第3章 数据分析入门

BIG DATA

Weka是一款免费的、基于Java环境下的开源的机器学习(Machine Learning)以及数据挖掘(Data Mining)软件。Weka是怀卡托智能分析环境(Waikato environment for knowledge analysis)的英文字首缩写。有趣的是,该缩写Weka也是新西兰独有的一种鸟名,而Weka的主要开发者恰好来自新西兰的怀卡托大学(the University of Waikato)。

3.1 Weka简介与数据预处理

3.1.1 软件下载

Weka软件的官方网址是https://www.cs.waikato.ac.nz/ml/weka/,网站首页如图3.1所示。

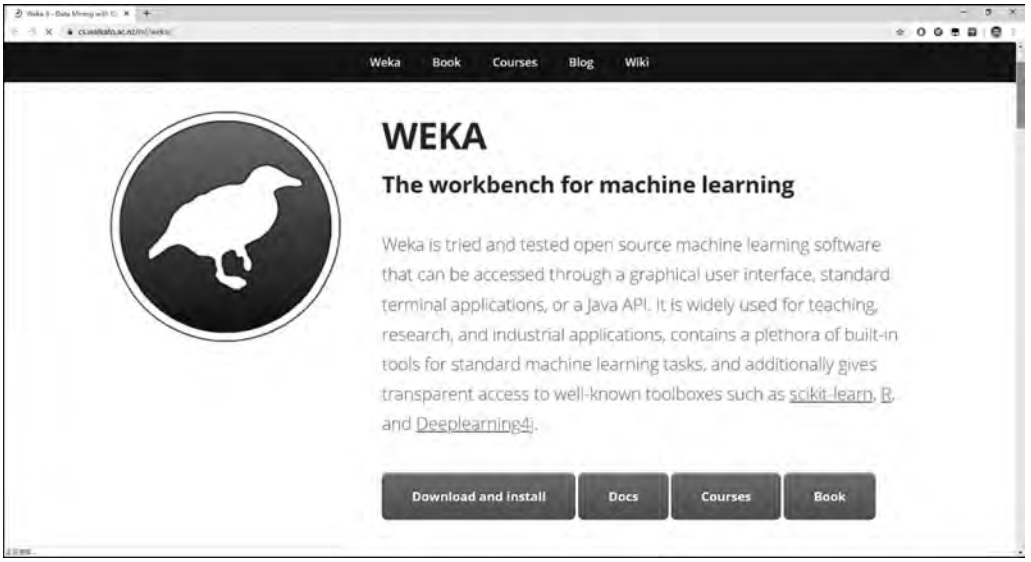


图3.1 Weka网站首页

Weka是集数据预处理、学习算法和评估方法等为一体的综合性数据挖掘工具,学习算法包括分类、回归、聚类、关联分析等。Weka经历了二十多年的发展,功能已经十分强大和成熟,代表了当今数据挖掘和机器学习领域的最高水平。

单击网站首页上的Download and install按钮,进入https://waikato.github.io/weka-wiki/downloading_weka/页面,如图3.2所示。选择Stable version(稳定版本)的Windows安装文件下载。如果用户使用的是其他操作系统,例如苹果的Mac OS或者开源的Linux,

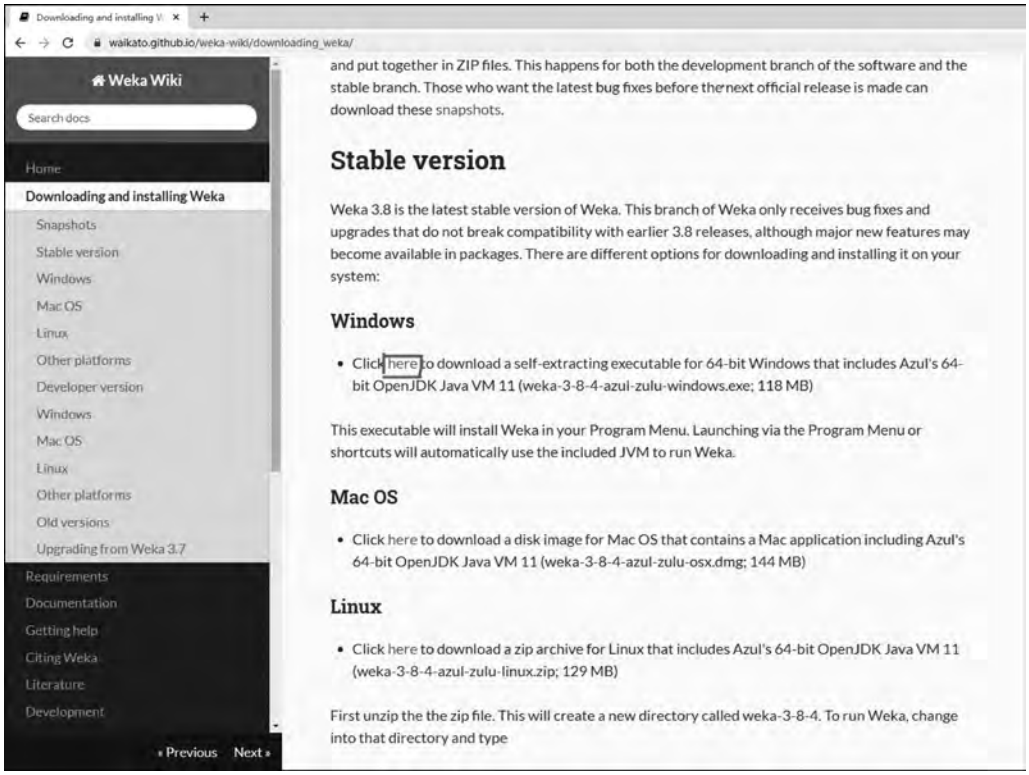


图 3.2　Windows 操作系统的 Stable 版本下载

则应选择对应的 Weka 版本。

本书以 Windows 系统下安装 Weka 为例。建议安装本书编写时使用的“稳定版本(Stable version)3.8.4”。如果使用的 Windows 系统未安装过 Java，则须下载自带 Java VM 的 Weka 版本，即安装文件名中带有 jre 字样的版本。安装结束后，启动 Weka，出现程序主界面，则表示安装成功，如图 3.3 所示。

图 3.3　Weka 主界面

3.1.2 文件与数据格式

Weka 用 ARFF(Attribute-Relation File Format)文件格式存储数据，这是一种 ASCII 文本文件。下载的 Weka 安装文件自带了多个示例用数据文件，其默认安装路径在“C:\Program Files\Weka-3-8-4\data”目录中，如图 3.4 所示。

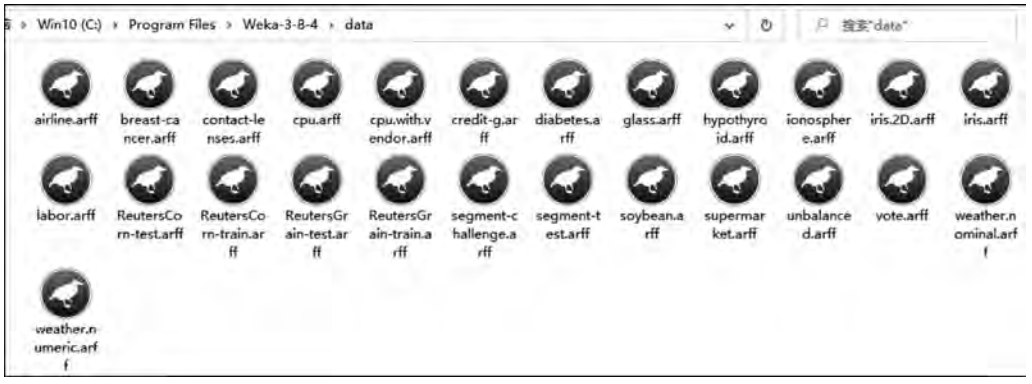


图 3.4 Weka 自带数据文件

下面以 Weka 自带的 weather.numeric.arff 数据文件为例，简单说明 ARFF 文件的格式和结构。如果直接双击该文件，Windows 会自动调用已经安装好的 Weka 软件的 Explorer 工具打开数据文件，如图 3.5 所示。

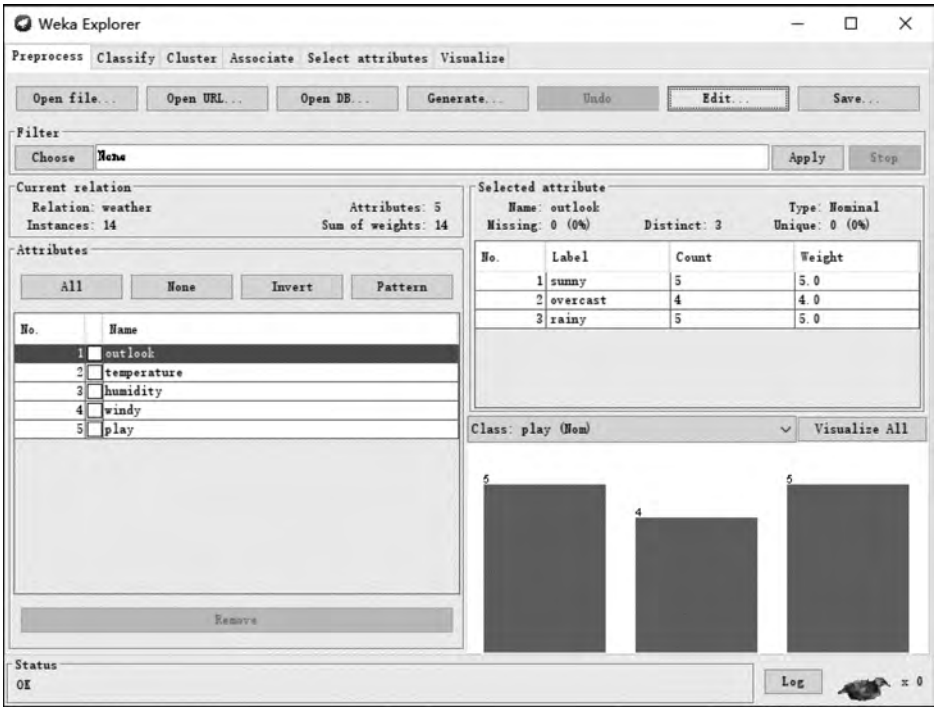
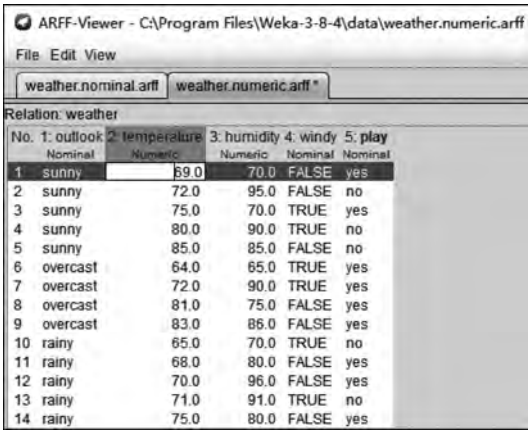


图 3.5 Explorer 自动打开 ARFF 数据文件

单击图 3.5 右上方的 Edit 按钮，则会看到数据的具体结构和值，如图 3.6 所示。



No.	1: outlook Nominal	2: temperature Numeric	3: humidity Numeric	4: windy Nominal	5: play Nominal
1	sunny	69.0	70.0	FALSE	yes
2	sunny	72.0	95.0	FALSE	no
3	sunny	75.0	70.0	TRUE	yes
4	sunny	80.0	90.0	TRUE	no
5	sunny	85.0	85.0	FALSE	no
6	overcast	64.0	65.0	TRUE	yes
7	overcast	72.0	90.0	TRUE	yes
8	overcast	81.0	75.0	FALSE	yes
9	overcast	83.0	86.0	FALSE	yes
10	rainy	65.0	70.0	TRUE	no
11	rainy	68.0	80.0	FALSE	yes
12	rainy	70.0	96.0	FALSE	yes
13	rainy	71.0	91.0	TRUE	no
14	rainy	75.0	80.0	FALSE	yes

图 3.6　浏览 numeric 数据

表格里的每个横行称作实例(instance),相当于统计学中的一个样本,或者数据库中的一条记录。每个竖行称作属性(attribute),相当于统计学中的一个变量,或者数据库中的一个字段。在 Weka 看来,这样的一个表格或者数据集,呈现了属性之间的一种关系(relation)。图 3.6 中一共有 14 个实例,5 个属性,关系名称为 weather。

如果直接使用文本编辑软件(例如,Windows 自带的“记事本”程序)打开 ARFF 数据文件,可以看到其内容如图 3.7 所示。

图 3.7　ARFF 文件以纯文本方式打开

ARFF 文件用分行来表示不同的数据域，因此不能在这种文件里随意断行。以“%”开始的行是注释，Weka 将忽略这些行。除去注释后，整个 ARFF 文件可以分为两个部分。第 1 部分给出了头信息(head information)，包括了对关系的声明和对属性的声明。

(1) 关系声明。关系声明在 ARFF 文件的第 1 个有效行，关系声明的格式为@relation < relation-name > 。< relation-name >是一个字符串。如果这个字符串包含空格，则必须加上引号(指英文标点的单引号或双引号)。

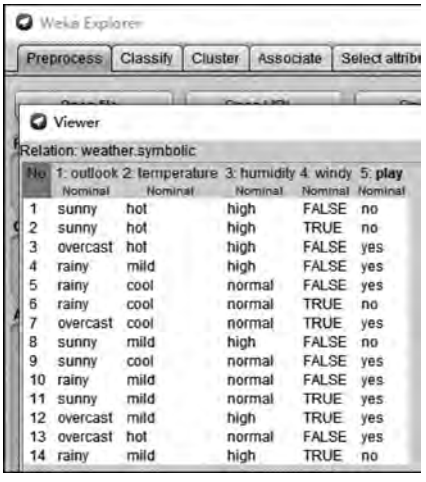
(2) 属性声明。属性声明用一列以@attribute 开头的语句表示。属性声明的格式为@attribute < attribute-name > < datatype >。其中，< attribute-name >是必须以字母开头的字符串。和关系名称一样，如果这个字符串包含空格，必须加上引号。数据集中的每一个属性都有它对应的@attribute 语句，来定义它的属性名称和数据类型。这些声明语句的顺序很重要。首先，它表明了该项属性在数据部分的位置。例如，humidity 是第 3 个被声明的属性，这说明数据部分那些被逗号分开的列中，第 3 列数据 85 90 86 96 ... 是相应的 humidity 值。其次，最后一个声明的属性被称作 class 属性，在分类或回归任务中，它是默认的目标变量。

第 2 部分给出了数据信息(data information)，即数据集中给出的数据。从@data 标记开始，后面的就是数据信息了。Weka 支持的< datatype >有以下 4 种：

- numeric 数值型，包括整数(integer)或者实数(real)；
- nominal-specification 标称型，只能取预定义值列表中的一个，常用于分类标识；
- string 字符串型；
- date 日期和时间型。

当然，对于大多数普通用户来说，直接使用图 3.6 的 Weka 自带数据浏览编辑工具就可以满足需求了。

在图 3.6 所示的数据集中，第 2 个属性 temperature(温度)和第 3 个属性 humidity(湿度)是数值型。第 1 个属性 outlook(天气状况)是标称型，可选的值是 sunny(晴天)、overcast(阴天)、rainy(下雨)。第 4 个属性 windy(刮风)是标称型，可选的值是 TRUE、FALSE。第 5 个属性 play(运动)是标称型，可选的值是 yes、no。



No	1: outlook Nominal	2: temperature Nominal	3: humidity Nominal	4: windy Nominal	5: play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

图 3.8　weather. nominal. arff 文件数据

Weka 自带了另外一个 weather. nominal. arff 数据文件，打开以后如图 3.8 所示。与图 3.6 相比，主要区别在于后者都是标称型数据，前者包含标称型和数值型两种数据。

在 Excel 和 Weka 之间交换数据，可以借助 csv 格式的文件，步骤如下。

(1) 将 Excel 文件另存为 csv 文件。使用 Excel 打开 xls 或者 xlsx 文件，将其另存为 csv 格式，如图 3.9 所示。

(2) 将 csv 文件在 ArffViewer 中另存为 arff 文件：打开 Weka 主界面，单击“工具”菜单，打开 ArffViewer 工具，打开刚刚生成的 csv 文件，文件类型选择 csv，另存为同名文件，文件类型选择 Arff data files 格式，如图 3.10 所示。

图 3.9　将 Excel 文件另存为 csv 文件

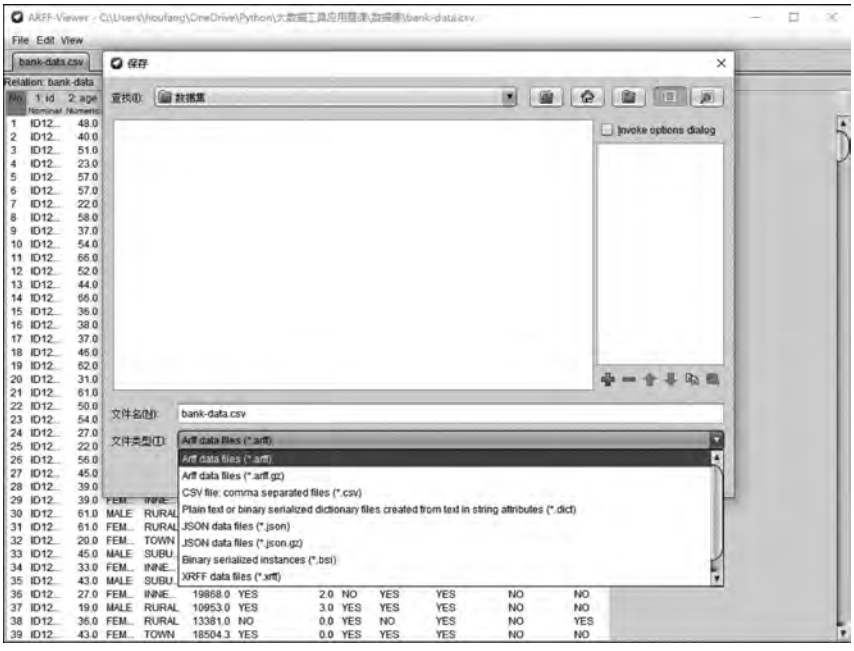


图 3.10　借助 ARFF-Viewer 将 csv 文件导出为 arff 文件

3.1.3 Weka 程序界面

本小节介绍 Weka 程序界面。Weka 主界面菜单包含以下 4 部分。

(1) Program(程序)菜单,如图 3.11 所示。

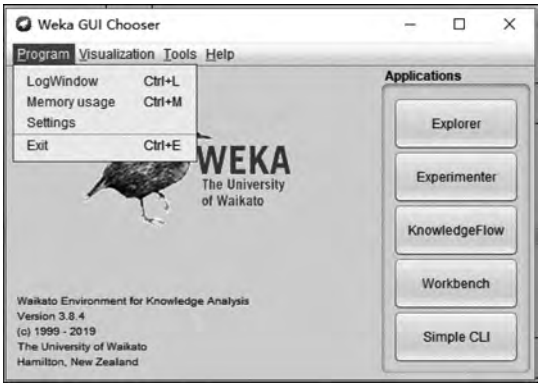


图 3.11 Weka Program(程序)菜单

Program(程序)菜单包含以下子菜单命令。

- LogWindow(日志窗口)。打开一个日志窗口,捕获所有的 stdout 或者 stderr 输出。
- Memory usage(内存使用)。显示 Weka 内存使用情况,同时执行 Java 垃圾回收。
- Settings(设置)。设置图形界面的风格和网络超时时间。
- Exit(退出)。退出 GUI 选择器。

(2) Visualization(可视化)菜单,如图 3.12 所示。

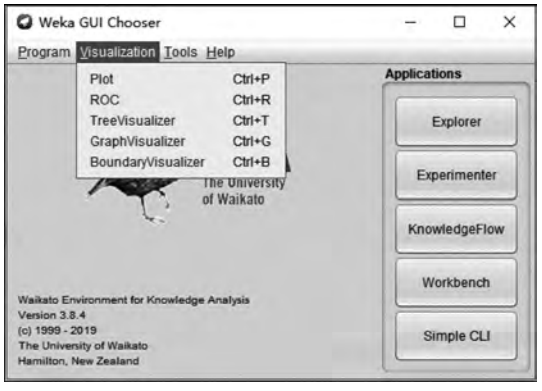


图 3.12 Weka Visualization(可视化)菜单

Visualization(可视化)菜单包含以下子菜单命令。

- Plot(散点图)。绘制数据集的 2D 散点图。
- ROC(受试者工作特征曲线)。显示 ROC 曲线。
- TreeVisualizer(树结构可视化)。显示有向图,例如决策树等。
- GraphVisualizer(图结构可视化)。显示 XML、BIF 或 DOT 格式的图,如贝叶斯网络。

• BoundaryVisualizer(边界可视化)。显示二维空间中分类器决策边界的可视化。

(3) Tools(工具)菜单,如图 3.13 所示。

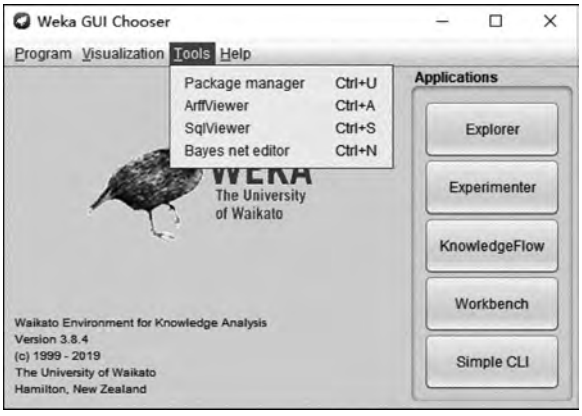


图 3.13 Weka Tools(工具)菜单

Tools(工具)菜单包含以下子菜单命令。

• Package manager(包管理器)。Weka 包管理系统的图形接口。
• ArffViewer(ARFF 文件查看器)。以电子表格形式查看 ARFF 文件的 MDI 应用。
• SqlViewer(SQL 查看器)。一个 SQL 工作表单,通过 JDBC 查询数据库。
• Bayes net editor(贝叶斯网络编辑器)。一个用于编辑、可视化和学习贝叶斯网络的应用。

(4) Help(帮助)菜单,如图 3.14 所示。

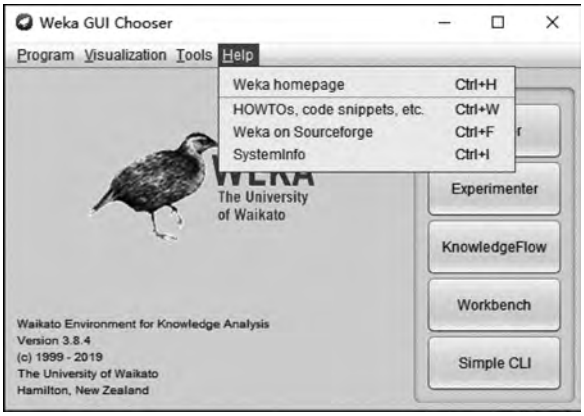


图 3.14 Weka Help(帮助)菜单

Help(帮助)菜单包含以下子菜单命令。

• Weka Homepage(Weka 主页)。在浏览器中打开 Weka 主页。
• HOWTOs,code snippets,etc.(WekaWiki)。包含许多关于 Weka 开发和使用的示例和指南。
• Weka on Sourceforge(Sourceforge.net 上的 Weka 项目)。Weka 项目在 Sourceforge.net 上的主页。

• SystemInfo(系统信息)。显示关于 Java/Weka 环境的内部信息,例如,CLASSPATH。

Weka 程序主页面窗口右侧共有 5 个按钮,如图 3.15(a)所示。

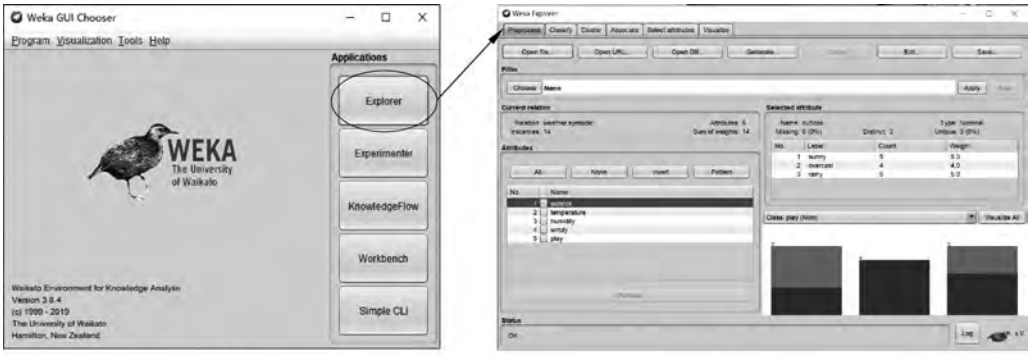


图 3.15　Explorer(探索者)界面

• Explorer 是用来进行数据实验、挖掘的环境,它提供了分类、聚类、关联规则、特征选择以及数据可视化的功能。
• Experimenter 是用来进行实验、对不同学习方案进行数据测试的环境。
• KnowledgeFlow 功能和 Explorer 差不多,不过提供的接口不同,用户可以使用拖曳的方式去建立实验方案。另外,它支持增量学习。
• Workbench 工作台界面包含了其他界面的组合。
• Simple CLI 是简单的命令行界面。

Explorer(探索者)界面是 Weka 主要图形用户界面(GUI),其全部功能都可以通过菜单选择或表单填写进行访问。后续操作以 Explorer 界面为主。单击 Explorer 按钮后,弹出如图 3.15(b)所示界面。

3.1.4　数据预处理

大数据处理一直遵循着一个理念,即高质量的数据才能产生高质量的数据挖掘结果。也就是高质量的输入才会得到高质量的输出。为了使输入的数据有比较好的质量,就必须预先对输入数据进行一些处理。数据挖掘前检测并纠正一些数据质量问题,称为数据预处理。下面以 Weka 自带的"C:\Program Files\Weka-3-8-4\data\weather.numeric.arff"文件为例,介绍对数据集进行属性删除、添加、赋值、离散化等操作的步骤。操作前请做好该文件的备份,以免原始数据被修改,影响后续操作。

1. 属性删除

属性删除可以直接在 Preprocess 标签页中进行。例如要将湿度(humidity)和刮风(windy)两个属性删除,则直接在左下方的 Attributes 区域选中 humidity 和 windy 两项前面的复选框,再单击 Remove 删除按钮即可。

[**注意**] 删除之后一定要选择 Save→Save as 命令将其另存为其他文件,否则会覆盖原来文件,如图 3.16 所示。

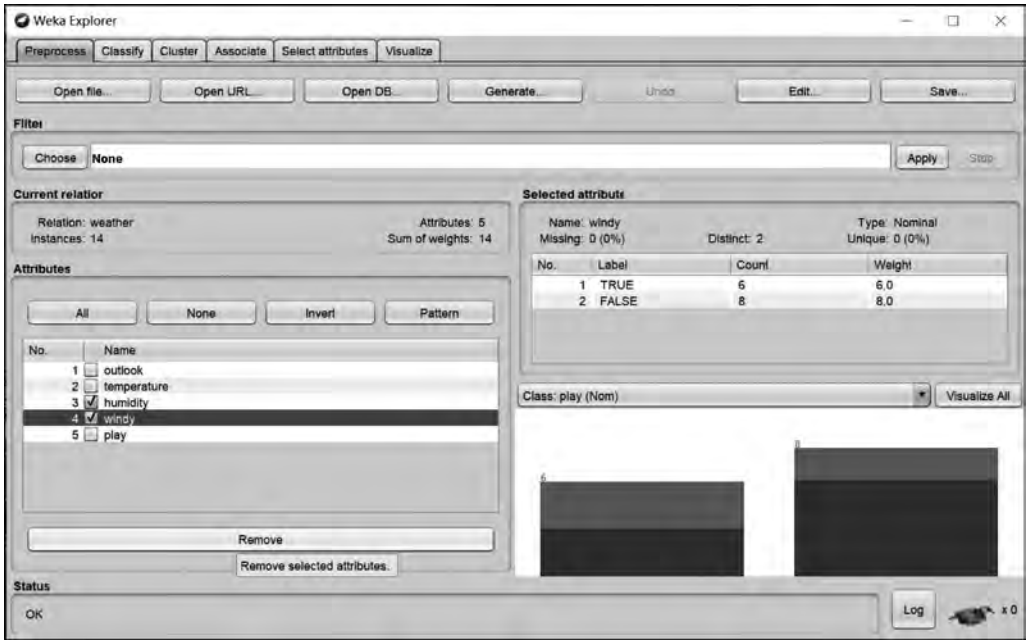


图 3.16　删除属性

2. 属性添加

假设要添加一个新属性“心情(mood)”,可选的值为“好(good)”或者“不好(bad)”。这时候需要使用 AddUserFields 过滤器来完成操作。Weka 中数据预处理工具称作 Filters 过滤器。顾名思义,过滤器是对输入数据集进行某种程度的过滤(转换)操作。Weka 过滤器分为无监督过滤器和有监督过滤器两种类型,每种类型又细分为属性过滤器和实例过滤器。有监督过滤器需要预先经过训练;无监督过滤器则无须预先训练。新属性“心情(mood)”添加步骤如下。

(1) 在 Explorer 对话框的 Preprocess 标签页中,可以找到 Filter 的下拉菜单,如图 3.17 所示。

(2) 单击 Choose 按钮,选择 filters→unsupervised→attribute→AddUserFields 过滤器,如图 3.18 所示。

(3) 单击 Filter 下方文本框中出现的 AddUserFields 字符串,会出现一个新的 weka.filters.unsupervised.attribute.AddUserFields 对话框。单击 New 按钮,在 Attribute name 中输入“mood”,Attribute type 下拉列表中选择 nominal 标称类型,如图 3.19 所示。

(4) 选好后,单击 OK 按钮。返回到 Preprocess 标签页中,单击 Apply 按钮,完成属性添加。添加了属性的数据如图 3.20 所示。

3. 属性赋值

“mood”属性是 nominal 标称型数据,还需设置允许的取值:“好(good)”或者“不好(bad)”,步骤如下。

(1) 保持图 3.20 中“mood”属性的选中状态,单击 Choose 按钮,找到 AddValues 过滤器,如图 3.21 所示。

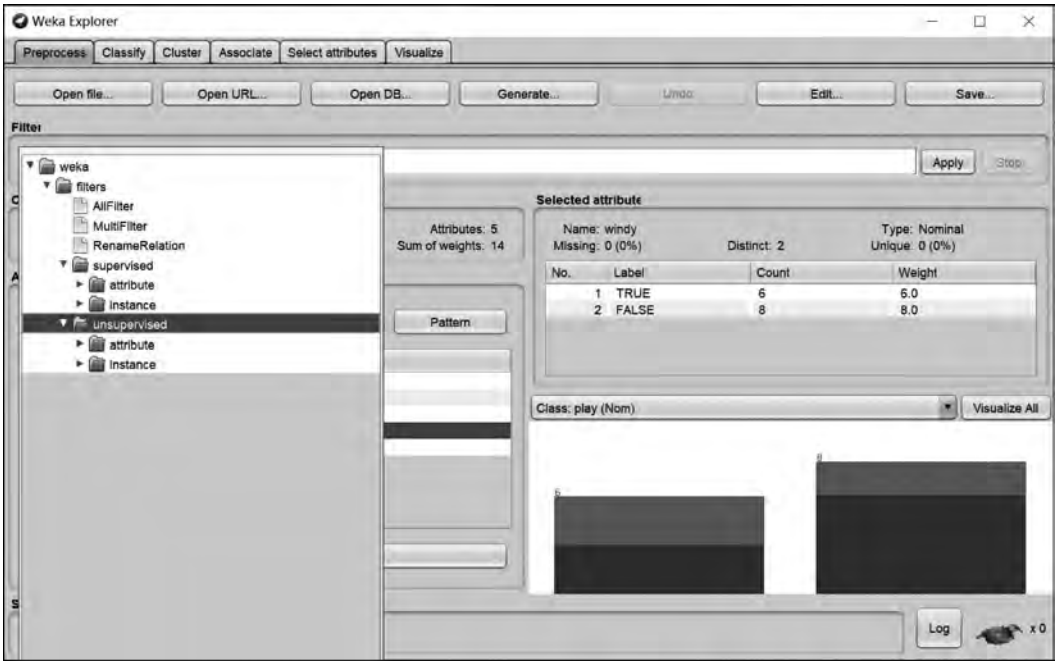


图 3.17　Filter 下拉菜单

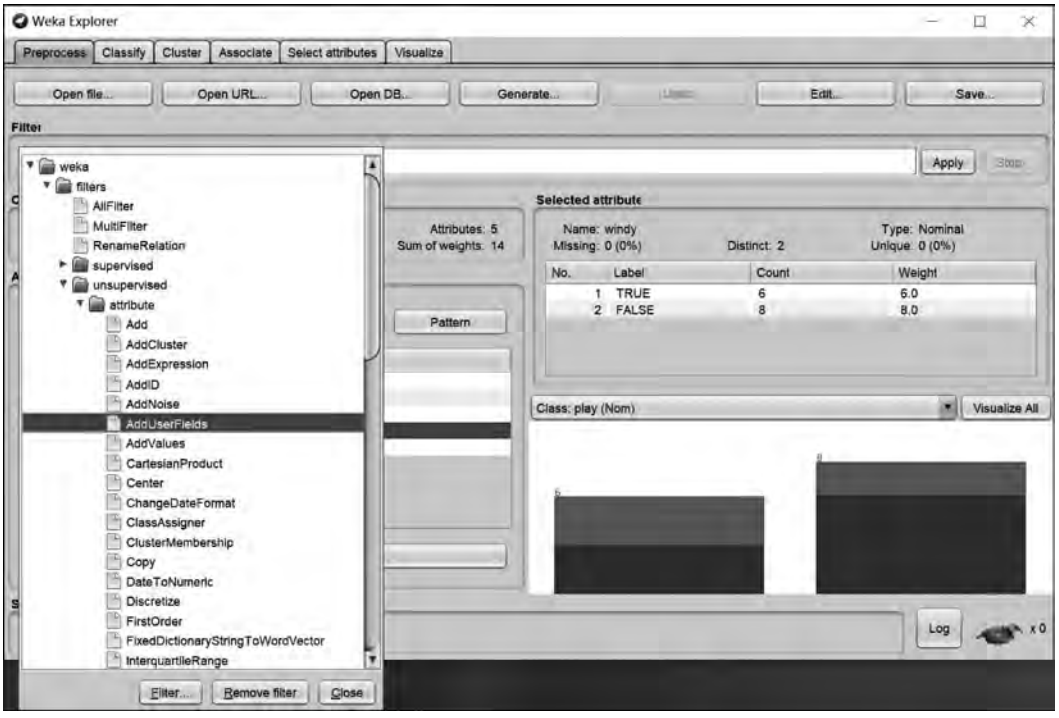


图 3.18　选择 AddUserFields 过滤器

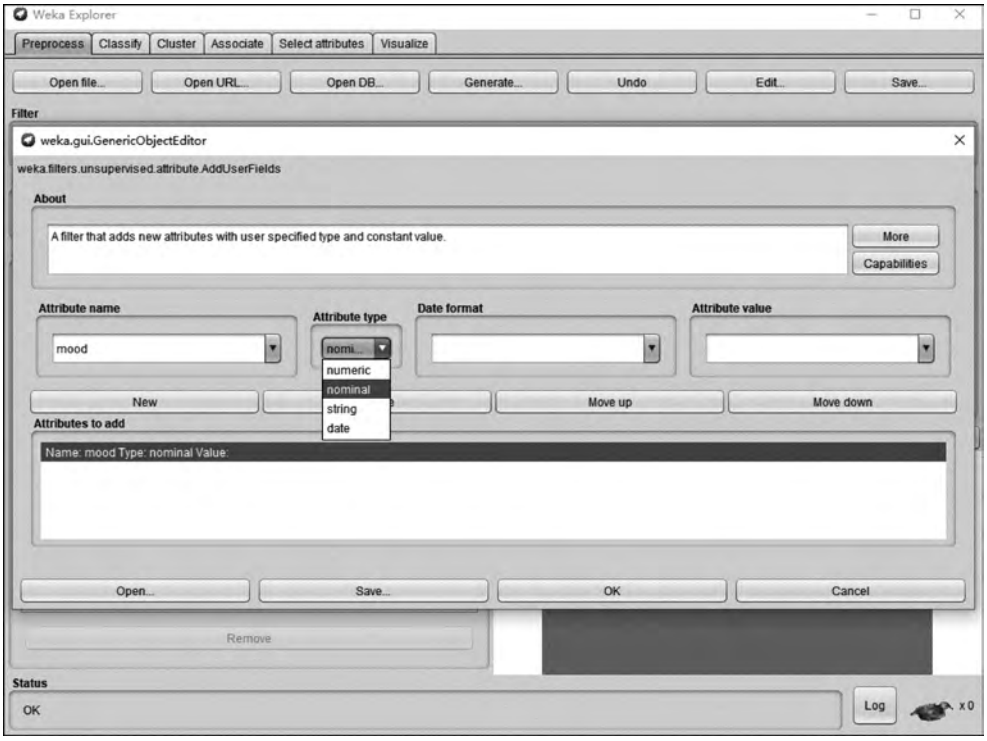


图 3.19　添加“mood”属性

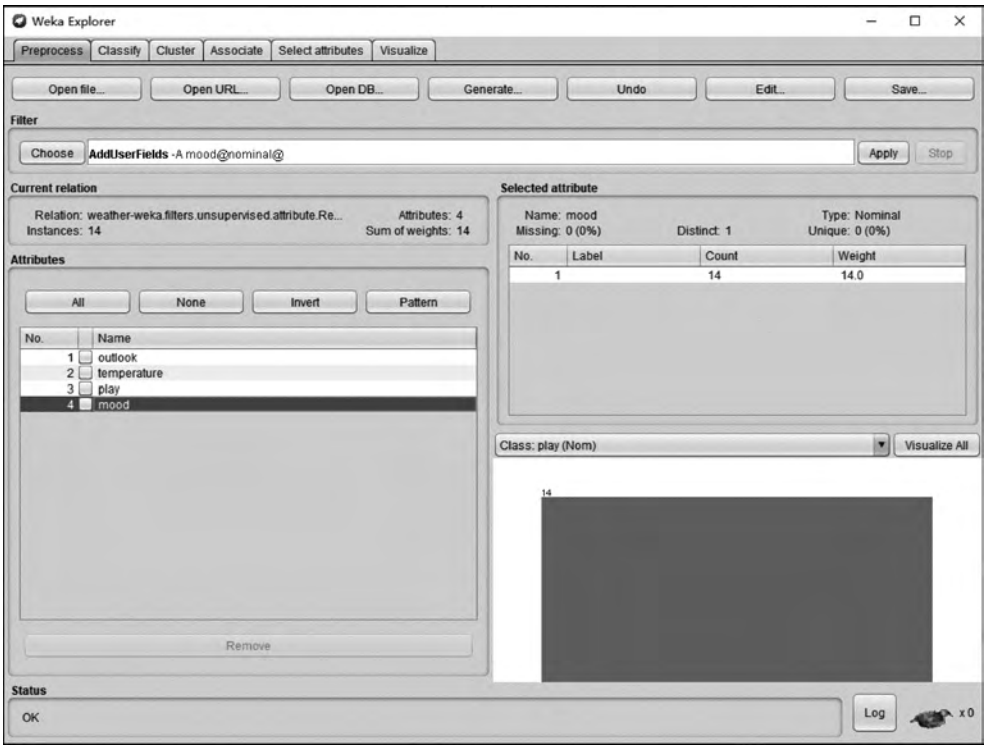


图 3.20　添加好的“mood”属性

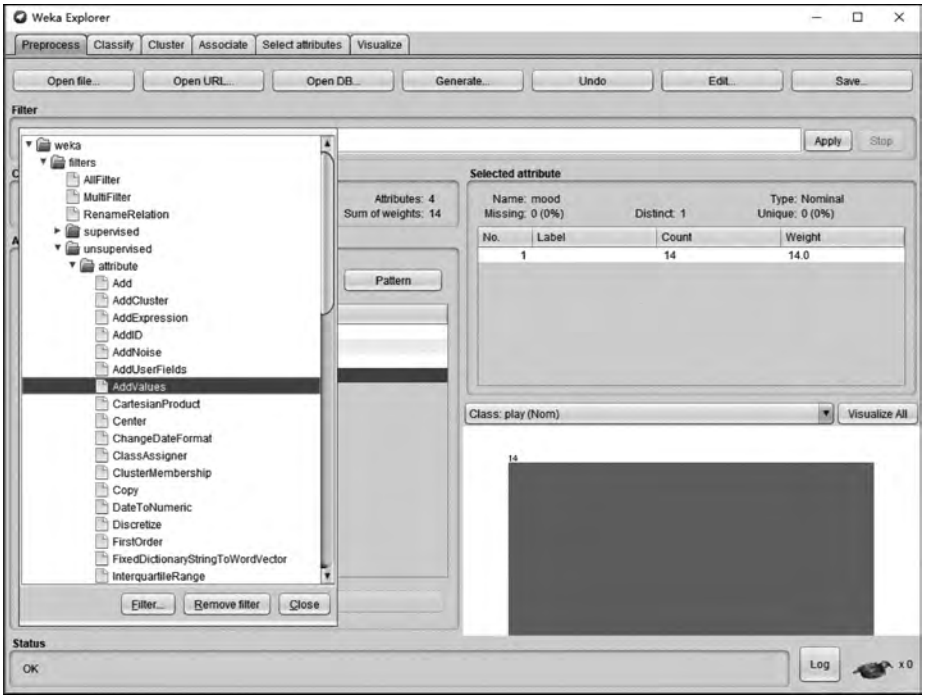


图 3.21　选择 AddValues 过滤器

（2）单击 Filter 下方的文本框，会出现一个新的 weka. filters. unsupervised. attribute. AddValues 对话框。在 labels 标签框中输入“good，bad”，单击 OK 按钮，如图 3.22 所示。

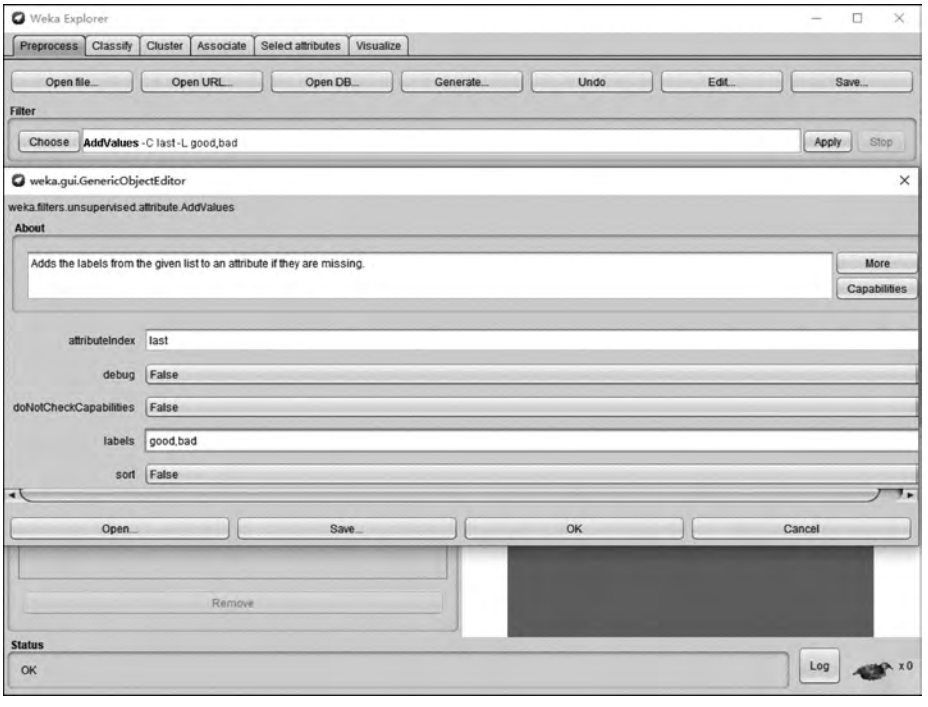


图 3.22　添加“mood”属性可能的取值

(3) 返回到 Preprocess 标签页中，单击 Apply 按钮。这样，该属性取值添加完成，如图 3.23 中部右侧的 Selected attribute 所示。

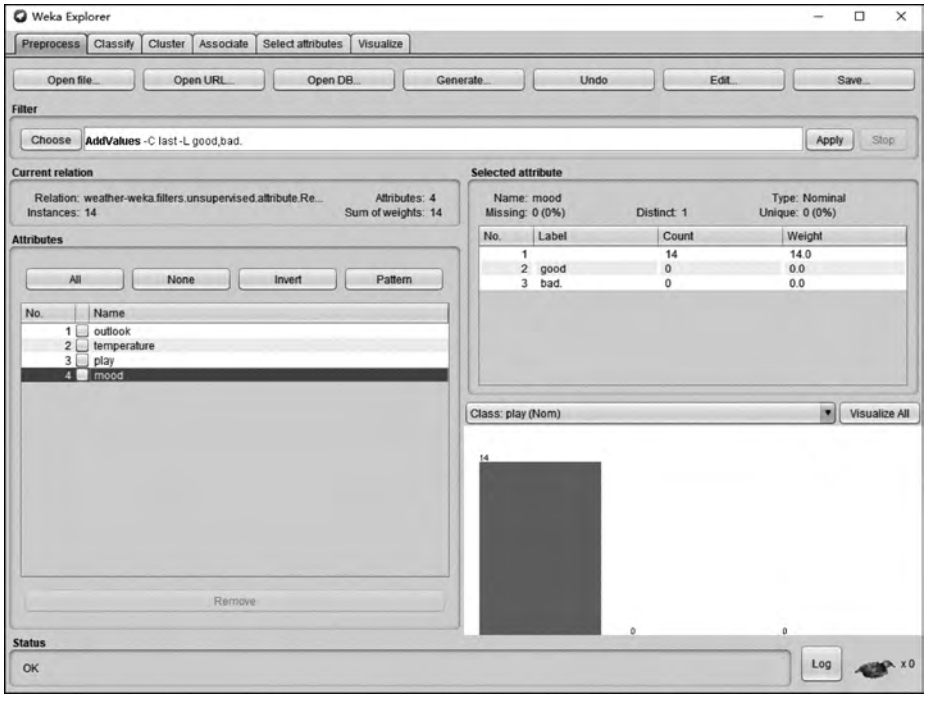


图 3.23 “mood”属性取值范围

(4) “mood”属性已经添加了，下面给每个实例的该属性设置具体的属性值。在图 3.23 所示的 Preprocess 标签页中，单击 Edit 编辑按钮。在 Viewer 界面，设置第 1 个实例的“心情(mood)”属性值为“好(good)”，设置第 2 个实例的“心情(mood)”属性值为“不好(bad)”，其他实例执行类似的操作，如图 3.24 所示。

4. 属性离散化

所谓属性离散化是将数值型属性转换为标称型属性。为什么需要离散化呢？主要是因为某些数据挖掘算法只能处理标称型属性，如关联分析等。离散化分为无监督离散化和有监督离散化。无监督离散化提供等宽和等频两种方法。等宽是指让间隔区间相等，等频是使一个区间内实例的数量相等。继续以 weather.numeric.arff 文件为例演示一个无监督、等宽离散化的操作。该文件中的 temperature 温度属性是数值型数据，现在需要将它们离散(转换)成低温(cool)、中温(mild)、高温(hot)3 个等级，步骤如下。

图 3.24 逐个设置“mood”属性值

(1) 单击 Choose 按钮，选择 filters→unsupervised→attribute→Discretize(离散化)过滤器，如图 3.25 所示。

(2) 单击 Filter 下方的文本框，会出现 weka.filters.unsupervised.attribute.Discretize 对话框，在这个对话框中，有两个关键参数：一个是 attributeIndices(属性编号)，

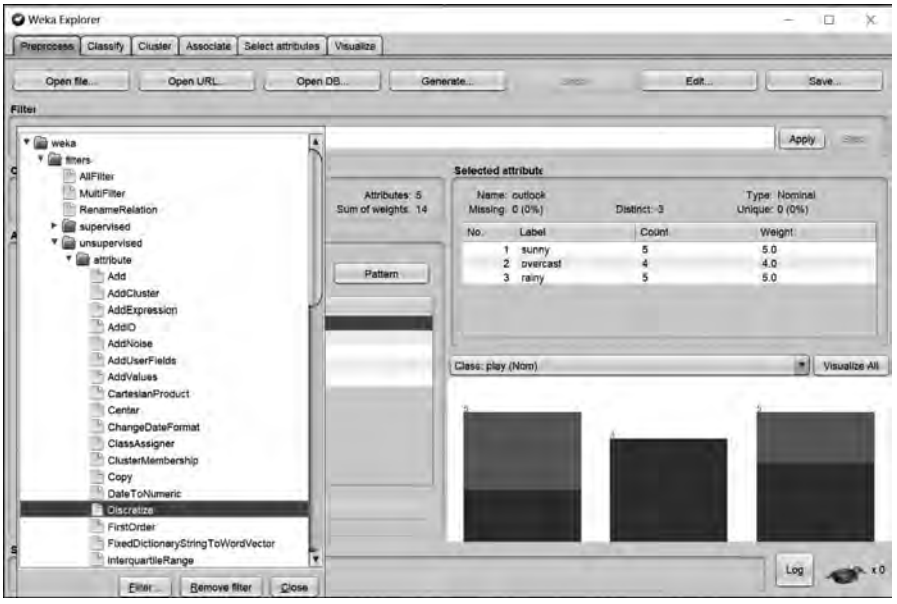


图 3.25　选择 Discretize 过滤器

指明需要进行离散化的属性是哪一列，因为 temperature 属性在第 2 列，则在属性编号 attributeIndices 文本框中输入编号 2。另一个关键参数是 bins(箱数)，指离散为多少个等级，由于需要离散成低温(cool)、中温(mild)、高温(hot)3 个等级，因此在箱数文本框输入 3。其他保持不变，单击 OK 按钮，如图 3.26 所示。

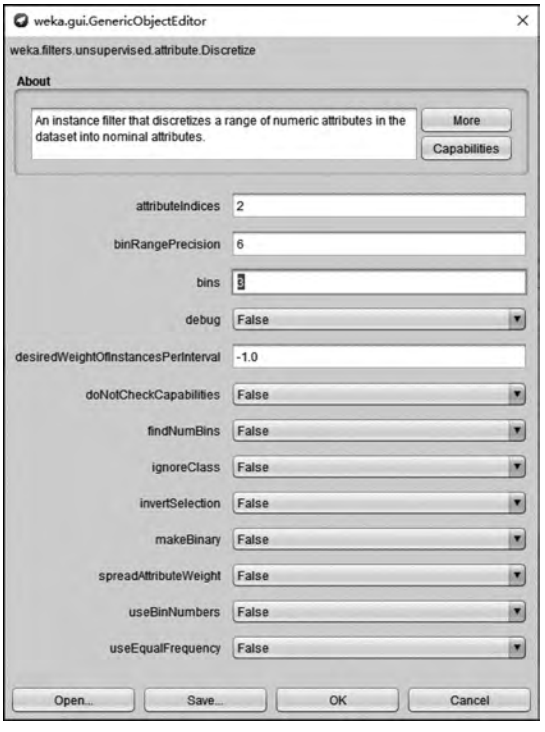


图 3.26　离散化参数设置

(3) 返回到 Preprocess 标签页中,单击 Apply 按钮。在 Attributes 中,选中 temperature(温度)属性,观察右侧 Selected attribute 区域,可以发现,原来的 temperature 被分为 3 个范围,如图 3.27 右侧中部所示。

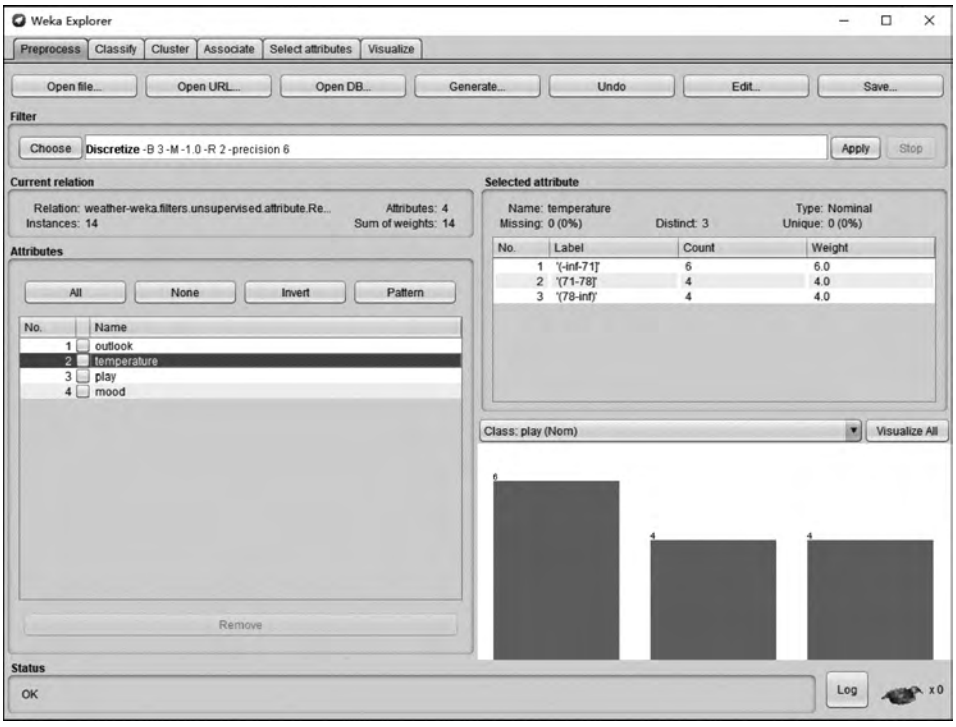


图 3.27 离散化后的 temperature 属性

(4) 如果觉得离散后的值可读性比较差,可以先将修改后的数据文件另存为其他文件,再使用 Word 打开该文件,用“替换”功能将图 3.27 中部右侧“Label”列下的 3 个值分别替换成低温(cool)、中温(mild)、高温(hot)。再用 Viewer 查看最后离散后的结果,此时已改为低温(cool)、中温(mild)、高温(hot)3 个等级了。

借助 Weka 的数据预处理功能,可以完成属性删除、添加、赋值、离散化等操作,数据预处理将为后续的数据挖掘算法实施奠定基础。

3.2 数据分类

分类(classify)是通过分类函数或分类模型(分类器),将未知类别实例划分到某个给定的类别,在第 1 章中,形象地称之为“贴标签”。即有一个由多个属性字段描绘的实例(也就是某个数据对象),分类算法通过预先制定好的规则,或者是通过前期的算法训练得到的一个数学模型,给这个实例贴上一个分类标签。分类在现实生活中有着广泛的应用。例如,通过前期的数据搜集,医院开发一套心血管疾病风险的预测系统。预测系统可以根据就诊者的身体指标、生活习惯、家族病史等一系列属性,对其罹患心血管疾病的可能性做分类预测,评估其为高风险、中风险或者低风险。再例如,在金融机构的风险控制领域,需要对客户是高风险、中风险、低风险进行分类,以便对不同类别的

客户采用不同的策略。

从分类的定义可以看出，分类是预测离散的值，主要预测未知类别实例所属类别。和分类类似的还有回归(regression)分析。回归分析也是通过已有的数据发现规律，从而预测未知属性的数值。分类和回归都是预测技术，但分类预测离散的值，回归预测连续的值。更直观的区别是，分类得到的结果是具体的类别，而回归得到的结果是具体的数值。

分类以及后续讲到的聚集等数据挖掘的方法，都是属于机器学习的范畴。为更好理解这些技术，在介绍具体的算法操作之前，先对机器学习的类型做一个分类：有监督学习(supervised learning)、无监督学习(unsupervised learning)、半监督学习(semi-supervised learning)。

(1) 有监督学习通过对已有样本分析建立模型。所谓已有样本，是指已知数据和输出。通俗地讲，就是算法“学习”的时候，有一个“监督者”事先提供了分好类的好样本和坏样本。算法目的在于找出这些样本的属性数据与“好或坏”标签之间的关系。

(2) 无监督学习直接对实例建立模型。这种“学习”方式，不存在一个可以提供分类结果的“监督者”，需要算法自己判断实例之间的关系。

(3) 半监督学习是介于有监督学习与无监督学习之间的一种类型。

显然，分类属于有监督学习的范畴，而 3.3 节介绍的聚类，则属于无监督学习。分类是利用已知的观测数据构建一个分类模型，常常称为分类器，用来预测未知类别实例所属类型。使用 Weka 进行分类分如下两个步骤。

(1) 通过分析已知类别数据集选择合适的分类器进行训练。

(2) 使用分类器对未知类别实例进行分类。

有可能需要根据分类的情况，返回到步骤(1)，重新选择另外的分类器。Weka 把分类和回归都放在 Weka Explorer 的 Classify 标签页中。下面介绍三个典型的分类器：J48 决策树、LinearRegression、M5P。

3.2.1 J48 决策树分类器

决策树是描述对实例进行分类的树形结构，由决策节点、叶节点和分支组成。决策节点表示一个属性，叶节点表示一个类别，分支表示某个决策节点的不同取值。现实生活中其实存在很多可以使用树形结构来做决策的例子，例如女孩找对象。

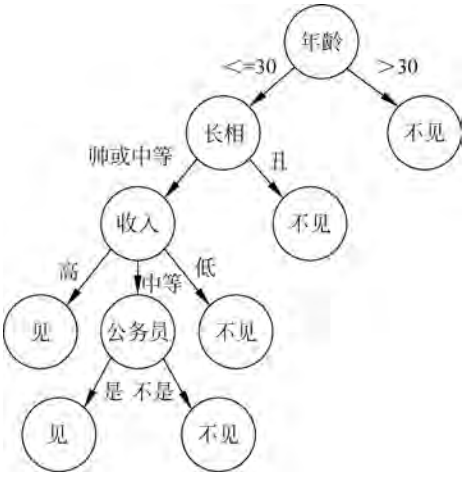


图 3.28 “见面”决策树

首先在女孩的心目中预先构建了一颗决策树，树中有年龄、长相、收入和公务员 4 个决策节点，代表男孩的 4 个属性。还有两个类别“见面”和“不见面”，这是叶子节点的取值范围。女孩依据男孩的具体情况，将男孩划分到“见面”还是“不见面”的类别。比方说母亲口中的男孩 26 岁、长相挺帅的、收入中等、公务员职业，依据女孩心中的决策树，这个男孩划分到“见面”的类别，如图 3.28 所示。

下面以天气数据集为例来介绍 J48 决策树分类器的使用。首先在 Preprocess 标签页中，

单击 Open file 按钮打开“C:\Program Files\Weka-3-8-4\data\ weather. nominal. arff”文件。切换到 Classify 标签页，单击 Choose 按钮，打开分类器分层列表。单击 trees 目录以展开子目录，然后找到 J48 目录并选择该分类器，如图 3. 29 所示。

图 3. 29　选择 J48 决策树分类器

回到 Classify 标签页，在左侧的 Test options(测试选项组)中，选中 Use training set(使用训练集)，即使用训练数据集作为测试数据集。单击 Start 按钮，训练和测试结果会以文本方式显示在右侧的 Classifier Output 分类器输出区域中，如图 3. 30 所示。

这是生成的用文本描述的决策树，可读性比较差。再来看可视化的决策树。在 Result list(结果列表)中，右击对应的 trees. J48 条目，并在弹出的快捷菜单中选择 Visualize tree(可视化树)菜单项，如图 3. 31 所示。

图 3. 32 是生成的可视化决策树，这个决策树就非常直观了。以该决策树最左侧的一支为例，决策路径是：先考察 outlook 属性的取值，如果该值是 sunny，则考察 humidity 属性的值，如果是 high，那么决策结果是 no；如果是 normal，那么决策结果是 yes。该决策树视图可以自动缩放和平移。

在图 3. 30 右侧的 Classifier output 中，算法运行总结 Summary 的 Correctly Classified Instance 的结果是 100%，表示这个分类器对所有的结果都判断正确了。在真实世界里，当然不会存在一个能 100%正确的预测算法。这里之所以出现这种情况，是因为用来训练分类器的数据和后来用分类器来预测的数据是相同的，所以出现了完全一致的情况。

现实中的业务一般会事先准备好一批带有分类标签的训练数据，将其分拆为两部分：一部分是保留分类结果的训练集数据，输入算法进行训练，得到训练好的算法模型；另一部

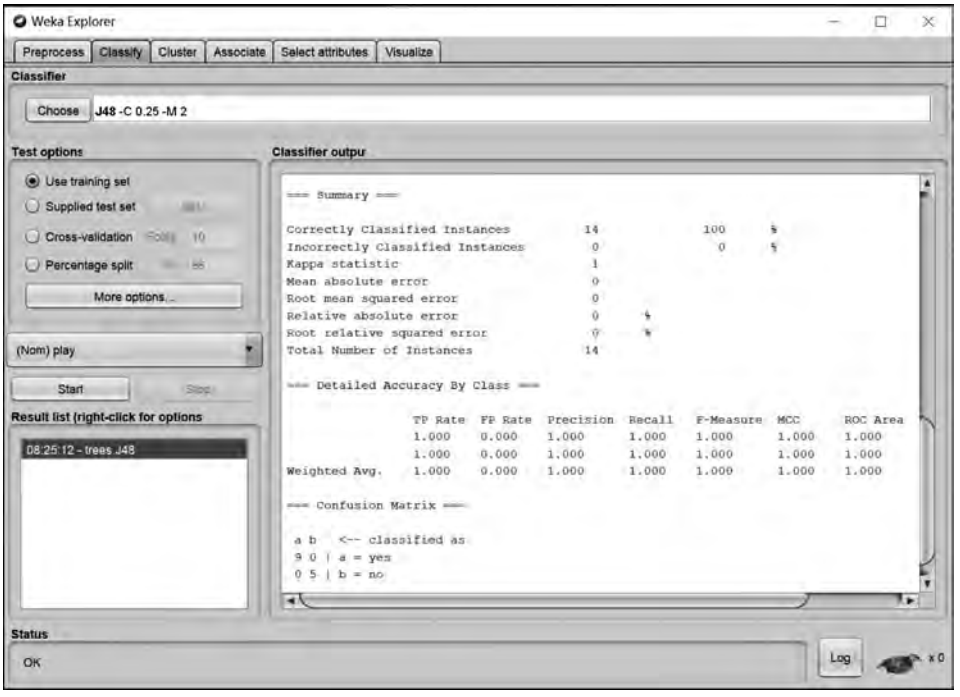


图 3.30　J48 分类器分类结果文本显示

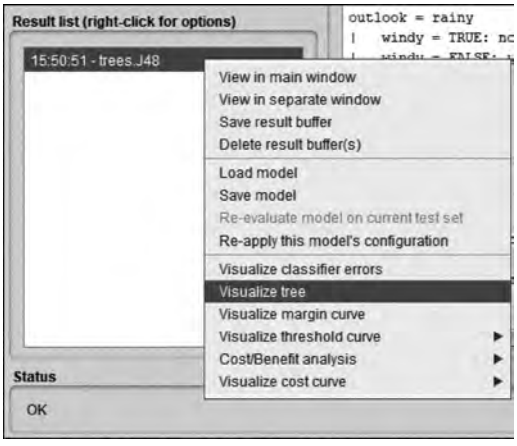


图 3.31　设置决策树可视化

分是去除了分类结果的测试集数据，用训练好的模型对测试集数据进行分类预测。然后比较测试集算法预测结果和原分类结果的误差，以此作为算法的评价标准。通过了评价的算法，就可以用于真实业务数据的分类了，如图 3.33 所示。

用于算法评估的方法有很多，混淆矩阵是其中比较简单直观的一个。图 3.34 是图 3.30 右下侧显示本例的混淆矩阵。大多数资料中的标准混淆矩阵见表 3.1。其中行表示真实分类结果，列表示预测分类，数字表示个数。本例中，全部 9 个真实类别为 yes 的实例都预测为 yes，全部 5 个真实类别为 no 的实例都预测为 no。主对角线（左上到右下）上的数值很大，非主对角线上的数值为 0，表示预测完全正确。

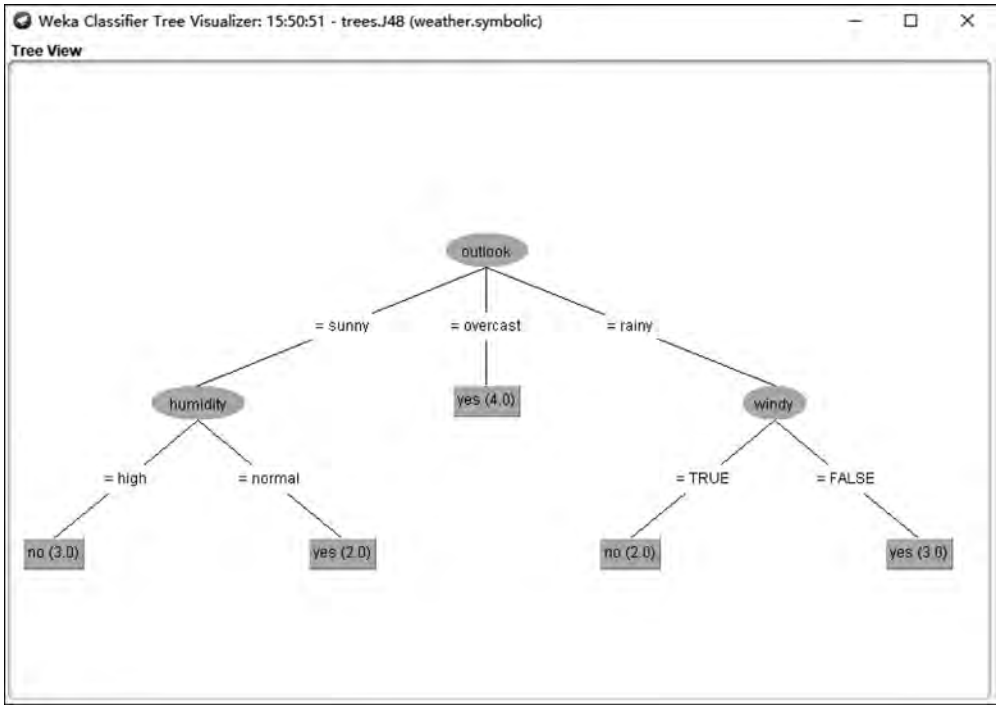


图 3.32　J48 分类器可视化决策树

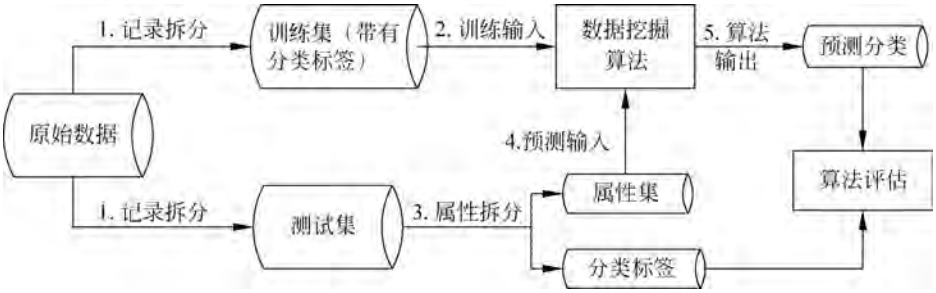


图 3.33　分类业务的一般流程

```
=== Confusion Matrix ===

 a b   <-- classified as
 9 0 | a = yes
 0 5 | b = no
```

图 3.34　Weka 输出的混淆矩阵

表 3.1　常见标准混淆矩阵

真 实 分 类	预 测 分 类	
	Yes	**No**
Yes	9	0
No	0	5

可以使用得到的决策树来进行预测。假设已知某天的天气状况是下雨(outlook＝rainy)，温度是低温(temperature＝cool)，湿度是高(humidity ＝high)，不刮风(windy＝FALSE)。根据上面这个决策树，这天能出去运动吗？按照图 3.32 决策树的决策路径，分类结果是 yes，那么这天是可以出去运动的。

这个实例也可以让 Weka 来自动进行分类。首先用记事本将已知天气指标构建一个 .arff 文件。头信息跟天气数据集完全一样；数据信息就是已知各个属性的值，play 值未知，设定为 yes。保存到素材文件夹，文件名为 test.arff，如图 3.35 所示。

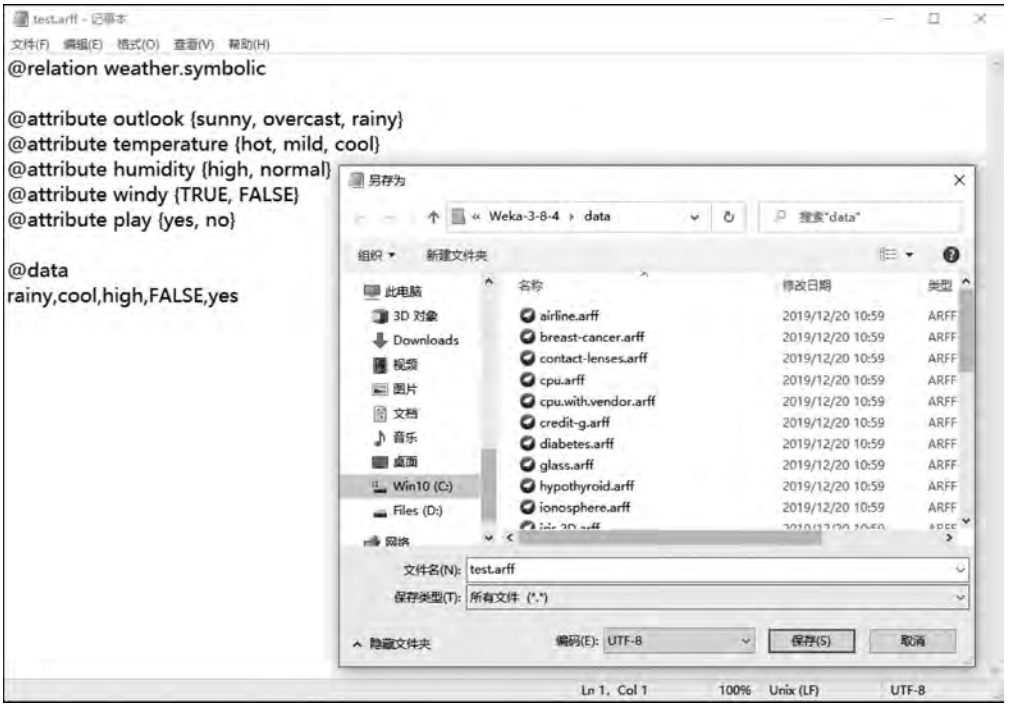


图 3.35　生成 test.arff 文件

准备好这个文件后，可以用 Weka 中已经生成的决策树来分类了。选择 Supplied test set(提供测试集)命令，选择 test.arff 文件，如图 3.36 所示。

单击 More option(更多选项)按钮，在弹出的对话框中，单击 Output Prediction 右侧的 Choose 按钮，选择 Plain Text，这样可以更好地观察预测结果。准备好后，单击 Start 按钮，右边的框显示分类结果，输出的测试结果如图 3.37 所示。重点来看 Predictions on test set 这一部分：第 1 列 inst＃为实例编号；第 2 列 actual 为真实类别，是在 test.arff 文件中填写的类别 yes；第 3 列为 J48 分类器利用之前训练出来的模型预测出来的类别，为 yes，跟刚才手工按照决策树路径进行判断的结果一致；第 4 列为预测结果置信度，为 1，如图 3.37 所示。

使用训练数据集作为测试数据集来判断分类器的性能不太可信，因为这个分类器本身就是由训练数据集的数据产生的。一般会认为把这个分类器用在新数据(非训练数据)上的效果更有说服力。这里介绍另外一种测试方法：十折交叉验证(10-fold cross-validation)。十折交叉验证用来测试算法的准确性。它的基本过程是：将数据集随机分成 10 份，轮流将其中 9

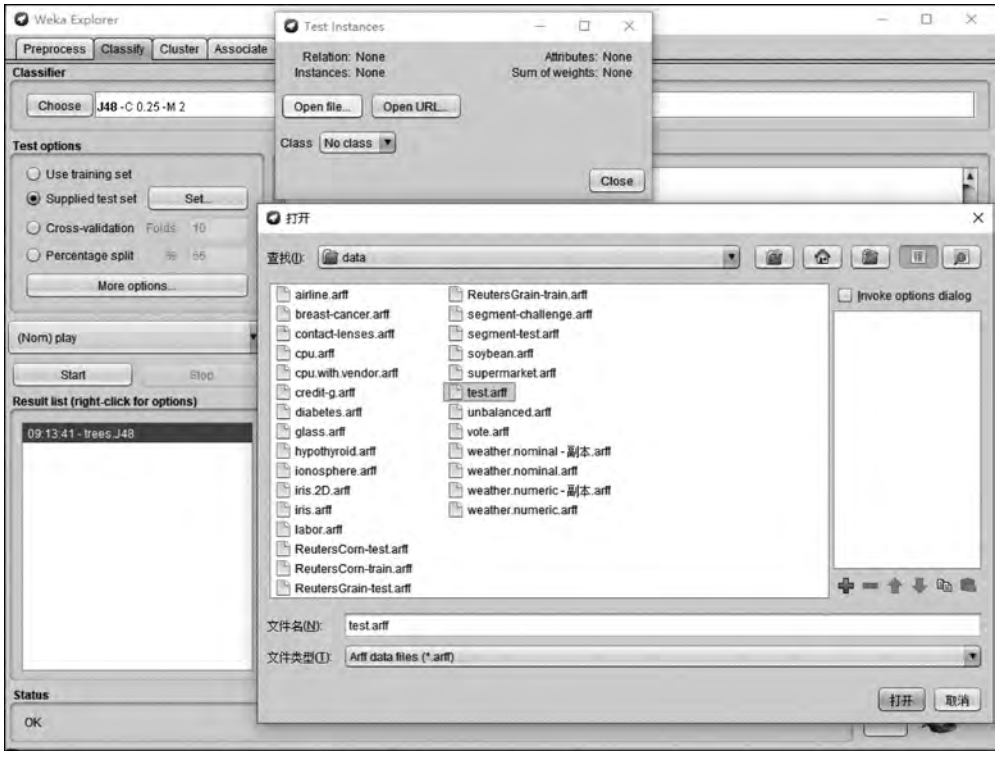


图 3.36　载入 test.arff 文件

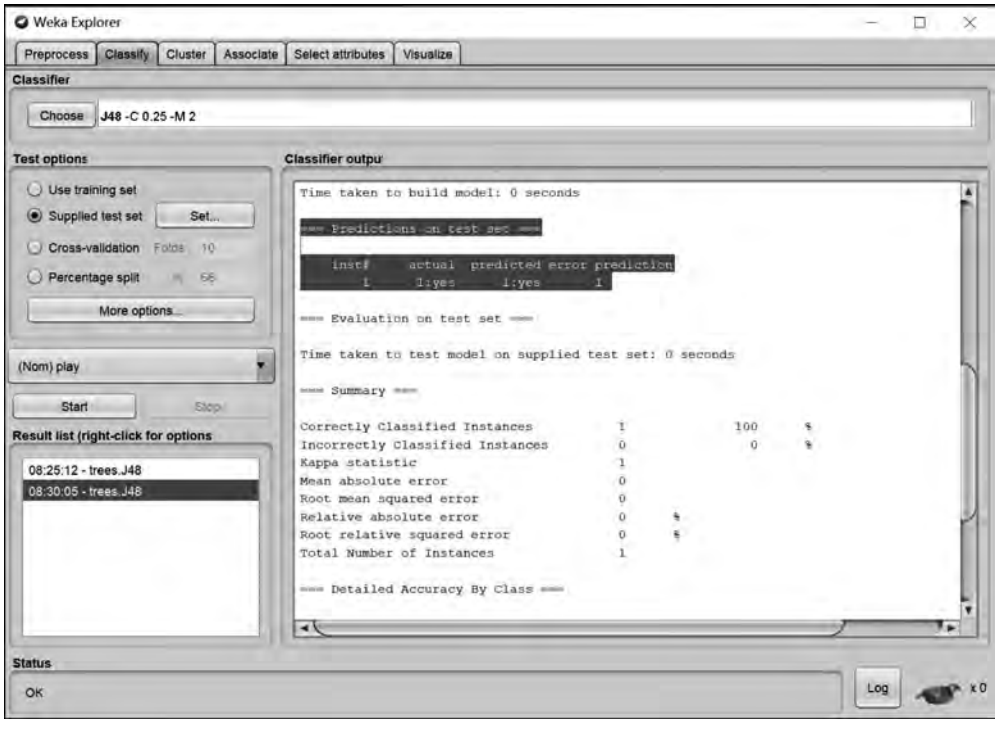


图 3.37　测试结果

份合成一个训练数据，1 份作为测试数据进行试验。这将进行 10 次试验。10 次试验得出 10 个正确率(或差错率)。那么，就将 10 次正确率的平均值作为对算法精度的估计。十折交叉验证使得训练数据集与测试数据集不同。

下面以 2.1 节中介绍过的鸢尾花(iris)数据集为例说明十折交叉验证的操作。鸢尾花数据集包含 150 个实例，每个类别有 50 个实例。定义了五个属性，分别是花萼长(sepal. length)，花萼宽(sepal. width)，花瓣长(petal. length)，花瓣宽(petal. width)，最后一个属性的字符串表示类别。进入 Explorer 界面，打开鸢尾花(iris)数据集，选择使用 J48 分类器。勾选十折交叉验证(10-fold cross-validation)。正确分类的测试实例数是 144 个，占比 96%。在右下侧的混淆矩阵中可以看到，有 1 个 a 被错误地分类到 b，有 3 个 b 被错误地分类到了 c，有 2 个 c 被错误地分类到了 b，如图 3.38 所示。

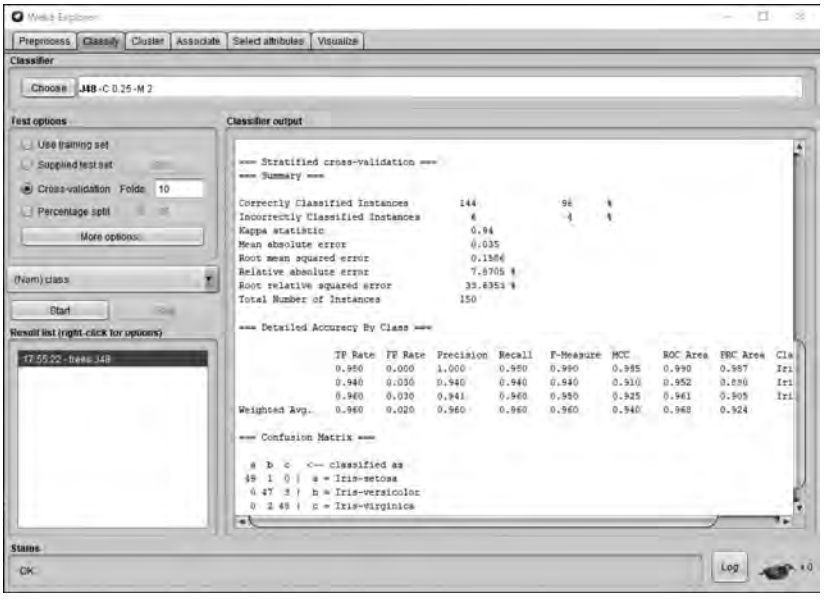


图 3.38　十折交叉验证

3.2.2　LinearRegression 分类器

本小节以 CPU 数据集为例介绍使用 LinearRegression 分类器构建线性回归公式。CPU 数据集呈现了 CPU 几个相关属性与其处理能力的关联，属性与类别都为数值型。CPU 数据集包含 209 个实例。定义了七个属性，分别是周期时间(MYCT)，内存最小值(MMIN)，内存最大值(MMAX)，高速缓存(CACH)，最小通道数(CHMIN)，最大通道数(CHMAX)，class 属性体现 CPU 性能的类别属性。

进入 Weka Explorer 界面，打开 CPU 数据集。切换到 Classify 标签页，单击 Choose 按钮，选择 functions 条目下的 LinearRegression 分类器，如图 3.39 所示。选择 Cross-validation Folds，保持默认值 10，单击 Start 按钮。

图 3.40 中，右边输出结果的中间部分 Linear Regression Model 是线性回归函数，其结果为 class=0.0491×MYCT+0.0152×MMIN+0.0056×MMAX+0.6298×CACH+

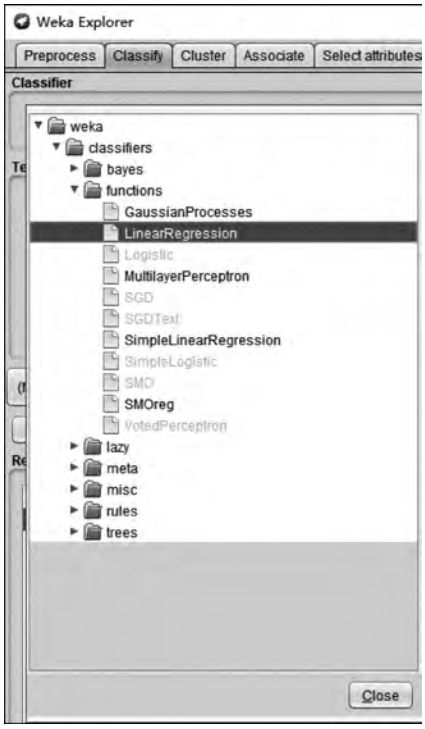


图 3.39　选择 LinearRegression 分类器

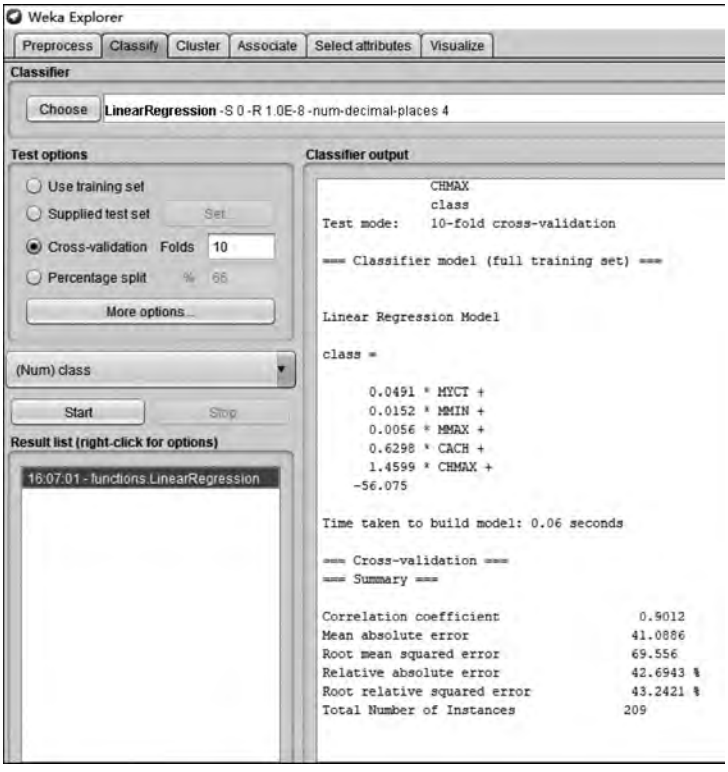


图 3.40　LinearRegression 线性回归结果

1.4599×CHMAX+(−56.075)。这说明 class 分类结果可以用这样一个线性方程表示，这也是 Linear Regression 名字的来源。

在 Result list 结果列表中，右击 functions LinearRegression，在弹出菜单中选择 Visualize classifier errors(可视化分类误差)命令，弹出一张误差可视化图，直观显示误差状况，如图 3.41 和图 3.42 所示。

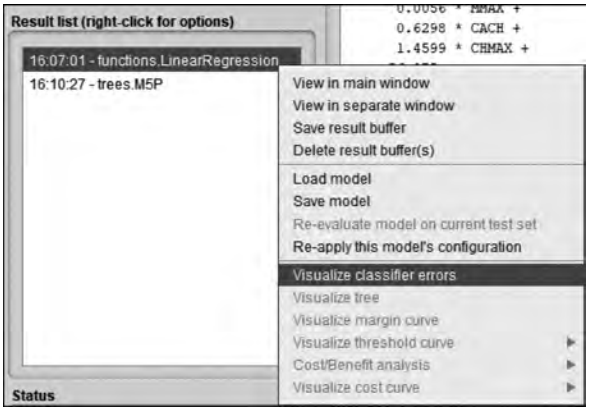


图 3.41　设置可视化分类误差

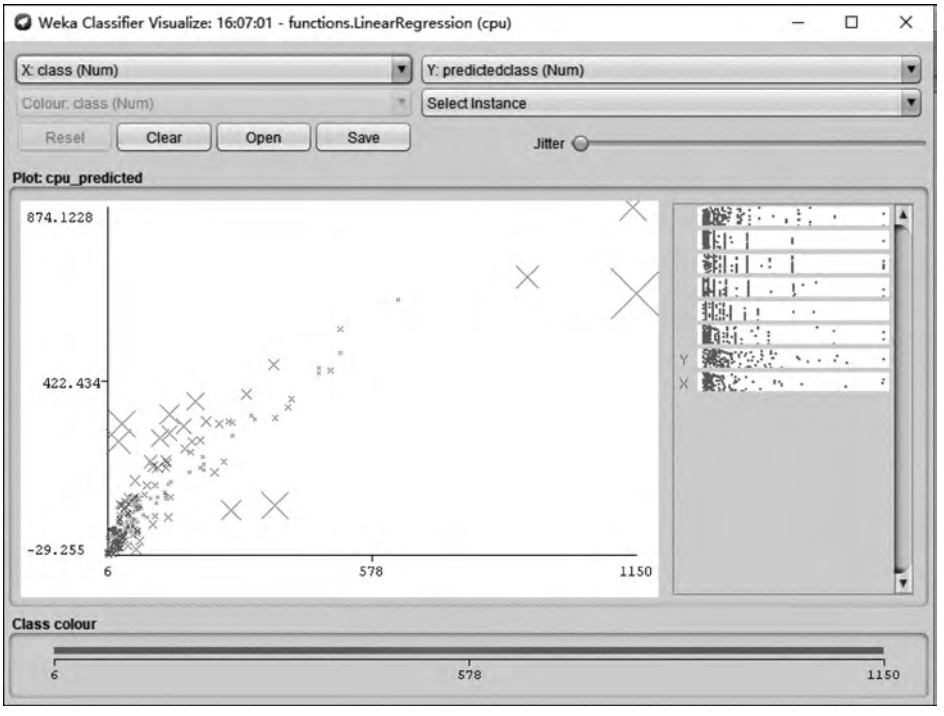


图 3.42　LinearRegression 可视化分类误差

这个线性回归函数中，分类 class 是线性函数的因变量，其他属性是自变量。平均绝对误差 MAE 41.0886 以及其他的性能指标值都表明，这个分类器在这个数据集上的性能无法满足需求。在数据挖掘的应用中，经常会碰到这种选用某种算法效果不好的情况，有可

能需要更换算法，或者调整参数。例如，如果选择下面介绍的 M5P 分类器，就会明显改善分类效果。

3.2.3 M5P 分类器

M5P 是决策树方案和线性回归方案的结合体。先构建决策树，后使用线性回归。

进入 Weka Explorer 界面，打开 CPU 数据集。切换到 Classify 标签页，单击 Choose 按钮，选择 trees 条目下的 M5P 分类器，如图 3.43 所示。选择 Cross-validation Folds，单击 Start 按钮。

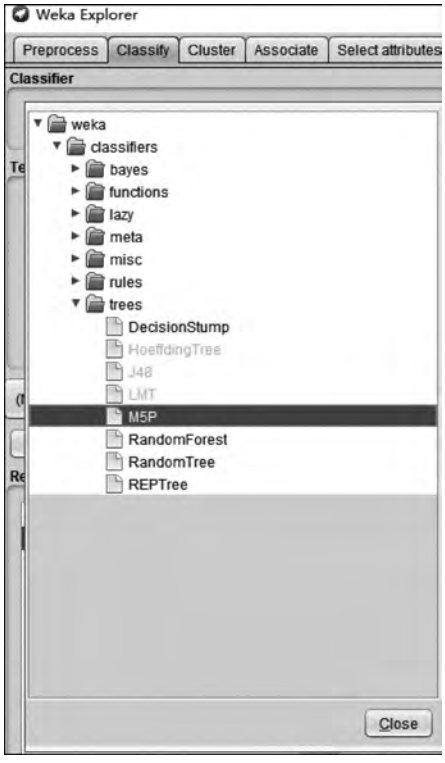


图 3.43 选择 M5P 分类器

在图 3.44 右侧的结果输出中，包括 5 个线性回归方程，LM1～LM5。M5P 的平均绝对误差(MAE)是 29.8309。

在 Result list 结果列表中，右击 trees.M5P 条目，在弹出的菜单中选择 Visualize tree(可视化树)命令菜单项，弹出决策树的可视化结果，如图 3.45 所示。

通过决策树，可以看到数据集中 6 个属性中有 4 个进行了分叉，一共产生 5 个叶节点，分支的每个叶节点对应一个线性回归方程。这意味着利用 M5P 分类器产生的线性模型不同的情况对应着不同的线性回归方程。叶节点括号中第 1 个数字代表到达该叶节点的实例个数，第 2 个数字代表使用对应线性模型的预测均方根误差。M5P 算法的可视化误差如图 3.46 所示。

简单比较 LinearRegression 和 M5P 两种分类器的性能。LinearRegression 和 M5P 的平均绝对误差(MAE)分别为 41.0886 和 29.8309，M5P 性能优于 LinearRegression。图 3.42

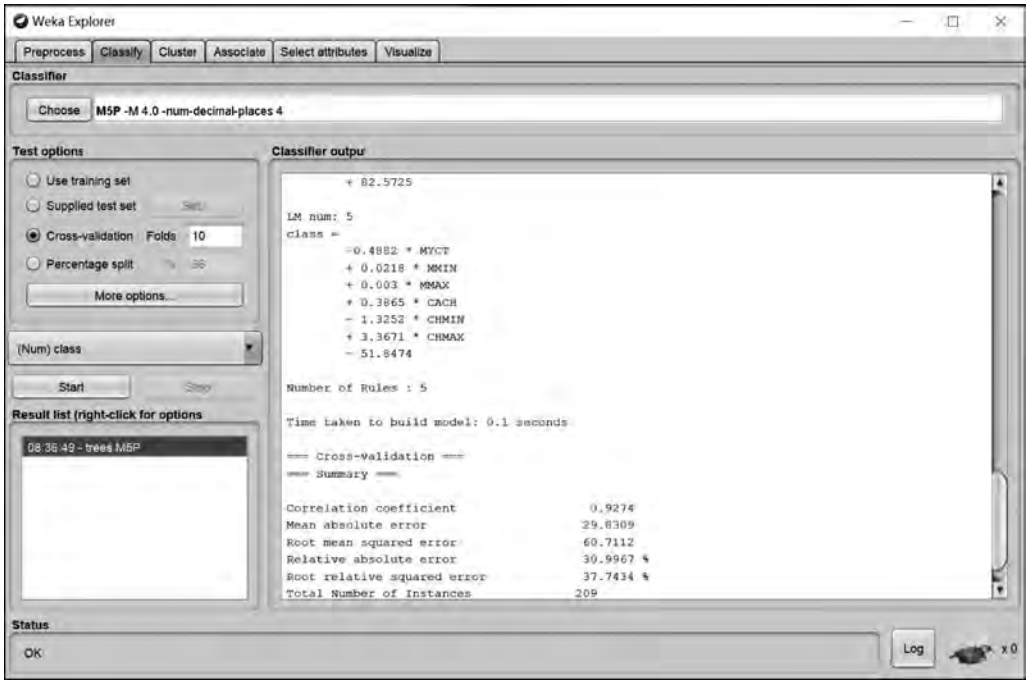


图 3.44　M5P 分类器分类结果

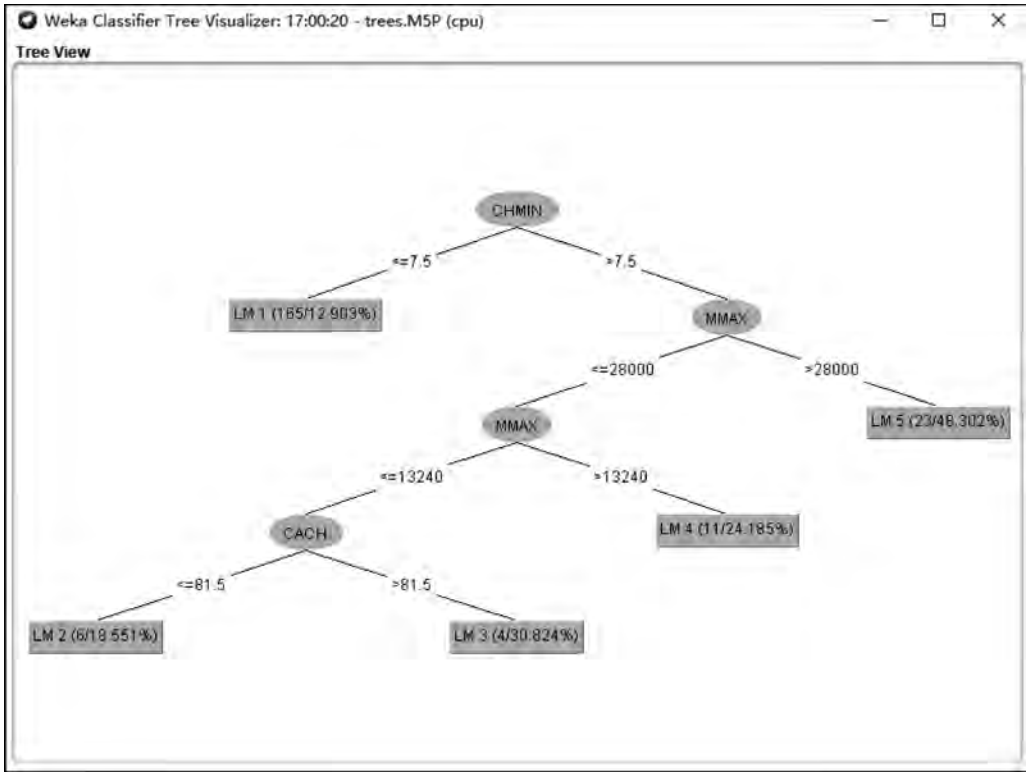


图 3.45　M5P 分类器可视化分类结果

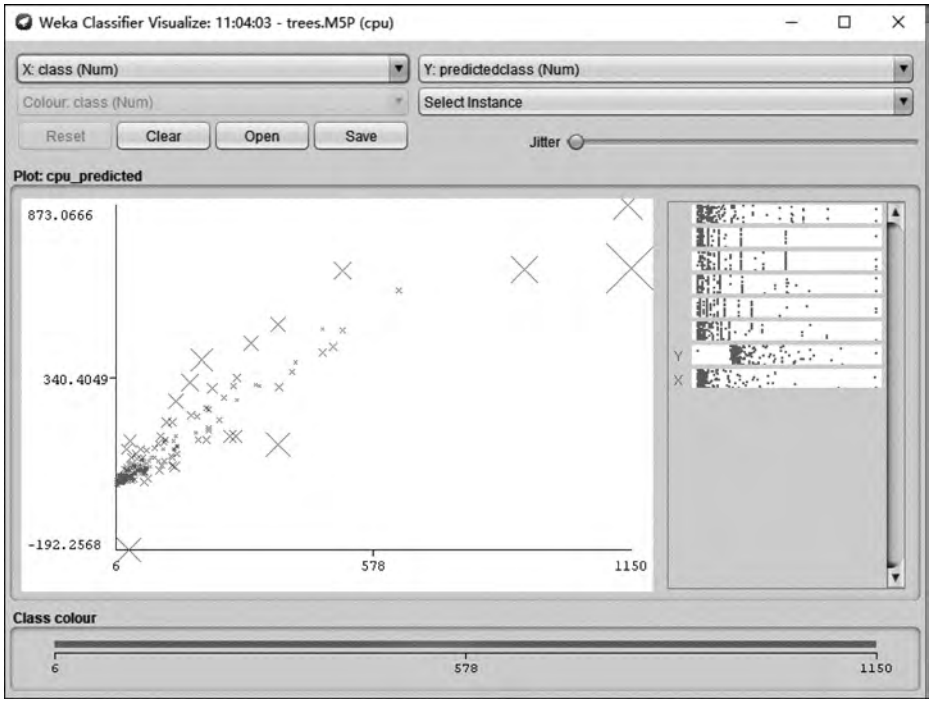


图 3.46　M5P 可视化分类误差

和图 3.46 分别是 LinearRegression 和 M5P 的可视化误差(visualize classify errors)结果。可视化误差图中,一个叉号代表存在一个误差,叉号大小代表误差的绝对值。LinearRegression 的叉号多于 M5P,因此,M5P 性能优于 LinearRegression。所以,在对 CPU 数据集进行分类预测的业务中,应该选择 M5P 分类器,而不使用 LinearRegression 分类器。

3.3　数据聚类

本节介绍数据聚类,主要内容分为两部分。第 1 部分概述数据聚类,简单介绍聚类的定义、聚类与分类的区别;第 2 部分介绍主要的聚类器及其操作,主要包括 3 个聚类器:SimpleKMeans、EM、DBSCAN。

什么是聚类呢?聚类是将数据集中相似的实例聚集在一起,而将不相似的实例划分到不同类别。在第 1 章中,形象地称之为“找朋友”。有一个由多个属性字段描绘的实例(也就是某个数据对象),聚类算法通过前期的算法训练得到的一个数学模型,将这些实例簇集成若干组,就像是俗话说的:“物以类聚,人以群分”一样。聚类对数据集进行分组,所生产的组称为簇(cluster)。簇内任意两个实例之间应该具有较高的相似度,而隶属于不同簇的两个实例之间应该具有较高的相异度。

聚类与分类是有区别的,其区别主要有以下两点。

(1) 分类是将数据集中的实例归结到某个已知的类别中,而聚类是将数据集中的实例聚集到某个预先不知的类别中。在聚类中,用户其实不太关心具体的类别是什么,而更关心的是哪些实例具有比较高的相似性,也就是属于一类。

(2) 分类是有监督学习,而聚类是无监督学习。

那么实例和实例之间如何度量相似度?最简单的方式是通过距离的远近来度量。假设将具有 n 个属性的实例当成是 n 维空间中的一个点的坐标,两个实例在这样一个 n 维空间中的距离就是它们的相似性强弱的度量值。

使用 Weka 进行聚类分两步。第 1 步是通过分析已知类别数据集,选择合适的聚类器进行训练;第 2 步是使用聚类器对新实例进行聚类。有可能需要根据聚类的效果,返回到第 1 步,重新选择另外的聚类器。现实中聚类有广泛应用,例如在商务上,聚类能帮助市场分析人员从客户信息库中发现不同的客户群体,从而针对不同的客户群体,制定不同的销售方案。

Weka 使用聚类器这个概念,它的任务是把所有的实例分配到若干簇,使得同一个簇的实例聚集在一个簇中心的周围,它们之间的距离比较近;而不同簇实例之间的距离比较远。本节介绍 3 个经典的聚类器,包括 SimpleKMeans、EM 和 DBSCAN。

3.3.1 SimpleKMeans 聚类器

SimpleKMeans 聚类器使用 k 均值算法或 k 中位数算法。SimpleKMeans 聚类器思想有直观的几何意义:将样本点聚集(归属)到距离它最近的那个聚类中心。算法的目标簇的数量由参数 k 指定,这 k 个聚类中心的坐标由与它相邻的若干个实例的坐标的距离平均值确定。这也是 SimpleKMeans 聚类器名字中间 K 和 Means 这两个元素的来源。

当选择 k 均值算法时,SimpleKMeans 聚类器使用欧氏距离度量相似度;当选择 k 中位数算法时,SimpleKMeans 聚类器使用曼哈顿距离度量相似度。

欧氏距离指在 n 维空间中两个点之间的真实距离。在二维和三维空间中的欧氏距离就是两点之间的实际距离,例如,两个点在二维空间中的欧氏距离 $d=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$,推广到更高维的空间中以后,其距离计算的方法仍然一样。而曼哈顿距离是两个点在标准坐标系上的绝对轴距总和。例如,两个点在二维空间中的曼哈顿距离 $d=|x_2-x_1|+|y_2-y_1|$,其计算方法同样可以推广到更高维的空间中。

SimpleKMeans 聚类器适用于处理标称型属性。本节继续以天气数据集为例介绍 SimpleKMeans 聚类器的使用。进入 Weka Explorer 界面,打开素材文件夹中数值型天气数据集“C:\Program Files\Weka-3-8-4\data\weather. numeric. arff”,注意是数值型天气数据集。切换到 Cluster 标签页,单击 Choose 按钮,选择 SimpleKMeans 聚类器,如图 3.47 所示。

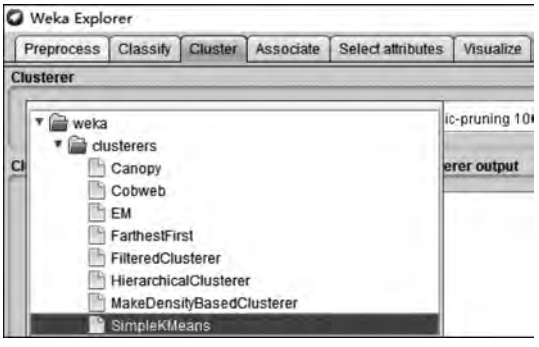


图 3.47 选择 SimpleKMeans 聚类器

在标签页单击 Choose 按钮右侧的文本框，弹出参数设置窗口，保持默认设置，即选择欧氏距离，numClusters 簇数目为 2，意味着要划分为两个类别；设置 seed 种子为 10，单击 OK 按钮，如图 3.48 所示。

图 3.48　SimpleKMeans 参数设置

回到 Cluster 标签页，选中 Use training set（使用训练集），勾选 Store clusters for visualization（存储聚类可视化）。单击 Ignore attributes（忽略属性）按钮，在弹出窗口中选择 play 属性，单击 Select 按钮，忽略类别属性，如图 3.49 所示。

单击 Start 按钮，运行结果如图 3.50 所示。

在图 3.50 所示的 Clusterer output（聚类器输出框）中，Within cluster sum of squared errors 缩写为 SSE，它是误差的平方数，用来度量聚类质量。SSE 值越小，表明聚类质量越高。本例 SSE 值是 11.23。Initial starting points (random)表示随机设定两个实例作为聚类的簇中心。Final cluster centroids 是算法反复调整簇中心，使得各实例点距离簇中心点

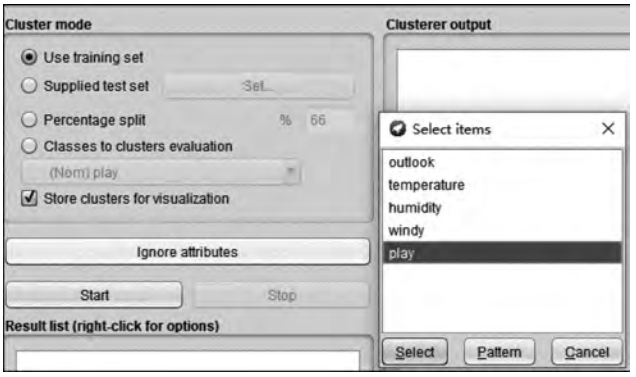


图 3.49　聚类模式设置

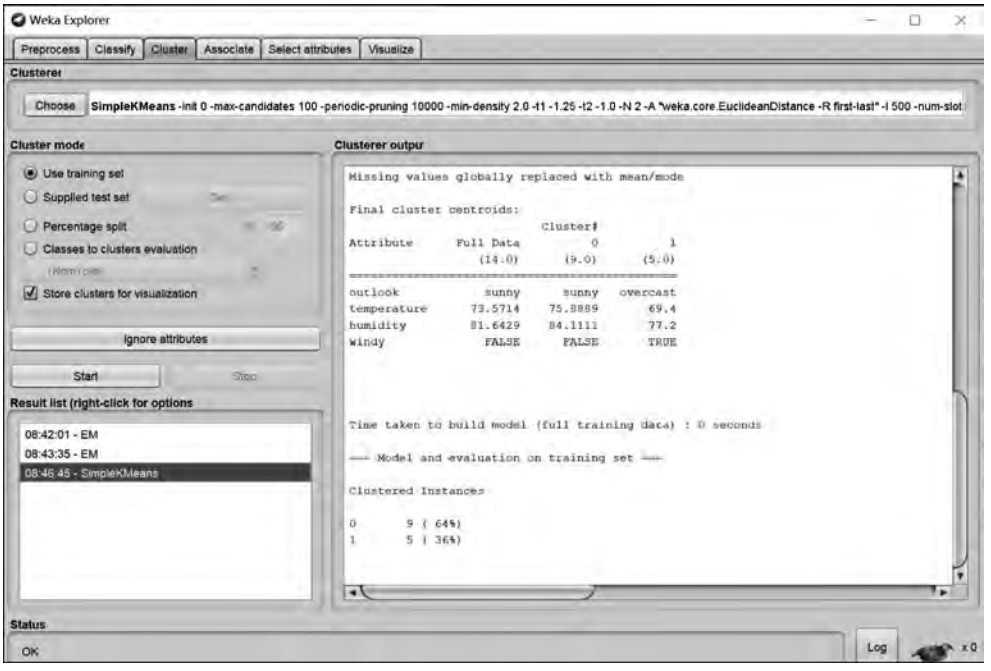


图 3.50　SimpleKMeans 聚类结果

的距离和最小。这就是算法找到的最终聚类中心。簇中心的每个值是如何计算出来的呢？标称型属性的簇中心是簇内数量最多的属性值；数值型属性的簇中心是簇内值的平均。Clustered Instances 是聚类后每个簇的实例数目以及百分比。第 0 个簇有 9 个实例，占比 64%；第 1 个簇有 5 个实例，占比 36%。

在 Result list(结果列表)中，右击 SimpleKMeans 条目，在弹出菜单中选择 Visualize cluster assignments(可视化簇分配)命令，如图 3.51 所示。

在弹出的对话框中，单击 Save 按钮，将簇分配的结果保存在桌面，文件名为 KM_Result 的.arff 文件，如图 3.52 所示。

在 Weka Explorer 中打开这个文件，单击 Edit 按钮，最后一列就是簇分类结果。可以非常清晰地看到每个实例是属于第 0 簇还是第 1 簇，如图 3.53 所示。

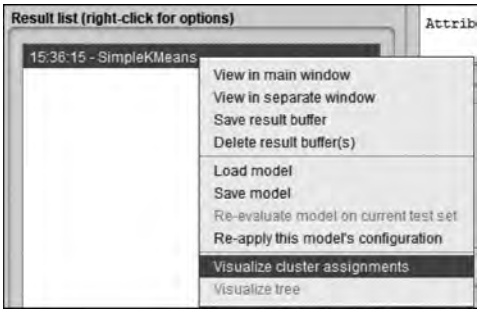


图 3.51　可视化簇分配

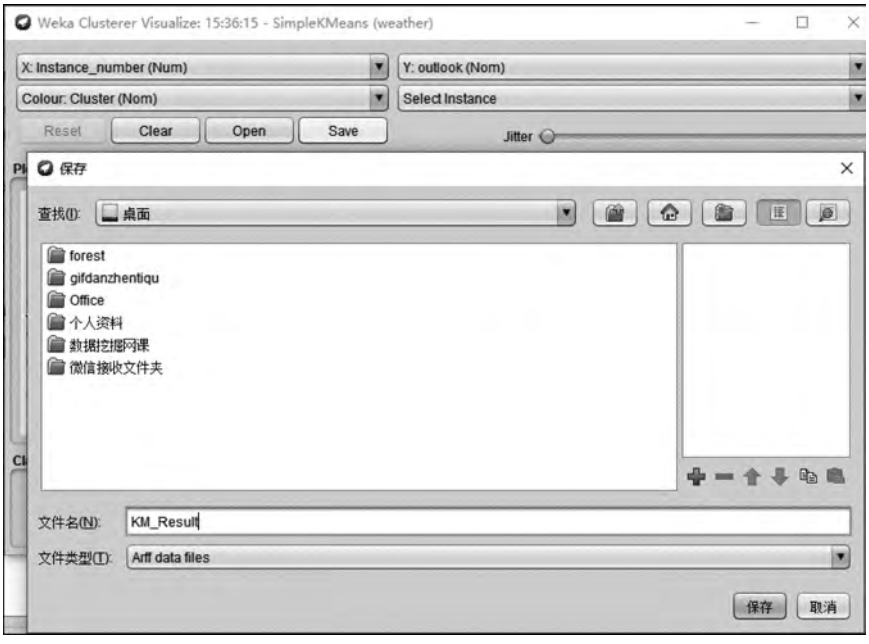
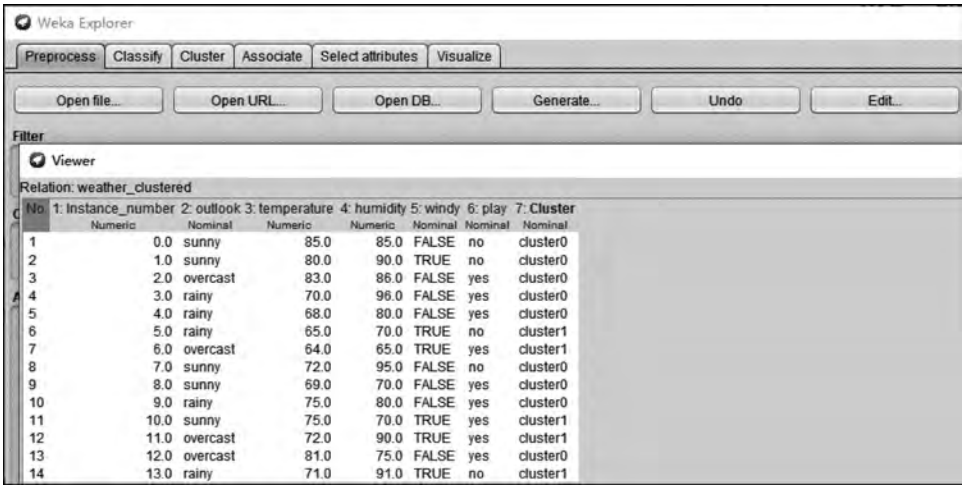


图 3.52　输出到文件



No.	1: Instance_number Numeric	2: outlook Nominal	3: temperature Numeric	4: humidity Numeric	5: windy Nominal	6: play Nominal	7: Cluster Nominal
1	0.0	sunny	85.0	85.0	FALSE	no	cluster0
2	1.0	sunny	80.0	90.0	TRUE	no	cluster0
3	2.0	overcast	83.0	86.0	FALSE	yes	cluster0
4	3.0	rainy	70.0	96.0	FALSE	yes	cluster0
5	4.0	rainy	68.0	80.0	FALSE	yes	cluster0
6	5.0	rainy	65.0	70.0	TRUE	no	cluster1
7	6.0	overcast	64.0	65.0	TRUE	yes	cluster1
8	7.0	sunny	72.0	95.0	FALSE	no	cluster0
9	8.0	sunny	69.0	70.0	FALSE	yes	cluster0
10	9.0	rainy	75.0	80.0	FALSE	yes	cluster0
11	10.0	sunny	75.0	70.0	TRUE	yes	cluster1
12	11.0	overcast	72.0	90.0	TRUE	yes	cluster1
13	12.0	overcast	81.0	75.0	FALSE	yes	cluster0
14	13.0	rainy	71.0	91.0	TRUE	no	cluster1

图 3.53　SimpleKMeans 聚类结果

本例中，已知 play 属性值，也就是每个属性的类别。可以借此来检验聚类器的性能。方法是：不选中 Use training set，而是选中 Classes to clusters evaluation(类别作为簇的评估准则)，比较聚类器所得到的簇与预先指定类别的匹配程度，如图 3.54 所示。

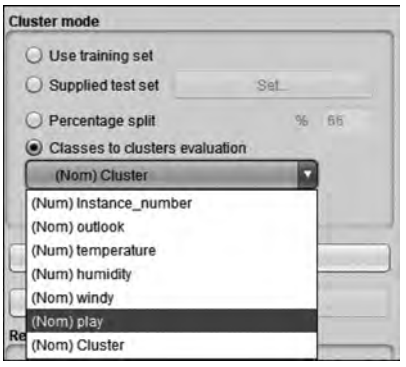


图 3.54　设置评估聚类结果

单击 Start 按钮，得到聚类结果，如图 3.55 所示。

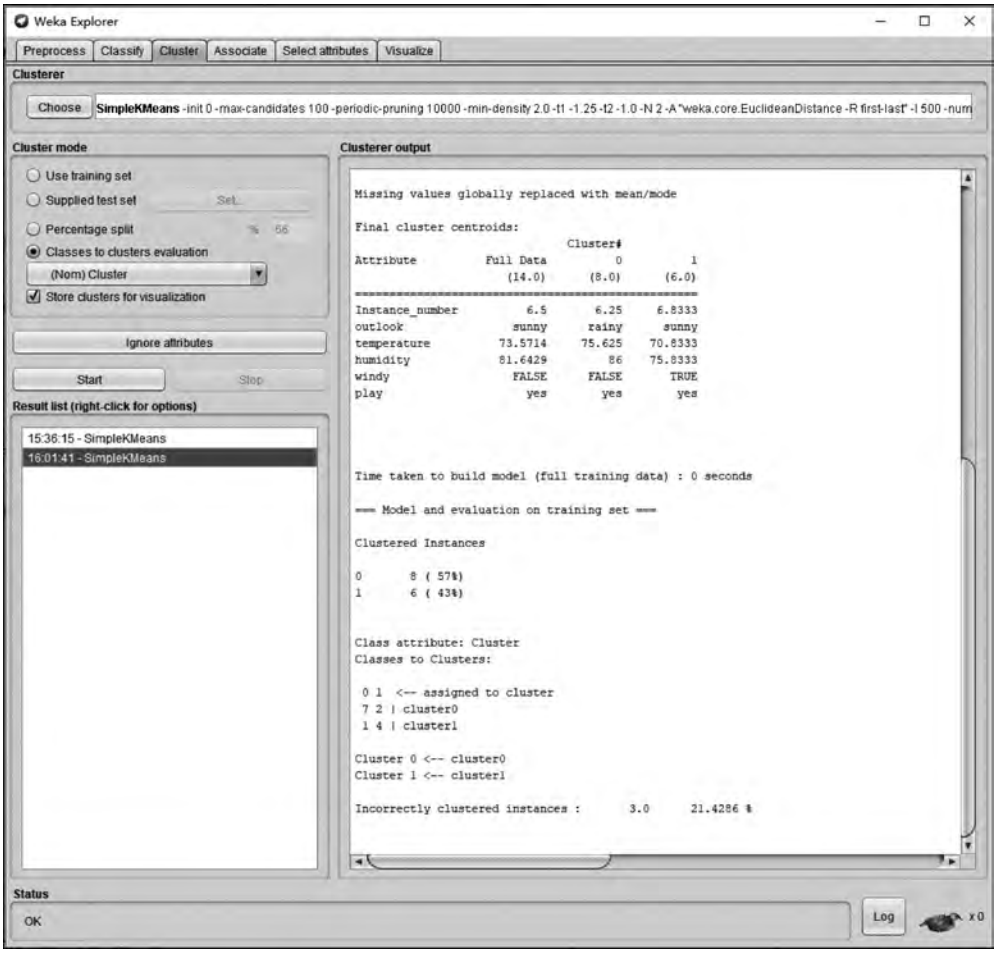


图 3.55　聚类结果评估

图 3.55 右下方 Incorrectly clustered instance 值为 3，表示有 3 个实例的聚类结果与原始结果不一致。另外，在图 3.48 的参数设置页面中，如果 seed 设置一个随机种子，产生一个随机数，对聚类的质量有比较大的影响。调整这个参数的值，可能会得到更好的聚类结果，但是也有可能会得到更差的聚类结果。

3.3.2 EM 聚类器

下面介绍另外一个重要的聚类器 EM(Expectation Maximization，期望最大化)。EM 聚类器使用 EM 算法，EM 算法是解决数据缺失时聚类问题的一种出色算法，它在概率模型中寻求最大似然估计或者最大后验估计，可以根据实例与簇之间隶属关系发生的概率来分配实例。

进入 Weka Explorer 界面，继续使用 3.3.1 小节使用的天气数据集。切换到 Cluster 标签页，单击 Choose 按钮，选择 EM 聚类器，如图 3.56 所示。

单击 Choose 按钮右侧文本框，弹出参数设置窗口，将 numClusters 簇数目设置为 2，意味着要划分为两个类别。其他参数保持不变。选中 Classes to clusters evaluation 类别作为簇的评估准则，单击 Ignore attributes 按钮，在弹出窗口中选择 play 属性，单击 Select 按钮，忽略 play 属性。以上步骤，可以参见图 3.48 和图 3.49。单击 Start 按钮，得到聚类结果如图 3.57 所示。

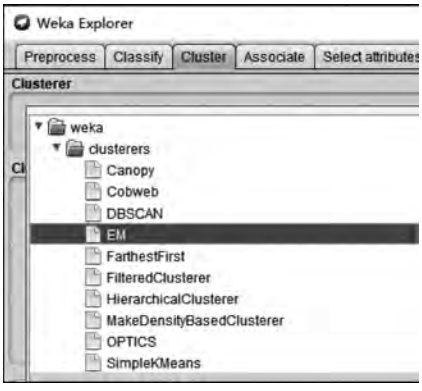


图 3.56 选择 EM 聚类器

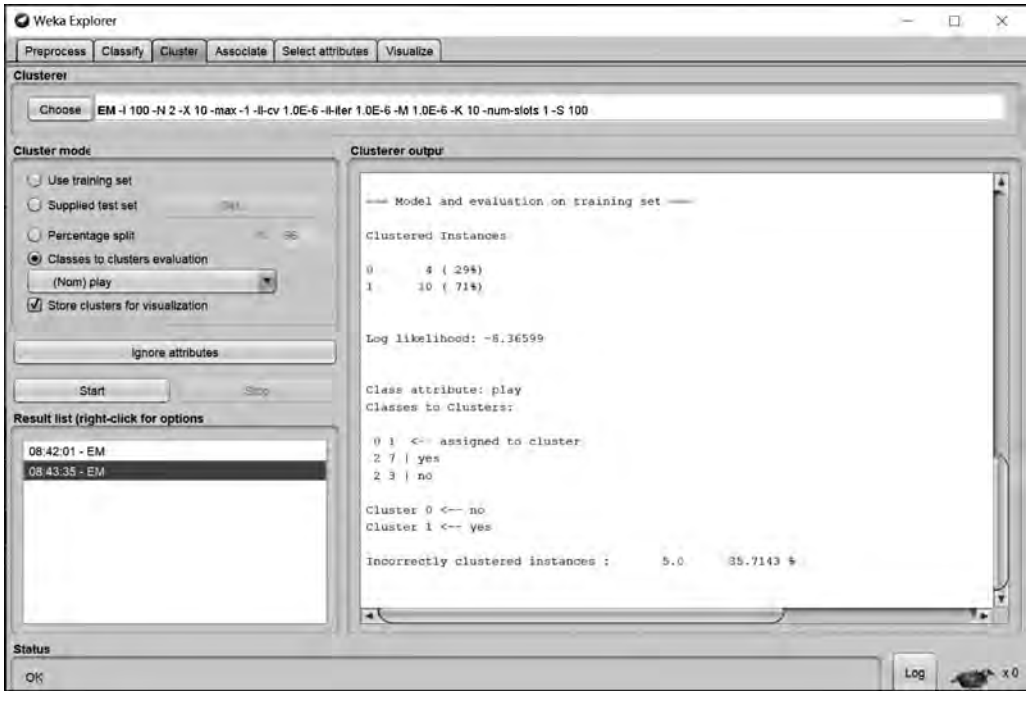


图 3.57 EM 聚类结果

图 3.57 右侧 Clustered Instance 是各个簇中实例的数目及百分比。第 0 个簇有 4 个实例，占比 29%；第 1 个簇有 10 个实例，占比 71%。Incorrectly clustered instance 输出了聚类器所得到的簇与预先指定类别的不匹配实例数目，一共有 5 个，占比 35.7%。查看或保存聚类的具体结果与图 3.50～图 3.52 记录的步骤类似，请参考执行。

3.3.3 DBSCAN 聚类器

DBSCAN 使用欧氏距离度量，以确定哪些实例属于同一个簇。但是，不同于 k 均值算法，DBSCAN 可以自动确定簇的数量。DBSCAN 聚类器属于包 optics_dbScan，最新版本为 1.0.5。

切换到 Cluster 标签页，如果找不到 DBSCAN 聚类器，需要另外安装 optics_dbScan 包。回到 Weka 主界面，依次选择 Tools→Package manager→optics_dbScan→Install 命令，如图 3.58 和图 3.59 所示。

图 3.58 包管理器

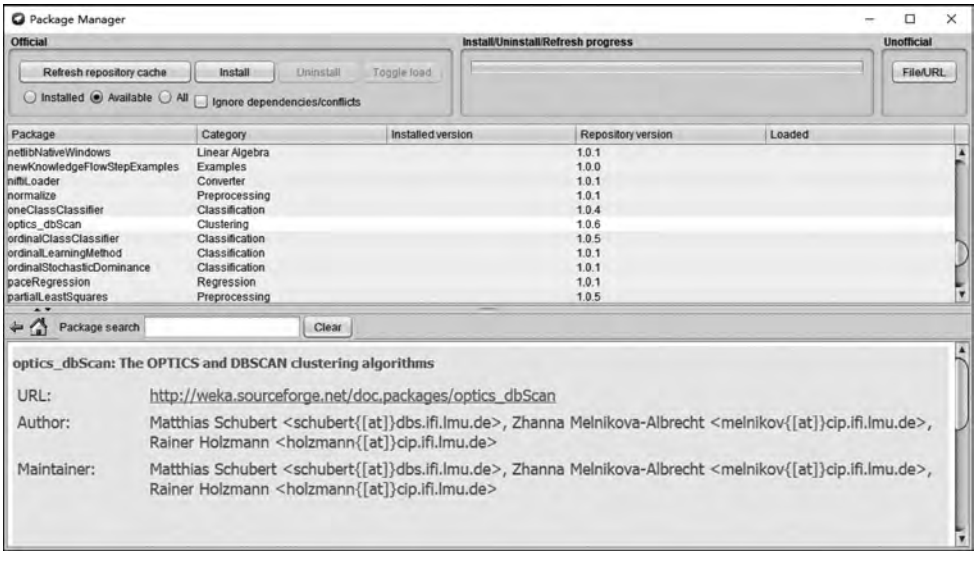


图 3.59 安装 optics_dbScan 包

DBSCAN 聚类器有两个重要参数，一个是参数 ε(epsilon)，它设定簇内点对之间的最小距离，ε 的值越小，产生的簇越密集，这是因为实例必须靠得更紧密，彼此才能同属于一个簇；另外一个参数是 minPoints，指簇内实例数的最小值。参数 ε 和 minPoints 的设定，对聚类的结果有很大的影响。根据设定的 ε 值和簇的最小值，有可能存在某些不属于任何簇的实例，这些实例称为离群值。下面以鸢尾花(iris)数据集为例，介绍 DBSCAN 聚类器的使用。

进入 Weka Explorer 界面，打开素材文件夹中鸢尾花数据集。切换到 Cluster 标签页，单击 Choose 按钮，选择 DBSCAN 聚类器，如图 3.60 所示。

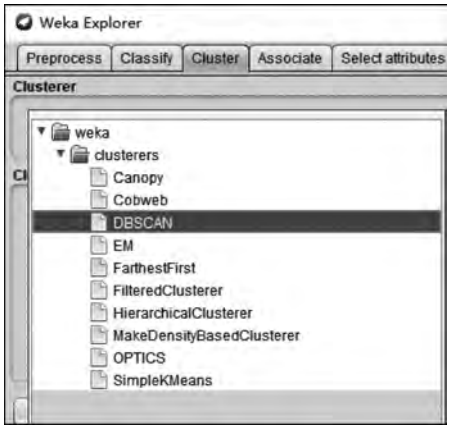


图 3.60　选择 DBSCAN 聚类器

单击 Choose 按钮右侧文本框，弹出参数设置窗口，将 ε 设置为 0.2，minPoints 设置为 5，其他参数保持不变，如图 3.61 所示。

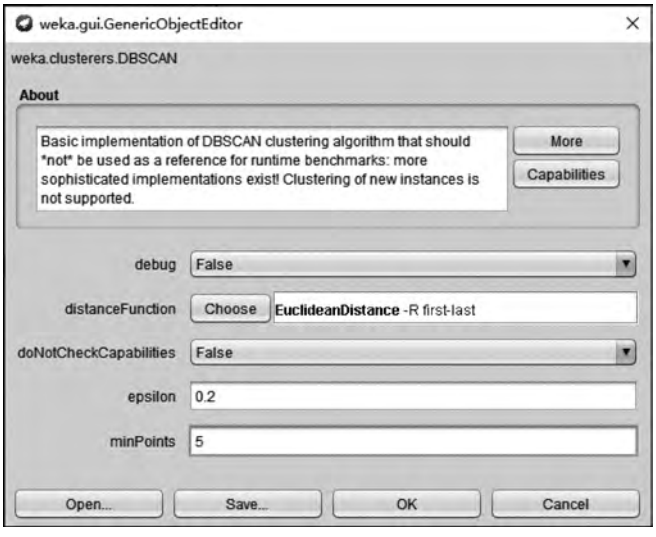


图 3.61　DBSCAN 参数设置

选中 Classes to clusters evaluation(类别作为簇的评估准则)，勾选 Store clusters for visualization(存储聚类可视化)，忽略 class 属性，如图 3.62 所示。

单击 Start 按钮，得到聚类结果，如图 3.63 所示。

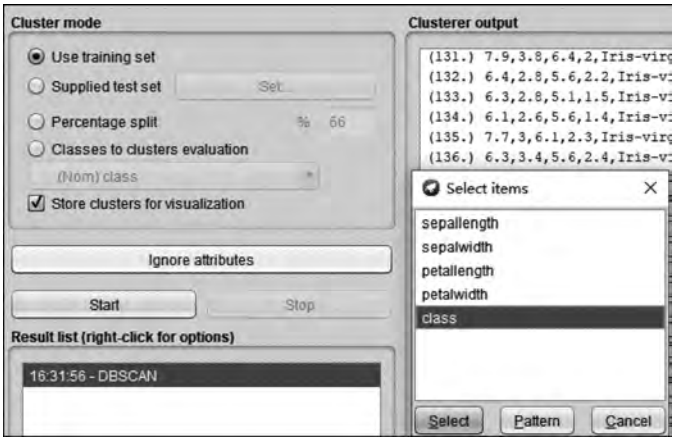


图 3.62　聚类模式设置

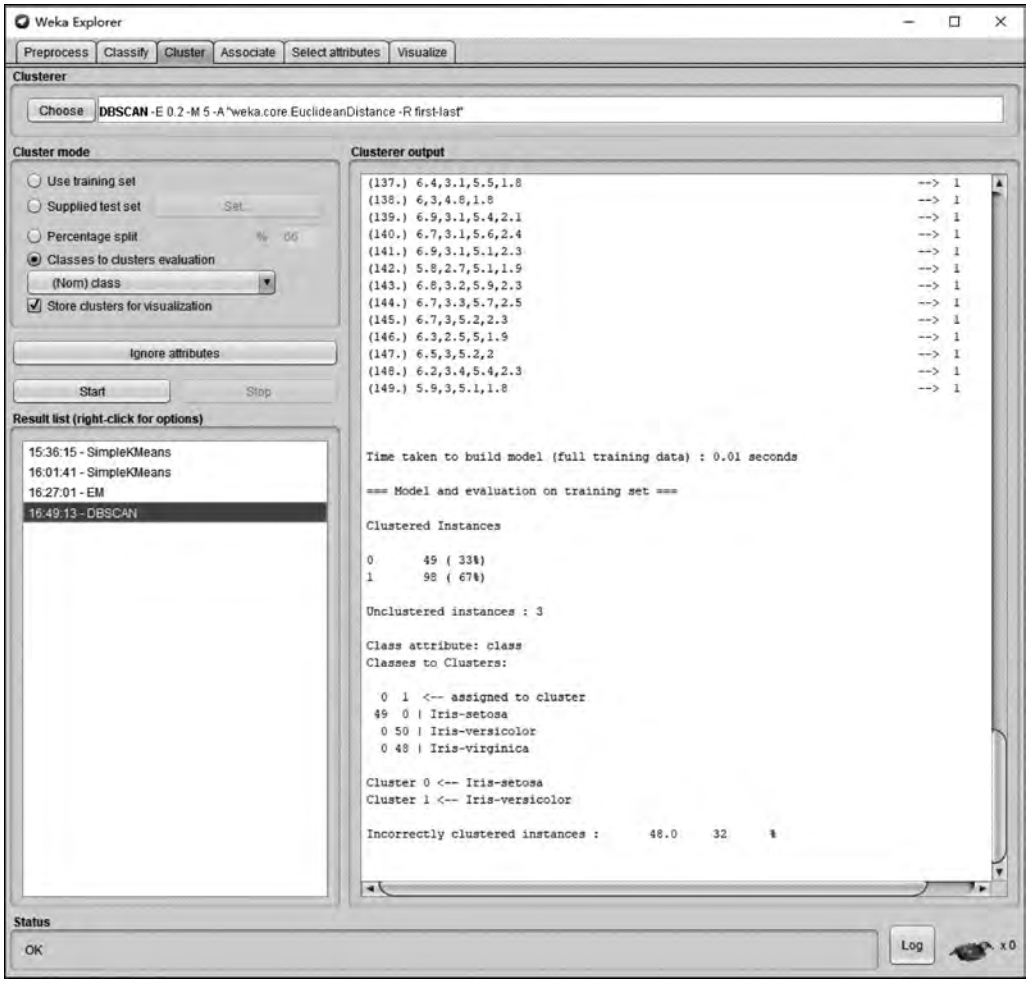


图 3.63　DBSCAN 聚类结果

图 3.63 右侧在 Clusterer output(聚类器输出)中可以看到,DBSCAN 只发现了两个簇,一个簇有 49 个实例,另一个簇有 98 个实例,还有 3 个实例未能聚类。未正确聚类的实例有 48 个,占比 32%。查看或保存聚类具体结果的方法与前述例子类似,读者可以自己尝试。

还可以将聚类的结果进行可视化输出。右击 Result list 中 DBSCAN 算法得到的结果,在弹出的菜单中选择 Visualize cluster assignments,如图 3.64 所示。

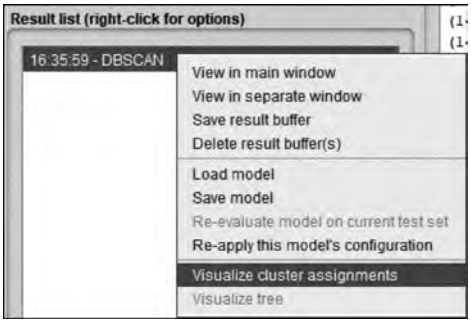


图 3.64 设置可视化聚类结果

可视化结果如图 3.65 所示。

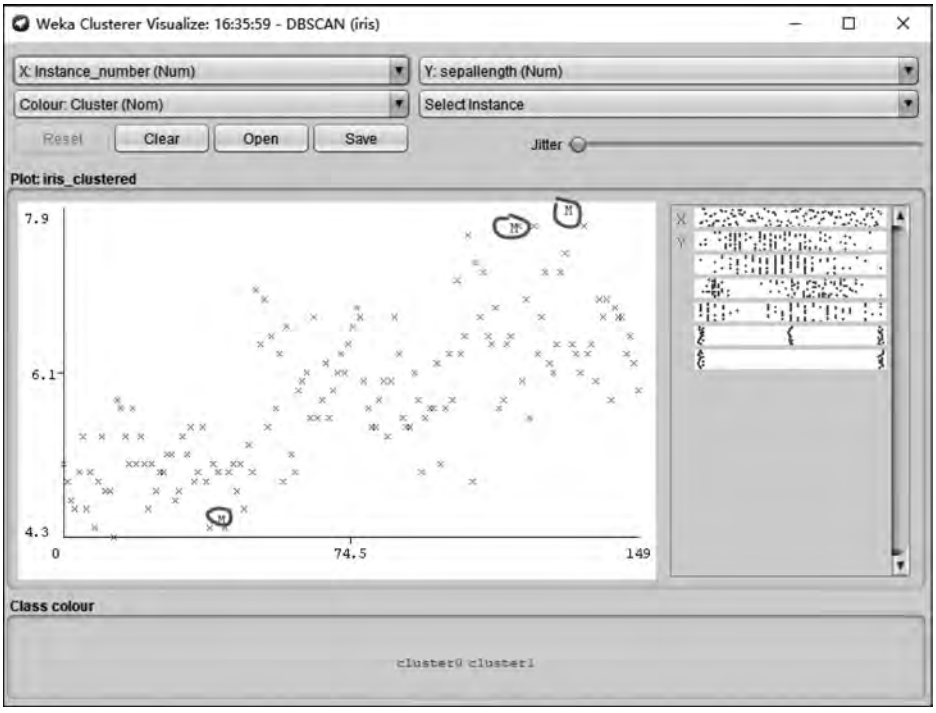


图 3.65 可视化聚类结果

因为 Weka 使用颜色来区分聚类结果,因此图 3.65 的灰度图像中难以查看两个不同的聚类结果信息,建议读者在 Weka 中自行验证。图 3.65 中单个圈注的 M 符号表示有 3 个实例未能聚类。

3.4 数据关联

随着数据库技术的飞速发展，快速增长的海量数据被收集并存放在一定的数据库中。从这些海量数据中分析并发现有用知识的数据挖掘技术目前已经发展得较为成熟。数据挖掘的任务是从大量的数据中发现模式。根据数据挖掘的任务分类，必然可以获知数据挖掘的方法也分很多种。

关联分析是当前数据挖掘研究的主要方法之一，关联规则描述了一组数据项之间的密切度或关系，它反映一个事物与其他事物之间的相互依存性和关联性。如果两个或者多个事物之间存在一定的关联关系，那么其中一个事物就能够通过其他事物预测到。

一个典型的使用关联规则发现问题的案例是超市中购物篮数据分析。通过发现顾客放入购物篮中的不同商品之间的关系来分析顾客的购买习惯。数据挖掘领域最经典的案例就是“啤酒与尿布”的故事。该故事中体现的就是关联规则的妙用和功力。美国的沃尔玛超市管理人员通过对超市一年多的销售数据进行详细的分析，发现了一个令人难于想象的现象，与尿布一起被购买最多的商品竟然是啤酒。借助于数据仓库和关联规则，商家发现了这个隐藏在背后的事实：每逢周五，美国的妇女们经常会嘱咐她们的丈夫下班以后去超市为孩子买尿布，而 30%～40%的丈夫在购买尿布的同时，会顺便购买自己爱喝的啤酒，因此啤酒和尿布就经常出现在一个购物车里了。根据发现的该事实，沃尔玛超市及时调整了货架的摆放位置，把啤酒和尿布摆在相邻的货架，最后经过一段时间的实施，发现二者的销售量都大大提高了。生活中其实还有很多类似的现象，例如 70%购买了牛奶的顾客将倾向于同时购买面包，买了一台 PC 之后人们习惯于继续购买音响、耳机、摄像头等，这些都是事物之间的关联现象。

3.4.1 关联规则相关概念

1. 支持度和置信度

关联规则挖掘的目的是在数据项目中找出所有的并发关系。关联规则可以采用与分类规则相同的方式产生。因为做数据挖掘时得到的关联规则数量可能很大，所以通常需要依据两个指标来对规则进行修剪，即支持度和置信度。

1）支持度

支持度又称覆盖率，通俗地说，即为几个相互关联的数据在数据集中出现的次数占总数据集的比重。给出一个形式化定义为，在 M 条交易集中，对于关联规则 R：$A \Rightarrow B$，其中 $A \subset I$，$B \subset I$，并且 $A \cap B = \varnothing$。规则 R 的支持度是交易集中同时包含 A 和 B 元素的交易数与所有交易数之比，其计算公式为

$$\text{Support}(A,B) = P(A,B) = \frac{\text{number}(AB)}{\text{number}(\text{AllSamples})}$$

其中，P 为概率。支持度是一种重要度量，低支持度的规则一般只是偶然出现，所以可以用支持度来删除无意义的规则。

2）置信度

置信度也称为准确率，指的是一个数据出现后，另一个数据出现的概率，或者说数据的

条件概率。即，在 M 条交易集中，对于关联规则 R：$A \Rightarrow B$，其中 $A \subset I$，$B \subset I$，I 是所有项集的集合，并且 $A \cap B = \varnothing$。规则 R 的置信度是指包含 A 和 B 的交易数与包含 A 的交易数之比。计算公式为

$$\text{Confidence}(A \Rightarrow B) = P(A \mid B) = \frac{P(AB)}{P(A)}$$

置信度一般用于度量规则，可用于推理的可靠性，因此置信度越高，推理越可靠。一般来说，只有支持度和置信度均较高的关联规则才是用户感兴趣的、有用的规则。

2. 项集、频繁项集及强关联规则

数据挖掘中最基本的模式是项集，它是指若干个项的集合。频繁模式是指数据集中频繁出现的项集、序列或子结构。频繁项集是指支持度大于或等于最小支持度的集合。频繁项集的经典应用就是前面提到的购物篮模型。一般来说，如果事件 A 中包含 k 个元素，那么称这个事件 A 为 k 项集；事件 A 满足最小支持度阈值的事件称为频繁 k 项集。关联规则表示的是在某个频繁项集的条件下推出另一个频繁项集的概率。如果该关联规则同时满足置信度大于或等于最小置信度阈值，则该规则被称为强关联规则。所以，在数据挖掘中一般找的是频繁项集和强关联规则。

3.4.2 Apriori 算法介绍

关联规则挖掘的目的就是发现隐藏在大型数据集中的数据之间的有价值的联系，这些联系可以采用关联规则的形式表示，所发现的联系，可用于医疗诊断、购物推荐和科学数据分析等领域。

最常用的一种关联规则挖掘算法是 Apriori 算法。Apriori 算法基于演绎原理，使用基于支持度的修剪技术来高效地产生所有频繁项集，控制项集的指数增长。算法基于逐级搜索的思想，采用多轮搜索的迭代方法，每一轮搜索扫描一遍整个数据集，最终生成所有的频繁项集。算法的基本思想如下所示。

(1) 找出频繁“1 项集”的集合。该集合记作 L1。L1 用于找频繁“2 项集”的集合 L2，而 L2 用于找 L3，“$k-1$ 项集”用于找“k 项集”。

(2) 如此下去，直到不能找到“k 项集”。找每个 Lk 需要一次数据集扫描。

(3) 最后利用频繁项集构造出满足用户最小置信度的规则。

Apriori 算法的原理是：如果某个项集是频繁项集，那么它所有的子集也是频繁的。即如果{0,1}是频繁的，那么{0}，{1}也一定是频繁的。所以，挖掘或识别出所有频繁项集是该算法的核心，占整个计算量的大部分。频繁项集的发现过程，一般经过如下步骤。

首先会生成所有单个物品的项集列表；然后，扫描交易记录并通过计数的方式来查看哪些项集满足最小支持度要求，那些不满足最小支持度的集合会被去掉，从而产生频繁项集；接着对剩下的集合进行组合，即通过连接、剪枝操作生成包含两个元素的项集；接下来重新扫描交易记录，去掉不满足最小支持度的项集，生成相应的频繁项集；最后重复进行以上几个操作直到不能发现更大频繁项集。

举例看一下 Apriori 算法的执行过程。现有 A、B、C、D、E 5 种商品的交易记录表(见表 3.2)，试找出 3 种商品关联销售情况($k=3$)，最小支持度=50%。

表 3.2　商品交易记录表

交　易　号	商 品 代 码	交　易　号	商 品 代 码
100	A、C、D	300	A、B、C、E
200	B、C、E	400	B、E

求解过程如下所示。

(1) 首先利用求解支持度的公式求出各 1 项集，即{A}、{B}、{C}、{D}、{E}的支持度，结果如表 3.3 所示。

表 3.3　1 项集的支持度

1　项　集	支持度/%
{A}	50
{B}	75
{C}	75
{D}	25
{E}	75

(2) 由表 3.3 可知 1 项集{D}的支持度为 25%，小于最小支持度阈值 50%，所以这时候去掉 D 商品，得出频繁 1 项集的列表，结果见表 3.4。

表 3.4　频繁 1 项集的支持度

频繁 1 项集	支持度/%
{A}	50
{B}	75
{C}	75
{E}	75

(3) 在剩余商品组合{A,B,C,E}中两两结合产生 2 项集{A,B}、{A,C}、{A,E}、{B,C}、{B,E}、{C,E}，同理继续利用公式计算支持度，结果见表 3.5。

表 3.5　2 项集的支持度

2　项　集	支持度/%
{A,B}	25
{A,C}	50
{A,E}	25
{B,C}	50
{B,E}	75
{C,E}	50

(4) 由表 3.5 可知 2 项集{A,B}、{A,E}的支持度为 25%，小于最小支持度阈值 50%，所以这时候去掉这两个组合，得出频繁 2 项集的列表，结果见表 3.6。

表 3.6　频繁 2 项集的支持度

频繁 2 项集	支持度/%
{A,C}	50
{B,C}	50
{B,E}	75
{C,E}	50

(5) 由剩余 4 个商品组合两两结合，生成 3 项集{A,B,C}、{A,C,E}、{B,C,E}，并继续利用公式计算支持度，结果见表 3.7。

表 3.7　3 项集的支持度

3　项　集	支持度/%
{A,B,C}	25
{A,C,E}	25
{B,C,E}	50

(6) 由表 3.7 可知 3 项集{A,B,C}、{A,C,E}的支持度为 25%，小于最小支持度阈值 50%，所以这时候去掉这两个组合，得到频繁 3 项集{B,C,E}，支持度为 50%，等于题目要求的最小支持度阈值 50%，所以最后得出题目的结果即为 B、C、E 3 种商品可以一起关联销售。

Weka 软件提供了 Apriori 算法的实现类，所以不需要写代码，只需要选择相应的算法，通过减少最小支持度进行迭代，直到找到所需数量的并且满足给定最小置信度的规则。

3.4.3　Weka 中 Apriori 关联规则挖掘

1. Associate 标签页

Weka 中通过 Associate(关联)标签页来实现数据关联问题，如图 3.66 所示。

Associate 标签页中包含了学习关联规则的方案。从图中可以看到，在 Associator 栏中设置关联规则学习器，可以通过单击 Choose 按钮进行选择，采用与前面章节中的聚类器、分类器等相同的方式来进行选择和配置。只不过这里点开后一般自动选择了 Apriori 算法。单击文本框，会弹出 Weka 的通用对象编辑器 GenericObjectEditor，可以对 Apriori 算法的各个参数进行相应的设置和修改，后面的案例分析中将对这些参数做出具体的解释和说明。

Weka 关联规则挖掘的一般步骤是，首先选择合适的关联规则学习器，并为关联规则学习器设置好合适的参数，然后单击 Start 按钮就可以启动学习器，学习完成后可右击 Result list 结果列表中的条目，从而查看或保存结果。

2. Apriori 关联规则挖掘

为了更好地理解并应用 Apriori 算法，在 Weka 软件中通过加载案例数据运行该算法并挖掘规则。这里要注意一个问题，Apriori 算法一般要求的是完全标称型数据，如果案例中有数值型属性，必须先进行离散化操作。

在 Preprocess 标签页加载 weather.nominal.arff 文件，切换至 Associate 标签页，并在

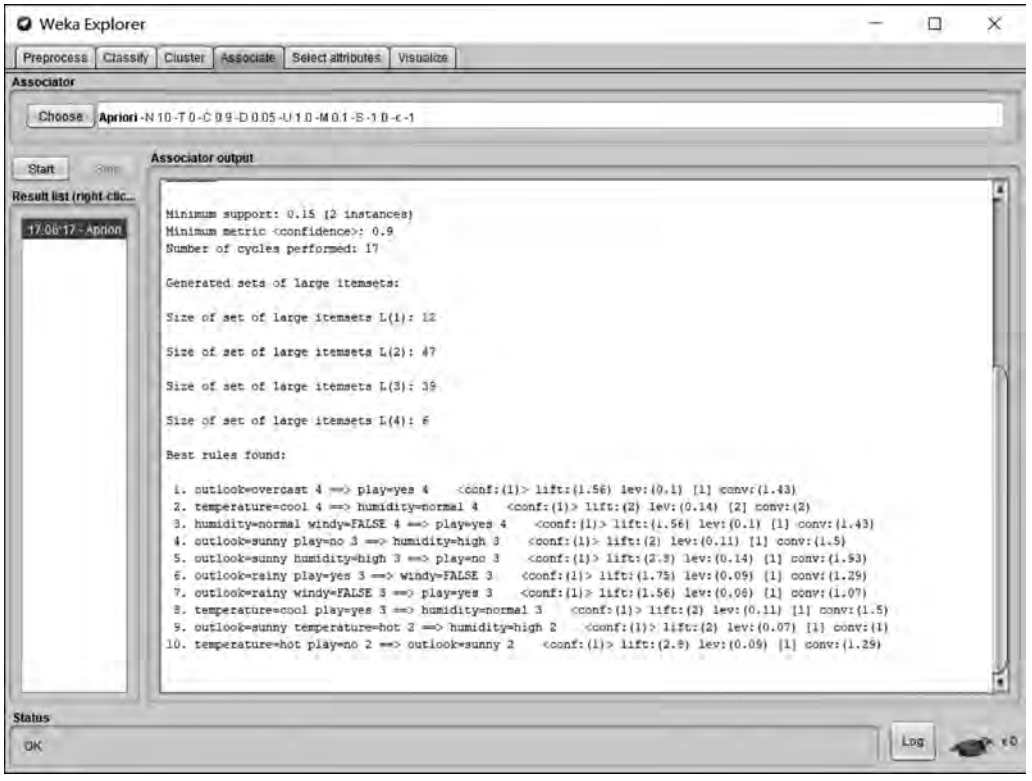


图 3.66　Associate 标签页

Associator 栏中选择 Apriori 算法，其他各参数先按照默认设置，接下来单击 Start 按钮，启动 Apriori 运行，结果如图 3.66 所示。

在图 3.66 中，可以看到，Result list 下面的列表中显示的是每次运行算法的时间点记录。右侧的 Associator output 区域中列出的是本次算法运行的结果。默认情况下，一次算法的运行会输出最优的 10 条规则，并按照每条规则后面尖括号中的置信度值进行排序。拿出其中的第 1 条规则解释，其规则如下：1. outlook＝overcast 4 ==> play＝yes 4 <conf:(1)> lift:(1.56) lev:(0.1)[1] conv:(1.43)。所有规则采用“前件 数字==>结论 数字”的形式表示，前件后面的数字表示有多少个实例满足前件，结论后面的数字表示有多少个实例满足整个规则，这就是规则的支持度。因为在结果给出的 10 条规则中，这两个数字相等，所以可以得出每个规则的置信度都为 1。在这 10 条规则的上面结果区，还给出了算法运行后达到的几个参数值，分别为如下。

- Minimum support(最小支持度)：0.15
- Minimum metric < confidence >(最小置信度)：0.9
- Number of cycles performed(为产生规则算法实际运行的次数)：17
- Generated sets of large itemsets：

```
Size of set of large itemsets L(1):12
Size of set of large itemsets L(2):47
Size of set of large itemsets L(3):39
Size of set of large itemsets L(4):6
```

这表示达到最小支持度 0.15 后，产生的各个频繁项集分别为：12 个大小为 1 的项集、47 个大小为 2 的项集、39 个大小为 3 的项集、6 个大小为 4 的项集。

前面提到，在实践中，需要通过最小支持度和置信度两个指标的衡量得出满意的结果。在 Weka 中，这一切是通过多次运行 Apriori 算法来得到的。当然，在执行算法之前需要用户指定最小置信度等参数的值。下面通过打开 Weka 软件的通用对象编辑器 GenericObjectEditor，如图 3.67 所示，研究一下 Apriori 算法中的各个参数的意义及如何设置。

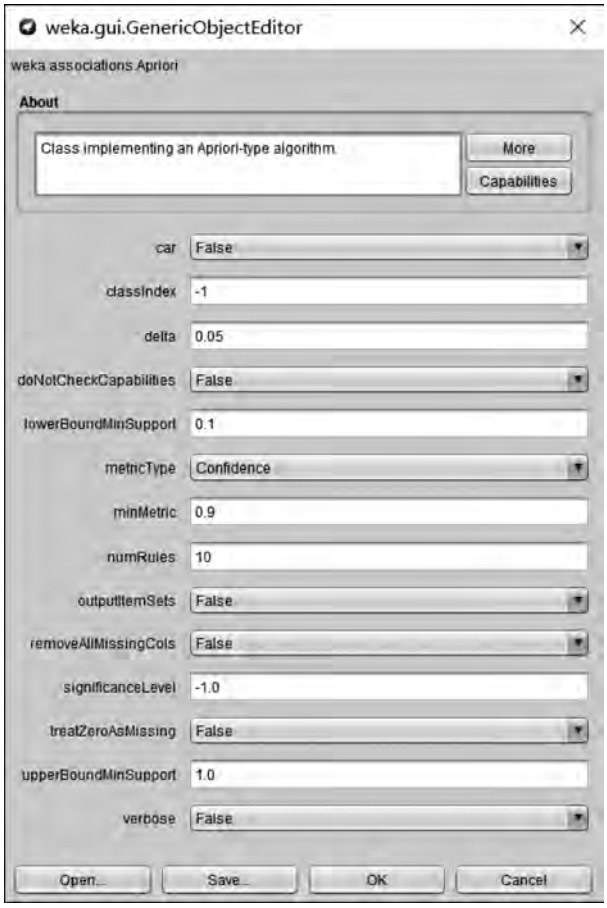


图 3.67　Apriori 算法的通用对象编辑器

针对跟实际应用相关性比较大的几个参数，解释如下。

- car 和 classindex 参数，跟类关联规则挖掘和类属性相关，这里不展开。
- delta，支持度变化的单位。以此数值为迭代递减单位，不断减小支持度直至达到最小支持度或产生了满足数量要求的规则。
- lowerBoundMinSupport，最小支持度下界。
- metricType，度量类型。设置对规则进行排序的度量依据。可以是置信度(confidence)、提升度(lift)、杠杆率(leverage)、确信度(conviction)。在 Weka 中可以通过 metricType 下拉列表框来设置这几个类似置信度的度量，从而衡量规则的关联程度。除置信度之外的其他几个标准在这里不做解释。

- minMtric，所选中的度量类型的最小值。
- numRules，结果中想要发现的规则数。
- outputItemSets，如果设置为 TRUE，会在结果中输出各项集的具体内容。
- upperBoundMinSupport，最小支持度上界。从这个值开始，以 delta 为单位，迭代减小最小支持度。

现在，更改通用对象资源管理器里的几个参数，比如设置 outputItemSets 为 True，numRules 为 5，minMetric 为 0.95，再次运行 Apriori 算法，观察一下结果有何不同，如图 3.68 所示。

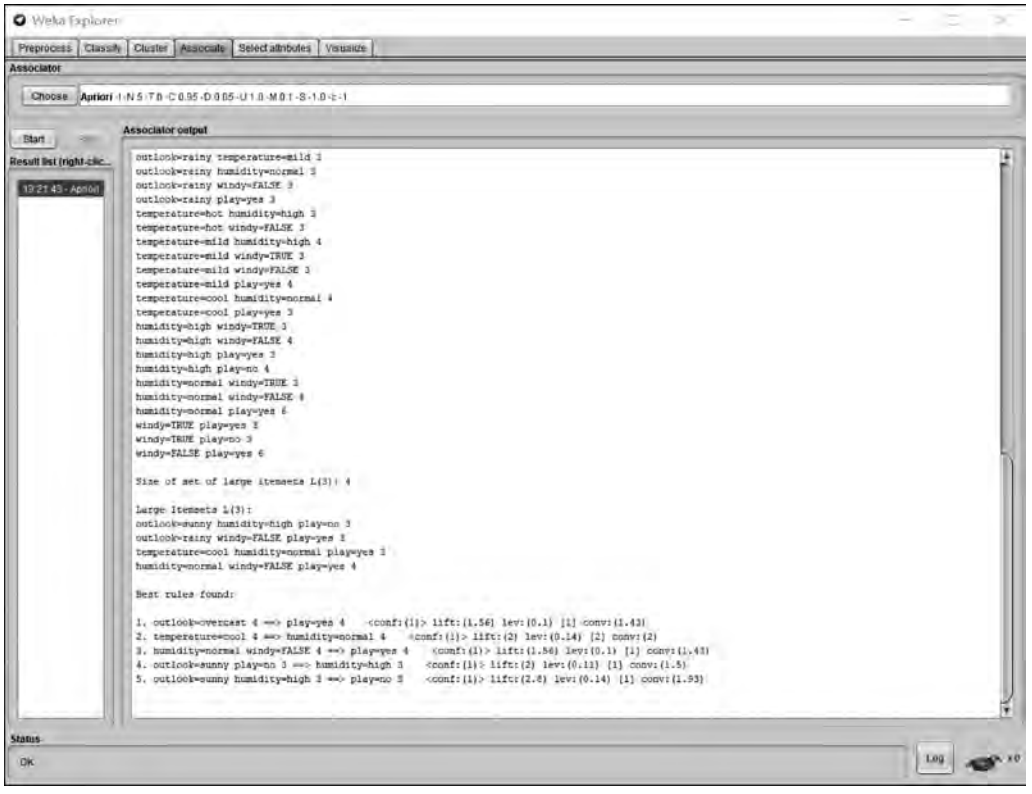


图 3.68　设置不同参数后的运行结果

Apriori 算法还有其他一些参数，更多信息可以通过在通用对象编辑器中单击 More 按钮获得。

3. 挖掘其他数据集

本案例挖掘乳腺癌的相关数据集，该数据集是从斯洛妮亚卢布尔雅那大学医疗中心乳腺癌肿瘤研究所获得的。数据集中一共有 286 个实例，9 个属性加 1 个类别属性。本来该数据集更多地用于分类问题，目的是可以根据病人的各项身体指标预测其癌症是否会复发。这里应用关联规则挖掘，看看能不能发现一些有趣的关联性，从而为病人的检查或医生的诊断提供有价值的建议。本数据集中所有的属性都被处理为标称型属性，并且有些属性具有一定的数据缺失。

首先在 Preprocess 标签页中加载 breast-cancer.arff 数据集，切换至 Associate 标签页，选择 Apriori 算法，保持默认选项不变，单击 Start 按钮，运行结果如图 3.69 所示。

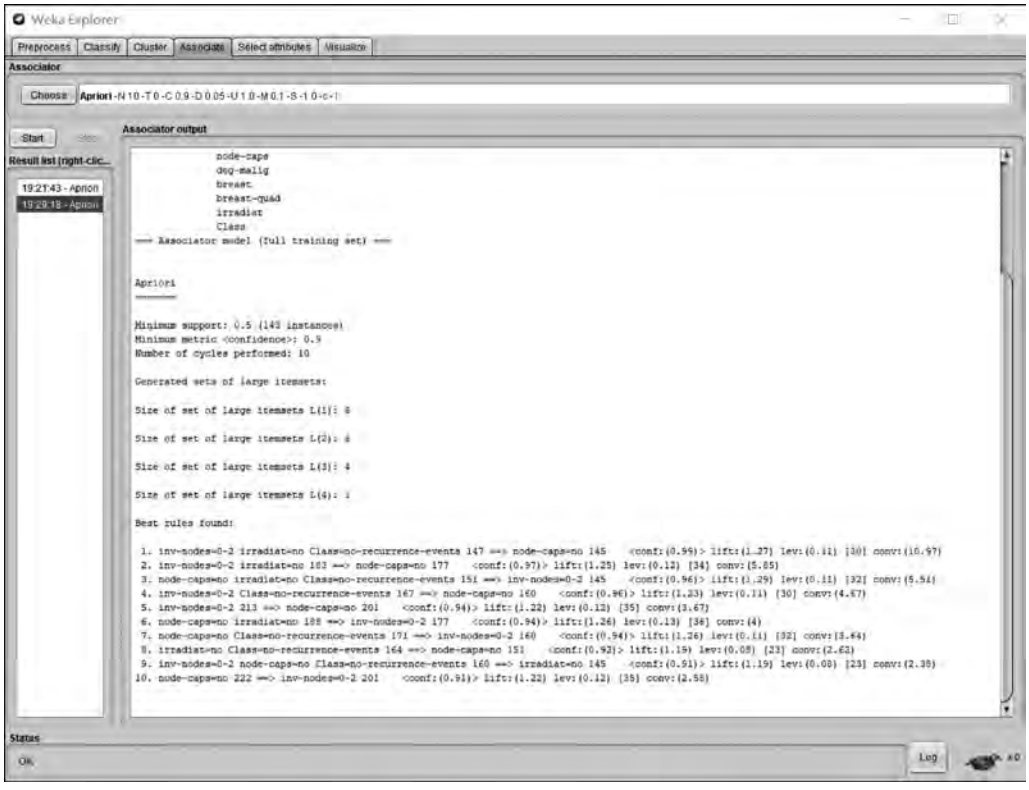


图 3.69　breast-cancer 数据集运行 Apriori 算法的结果

从图 3.69 的运行结果可以看到，输出的最小支持度达到 0.5，有 143 个实例最小置信度为 0.9，执行了 10 次迭代。达到最小支持度 0.5 时，产生的各个频繁项集分别为 6 个大小为 1 的项集、6 个大小为 2 的项集、4 个大小为 3 的项集、1 个大小为 4 的项集。

挖掘产生的 10 条规则如下。

- 第 1 条规则：受侵淋巴结数＝0～2，未放疗，无复发＝＝>无结节帽。
- 第 2 条规则：受侵淋巴结数＝0～2，未放疗＝＝>无结节帽。
- 第 3 条规则：无结节帽，未放疗，无复发＝＝>受侵淋巴结数＝0～2。
- 第 4 条规则：受侵淋巴结数＝0～2，无复发＝＝>无结节帽。
- 第 5 条规则：受侵淋巴结数＝0～2 ＝＝>无结节帽。
- 第 6 条规则：无结节帽，未放疗＝＝>受侵淋巴结数＝0～2。
- 第 7 条规则：无结节帽，无复发＝＝>受侵淋巴结数＝0～2。
- 第 8 条规则：未放疗，无复发＝＝>无结节帽。
- 第 9 条规则：受侵淋巴结数＝0～2，无结节帽，无复发＝＝>未放疗。
- 第 10 条规则：无结节帽＝＝>受侵淋巴结数＝0～2。

在这 10 条关联规则中，有 3 个指标是多次同时出现，而且实例数非常多。这给出了一条比较符合实际的结论：如果一个病人的受侵淋巴结小于两个，并且无结节帽，未做过放

疗，那这个病人复发的可能性就比较低。也就是说这四者之间的关联性非常大。

对于医生来说，这就是一条重要的参考信息，可以根据挖掘到的规则来给病人做出合理的判断，或者当病人来到医院检查时，也可以关注这几项指标，让整个看病过程更快捷。

最后建议读者试着借助于通用对象编辑器，修改 Apriori 算法的不同参数，针对以上数据集进行挖掘，看看挖掘效果，尝试能否从中分析出一些意外而又在情理之中的结论。

3.5 选择属性

事物的属性从多个角度描述了事物，然而有的属性对于目标是不重要的，或者起了反作用，这就需要从众多属性中，把不重要的识别出来，保留重要的属性。数据挖掘中，在考虑挖掘模型拟合效果、系统运行时间等前提下，对某些数据集来说，构建的包含所有或大多数属性的模型并不一定是最优模型。原因在于数据集中存在的跟挖掘任务不相关的属性或冗余的属性，可能会导致无效的挖掘过程或降低挖掘的效率。很多研究表明，一些常见的算法可能会因为不相关或冗余的数据而产生低质量的结果，甚至于冗余的属性也可能直接影响某些分类算法的表现。

因此我们需要对所有的属性进行甄别，在数据挖掘操作之前选择合适有效的属性。大量的实证研究表明，属性选择对于提高挖掘的效率以及挖掘结果的准确性是非常有效的，所以要重视属性的选择问题。

3.5.1 属性选择概述

1. 属性选择的定义

属性选择一般属于数据挖掘过程中的数据预处理阶段的工作。面对大量的高维数据，从理论上来说，当然是属性数目越多越有利于目标的分类，但实际情况却并非如此。一般在选取的样本数目有限的情况下，用很多属性去对数据做分类，设计分类器，从计算的复杂性或分类器的性能角度来说都是不适宜的。而样本的属性中既包括有效属性，又包括噪声属性，还包括问题无关属性以及冗余属性。很明显我们希望保留有效属性，尽可能地去掉噪声属性，对于无关属性或冗余属性也希望尽量减少到最少。

属性选择的目的就是从属性集中删除不具有预测能力或预测能力极其微弱的属性，从而建立高效的挖掘模型。属性选择也是对高维数据进行降维的一种有效方法。当然，在不同的应用问题中，属性选择的目标或标准也不一样。在本小节内容中，只研究数据挖掘中的属性选择问题。下面给出一个最典型的定义。

属性选择是指在属性个数为 n 的属性集合中，选择 m 个属性成为一个属性子集，其中 $m<n$，要求在所有属性个数为 m 的属性子集中，该子集的评估函数结果最优。

在数据挖掘的研究中，通常使用距离来衡量样本之间的相似度，而样本距离是通过属性值来计算的。因为不同的属性在样本空间的权重是不一样的，即它们与类别的关联度是不同的，所以有必要筛选一些属性或者对各个属性赋一定的权重，然后搜索数据集中全部属性的所有可能组合，找出预测效果最好的那一组属性。

2. 属性选择的关键因素

根据属性选择的目的和定义，在做属性选择的时候一般要考虑如下 3 个问题。

(1) 如何选择属性或属性子集?

(2) 如何判断一个属性或属性子集对于预测结果是最优的?

(3) 按照什么标准对属性进行选择和排序?

这些应该是在做属性选择时必须考虑的问题。解决了这些问题,属性选择就完成了。很显然,在某些情况下,手工选择属性是可选方法之一。但是,手工选择属性的缺陷在于,选择过程既烦琐又容易出错。所以有必要借助于一定的软件实现属性选择的自动化。要实现自动选择属性就需要考虑如何借助于计算机解决上面的 3 个问题。所以,在自动选择属性时设定两个对象:一个是属性评估器,即用什么样的方法给每个属性赋予一定的评估值,而该评估值能决定该属性的重要性;另一个是搜索方法,即当设定了判断每个属性的重要程度的标准后,采取什么样的方法或风格在整个属性集中搜索。

3.5.2 Weka 中 Select attributes 标签页

Weka 软件专门提供了如图 3.70 所示的 Select attributes 标签页,可以帮助用户实现选择属性自动化。结合前面提到的用于自动选择属性的两个对象,属性评估器在该标签页中的 Attribute Evaluator 选项组中设置,而搜索方法在 Search Method 选项组中设置。如图 3.70 所示,简单介绍一下这两个选项组的内容及设置。

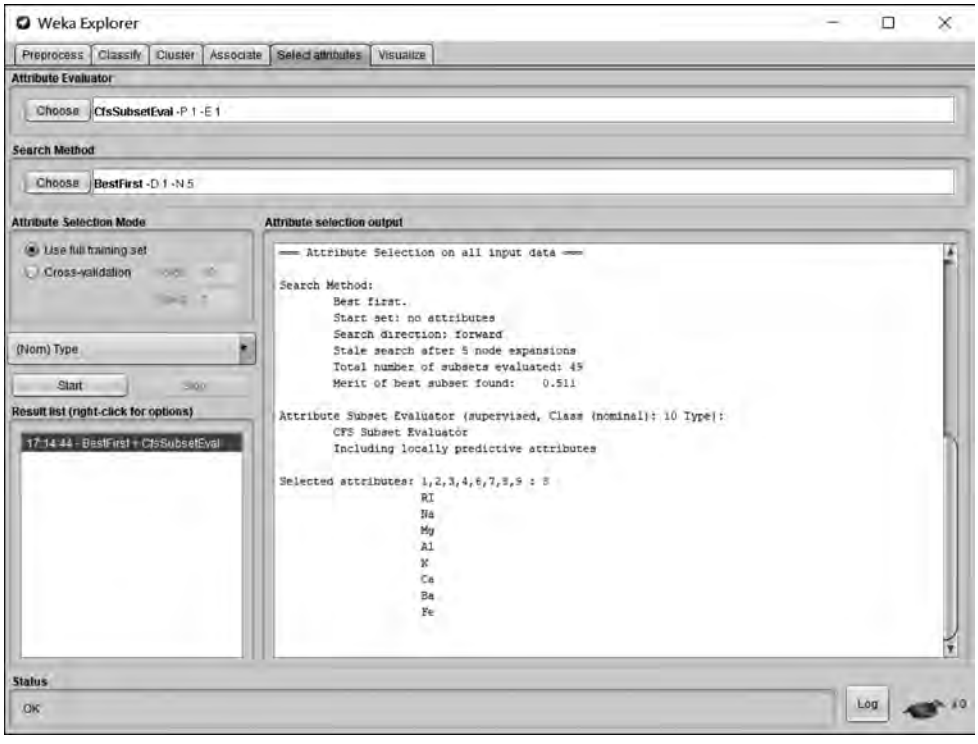


图 3.70 Select attributes 标签页

属性评估器和搜索方法是成组匹配的,在 3.5.3 小节中会进行解释。也就是说虽然用户在 Weka 中是分别设置这两个对象,但是如果用户选择的组合不匹配,Weka 会弹出一个错误消息提示框,也会帮忙自动定位一个匹配的内容。

Select attributes 标签页包含 4 部分内容，现解释如下。

- Attribute Evaluator 选项组用于设置属性评估器，可以通过单击 Choose 按钮选择不同的属性评估方法，具体方法的介绍，请见 3.5.4 小节内容。
- Search Method 选项组用于设置搜索方法，同理也可以通过 Choose 选择不同的搜索方法，具体见 3.5.4 小节。
- Attribute Selection Mode 选项组用于设置属性选择模式，有两种模式可以选择：Use full training set(使用完整的训练集)，该模式的意思即为使用所有的训练数据集来确定属性或属性子集的评估值；Cross-validation(交叉验证)，通过交叉验证来确定属性或属性子集的评估值。其中 Folds 选项用于设置交叉验证的折数，Seed 选项用于设置打乱数据时使用的随机种子。
- Select attributes 标签页有一个下拉列表框，用于设置在选择属性时哪个属性用作类别属性。

设置好上面内容后，单击 Start 按钮，即可启动自动选择属性的过程。当过程结束后，结果会显示在 Attribute selection output(属性选择输出)区域，同时会在 Result list(结果列表)区域添加一个关于本次执行的条目。同样，在结果列表区域中的条目上右击，弹出的快捷菜单可以设置以何种方式查看结果，例如，View in main window 表示在主窗口中查看；View in separate window 表示在单独窗口中查看。

需要提醒的是，选择属性操作除了可以在 Select attributes 标签页中完成之外，还可以在 Classify 标签页中使用元分类器 AttributeSelectedClassifier 完成，如图 3.71 所示。通过该元分类器可以在数据挖掘中做训练数据的分类前，先通过选择属性来减少维度。

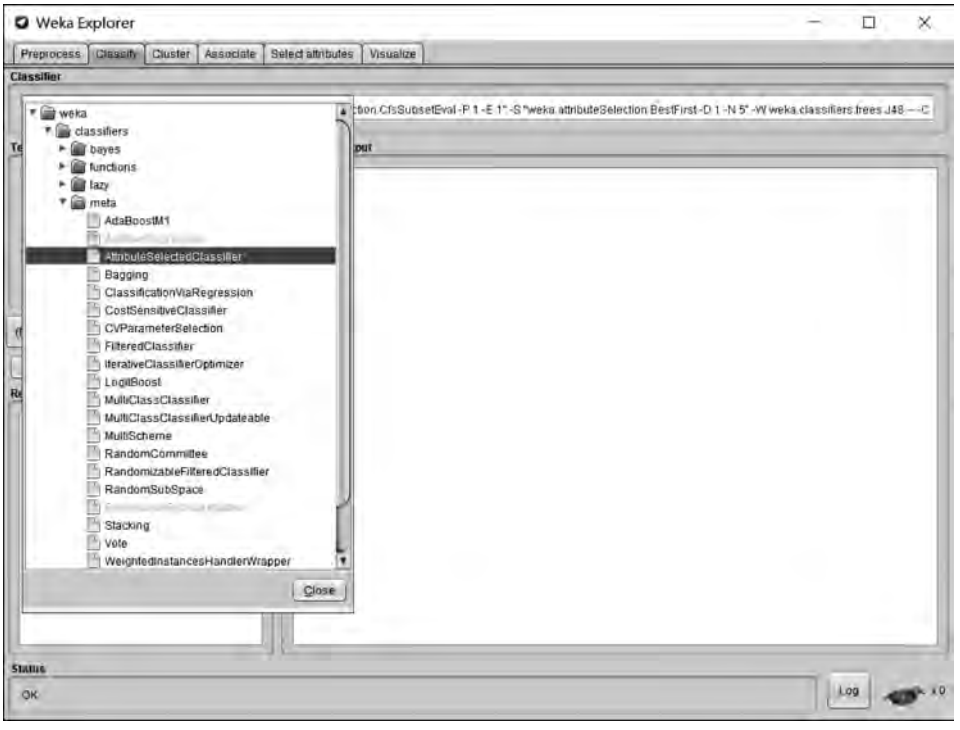


图 3.71　Classify 标签页中元分类器 AttributeSelectedClassifier 的设置

3.5.3 选择属性模式介绍

从图 3.70 可以看到使用 Weka 软件实现自动选择属性的关键之处在于设置合适的属性评估器和搜索方法。选择属性的目的在于从所有属性集中搜索出满足需要的属性子集空间，然后评估每一个空间；或者也可以考虑针对属性全集中的每一个属性分别给出一定的评估值，然后按照评估值进行排序，舍弃低于一定标准的属性。Weka 中刚好提供了这样的两种属性选择模式，一种是属性子集评估器＋搜索方法的模式，后者可以说是一种循环，而前者则是循环中每个环节需要做的操作；另一种模式是单一属性评估器＋排序方法，前者针对每个属性给出一个评估值，后者对所有评估值做排序。操作软件时，由用户自己选择属性评估器和搜索方法的组合，当用户选择的组合不恰当时，Weka 软件会给出一个错误提示信息。下面具体介绍各个属性评估器以及搜索方法。

1. 属性子集评估器

属性子集评估器，是选取属性的一个子集，并且给出一个用于后续搜索方法的度量数值。所有的属性子集评估器，都可以通过 Weka 的 GenericObjectEditor（通用对象编辑器）进行相关参数的配置，如图 3.72 所示。

图 3.72 属性子集评估器的配置

下面解释一下 Weka 中常见的几种属性子集评估器。

- CfsSubsetEval 评估器，根据属性子集中每一个特征的预测能力以及它们之间的关联性进行评估，与类具有高相关性的属性子集被推荐选择。循环搜索的过程中，在已有推荐的基础上迭代添加与类别属性相关度最高的属性，同时还要注意选出的属性要与当前已有属性相关度低。
- ClassifierSubsetEval 评估器，根据训练集或测试集之外的数据评估属性子集。
- WrapperSubsetEval 评估器，是一种包装器方法，使用一种学习模式或者分类器对属性集进行评估，该评估器会对每个子集采用交叉验证来估计学习方案的准确性。由于该

方法使用分类器进行评估，所以需要在它的 GenericObjectEditor（通用对象编辑器）中设置具体的分类器方法，其中包括之前学过的贝叶斯、决策树、线性回归、Logistic 回归以及元分类器等，可以根据自己的需要进行选择。具体设置方法如图 3.73 所示。

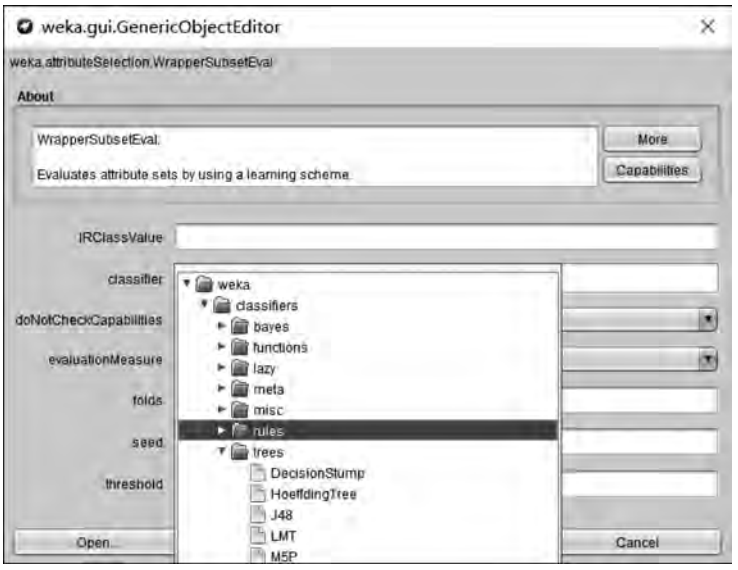


图 3.73　WrapperSubsetEval 评估器的配置

2. 单个属性评估器

单个属性评估器，是针对每个属性给出一个评估值，该评估器必须跟后续的排序 Ranker 方法一同使用，排序方法会通过舍弃若干属性后得出一个具有一定数目属性的排名列表。Weka 中经常用到的单个属性评估器如下。

- ClassifierAttributeEval 评估器，使用用户指定的分类器来评估属性。
- CorrelationAttributeEval 评估器，通过测量单个属性与类之间的相关性（Pearson 相关系数）来评估属性。
- GainRatioAttributeEval 评估器，通过测量相应类别的每一个属性的增益比来评估属性。
- InfoGainAttributeEval 评估器，通过测量类别对应的每一个属性信息增益来评估属性。如果数据集中的属性为数值型，该评估器会先使用基于最小描述长度的离散化方法对数值型属性进行离散化。
- OneRAttributeEval 评估器，使用简单的 OneR 分类器采用的准确性度量来评估属性。
- ReliefFAttributeEval 评估器，基于实例的评估器，它会随机抽取实例样本，并检查具有相同和不同类别的邻近实例。该评估器可以针对离散型数据，也可以运行于连续型数据。其中的参数包括指定抽样实例的数量、要检查的邻近实例的数量等。
- SymmetricalUncertAttributeEval 评估器，通过测量属性的对称不确定性来评估属性。
- PrincipalComponents 评估器，使用主成分分析法进行属性评估。该评估器不同于其他单个属性评估器的地方在于，它会转换成属性集，新属性按照各特征值进行排序。

3. 搜索方法

属性评估器只是针对属性的评估方法，要想得到需要的属性子集，还需要借助于一定的搜索方法，在属性全集中进行相应的搜索。搜索方法是指，遍历属性空间以搜索好的子集，通过所选的属性子集评估器来衡量其质量。同样，每个搜索方法都可以使用 Weka 通用对象编辑器进行配置。常用的搜索方法介绍如下：

(1) BestFirst 最佳优先搜索方法。BestFirst 是一种可以回溯的贪婪上升的搜索方法。它可以从空属性开始向前通过一步步增加属性进行搜索，也可以从全集开始向后通过一步步减少属性个数进行搜索，还可以从中间点开始双向搜索，同时考虑所有可能的单个属性的增删操作。

(2) GreedyStepwise。GreedyStepwise 是一种向前或向后的单步搜索方法，也是贪婪搜索属性的子集空间，可以从空集开始向前搜索，也可以从全集开始向后搜索，但是不回溯。在搜索过程中，如果加或减剩余的最佳属性会导致评估指标下降，那么搜索就会立即终止。在该方法中，用户可以指定要保留的属性数目，或者设置一个阈值，舍弃所有低于该阈值的属性。前面介绍的属性子集评估器，就是要与 BestFirst 和 GreedyStepwise 两种搜索方法其中之一相结合，构成一种选择属性的模式。

(3) Ranker 排序搜索方法。严格来说，这不是一个搜索属性的方法，而是一个对属性排序的方法。通过使用单个属性评估器对属性评估，然后按照评估值进行排序，所以该方法只能和单个属性评估器组合，不能与属性子集评估器匹配。而且，Ranker 方法不仅可以对所有属性进行排序，还能把排名比较低的属性删掉，实现选择属性的目的。用户也可以在通用对象编辑器中设置一个截止阈值，从而舍弃低于该阈值的属性，或者指定选择留下多少个属性。用户甚至可以指定保留某些属性，不管它的排名如何。

3.5.4 Weka 中选择属性操作示例

1. 手工选择属性

前面提到选择属性可以采取手工选择，也可以采取自动选择。手工选择过程既烦琐又容易出错，一般在实践中面对大量数据时，很少选择手工选择。不过在这一节做一个手工选择属性的简单实验，目的在于理解选择属性的工作原理和工作过程。当然本节的手工选择过程也是需要借助 Weka 软件的帮助。使用 Weka 提供的玻璃数据集，使用 IBk 算法，来验证一下哪个属性子集可以产生较好的分类准确率。

启动 Weka，在 Preprocess 标签页中加载 glass.arff 文件，可以看到，除最后一个类别属性外，该数据集共有 9 个属性，如图 3.74 所示。在 Attributes 选项组中，通过选中属性表格里属性名称前面的复选框选择想要移除的属性，然后单击 Remove 按钮移除选中的属性，使用剩下的属性子集进行测试。

采用逐步消去属性的方法，从完整数据集中移除一个属性，形成一个属性子集，对每个属性子集进行交叉验证，可以确定最佳的 8 个属性的数据集，以此类推，移除两个属性，3 个属性等。具体手工测试的步骤如下。

(1) 先以 9 个属性为属性子集，单击 Classify 标签页。

(2) 单击 Choose 按钮，选择 IBk 分类器。

(3) 在 Test options 选项中，选择十折交叉验证。

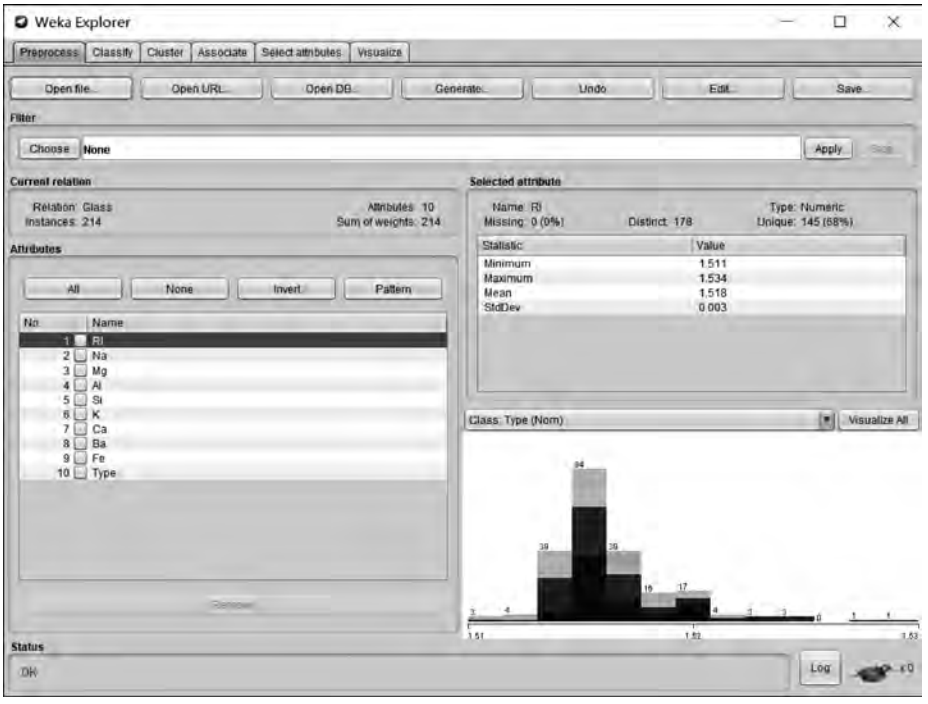


图 3.74　玻璃数据集的预处理 Preprocess 标签页

（4）单击 Start 按钮，启动分类器，得出结果，记录正确分类的百分比，可以看到 9 个属性分类准确率为 70.560 7%，如图 3.75 所示，同时在准确率表 3.8 中记录下来。

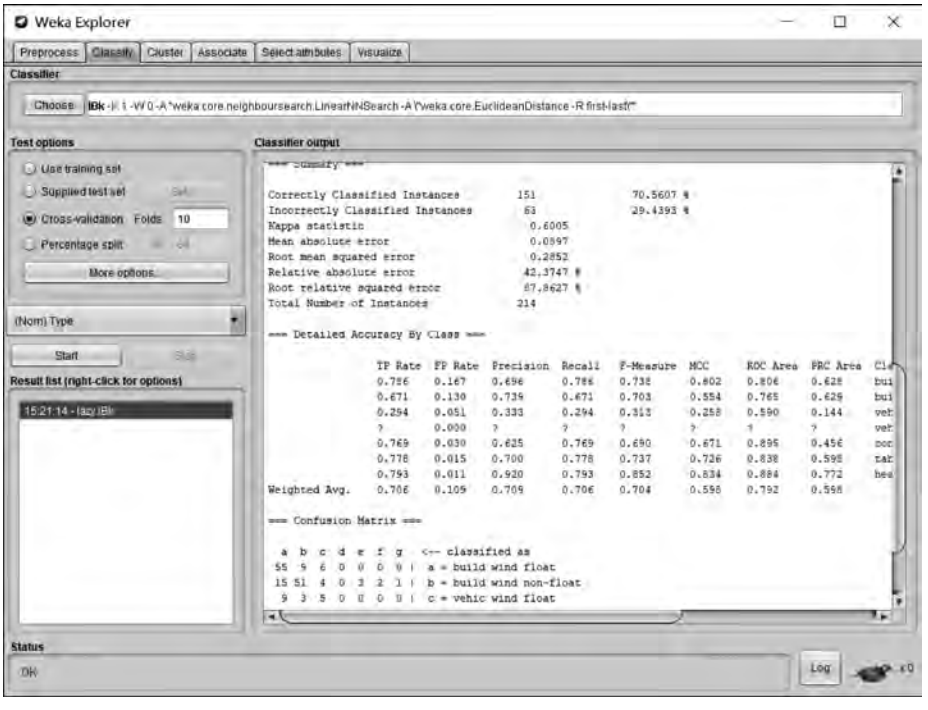


图 3.75　玻璃数据集第一次分类后的结果

(5) 回到预处理标签页,移除一个属性 Fe,再单击 Classify 标签页中的 Start 按钮,得出结果,可以看到 8 个属性的分类准确率为 77.102 8%。以此类推,移除其他单个属性,运行结果,发现在 8 个属性的分类准确率中,只有移除 Fe 属性的准确率最高,记录该准确率于表 3.8 的第 2 行。

(6) 回到预处理标签页,进行移除两个属性的运算,最后得出 7 个属性的最大准确率为 77.570 1%,记录于表 3.8 的第 3 行。如此反复,得出最终结果,如表 3.8 所示。

表 3.8 玻璃数据集不同属性子集分类的准确率

数据子集大小(属性数量)	最佳子集的属性	分类准确率/%
9	RI,Na,Mg,Al,Si,K,Ca,Ba,Fe	70.560 7
8	RI,Na,Mg,Al,Si,K,Ca,Ba	77.102 8
7	RI,Na,Mg,Al,K,Ca,Ba	77.570 1
6	RI,Na,Mg,K,Ca,Ba	78.972 0
5	RI,Mg,Al,K,Ca	79.439 3
4	RI,Mg,K,Ca	77.102 8
3	RI,K,Ca	73.831 8
2	RI,Mg	65.887 9
1	Al	52.336 4
0		35.514 0

从表 3.8 可以看出,玻璃数据集用 IBk 分类器进行分类时,当在 9 个属性中选择 5 个属性时可以达到最高分类准确率 79.439 3%,更多或更少的属性均会降低分类的准确率,即最佳的属性子集为 5 个属性,分别为 RI、Mg、Al、K、Ca。所以,最佳属性子集的分类准确率 79.439 3%,优于属性全集的分类准确率 70.560 7%。这再次证实了前面提到的“选择属性”操作的意义——去除冗余的属性,筛选出对预测学习结果最好的一组属性。

通过本次实验可以获知,在数据集中手工选择属性是很烦琐和复杂的,单纯从每次去除属性来看,实验的次数是巨大的,因为需要从 9 个属性全集中把任意 1 个或任意 2 个以至任意 8 个都要去掉,分别做实验,从而得出分类准确率,然后再选择最高的准确率填入表格。所以这样的实验不仅任务量很大,也会花费很长时间。要在数据量更大的数据集中实现显然是很不现实的,所以需要借助于一定的软件帮助进行自动属性选择。研究表明,自动的属性选择方法通常更快更好。

2. 自动选择属性

自动选择属性有两种模式:搜索方法+属性子集评估器和单个属性评估器+排序。下面就在 Weka 软件中用这两种模式来进行实验。

首先使用 CfsSubsetEval 评估器来评估属性子集,选择 GreedyStepwise 搜索方法,具体步骤如下。

继续使用玻璃数据集,便于将得出的结果跟手工选择属性的结果做对比。先加载 glass.arff 文件,然后选择 Select attributes 标签页。在 Attribute Evaluator 选项组中,单击 Choose 按钮,选择评估器,默认是 CfsSubsetEval 评估器,单击 Search Method 选项组中的 Choose 按钮,选择 GreedyStepwise 搜索方法,然后单击 Start 按钮,启动自动选择属性模

式，运行结果如图 3.76 所示。从运行结果可以看出，已经选出最佳属性子集，共有 7 个属性，分别为 RI、Mg、Al、K、Ca、Ba、Fe。结果中显示的 1，3，4，6，7，8，9 是这 7 个属性在属性全集中的排位。

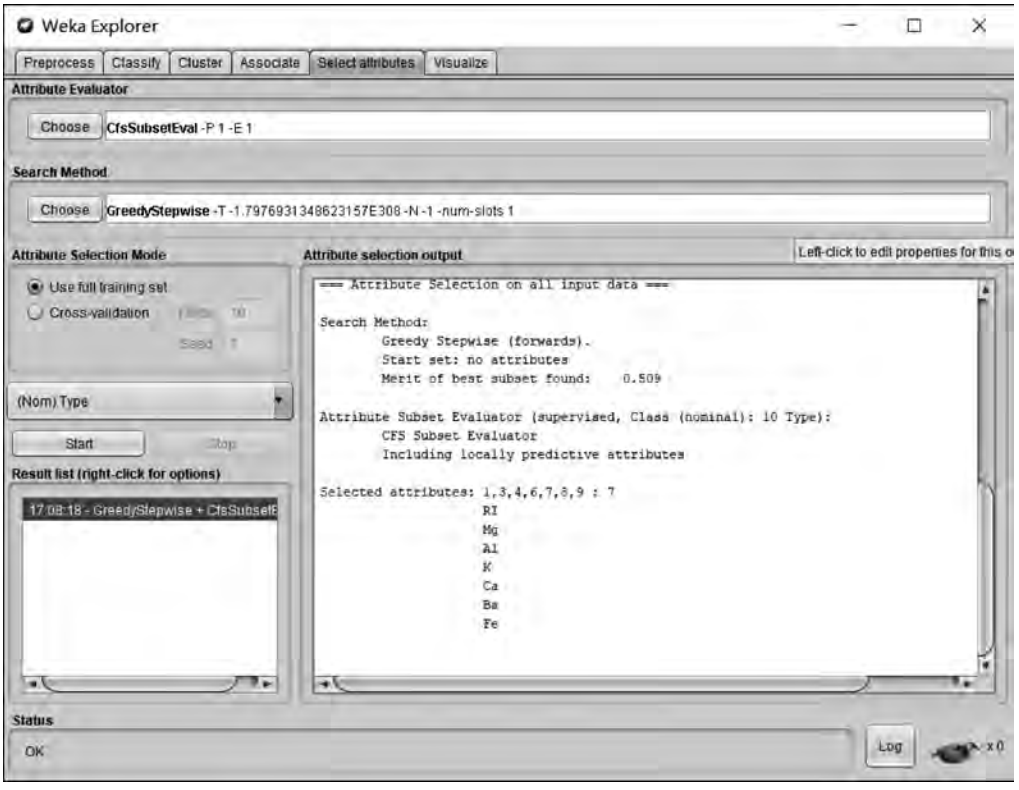


图 3.76　CfsSubsetEval 评估器运行结果

接下来换一种属性子集评估器再做一次实验，选择 WrapperSubsetEval 评估器，同时设置 BestFirst 搜索方法。注意选择此评估器时，需要在该评估器的通用对象编辑器中选择一种分类算法，这里选择 J48 分类算法，并设置搜索算法为 BestFirst，然后单击 Start 按钮，结果如图 3.77 所示。观察运行结果可以看到，该评估器选择出的最佳属性子集共有 5 个，分别为 RI、Mg、Al、K、Ba。5 个属性的排位标号分别为 1、3、4、6、8。

前面是使用属性子集评估器进行属性选择，接下来，换第 2 种属性选择模式进行实验。单个属性评估器使用 InfoGainAttributeEval 评估器，同时使用 Ranker 搜索方法对属性进行排名，具体步骤如下。

在 Attribute Evaluator 选项组中，单击 Choose 按钮，选择 InfoGainAttributeEval 评估器，此时会弹出一个警告对话框，如图 3.78 所示，帮用户询问是否需要选择 Ranker 搜索方法，单击“是”按钮，会发现 Search Method 选项组自动切换成了 Ranker 方法。然后单击 Start 按钮，启动自动选择属性的第二种模式，运行结果如图 3.79 所示。

从图 3.79 的属性选择输出结果可以看到，9 个属性已经按照信息增益的重要程度进行了排序，顺序是 Al、Mg、K、Ca、Ba、Na、RI、Fe、Si，各个属性的编号记为 4、3、6、7、8、2、1、9、5。用户可以按照这个排名选择实际的属性子集。

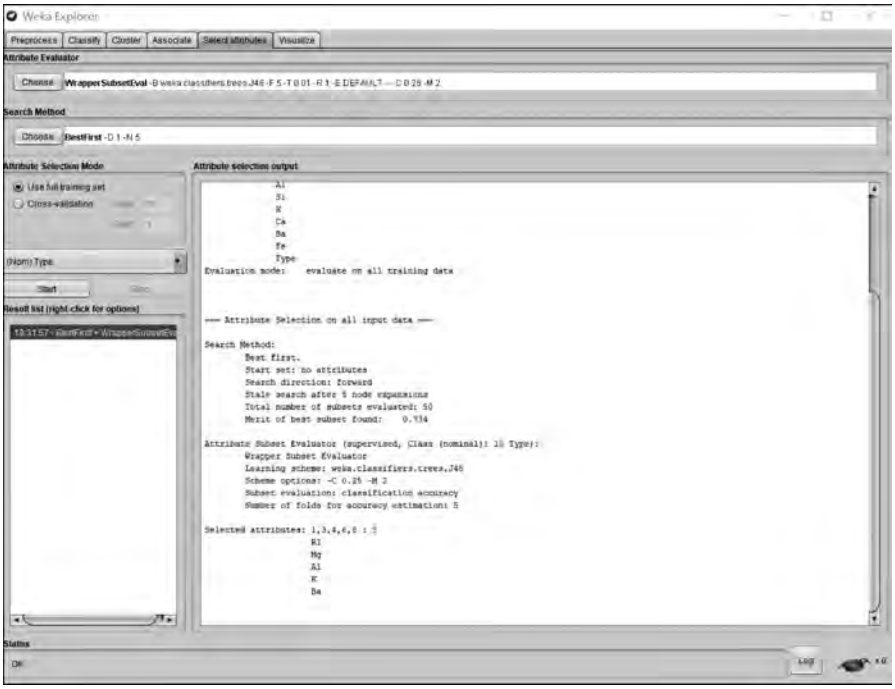


图 3.77　WrapperSubsetEval 评估器运行结果

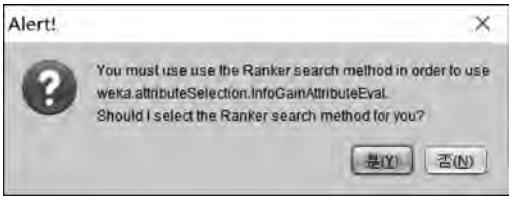


图 3.78　警告对话框

接下来，将两种属性子集评估器以及一种单个属性评估器的实验结果做一个简单的比较，CfsSubsetEval 评估器选出 1、3、4、6、7、8、9 共 7 个属性，WrapperSubsetEval 评估器选出 1、3、4、6、8 共 5 个属性。两种方法都选出了 1、3、4、6、8 这 5 个属性。而这 5 个属性在信息增益评估器中的排位分别为第 7 位、第 2 位、第 1 位、第 3 位、第 5 位。除了 1 和 8 属性，3、4、6 都是排位很靠前的属性。这也说明虽然 3 种方法得出的结果不完全相同，但各种算法都选中了跟类别属性相关性很强的属性。

最后参考刚才手工选择属性时使用的属性子集对分类准确度的影响的相关操作，比较一下两种不同的属性子集评估算法所选出的属性子集对分类准确度的影响，结果见表 3.9。

表 3.9　不同属性子集评估算法的分类准确率

序　号	属 性 子 集	分 类 算 法	评 估 策 略	分类准确率/%
1	全集	IBk	十折交叉验证	70.560 7
2	1,3,4,6,7,8,9	IBk	十折交叉验证	72.429 9
3	1,3,4,6,8	IBk	十折交叉验证	76.168 2

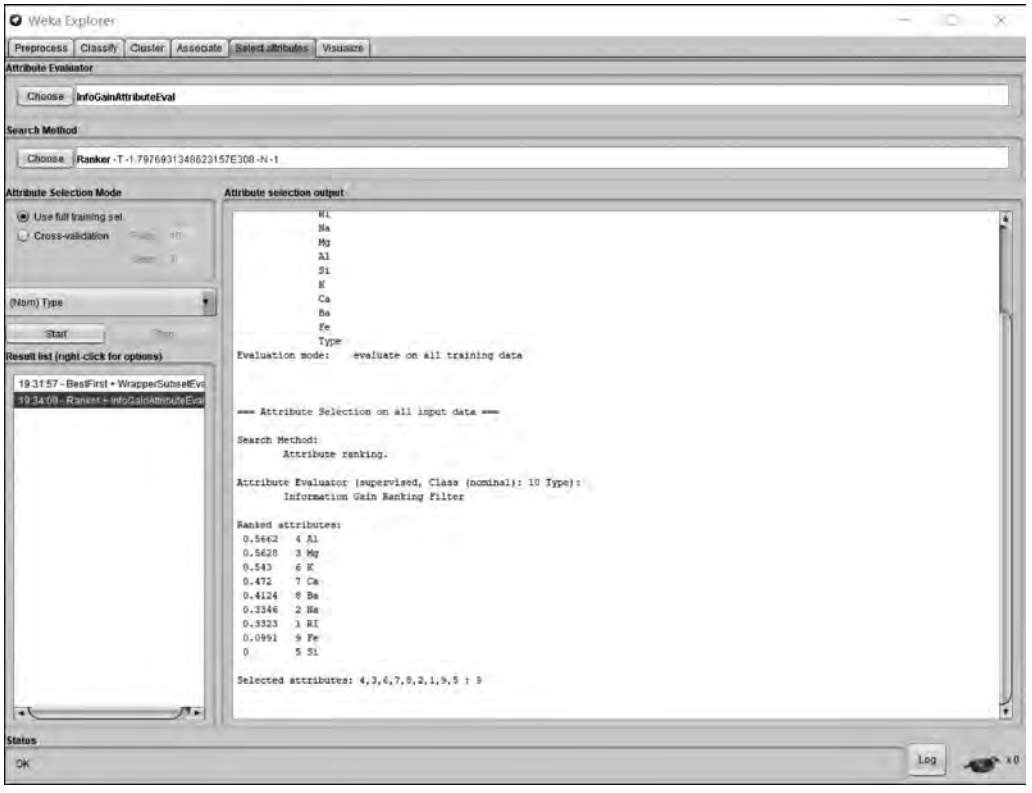


图 3.79　InfoGainAttributeEval 评估器运行结果

从表 3.9 中可以发现，经过属性选择后，分类准确度有了一定程度的提高，而且相对于玻璃数据集这个案例来说，WrapperSubsetEval 评估器选出的 5 个属性的子集，要比全集和 CfsSubsetEval 评估器选出的 7 个属性的分类效果更好一些。

综上所述，通过实验可以看到，自动属性选择得到的结果可以为数据挖掘工作的前期数据预处理工作提供很大的帮助。

3.6　数据可视化

所谓的可视化(Visualization)，是利用计算机相关技术，以图形、图像或表格的形式将数据在屏幕上显示出来。通过分析数据的特征或属性相互之间的关系来更好地研究数据，以发现其中所包含的信息。因为可视化的结果本身比较直观，所以更便于相关研究者发现其中冗余或无意义的属性及数据，从而更好地发现数据里所包含的模式。

本节借助 Weka 软件提供的工具，侧重于研究多属性的数据，通过构造两两属性之间的散点图来显示属性之间的关系，便于用户发现属性之间的关联性。

Weka 软件提供了两种途径进行数据的可视化。第 1 是 Weka GUI 窗口中提供的 Visualization 菜单项，其中包括 Plot、ROC、TreeVisualizer、GraphVisualizer 和 Boundary Visualizer，如图 3.80 所示；第 2 是 Weka Explorer 界面中的 Visualize 标签页。

简单解释图 3.80 中 Visualization 菜单项。

- Plot，散点图，可画出数据集的二维散点图。

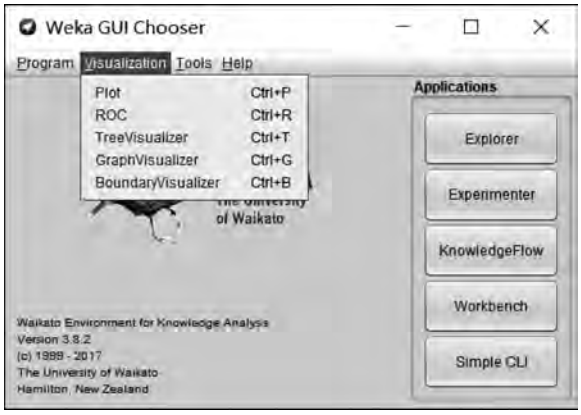


图 3.80 Weka GUI 窗口中的 Visualization 菜单项

- ROC，接受者操作特性曲线，如果打开预先保存的文件，选择该菜单项可显示 ROC 曲线。
- TreeVisualizer，树结构可视化工具，打开保存的数据文件，可显示一个有向图，例如决策树。
- GraphVisualizer，图可视化工具，显示为 XML、BIF 或 DOT 格式的图片，例如贝叶斯网络。
- BoundaryVisualizer，边界可视化工具，允许在二维空间中对分类器的决策边界进行可视化，从而可以比较直观地了解分类器的工作原理。

3.6.1 Visualize 标签页

Weka 软件除了通过其 GUI 窗口中的 Visualization 菜单项做比较简单的数据可视化外，通过其 Explorer 界面的 Visualize 标签页还可从散点图的角度更为详细地对当前数据集做可视化浏览。要注意这里的可视化对象并非分类或聚类模型的运算结果，而是数据集本身。它会把数据显示在一个二维散点图矩阵中，每个单元格对应两个属性。该矩阵的用途是：第一，可以直观地以二维矩阵图方式显示属性两两之间的关系；第二，当给定类别标签后，可以通过它看到所有不同类别的数据实例有两个属性分散的程度。在其中可以借助于一条直线或曲线选择并区分显示在矩阵中的数据点，可为基于该属性的分类做一定的铺垫。下面认识一下 Visualize 标签页中各个选项。

1. Visualize 标签页的组成

首先在 Preprocess 标签页加载数据集，这里使用 Weka 中自带的鸢尾花数据集 iris.arff。然后单击 Visualize 标签页，如图 3.81 所示。该标签页可分为三部分，最上方是由数据中的各属性两两组成的二维矩阵图区，因为 iris 有 5 个属性，所以构成一个 5×5 的矩阵；中间是按钮调节区；下面是 Class Colour 和 Status 的状态显示区。

先选择一个属性（一般选择类别 class 属性），用于对二维矩阵图中的数据点着色。通过单击窗口下方的 Colour：class 下拉列表框，选择一个类别属性，就会依据该属性的值对数据点进行着色；Class Colour 选项组显示的是不同类别对应的颜色，本例中鸢尾花的种类有

三种，依次为山鸢尾、杂色鸢尾和弗吉尼亚鸢尾，所以 Class Colour 选项组里显示了 3 种颜色，单击其中的任一个类别属性名，会弹出 Select new Color 对话框，选择一种颜色，对应的类别名就会显示为该色。以此类推，为每一个类别属性设置一种颜色。如果类别属性属于标称型数据，则会显示离散的着色，如图 3.81 中显示了 3 种类别的不同颜色；如果类别属性是数值型数据，则会显示为颜色渐变的彩色条，根据属性值由低到高，对应的颜色会从蓝色变化到橙色。没有类别取值的数据点会显示为黑色。

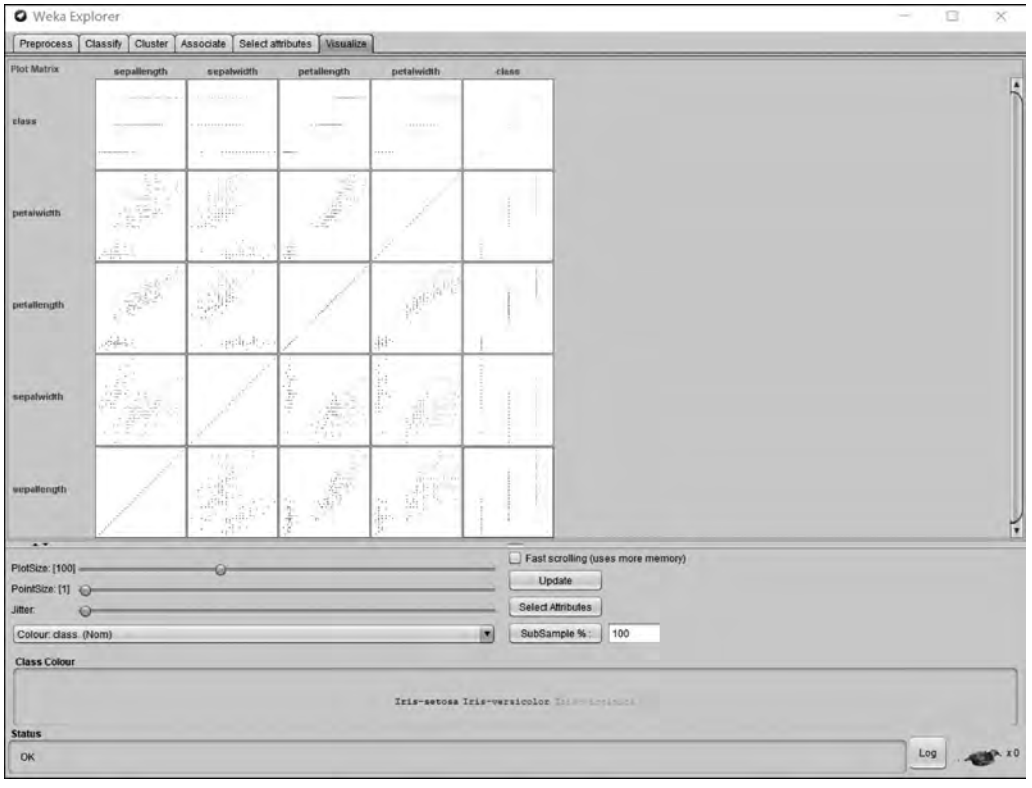


图 3.81　iris 数据集的可视化

在设置好类别属性各个值的颜色后，二维矩阵图区的各个数据点就会依据其所属的类别而显示相应的颜色。观察该二维矩阵区，可以发现数据集的各个属性分别显示在 X 轴和 Y 轴，同时 X 轴和 Y 轴的每个属性两两相交，形成一个单元格，整个二维矩阵图就是由多个单元格组成的，所有的数据点以各种颜色密集地分布在这些单元格中。

下面介绍中间的按钮调节区各个选项的意义。PlotSize 滑块和 PointSize 滑块分别用于改变单个二维散点图（即单元格）的大小和数据点大小；Jitter（抖动）滑块通过由左至右调整可以增加 X 轴、Y 轴上点的随机性，这样重叠的点随着抖动的增加，将不再重叠，会更清晰地显示出来。因此，抖动越大，数据点越多；Fast Scrolling 复选框用于加快数据滚动的速度；Select Attributes 按钮用于对散点图矩阵中的数据点进行属性子集的选择，可以只选择一组属性的子集放在散点图矩阵中，还可以取出数据的一个子样本；SubSample 按钮用于对数据进行二次抽样，可以在旁边的文本框设置随机数种子和抽样的百分比。最后注意，只有在单击 update 按钮后，按钮调节区的所有按钮内容的更改才会生效。

2. 单个二维散点图

当单击二维矩阵图中的一个单元格时，将会弹出一个单独的二维散点图，显示选定的两个属性交叉形成的数据散点图，如图 3.82 所示。

图 3.82　可视化 iris 数据集的单个散点图

从图 3.82 可以看到，跟两个属性相关的数据点散布在窗口的主要区域里，即 Plot：iris 区域。窗口最上方是两个下拉列表框，用来设置图中的 X 轴和 Y 轴。左边的下拉列表框用于设置 X 轴要显示的属性，右边的下拉列表框用于设置 Y 轴要显示的属性。本例需要在两个下拉列表框中把 5 个属性都显示出来，因此可以选择任意两个属性作为该散点图的横纵坐标。在 X 轴选择器下方是一个下拉列表框，用来选择着色的方案，它可以根据所选的属性给点着色，最下方的 Class Colour 选项组中图例颜色的设置跟 Visualize 标签页中的 Class Colour 选项组的设置是一样的，单击也会弹出 Select new Color 对话框。

中间 Plot：iris 区的右边有一些水平横条。每一条代表着 iris 数据集的一个属性，自上而下属性的顺序是与 Visualize 标签页中二维矩阵图中自左至右属性的顺序是一一对应的，横条中的点代表了属性值的分布。这些点随机地在竖直方向散开，使得点的密集程度能被看出来。

单击这些横条可以改变左侧 Plot：iris 区的坐标轴。单击可以改变 X 轴的属性；右击改变 Y 轴的属性。横条旁边的“X”和“Y”代表了当前的坐标轴使用的属性（如果某个横条旁边显示的是“B”，则说明 X 轴和 Y 轴都使用该属性）。

属性横条的上方是一个标着 Jitter 的抖动滑块。它能随机地使散点图中各点的位置发生偏移，也就是抖动。把它拖到右边可以增加抖动的幅度，这对识别点的密集程度很有用。如果不使用这样的抖动，几万个点放在一起和单独的一个点用肉眼看会没什么区别。

在 Y 轴选择器的下方是一个标着 Select Instance 的下拉列表框，它决定选取数据点的

方法，Weka 软件提供了 4 种选择数据点的方法。

（1）Select Instance。单击图中某个数据点，会打开一个窗口列出它的属性值，若单击处的点大于一个，则其他组的属性值也会被列出来，如图 3.83 和图 3.84 所示。

图 3.83　只有 1 个实例的数据点

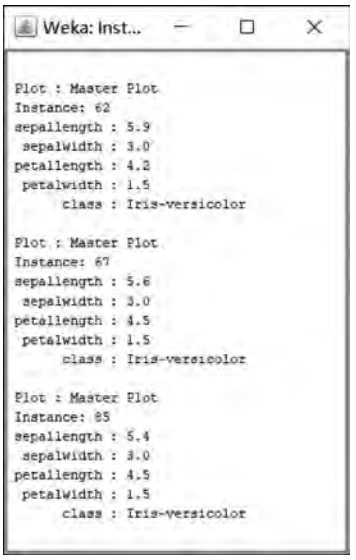


图 3.84　包含 3 个实例的数据点

（2）Rectangle。通过按住鼠标左键并拖动会创建一个矩形，可以把要选择的点框在该矩形中。

（3）Polygon。通过单击并拖动鼠标，会创建一个形式自由的多边形并选取其中的点。单击添加多边形的顶点，右击完成顶点设置即结束选择。起始点和最终点会自动连接起来，因此多边形总是闭合的。

（4）Polyline。可以创建一条折线把它两边的点区分开。单击添加折线顶点，右击结束设置。折线总是打开的（与 Polygon 中创建的闭合多边形相反）。

使用 Rectangle、Polygon 或 Polyline 圈定了散点图的一个区域后，该区域会变成灰色。这时单击散点图上方的 Submit 按钮会移除落在灰色区域之外的所有数据点，只显示所选中的数据点。同时可以发现 X 轴和 Y 轴的数据值范围也会发生改变，变成所选区域对应的数值范围。如果选中某个区域后，单击 Clear 按钮，会清除所选区域而对原有图形不产生任何影响。如果图中所有的数据点都被移除，则 Submit 按钮会变成 Reset 按钮。这个按钮能取消前面所做的全部移除操作，图形回到所有点都在的初始状态。最后，单击 Save 按钮可把当前能看到的数据集保存到一个新的.arff 数据集文件中。单击 Open 按钮可以打开保存后的数据集文件。

3.6.2　数值型类别属性可视化

在 iris 数据集中，类别属性 class 是标称型数据，可以看到图 3.81 中的点是有 3 种不同的灰度。再通过一个类别属性是数值型数据的例子，与图 3.81 对比一下。加载 Weka 中的 cpu.arff 数据集，它的类别属性是数值型数据，加载数据集后打开 Visualize 标签页，如图 3.85 所示。

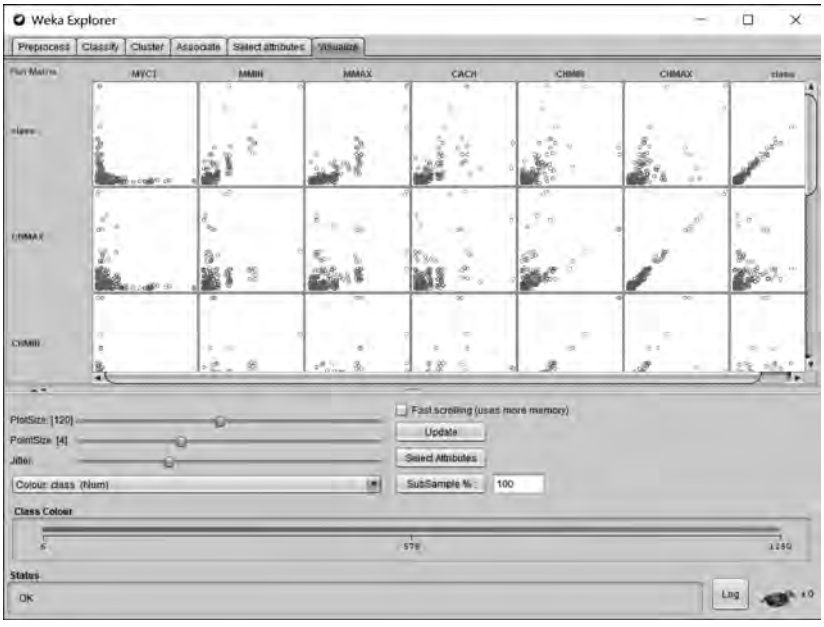


图 3.85　cpu 数据集的可视化

可以看到，该标签页窗口底部的 Class Colour 显示这里显示的不再是离散的几个颜色和属性名，而是显示为一个彩色条，伴随着属性值从左到右由低到高，颜色从蓝色一直逐渐变化到橙色（详见微课视频演示）。这正是与图 3.81 不一样的地方。除此之外，窗口中所有其他按钮的功能均与类别属性为标称型数据集（图 3.81）的功能完全相同。这里不再一一解释，读者可以使用 cpu 数据集自行尝试一下前面在图 3.81 中所做的所有实验操作。

值得一提的是，使用 Weka GUI 窗口中 Visualization 菜单下的 Plot 菜单项得出的二维散点图（图 3.86），与 Visualize 标签页中的单个二维散点图（图 3.82）相比，除了窗口的标题栏显示不一样以外，其他都完全一样。

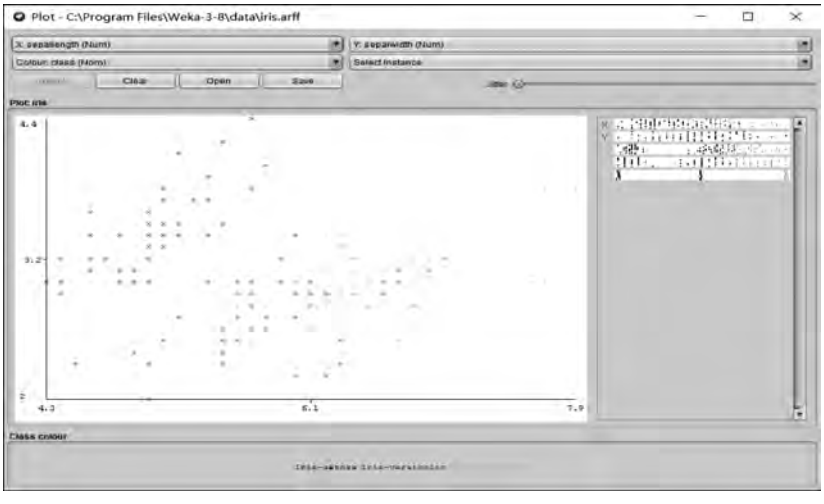


图 3.86　Visualization 菜单下 Plot 菜单项得出的二维散点图