在学习过程中采用学习率衰减 (learning rate decay) 的策略，之后依然保持其他参数不变，MLP 在测试集上的准确率达到了所有模型中最高的 97.90%，仅仅只错分了 56 个样本。关于学习率衰减的一个形象解释是：在训练初期，我们希望学习的步长（即学习率）在可承受的范围内尽量大一些，随着训练轮数的增加，为了避免梯度下降无法收敛到全局最优点，甚至发散，我们期望学习的步长越来越小，逐步逼近最优点，至少是在一个最优的范围内来回振荡。

IBk 方法中，距离度量选择 Filtered Distance，其 Filter 也保持默认的 Random Projection 方法（默认参数），再次使用 IBk 方法训练的准确率为 97.56%。Weka 的 Java 文档介绍 Random Projection 方法用随机矩阵将数据投影到低维子空间以降低数据的维度，它首先应用 NominalToBinary 过滤器，在降维之前将所有属性转换为数字（numeric 类型）。默认的设置是将数据降到 10 维，在尝试之后我们将降维的目标维度设定为 5，准确率被进一步提升到了 97.82%。在 2003 年的 Fradkin 和 Madigan 的对比实验报告 [7]中，指出了 Random Projection 具有较 PCA 而言更低的计算成本。

在分析中，我们发现 AdaBoost.M1 的准确率和单独使用决策树桩的准确率一致（97.86%），输出的日志中可以看到决策树桩选取了光照度（Light）和 365.125 Lux（光照度单位）的大小关系作为决策依据。在对样本属性的观察中可以注意到一个有趣的事实：在训练集中确实几乎可以仅仅根据光照度将 2 类样本分开，在测试集中观察到了同样的现象，但在测试集中采取这样的决策规则会比在训练集中错分更多数据点。

3. 小结

本次实验中，经过结果分析和参数调整后，5 种分类器基本都在各数据集的具体任务上取得了较良好的准确率表现。通过对算法基本原理的整理和重新学习，让我更加深刻地体会到了具体问题具体分析的重要性。面对具体任务首先要了解、分析数据集与相应的任务要求及其内容，对数据集的分布、实际场景中的情况和模式识别任务都要先认识清楚之后，才能更好地利用学习过的基本方法去处理问题。了解应用场景是解决“从哪里来”的问题，分析数据集对应“是谁”的问题，把握具体任务的要求和内容又是回答“到哪里去”问题的

过程。模式识别与机器学习不是简单算法的堆积，将其拿过来和数据集挨个配对处理之后比较一下就高枕无忧了，更多时候需要依据任务特点对现有的方法进行一定程度的改变和融合（如使用 IBk 方法处理 Occupancy Detection 数据集的过程）。

参考文献

[1] R. R. Bouckaert, E. Frank, M. Hall, R. Kirkby, P. Reutemann, A. Seewald, and D. Scuse. Weka manual for version 3-8-4. *University of Waikato, Hamilton, New Zealand*, 2019. 2, 7

[2] L. M. Candanedo and V. Feldheim. Accurate occupancy detection of an office room from light, temperature, humidity and co2 measurements using statistical learning models. *Energy and Buildings*, 112:28–39, 2016. 9

[3] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. S. Torr, and S.-M. Hu. Global contrast based salient region detection. In *IEEE CVPR*, pages 409–416, 2011. 1

[4] D. Dua and C. Graff. UCI machine learning repository, 2017. 6, 7, 8, 9

[5] T. Ekwevugbe, N. Brown, and V. Pakka. Real-time building occupancy sensing for supporting demand driven hvac operations. *13th International Conference for Enhanced Building Operations*, 2013. 9

[6] T. Ekwevugbe, N. Brown, V. Pakka, and D. Fan. Real-time building occupancy sensing using neural-network based sensor network. In *2013 7th IEEE international conference on digital ecosystems and technologies (DEST)*, pages 114–119. IEEE, 2013. 9

[7] D. Fradkin and D. Madigan. Experiments with random projections for machine learning. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 517–522, New York, NY, USA, 003. ACM Press. 9

[8] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009. 1

[9] R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993. 5

[10] WekaMOOC. Data mining with weka. http://www.youtube.com/playlist?list=PLm4W7_iX_v4NqPUjceOGd-OKNVO4c_cPD/. Accessed October, 2017. 5

[11] 周志华. 机器学习. 清华大学出版社, 北京, 2016. 1, 6

[12] 张学工，汪小我. 模式识别——模式识别与机器学习（第 4 版）. 清华大学出版社, 北京, 2021. 1

[13] 李航. 统计学习方法（第 2 版）. 清华大学出版社, 北京, 2016. 1

[14] 邱锡鹏. 神经网络与深度学习. 机械工业出版社, 北京, 2020. 1