数据挖掘开源软件包

Weka 基础教程

本文档部分资料摘自于网络，随着自己学习的深入将会不断地加以修订和完善。

整理者：韩光辉

版本信息：2010-1-15 Version 1.0

2010-5-26 Version 1.1

贵州大学
中文信息处理与应用系统软件实验室
韩光辉

WEKA 基础教程

目录

1.简介

WEKA 的全名是怀卡托智能分析环境（Waikato Environment for Knowledge Analysis），它的源代码可通过 http://www.cs.waikato.ac.nz/ml/weka 得到。同时 weka 也是新西兰的一种鸟名，而 WEKA 的主要开发者来自新西兰。

WEKA 作为一个公开的数据挖掘工作平台，集合了大量能承担数据挖掘任务的机器学习算法，包括对数据进行预处理，分类，回归、聚类、关联规则以及在新的交互式界面上的可视化。

如果想自己实现数据挖掘算法的话，可以看一看 weka 的接口文档。在 weka 中集成自己的算法甚至借鉴它的方法自己实现可视化工具并不是件很困难的事情。

2005 年 8 月，在第 11 届 ACM SIGKDD 国际会议上，怀卡托大学的 Weka 小组荣获了数据挖掘和知识探索领域的最高服务奖，Weka 系统得到了广泛的认可，被誉为数据挖掘和机器学习历史上的里程碑，是现今最完备的数据挖掘工具之一（已有 11 年的发展历史）。Weka 的每月下载次数已超过万次。

目前(2010-5-26)weka 的最新版本是 3.7.1,SourceForgeweka 站点地址是

http://sourceforge.net/projects/weka/

2.数据格式

2.1.数据文件格式

首先我们来看看 WEKA 所用的数据应是什么样的格式。跟很多电子表格或数据分析软件一样，WEKA 所处理的数据集是图 1 那样的一个二维的表格。

weather.arff

Relation: weather

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

图 1 新窗口打开

这里我们要介绍一下 WEKA 中的术语。表格里的一个横行称作一个实例（Instance），相当于统计学中的一个样本，或者数据库中的一条记录。竖行称作一个属性（Attrbute），相当于统计学中的一个变量，或者

数据库中的一个字段。这样一个表格，或者叫数据集，在 WEKA 看来，呈现了属性之间的一种关系(Relation)。图 1 中一共有 14 个实例，5 个属性，关系名称为“weather”。WEKA 存储数据的格式是 ARFF（Attribute-Relation File Format）文件，这是一种 ASCII 文本文件。图 1 所示的二维表格存储在如下的 ARFF 文件中。这也就是 WEKA 自带的“weather.arff”文件，在 WEKA 安装目录的“data”子目录下可以找到。

Code:

```
% ARFF file for the weather data with some numric features
%
@relation weather

@attribute outlook {sunny, overcast, rainy}
@attribute temperature real
@attribute humidity real
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
%
% 14 instances
%
sunny,85,85,FALSE,no
sunny,80,90,TRUE,no
overcast,83,86,FALSE,yes
rainy,70,96,FALSE,yes
rainy,68,80,FALSE,yes
rainy,65,70,TRUE,no
overcast,64,65,TRUE,yes
sunny,72,95,FALSE,no
sunny,69,70,FALSE,yes
rainy,75,80,FALSE,yes
sunny,75,70,TRUE,yes
overcast,72,90,TRUE,yes
overcast,81,75,FALSE,yes
rainy,71,91,TRUE,no
```

需要注意的是，在 Windows 记事本打开这个文件时，可能会因为回车符定义不一致而导致分行不正常。推荐使用 UltraEdit 这样的字符编辑软件察看 ARFF 文件的内容。

2.2.数据文件格式的详细描述

识别 ARFF 文件的重要依据是分行，因此不能在这种文件里随意的断行。空行（或全是空格的行）将被忽略。 以“%”开始的行是注释，WEKA 将忽略这些行。如果你看到的“weather.arff”文件多了或少了些“%”开始的行，是没有影响的。 除去注释后，整个 ARFF 文件可以分为两个部分。第一部分给出了头信息（Head information），包括了对关系的声明和对属性的声明。第二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。

2.2.1. 关系声明

关系名称在 ARFF 文件的第一个有效行来定义，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，它必须加上引号（指英文标点的单引号或双引号）。

2.2.2. 属性声明

属性声明用一列以"@attribute"开头的语句表示。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"是第三个被声明的属性，这说明数据部分那些被逗号分开的列中，第三列数据 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。

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，它必须加上引号。

WEKA 支持的<datatype>有四种，分别是

numeric-------------------------数值型

<nominal-specification>-----分类（nominal）型

string----------------------------字符串型

date [<date-format>]-------- 日期和时间型

其中<nominal-specification> 和<date-format> 将在下面说明。还可以使用两个类型"integer"和"real"，但是 WEKA 把它们都当作"numeric"看待。注意"integer"，"real"，"numeric"，"date"，"string"这些关键字是区分大小写的，而"relation""attribute "和"date"则不区分。

数值属性

数值型属性可以是整数或者实数，但 WEKA 把它们都当作实数看待。

分类属性

分类属性由<nominal-specification>列出一系列可能的类别名称并放在花括号中：{<nominal-name1>, <nominal-name2>, <nominal-name3>, ...} 。数据集中该属性的值只能是其中一种类别。

例如如下的属性声明说明"outlook"属性有三种类别："sunny"，" overcast"和"rainy"。而数据集中每个实例对应的"outlook"值必是这三者之一。

@attribute outlook {sunny, overcast, rainy}

如果类别名称带有空格，仍需要将之放入引号中。

字符串属性

字符串属性中可以包含任意的文本。这种类型的属性在文本挖掘中非常有用。
示例：

@ATTRIBUTE LCC string

日期和时间属性

日期和时间属性统一用“date”类型表示，它的格式是

@attribute <name> date [<date-format>]

其中<name>是这个属性的名称，<date-format>是一个字符串，来规定该怎样解析和显示日期或时间的格式，默认的字符串是 ISO-8601 所给的日期时间组合格式“yyyy-MM-ddTHH:mm:ss”。
数据信息部分表达日期的字符串必须符合声明中规定的格式要求（下文有例子）

数据信息

数据信息中“@data”标记独占一行，剩下的是各个实例的数据。每个实例占一行。实例的各属性值用逗号“,”隔开。如果某个属性的值是缺失值（missing value），用问号“?”表示，且这个问号不能省略。例如：

@data
sunny,85,85,FALSE,no
?,78,90,?,yes

字符串属性和分类属性的值是区分大小写的。若值中含有空格，必须被引号括起来。例如：

@relation LCCvsLCSH
@attribute LCC string
@attribute LCSH string
@data
AG5, 'Encyclopedias and dictionaries.;Twentieth century.'
AS262, 'Science -- Soviet Union -- History.'

日期属性的值必须与属性声明中给定的相一致。例如：

@RELATION Timestamps
@attribute timestamp DATE "yyyy-MM-dd HH:mm:ss"
@data
2001-04-03 12:12:12"
2001-05-03 12:59:55"

稀疏数据

有的时候数据集中含有大量的 0 值（比如购物篮分析），这个时候用稀疏格式的数据存贮更加省空间。稀疏格式是针对数据信息中某个实例的表示而言，不需要修改 ARFF 文件的其它部分。看如下的数据：

@data
0, X, 0, Y, "class A"
0, 0, W, 0, "class B"

用稀疏格式表达的话就是

@data
1 X, 3 Y, 4 "class A"}

{2 W, 4 "class B"}
每个实例用花括号括起来。实例中每一个非 0 的属性值用<index> <空格> <value>表示。<index>是属性的序号，从 0 开始计；<value>是属性值。属性值之间仍用逗号隔开。这里每个实例的数值必须按属性的顺序来写，如 {1 X, 3 Y, 4 "class A"}，不能写成{3 Y, 1 X, 4 "class A"}。
注意在稀疏格式中没有注明的属性值不是缺失值，而是 0 值。若要表示缺失值必须显式的用问号表示出来。

Relational 型属性
在 WEKA 3.5 版中增加了一种属性类型叫做 Relational，有了这种类型我们可以像关系型数据库那样处理多个维度了。但是这种类型目前还不见广泛应用，暂不作介绍。

整理自 http://www.cs.waikato.ac.nz/~ml/weka/arff.html 和
http://weka.sourceforge.net/wekadoc/index.php/en:ARFF_%283.5.3%29

3.数据准备

使用 WEKA 作数据挖掘，面临的第一个问题往往是我们的数据不是 ARFF 格式的。幸好，WEKA 还提供了对 CSV 文件的支持，而这种格式是被很多其他软件所支持的。此外，WEKA 还提供了通过 JDBC 访问数据库的功能。

在这一节里，我们先以 Excel 和 Matlab 为例，说明如何获得 CSV 文件。然后我们将知道 CSV 文件如何转化成 ARFF 文件，毕竟后者才是 WEKA 支持得最好的文件格式。面对一个 ARFF 文件，我们仍有一些预处理要做，才能进行挖掘任务。

3.1.* -> .csv

我们给出一个 CSV 文件的例子（bank-data.csv）。用 UltraEdit 打开它可以看到，这种格式也是一种逗号分割数据的文本文件,储存了一个二维表格。

Excel 的 XLS 文件可以让多个二维表格放到不同的工作表（Sheet）中，我们只能把每个工作表存成不同的 CSV 文件。打开一个 XLS 文件并切换到需要转换的工作表，另存为 CSV 类型，点"确定"、"是"忽略提示即可完成操作。

3.2.把 txt 的中的矩阵变成 arff 格式

Txt 中的文本如下：

市 流通 上市 控股
市 0.00000587 0.00133928 0.00058130 0.00000360
流通 0.00133928 0.00060736 0.00945238 0.00028522
上市 0.00058130 0.00945238 0.00003065 0.00318088
控股 0.00000360 0.00028522 0.00318088 0.00014110

用 excel 打开用 table 键分开的 txt 文件,然后另存为.csv,在用 wekaviewer 打开,另存为 arff。

3.3.Matlab 中的二维表格矩阵存储为 CSV

在 Matlab 中的二维表格是一个矩阵，我们通过这条命令把一个矩阵存成 CSV 格式。

csvwrite('filename',matrixname)

需要注意的是，Matllab 给出的 CSV 文件往往没有属性名（Excel 给出的也有可能没有）。而 WEKA 必须从 CSV 文件的第一行读取属性名，否则就会把第一行的各属性值读成变量名。因此我们对于 Matllab 给出的 CSV 文件需要用 UltraEdit 打开，手工添加一行属性名。注意属性名的个数要跟数据属性的个数一致，仍用逗号隔开。

3.4.csv -> .arff

将 CSV 转换为 ARFF 最迅捷的办法是使用 WEKA 所带的命令行工具。

运行 WEKA 的主程序，出现 GUI 后可以点击下方按钮进入相应的模块。我们点击进入“Simple CLI”模块提供的命令行功能。在新窗口的最下方（上方是不能写字的）输入框写上

java weka.core.converters.CSVLoader filename.csv > filename.arff

即可完成转换。

在 WEKA 3.5中提供了一个“Arff Viewer”模块，我们可以用它打开一个 CSV 文件将进行浏览，然后另存为 ARFF 文件。

进入“Exploer”模块，从上方的按钮中打开 CSV 文件然后另存为 ARFF 文件亦可。

3.5.Exploer 界面

我们应该注意到，“Exploer”还提供了很多功能，实际上可以说这是 WEKA 使用最多的模块。现在我们先来熟悉它的界面，然后利用它对数据进行预处理。

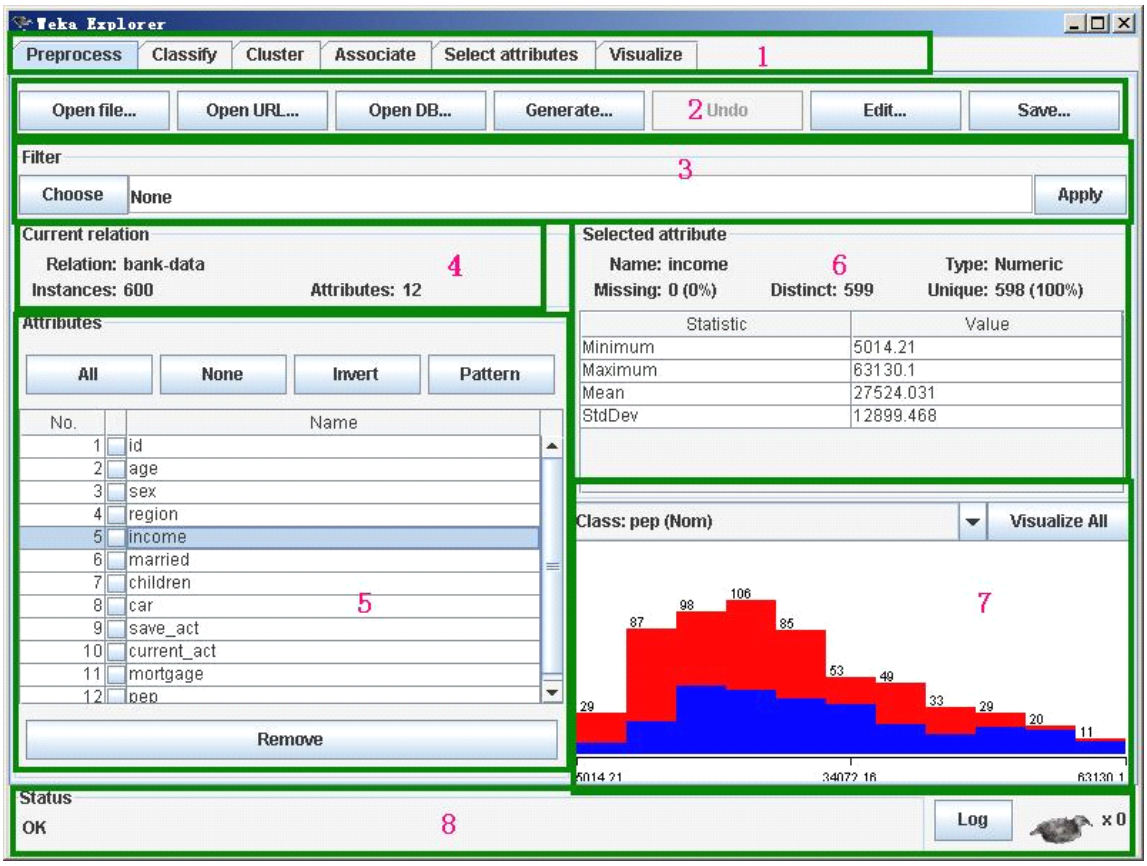


图2 新窗口打开

图2显示的是使用3.5版"Exploer"打开"bank-data.csv"的情况。我们根据不同的功能把这个界面分成8个区域。

区域1的几个选项卡是用来切换不同的挖掘任务面板。这一节用到的只有“Preprocess”，其他面板的功能将在以后介绍。

区域2是一些常用按钮。包括打开数据，保存及编辑功能。我们在这里把"bank-data.csv"另存为"bank-data.arff"。

在区域3中“Choose”某个“Filter”，可以实现筛选数据或者对数据进行某种变换。数据预处理主要就利用它来实现。

区域4展示了数据集的一些基本情况。

区域5中列出了数据集的所有属性。勾选一些属性并“Remove”就可以删除它们，删除后还可以利用区域2的“Undo”按钮找回。区域5上方的一排按钮是用来实现快速勾选的。

在区域5中选中某个属性，则区域6中有关于这个属性的摘要。注意对于数值属性和分类属性，摘要的方式是不一样的。图中显示的是对数值属性“income”的摘要。

区域7是区域5中选中属性的直方图。若数据集的最后一个属性（我们说过这是分类或回归任务的默认目标变量）是分类变量（这里的“pep”正好是），直方图中的每个长方形就会按照该变量的比例分成不同颜色的段。要想换个分段的依据，在区域7上方的下拉框中选个不同的分类属性就可以了。下拉框里选上“No Class”或者一个数值属性会变成黑白的直方图。

区域8是状态栏，可以查看 Log 以判断是否有错。右边的 weka 鸟在动的话说明 WEKA 正在执行挖掘任务。右键点击状态栏还可以执行 JAVA 内存的垃圾回收。

3.6.预处理

bank-data 数据各属性的含义如下：
id a unique identification number
age age of customer in years (numeric)
sex MALE / FEMALE
region inner_city/rural/suburban/town
income income of customer (numeric)
married is the customer married (YES/NO)
children number of children (numeric)
car does the customer own a car (YES/NO)
save_acct does the customer have a saving account (YES/NO)
current_acct does the customer have a current account (YES/NO)
mortgage does the customer have a mortgage (YES/NO)
pep did the customer buy a PEP (Personal Equity Plan) after the last mailing (YES/NO)

通常对于数据挖掘任务来说，ID 这样的信息是无用的，我们将之删除。在区域5勾选属性“id”，并点击“Remove”。将新的数据集保存一次，并用 UltraEdit 打开这个 ARFF 文件。我们发现，在属性声明部分，WEKA 已经为每个属性选好了合适的类型。

我们知道，有些算法，只能处理所有的属性都是分类型的情况。这时候我们就需要对数值型的属性进行离散化。在这个数据集中有3个变量是数值型的，分别是“age”，“income”和“children”。
其中“children”只有4个取值：0，1，2，3。这时我们在 UltraEdit 中直接修改 ARFF 文件，把
@attribute children numeric
改为
@attribute children {0,1,2,3}
就可以了。
在“Explorer”中重新打开“bank-data.arff”，看看选中“children”属性后，区域6那里显示的“Type”是不是变成“Nominal”了？

“age”和“income”的离散化我们需要借助 WEKA 中名为“Discretize”的 Filter 来完成。在区域2中点“Choose”，出现一棵“Filter 树”，逐级找到“weka.filters.unsupervised.attribute.Discretize”，点击。若无法关闭这个树，在树之外的地方点击“Explorer”面板即可。
现在“Choose”旁边的文本框应该显示“Discretize -B 10 -M -0.1 -R first-last”。 点击这个文本框会弹出新窗口以修改离散化的参数。
我们不打算对所有的属性离散化，只是针对对第1个和第4个属性（见区域5属性名左边的数字），故把 attributeIndices 右边改成“1,4”。计划把这两个属性都分成3段，于是把“bins”改成“3”。其它框里不用更改，关于它们的意思可以点“More”查看。点“OK”回到“Explorer”，可以看到“age”和“income”已经被离散化成分类型的属性。若想放弃离散化可以点区域2的“Undo”。
如果对“"(-inf-34.333333]"”这样晦涩的标识不满，我们可以用 UltraEdit 打开保存后的 ARFF 文件，把所有的“'\'(-inf-34.333333]\'"”替换成“0_34”。其它标识做类似地手动替换。

经过上述操作得到的数据集我们保存为 bank-data-final.arff。
----整理自 http://maya.cs.depaul.edu/~classes/ect584/WEKA/preprocess.html

4.关联规则

注意：目前，WEKA 的关联规则分析功能仅能用来作示范，不适合用来挖掘大型数据集。

我们打算对前面的“bank-data”数据作关联规则的分析。用“Explorer”打开“bank-data-final.arff”后，切换到“Associate”选项卡。默认关联规则分析是用 Apriori 算法，我们就用这个算法，但是点“Choose”右边的文本框修改默认的参数，弹出的窗口中点“More”可以看到各参数的说明。

4.1.背景知识

首先我们来温习一下 Apriori 的有关知识。对于一条关联规则 L->R，我们常用支持度（Support）和置信度（Confidence）来衡量它的重要性。规则的支持度是用来估计在一个购物篮中同时观察到 L 和 R 的概率 P(L,R)，而规则的置信度是估计购物栏中出现了 L 时也出会现 R 的条件概率 P(R|L)。关联规则的目标一般是产生支持度和置信度都较高的规则。

有几个类似的度量代替置信度来衡量规则的关联程度，它们分别是

Lift（提升度？）： P(L,R)/(P(L)P(R))

Lift=1时表示 L 和 R 独立。这个数越大，越表明 L 和 R 存在在一个购物篮中不是偶然现象。

Leverage（不知道怎么翻译）：P(L,R)-P(L)P(R)

它和 Lift 的含义差不多。Leverage=0时 L 和 R 独立，Leverage 越大 L 和 R 的关系越密切。

Conviction（更不知道译了）：P(L)P(!R)/P(L,!R) （!R 表示 R 没有发生）

Conviction 也是用来衡量 L 和 R 的独立性。从它和 lift 的关系（对 R 取反，代入 Lift 公式后求倒数）可以看出，我们也希望这个值越大越好。

值得注意的是，用 Lift 和 Leverage 作标准时，L 和 R 是对称的，Confidence 和 Conviction 则不然。

4.2.参数设置

现在我们计划挖掘出支持度在10%到100%之间，并且 lift 值超过1.5且 lift 值排在前100位的那些关联规则。我们把“lowerBoundMinSupport”和“upperBoundMinSupport”分别设为0.1和1，“metricType”设为 lift，“minMetric”设为1.5，“numRules”设为100。其他选项保持默认即可。“OK” 之后在“Explorer”中点击“Start”开始运行算法，在右边窗口显示数据集摘要和挖掘结果。

下面是挖掘出来的 lift 排前5的规则。

Best rules found:

1. age=52_max save_act=YES current_act=YES 113 ==> income=43759_max 61 conf:(0.54) < lift:(4.05)> lev:(0.0 [45] conv:(1.85)

2. income=43759_max 80 ==> age=52_max save_act=YES current_act=YES 61 conf:(0.76) < lift:(4.05)> lev:(0.0 [45] conv:(3.25)

3. income=43759_max current_act=YES 63 ==> age=52_max save_act=YES 61 conf:(0.97) < lift:(3.85)> lev:(0.0 [45] conv:(15.72)

4. age=52_max save_act=YES 151 ==> income=43759_max current_act=YES 61 conf:(0.4) < lift:(3.85)> lev:(0.0 [45] conv:(1.49)

5. age=52_max save_act=YES 151 ==> income=43759_max 76 conf:(0.5) < lift:(3.77)>

lev:(0.09) [55] conv:(1.72)
对于挖掘出的每条规则，WEKA 列出了它们关联程度的四项指标。

4.3.命令行方式

我们也可以利用命令行来完成挖掘任务，在"Simlpe CLI"模块中输入如下格式的命令：
java weka.associations.Apriori options -t directory-path\bank-data-final.arff
即可完成 Apriori 算法。注意，"-t"参数后的文件路径中不能含有空格。
在前面我们使用的 option 为
-N 100 -T 1 -C 1.5 -D 0.05 -U 1.0 -M 0.1 -S -1.0 命令行中使用这些参数得到的结果和前面利用 GUI 得到的一样。
我们还可以加上"- I"参数，得到不同项数的频繁项集。我用的命令如下：
java weka.associations.Apriori -N 100 -T 1 -C 1.5 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -I -t d:\weka\bank-data-final.arff
挖掘结果在上方显示，应是这个文件的样子。

----整理自 http://maya.cs.depaul.edu/~classes/ect584/WEKA/associate.html

5.分类与回归

5.1.背景知识

WEKA 把分类(Classification)和回归(Regression)都放在"Classify"选项卡中，这是有原因的。
在这两个任务中，都有一个目标属性（输出变量）。我们希望根据一个样本(WEKA 中称作实例)的一组特征（输入变量），对目标进行预测。为了实现这一目的，我们需要有一个训练数据集，这个数据集中每个实例的输入和输出都是已知的。观察训练集中的实例，可以建立起预测的模型。有了这个模型，我们就可以新的输出未知的实例进行预测了。衡量模型的好坏就在于预测的准确程度。
在 WEKA 中，待预测的目标（输出）被称作 Class 属性，这应该是来自分类任务的"类"。一般的，若 Class 属性是分类型时我们的任务才叫分类，Class 属性是数值型时我们的任务叫回归。

5.2.选择算法

这一节中，我们使用 C4.5 决策树算法对 bank-data 建立起分类模型。
我们来看原来的"bank-data.csv"文件。"ID"属性肯定是不需要的。由于 C4.5 算法可以处理数值型的属性，我们不用像前面用关联规则那样把每个变量都离散化成分类型。尽管如此，我们还是把"Children"属性转换成分类型的两个值"YES"和"NO"。另外，我们的训练集仅取原来数据集实例的一半；而从另外一半中抽出若干条作为待预测的实例，它们的"pep"属性都设为缺失值。经过了这些处理的训练集数据在这里下载；待预测集数据在这里下载。

我们用"Explorer"打开训练集"bank.arff"，观察一下它是不是按照前面的要求处理好了。切换到"Classify"

选项卡，点击“Choose”按钮后可以看到很多分类或者回归的算法分门别类的列在一个树型框里。3.5 版的 WEKA 中，树型框下方有一个“Filter...”按钮，点击可以根据数据集的特性过滤掉不合适的算法。我们数据集的输入属性中有“Binary”型（即只有两个类的分类型）和数值型的属性，而 Class 变量是“Binary”的；于是我们勾选“Binary attributes”“Numeric attributes”和“Binary class”。点“OK”后回到树形图，可以发现一些算法名称变红了，说明它们不能用。选择“trees”下的“J48”，这就是我们需要的 C4.5 算法，还好它没有变红。点击“Choose”右边的文本框，弹出新窗口为该算法设置各种参数。点“More”查看参数说明，点“Capabilities”是查看算法适用范围。这里我们把参数保持默认。
现在来看左中的“Test Option”。我们没有专门设置检验数据集，为了保证生成的模型的准确性而不至于出现过拟合（overfitting）的现象，我们有必要采用 10 折交叉验证（10-fold cross validation）来选择和评估模型。若不明白交叉验证的含义可以 Google 一下。

5.3.建模结果

OK，选上“Cross-validation”并在“Folds”框填上“10”。点“Start”按钮开始让算法生成决策树模型。很快，用文本表示的一棵决策树，以及对这个决策树的误差分析等等结果出现在右边的“Classifier output”中。同时左下的“Results list”出现了一个项目显示刚才的时间和算法名称。如果换一个模型或者换个参数，重新“Start”一次，则“Results list”又会多出一项。

我们看到“J48”算法交叉验证的结果之一为
Correctly Classified Instances 206 68.6667 %
也就是说这个模型的准确度只有 69%左右。也许我们需要对原属性进行处理，或者修改算法的参数来提高准确度。但这里我们不管它，继续用这个模型。

右键点击“Results list”刚才出现的那一项，弹出菜单中选择“Visualize tree”，新窗口里可以看到图形模式的决策树。建议把这个新窗口最大化，然后点右键，选“Fit to screen”，可以把这个树看清楚些。看完后截图或者关掉

这里我们解释一下“Confusion Matrix”的含义。

```
=== Confusion Matrix ===
 a b <-- classified as
 74 64 | a = YES
 30 132 | b = NO
```

这个矩阵是说，原本“pep”是“YES”的实例，有 74 个被正确的预测为“YES”，有 64 个错误的预测成了“NO”；原本“pep”是“NO”的实例，有 30 个被错误的预测为“YES”，有 132 个正确的预测成了“NO”。74+64+30+132 = 300 是实例总数，而(74+132)/300 = 0.68667 正好是正确分类的实例所占比例。这个矩阵对角线上的数字越大，说明预测得越好。

5.4.模型应用

现在我们要用生成的模型对那些待预测的数据集进行预测了。注意待预测数据集和训练用数据集各个属性的设置必须是一致的。即使你没有待预测数据集的 Class 属性的值，你也要添加这个属性，可以将该属性在各实例上的值均设成缺失值。
在“Test Opion”中选择“Supplied test set”，并且“Set”成你要应用模型的数据集，这里是“bank-new.arff”文件。

现在，右键点击“Result list”中刚产生的那一项，选择“Re-evaluate model on current test set”。右边显示结果的区域中会增加一些内容，告诉你该模型应用在这个数据集上表现将如何。如果你的 Class 属性都是些缺失值，那这些内容是无意义的，我们关注的是模型在新数据集上的预测值。

现在点击右键菜单中的“Visualize classifier errors”，将弹出一个新窗口显示一些有关预测误差的散点图。点击这个新窗口中的“Save”按钮，保存一个 Arff 文件。打开这个文件可以看到在倒数第二个位置多了一个属性（predictedpep），这个属性上的值就是模型对每个实例的预测值。

5.5.使用命令行（推荐）

虽然使用图形界面查看结果和设置参数很方便，但是最直接最灵活的建模及应用的办法仍是使用命令行。 打开“Simple CLI”模块，像上面那样使用“J48”算法的命令格式为：

java weka.classifiers.trees.J48 -C 0.25 -M 2 -t directory-path\bank.arff -d directory-path\bank.model

其中参数“ -C 0.25”和“-M 2”是和图形界面中所设的一样的。“-t ”后面跟着的是训练数据集的完整路径（包括目录和文件名），“-d ”后面跟着的是保存模型的完整路径。注意！这里我们可以把模型保存下来。

输入上述命令后，所得到树模型和误差分析会在“Simple CLI”上方显示，可以复制下来保存在文本文件里。误差是把模型应用到训练集上给出的。

把这个模型应用到“bank-new.arff”所用命令的格式为：

java weka.classifiers.trees.J48 -p 9 -l directory-path\bank.model -T directory-path\bank-new.arff

其中“-p 9”说的是模型中的待预测属性的真实值存在第 9 个（也就是“pep”）属性中，这里它们全部未知因此全部用缺失值代替。“-l”后面是模型的完整路径。“-T”后面是待预测数据集的完整路径。

输入上述命令后，在“Simple CLI”上方会有这样一些结果：

0 YES 0.75 ?
1 NO 0.7272727272727273 ?
2 YES 0.95 ?
3 YES 0.8813559322033898 ?
4 NO 0.8421052631578947 ?
...

这里的第一列就是我们提到过的“Instance_number”，第二列就是刚才的“predictedpep”，第四列则是“bank-new.arff”中原来的“pep”值（这里都是“?”缺失值）。第三列对预测结果的置信度（confidence ）。比如说对于实例 0，我们有 75%的把握说它的“pep”的值会是“YES”，对实例 4 我们有 84.2%的把握说它的“pep”值会是“NO”。

我们看到，使用命令行至少有两个好处。一个是可以把模型保存下来，这样有新的待预测数据出现时，不用每次重新建模，直接应用保存好的模型即可。另一个是对预测结果给出了置信度，我们可以有选择的采纳预测结果，例如，只考虑那些置信度在 85%以上的结果。

----整理自 http://maya.cs.depaul.edu/~classes/ect584/WEKA/classify.html

6.聚类分析

6.1.原理与实现

聚类分析中的“类”（cluster）和前面分类的“类”（class）是不同的，对 cluster 更加准确的翻译应该是“簇”。聚类的任务是把所有的实例分配到若干的簇，使得同一个簇的实例聚集在一个簇中心的周围，它们之间距离的比较近；而不同簇实例之间的距离比较远。对于由数值型属性刻画的实例来说，这个距离通常指欧氏距离。

现在我们对前面的“bank data”作聚类分析，使用最常见的 K 均值（K-means）算法。下面我们简单描述一下 K 均值聚类的步骤。

K 均值算法首先随机的指定 K 个簇中心。然后：1)将每个实例分配到距它最近的簇中心，得到 K 个簇；2)计分别计算各簇中所有实例的均值，把它们作为各簇新的簇中心。重复 1)和 2)，直到 K 个簇中心的位置都固定，簇的分配也固定。

上述 K 均值算法只能处理数值型的属性，遇到分类型的属性时要把它变为若干个取值 0 和 1 的属性。WEKA 将自动实施这个分类型到数值型的变换，而且 WEKA 会自动对数值型的数据作标准化。因此，对于原始数据“bank-data.csv”，我们所做的预处理只是删去属性“id”，保存为 ARFF 格式后，修改属性“children”为分类型。这样得到的数据文件为“bank.arff”，含 600 条实例。

用“Explorer”打开刚才得到的“bank.arff”，并切换到“Cluster”。点“Choose”按钮选择“SimpleKMeans”，这是 WEKA 中实现 K 均值的算法。点击旁边的文本框，修改“numClusters”为 6，说明我们希望把这 600 条实例聚成 6 类，即 K=6。下面的“seed”参数是要设置一个随机种子，依此产生一个随机数，用来得到 K 均值算法中第一次给出的 K 个簇中心的位置。我们不妨暂时让它就为 10。

选中“Cluster Mode”的“Use training set”，点击“Start”按钮，观察右边“Clusterer output”给出的聚类结果。也可以在左下角“Result list”中这次产生的结果上点右键，“View in separate window”在新窗口中浏览结果。

6.2.结果解释

首先我们注意到结果中有这么一行：

Within cluster sum of squared errors: 1604.7416693522332

这是评价聚类好坏的标准，数值越小说明同一簇实例之间的距离越小。也许你得到的数值会不一样；实际上如果把“seed”参数改一下，得到的这个数值就可能会不一样。我们应该多尝试几个 seed，并采纳这个数值最小的那个结果。例如我让“seed”取 100，就得到

Within cluster sum of squared errors: 1555.6241507629218

我该取后面这个。当然再尝试几个 seed，这个数值可能会更小。

接下来“Cluster centroids:”之后列出了各个簇中心的位置。对于数值型的属性，簇中心就是它的均值（Mean）；分类型的就是它的众数（Mode）， 也就是说这个属性上取值为众数值的实例最多。对于数值型的属性，还给出了它在各个簇里的标准差（Std Devs）。

最后的“Clustered Instances”是各个簇中实例的数目及百分比。

为了观察可视化的聚类结果，我们在左下方“Result list”列出的结果上右击，点“Visualize cluster assignments”。弹出的窗口给出了各实例的散点图。最上方的两个框是选择横坐标和纵坐标，第二行的“color”是散点图着色的依据，默认是根据不同的簇“Cluster”给实例标上不同的颜色。

可以在这里点“Save”把聚类结果保存成 ARFF 文件。在这个新的 ARFF 文件中，“instance_number”属性表示某实例的编号，“Cluster”属性表示聚类算法给出的该实例所在的簇。

----整理自 http://maya.cs.depaul.edu/~classes/ect584/WEKA/k-means.html

7.Weka 连接数据库

7.1.Weka 连接 SQL Server2000 数据库

1 安装驱动程序，SQL Server2000将三个.jar 加到环境变量。

2 修改 weka\experiment 下的 DatabaseUtils.props 文件。

我们可以看到有 DatabaseUtils.props.odbc DatabaseUtils.props.oracle 等

我们先将 DatabaseUtils.props 随便改成一个其他的名字，然后将 DatabaseUtils.props.mssqlserver 改成 DatabaseUtils.props，

打开现在的 DatabaseUtils.props 可以看到以下部分：（#表示注释）

2.1驱动加载

```
# JDBC driver (comma-separated list)

jdbcDriver=com.microsoft.jdbc.sqlserver.SQLServerDriver
```

2.2数据库连接，如果在本机上可以将 server_name 改为127.0.0.1或者 localhost

```
# database URL

jdbcURL=jdbc:sqlserver://127.0.0.1:1433
```

2.3数据类型的转换。由于 weka 仅支持名词型（nominal）、数值型（numeric）、字符串、日期（date）。所以我们要将现在数据库中的数据类型对应到这四种类型上来。

将以下数据类型对应的句子前面的注释符合去掉。由于 SQL Server2000有其他的数据类型 Weka 尚不能识别，所以我们在下面再添加上

smallint=3

datetime=8等等

string,getString()= 0; -->nominal

boolean,getBoolean() = 1; -->nominal

double,getDouble() = 2; -->numeric

byte,getByte() = 3; -->numeric

short,getByte()= 4; -->numeric

int,getInteger() = 5; -->numeric

long,getLong() = 6; -->numeric

gloat,getFloat() = 7; -->numeric

date,getDate() = 8; -->date

varchar=0

float=2

tinyint=3

int=5

3其他说明，我们暂时用不到，不用去管了

other options

CREATE_DOUBLE=DOUBLE PRECISION

CREATE_STRING=VARCHAR(8000)

CREATE_INT=INT

checkUpperCaseNames=false

checkLowerCaseNames=false

checkForTable=true

4 OK，下面可以操作了！运行 weka 的 Explore 界面后，通过 Open DB..打开 SQL Viewer 工作界面（3.5.5 版本比3.4.10在这里精细了许多）。

通过 user 我们设置好用户名和密码后即可 connect；连接成功后，可以通过书写 sql 语句查询出想要的结果后，OK 即可在 Explore 界面的

preprocoss 面板中看到了输入的数据。

在连接读取数据库的数据时，SQL Viewer 面板也提供了 Info，相当与我们单纯用 jdbc 连接数据库时的调试信息。

下面是我在网上下的一个工具（java 源码，下载于 souceforge 网站），可以把数据库中的数据转换为 Weka

使用的.arff 文件。当然，前提是安装了该数据库的驱程。有兴趣者可以一看。

7.2.WEKA 连接 mysql 数据库

按照 WEKA 官方的说法，要让 WKEA 直接读取数据库中的记录，只需要找到该数据库的 JDBC 驱动，然后适当修改 DatabaseUtils.props 文件即可。（http://weka.sourceforge.net/wiki/index.php/Databases）

① Windows 下 mysql 5.0的配置

② 1.下载 mysql-connector-java-3.1.12-bin.jar,据说 weka 不支持太高的版本,所以我下载这个比较早期的版本.

③ 2.将 mysql-connector-java-3.1.12-bin.jar 加入到 classpath 中.

④ 在 windows 中:我的电脑/属性/高级/环境变量/在 classpath 的最后加上;\YourPath\mysql-connector-java-3.1.12-bin.jar(最前面有个;不能掉)

⑤ 在 Linux 中,su - vi /etc/profile,在 classpath 最后加上:/YourPath/mysql-connector-java-3.1.12-bin.jar(最前面有个:不能掉)

⑥ Windows 最好重启一下,Linux 退出控制台,然后再进入控制台即可.

⑦ 3. (本步不是必须的).提取 DatabaseUtils.props,解压 weka.jar,将 experiment/DatabaseUtils.props 拷贝到 weka 的安装目录下.其它的文件可删除.修改 DatabaseUtils.props,(以 Linux 为例,Windows 可参照进行)

⑧ 1)在 jdbcDriver=RmiJdbc.RJDriver,jdbc.idbDriver,org.gjt.mm.mysql.Driver,com.mckoi.... 这行前加一个#号注释掉.

⑨ 2)去掉 jdbcDriver=org.gjt.mm.mysql.Driver 前面的#.

⑩ 3)在#jdbcURL=jdbc:idb=experiments.prp 前加#号,注释掉本行.

⑪ 4)jdbcURL=jdbc:mysql://localhost:3306/test 去掉本行前的#号,使之起作用.请将 localhost 改成你 mysql 服务器的地址,test 改成你要链接的数据库实例.

⑫ 4. 此时应该能正常使用 WeKa 的数据库功能.点击 Explorer 时如果提示找不到 org.gjt.mm.mysql.Driver,将 mysql- connector-java-3.1.12-bin.jar 拷贝到$JAVA_HOME/jre/lib/ext/下面即可.

⑬

⑭

⑮ 5.链接数据库成功后,如果数据库表字段与 weka 不同时,要在 DatabaseUtils.props 里做转换,方法如下:

⑯ 1)weka 连接数据库

⑰ 2)desc tableName,这样就知道了该表中有什么字段.对照 DatabaseUtils.props 中已经定义的转换,就知道还有什么字段是没转换的.

⑱ 3)查找 TIMESTAMP=8,然后将要转换的字段列在下面:

⑲ ========

⑳ TIMESTAMP=8

㉑ BLOB=1

㉒ .....

㉓ ========

㉔ 4)如果字段名里有空格,用下划线或反斜线代替空格即可,如:

㉕ INTEGER_UNSIGNED=5

㉖ 然后重新连接数据库,select * from yourTable.点击 ok.数据就导入了.

㉗ Linux 下 weka 3.5访问本机 mysql 数据库，数据库名为 stocks。注意 weka 3.4版的界面和功能都有所区别，例如3.4版无法支持数据库 DATETIME 型的字段。

第一步，下载 mysql 的 jdbc 驱动，在这个页面：http://dev.mysql.com/downloads/connector/j/3.1.html。注意 weka 不支持5.0/5.1版的驱动，要下这个3.1版的。下载得到的文件中找到"mysql-connector-java-3.1.xx-bin.jar"，这里 xx 可能是某个数字，当前的最高版本是14。这个就是 JDBC 驱动了。

把 weka.jar 文件解压缩到一个目录，例如解压到 weka_3.5这个目录。在 weka_3.5下新建一个目录为 lib，lib 下放入驱动文件。Weka_3.5中新建一个可执行文件为 weka.sh（windows 下面就是批处理文件 weka.bat）。这样，目录结构为

```
weka_3.5
|__weka
    |__associations
    |__attributeSelection
    |__classifiers
    |__...
|__lib
    |__mysql-connector-java-3.1.14-bin.jar
|__weka.sh
```

第二步，找到 weka/experiment/DatabaseUtils.props 文件，在开始一段是如下内容

代码:

```
# The comma-separated list of jdbc drivers to use
#jdbcDriver=RmiJdbc.RJDriver,jdbc.idbDriver
#jdbcDriver=jdbc.idbDriver
jdbcDriver=RmiJdbc.RJDriver,jdbc.idbDriver,org.gjt.mm.mysql.Driver,com.mckoi.JDBCDrive
r,org.hsqldb.jdbcDriver
#jdbcDriver=org.gjt.mm.mysql.Driver

# The url to the experiment database
#jdbcURL=jdbc:rmi://expserver/jdbc:idb=experiments.prp
jdbcURL=jdbc:idb=experiments.prp
#jdbcURL=jdbc:mysql://mysqlserver/username
```

我们将之修改为适合 mysql 的内容，也就是：

代码:

```
# url:        http://www.mysql.com/
# jdbc:       http://www.mysql.com/products/connector/j/
# author:   Fracpete (fracpete at waikato dot ac dot nz)
# version: $Revision: 1.2 $

# JDBC driver (comma-separated list)
jdbcDriver=org.gjt.mm.mysql.Driver

# database URL
jdbcURL=jdbc:mysql://localhost:3306/stocks
```

如果数据库中有某些类型的字段在这个文件中没有声明，例如 DATETIME 型的字段，则还需要在

代码:

```
LONG=6
REAL=7
DATE=8
TIME=8
TIMESTAMP=8
```

之后添加上

代码:

```
DATETIME=8
```

这里8是 WEKA 里日期型的意思。

第三步，修改 weka.sh（或者 windows 下的 weka.bat）。我这里 weka_3.5目录的路径是 "/opt/weka/weka_3.5"，那么 weka.sh 的内容为

代码:

```
#!/bin/sh

WEKA_HOME="/opt/weka/weka_3.5"
JDBC_PATH="$WEKA_HOME/lib/mysql-connector-java-3.1.14-bin.jar"
java -Xmx512M -classpath $WEKA_HOME:$JDBC_PATH weka.gui.Main
```

windows 下上述代码要稍作修改。

第四步，运行 weka.sh，选择 explorer，点击"Open DB..."，出现"SQL-Viewer"。点击"User"按钮填入数据库的用户名和密码，然后点击"Connect"按钮。连接成功的话在下方"Info"一栏中会有提示。

连上后在"Query"一栏中写入 SQL 查询语句，例如

代码：全选

```
select * from small;
```

然后点右边的"Excute"按钮。（如果连接数据库不成功是无法 Excute 的）

"Result"一栏中会显示有关结果，满意后点击最下方的"OK"按钮。Explorer 就从数据库中载入数据集了。

7.3.WEKA 连接 oracle 数据库

7.3.1.Windows 版本的 Oracle

操作系统：windows xp sp2，weka-3-5-7 连接 oracle 数据库说明
下载的是 weka-3-5-7.zip，直接解压缩。
weka-3-5-7目录下有 README 文件，这里面有一些有用的信息。

1、copy oracle 的驱动程序 ojdbc14.jar 到 weka-3-5-7目录，然后把 D:\Program Files\Weka-3-5\ojdbc14.jar 添加的系统的 CLASSPATH 里面。
2、解压缩 weka.jar（用 winrar 打开即可），找到 weka.jar\weka\experiment 目录，里面有 DatabaseUtil.props 文件。因为我们连接的是 oracle，所以 copy DatabaseUtil.props.oracle 出来就可以了。
3、修改 DatabaseUtil.props.oracle
jdbcURL=你自己的连接串
打开 # specific data types 一行下面对应配置的注释(就是去掉 #)
我的是这样：

```
# Database settings for Oracle 10g Express Edition
#
# url: http://www.oracle.com/
# jdbc: http://www.oracle.com/technology/softwa ... sqlj_jdbc/
# author: Fracpete (fracpete at waikato dot ac dot nz)
# version: $Revision: 1.3 $

# JDBC driver (comma-separated list)
```

```
jdbcDriver=oracle.jdbc.driver.OracleDriver

# database URL
jdbcURL=jdbc:oracle:thin:@localhost:1521:oracle

# specific data types
string, getString() = 0; --> nominal
boolean, getBoolean() = 1; --> nominal
double, getDouble() = 2; --> numeric
byte, getByte() = 3; --> numeric
short, getByte()= 4; --> numeric
int, getInteger() = 5; --> numeric
long, getLong() = 6; --> numeric
gloat, getFloat() = 7; --> numeric
date, getDate() = 8; --> date
text, getString() = 9; --> string
VARCHAR2=0
NUMBER=2
DOUBLE_PRECISION=2
TIMESTAMP=8

# other options
CREATE_INT=INTEGER
CREATE_STRING=VARCHAR2(4000)
CREATE_DOUBLE=NUMBER
checkUpperCaseNames=true
checkForTable=true
```

4、把修改后的 DatabaseUtil.props.oracle 改名为 DatabaseUtil.props 再放回 weka.jar\weka\experiment 目录里覆盖原来的文件。README 里提到了如何在外部目录的放置，但我没试验通过，只好把它 copy 回 weka.jar 里了。

5、启动 weka.jar

写个 RunWekaEx.bat，放置到 weka-3-5-7目录下

注意 javaw 的路径是你机器的实际路径

```
D:\jdk\jdk1.5.0_09\bin\javaw.exe -cp .;ojdbc14.jar;weka.jar weka.gui.GUIChooser
```

6、启动之后现在打开 weka--〉applications-->explor-->opendb 就会发现连接信息了。点 USER user/password 输入，点 Connect 连接成功。

注意：网上有很多方式是需要修改 weka.jar 并需要重新打包的，我觉得对我们初学的人来说并不实用。而且我的这个方法也确实是有效的。还有一点就是有的方法只能在 witn console 下使用，但是我经过测试，两种打开方式都是可以的。

7.3.2.Linux 版本的 Oracle

jdk 和 oracle 版本和 windows 是一样的，基本上和 windows 下是一致的，但是也有问题，就是在 windows 下，当我们通过快捷方式打开 weka 的时候，其实很多配置信息已经在 RunWeka.bat 以及 RunWeka.ini 里面预先设置好了，所以如果你通过命令行执行 java -jar weka.jar 的时候，由于绕过了前面两个文件所以连接数据库的时候是会报找不到合适的驱动的错误的。而在 linux 系统下，就只有 java -jar weka.jar 这么一个方法启动程序，所以我们需要手工写一个脚本来启动 weka，以便在其中配置信息。

具体步骤：

(1)新建一个 weka.sh 文件

(2)在里面输入

```
WEKA_HOME="/usr/local/source/weka/weka-3-5-8"
JDBC_PATH="$WEKA_HOME/ojdbc.jar"
java -Xmx512M -classpath $WEKA_HOME:$JDBC_PATH weka.gui.Main
```

保存好以后，通过./weka.sh 就可以打开 weka，并且联接数据库了。

不足：我在两个系统上面配置的这两个 weka，不知道为什么，都只能够连接本机的数据库，一旦通过网络连接其他数据库，必定报错：

exception: java.sql.SQLException: Io 异常: The Network Adapter could not establish the connection

7.4.一个操作实例

1.准备

Windows XP

jdk-1_5_0_14

weka-3-5-7.exe

SQLServer2005

mysql-6.0.0

Oracle10.2.0.1.0

Microsoft SQL Server 2005 JDBC Driver 1.2--->sqljdbc.jar

MySQL Driver for JDBC--->mysql-connector-java-5.1.6-bin.jar

Oracle Driver for JDBC--->ojdbc14.jar

2.双击 weka-3-5-7.exe 安装 weka

3.进入 weka 安装目录

3.1.解压缩 weka.jar

解压后的目录结构

[Weka-3-5]

|____...

|____[weka]

|____[META-INF]

|____...

|____[weka]

|____...

|____...

3.2.新建 lib 目录,将数据库 Driver for JDBC(jar 包)拷贝进\lib

完成后的目录结构

[Weka-3-5]

|____...

|____[weka]

|____[META-INF]

|____...

|____[weka]

|____...

|____[lib]

|____mysql-connector-java-5.1.6-bin.jar

|____ojdbc14.jar

|____sqljdbc.jar

|____...

4.设置环境变量

WEKA_HOME

C:\Program Files\Weka-3-5

ClassPath

.;%WEKA_HOME%\lib\sqljdbc.jar;%WEKA_HOME%\lib\mysql-connector-java-5.1.6-bin.jar;%WEKA_HOME%\lib\ojdbc14.jar;%JAVA_HOME%\lib\tools.jar;%JAVA_HOME%\lib\dt.jar

设置完成后,weka 就能找到放在\lib 中的数据库 jar 包了.

5.修改 DatabaseUtils.props

进入%WEKA_HOME%\weka\weka\experiment\你会看到:...

DatabaseUtils.props

DatabaseUtils.props.hsql

DatabaseUtils.props.mssqlserver2005

DatabaseUtils.props.mssqlserver

DatabaseUtils.props.mysql

DatabaseUtils.props.odbc

DatabaseUtils.props.oracle

DatabaseUtils.props.postgresql

...

weka 运行时会使用 DatabaseUtils.props

其他的如:'DatabaseUtils.props.数据库名称'(这些是 weka 提供的针对不同数据库提供的模板)

我们先将 DatabaseUtils.props 随便改成一个其他的名字,如:DatabaseUtils.props.sample

然后将 DatabaseUtils.props.mysql 改成 DatabaseUtils.props(假设我们需要连接 mysql 数据库)

打开现在的 DatabaseUtils.props 可以看到以下部分:(#表示注释)[小弟的注释]

[版本信息]

Database settings for MySQL 3.23.x, 4.x

[小弟连接的是 MySQL6所以改成-->]# Database settings for MySQL 6.x]

#

url: http://www.mysql.com/

jdbc: http://www.mysql.com/products/connector/j/

author: Fracpete (fracpete at waikato dot ac dot nz)

version: $Revision: 1.3 $

[JDBC 版本-->]# version: $Revision: 5.1 $]

JDBC driver (comma-separated list)

jdbcDriver=org.gjt.mm.mysql.Driver

[修改为-->jdbcDriver=com.mysql.jdbc.Driver]

database URL

jdbcURL=jdbc:mysql://server_name:3306/database_name

[这 个建议不修改,方便后面进入 weka 后,通过修改相应的'server_name','datebase_name'来连接相应的 mysql 数据库.其实 大家在这里像这样子 jdbcURL=jdbc:mysql://localhost:3306/foodmart 写死了也没什么,进入 weka 后同样可以修改,但显得不够专业不是!~]

```
# specific data types
# string, getString() = 0;    --> nominal
# boolean, getBoolean() = 1;  --> nominal
# double, getDouble() = 2;    --> numeric
# byte, getByte() = 3;        --> numeric
# short, getByte()= 4;        --> numeric
# int, getInteger() = 5;      --> numeric
# long, getLong() = 6;        --> numeric
# gloat, getFloat() = 7;      --> numeric
# date, getDate() = 8;        --> date
# text, getString() = 9;      --> string
```

[呵呵,这里是重点!由于 weka 仅支持名词型(nominal),数值型(numeric),字符串(string),日期(date).所以我们要将现在数据库中的数据类型对应到这四种类型上来.]

[将上面的内容改成:

```
# specific data types
 string, getString() = 0;    --> nominal
 boolean, getBoolean() = 1;  --> nominal
 double, getDouble() = 2;    --> numeric
 byte, getByte() = 3;        --> numeric
 short, getByte()= 4;        --> numeric
 int, getInteger() = 5;      --> numeric
 long, getLong() = 6;        --> numeric
 gloat, getFloat() = 7;      --> numeric
```

```
 date, getDate() = 8;         --> date
 text, getString() = 9;       --> string
TINYINT=3
SMALLINT=4
#SHORT=4
SHORT=5
INTEGER=5
INT=5
BIGINT=6
LONG=6
REAL=7
NUMERIC=2
DECIMAL=2
FLOAT=2
DOUBLE=2
CHAR=0
TEXT=0
VARCHAR=0
LONGVARCHAR=9
BINARY=0
VARBINARY=0
LONGVARBINARY=9
BIT=1
BLOB=9
DATE=8
TIME=8
DATETIME=8
TIMESTAMP=8
```

这里参考了一些网友的帖子,自己 google 了一些,这里 MySQL 常用的数据类型都设置好了,再也不用担心 weka 不识别对应的数据类型了

大家注意,上面有部分'#'要去掉哦!

在附录中会提供小弟为大家精心准备的 DatabaseUtils.props 文件:

DatabaseUtils.props.mssqlserver2005_ok

DatabaseUtils.props.mysql6_ok

DatabaseUtils.props.oracle10g_ok

文件名大家随意,使用的时候记得改成 DatabaseUtils.props 就好]

```
# other options
CREATE_DOUBLE=DOUBLE
```

```
CREATE_STRING=TEXT
CREATE_INT=INT
checkUpperCaseNames=false
checkLowerCaseNames=false
checkForTable=true
```

[其他设置,暂时不用修改]

6.制作 weka.jar 并替换原来的 jar

因为 weka 软件运行时需要读取 weka.jar,所以你修改之后要重新打包 jar 文件替换原来的 jar 才可以运行 weka 软件成功连接数据库.

6.1.从命令行进入%WEKA_HOME%\weka

6.2.执行 jar cvf weka.jar weka*.*

6.3.进入%WEKA_HOME%\weka 会发现打包好了的 weka.jar(没有的请刷新一下)

6.4.将%WEKA_HOME%\weka 下的 weka.jar 复制到%WEKA_HOME%(建议将原来的 weka.jar 改名成 weka.jar.sample 备用,大家今后如果针对不同数据库创建了多个 weka.jar 不妨将其改名成 -->'weka.jar.数据库名 ',用的时候将后缀去掉就行,体力活咱做一次就够了!

7.运行 weka

奇怪的问题:运行-->Weka 3.5(不带控制台)进入 weka 连不上数据库(mysql,oracle,sqlserver 都不行),说找不到合适的 JDBC DRIVER.但运行-->Weka 3.5 (with console)则全部正常.期待达人解答!

不理它,能用就行,毕竟现在还附送个'控制台'!

7.1.运行-->Weka 3.5 (with console)

7.2.选择 Applications--->Explorer

7.3.选择 Open DB...

7.4.选择 User...

根据自己的情况修改 Database URL,Username,Password.

7.5.选择 Connect

注意窗口下方的 Info 里的信息!

... = true ---->恭喜你,连接成功!~

... = false ---->失败!~别灰心,向上一步步地检查,你离 true 不远了!~

7.6.连接成功后光标会自动选择 Query 栏,等着各位兄台来输入 sql 语句.小弟输入一个超简单的,然后选择 Execute 执行 sql 语句.

7.7.执行成功后在 Result 栏中会有数据显示.

7.8.选择 OK,呵呵!~weka 已经捕获了相关数据,并显示相关信息,接下来各位爱怎么玩,就怎么玩!~

7.9.如果我不写 sql 语句,在连接成功后直接选择 OK,会怎么样?嘿嘿,weka 会说连接数据库有问题,没有合适的驱动.什么也不显示.所以还是告诉它我们需要哪些数据,不然接下来就没得玩了啦

7.5.Weka 数据库配置文件实例

这里给大家提供较为完整的配置方案,基本上够用.其中 specific data types 部分参照了各个数据库的数据类型说明,不常用的数据类型没有列出.(当然,有一部分比较 bt 的数据类型实在是不知道让 weka 如何对应,如果哪 位高手有更全面的设置,欢迎提供!~)

7.5.1.DatabaseUtils.props.mssqlserver2005_ok

Database settings for Microsoft SQL Server 2005 Express Edition

```
#
# url:      http://www.microsoft.com/
# jdbc:     http://msdn2.microsoft.com/en-us/data/aa937724.aspx
# author:   Fracpete (fracpete at waikato dot ac dot nz)
# version: $Revision: 1.2 $
# JDBC driver (comma-separated list)
jdbcDriver=com.microsoft.sqlserver.jdbc.SQLServerDriver
# database URL
jdbcURL=jdbc:sqlserver://server_name;databaseName=database_name
# specific data types
 string, getString() = 0;     --> nominal
 boolean, getBoolean() = 1;   --> nominal
 double, getDouble() = 2;     --> numeric
 byte, getByte() = 3;         --> numeric
 short, getByte()= 4;         --> numeric
 int, getInteger() = 5;       --> numeric
 long, getLong() = 6;         --> numeric
 gloat, getFloat() = 7;       --> numeric
 date, getDate() = 8;         --> date
 text, getString() = 9;       --> string
bit=1
tinyint=3
smallint=4
int=5
```

```
bigint=6
smallmoney=2
money=2
numeric=2
decimal=2
float=2
real=2
smalldatetime=8
datetime=8
timestamp=8
char=0
text=0
varchar=0
nchar=0
ntext=0
nvarchar=0
binary=0
varbinary=0
image=0
uniqueidentifier=9
rowversion=9
# other options
CREATE_DOUBLE=DOUBLE PRECISION
CREATE_STRING=VARCHAR(8000)
CREATE_INT=INT
checkUpperCaseNames=false
checkLowerCaseNames=false
checkForTable=true
```

7.5.2.DatabaseUtils.props.mysql6_ok

```
# Database settings for MySQL 6.x
#
# url:     http://www.mysql.com/
# jdbc:    http://www.mysql.com/products/connector/j/
# author:  Fracpete (fracpete at waikato dot ac dot nz)
# version: $Revision: 5.1 $
# JDBC driver (comma-separated list)
jdbcDriver=com.mysql.jdbc.Driver
# database URL
jdbcURL=jdbc:mysql://server_name:3306/database_name
# specific data types
 string, getString() = 0;    --> nominal
 boolean, getBoolean() = 1;  --> nominal
```

```
 double, getDouble() = 2;     --> numeric
 byte, getByte() = 3;         --> numeric
 short, getByte()= 4;         --> numeric
 int, getInteger() = 5;       --> numeric
 long, getLong() = 6;         --> numeric
 gloat, getFloat() = 7;       --> numeric
 date, getDate() = 8;         --> date
 text, getString() = 9;       --> string
TINYINT=3
SMALLINT=4
#SHORT=4
SHORT=5
INTEGER=5
INT=5
BIGINT=6
LONG=6
REAL=7
NUMERIC=2
DECIMAL=2
FLOAT=2
DOUBLE=2
CHAR=0
TEXT=0
VARCHAR=0
LONGVARCHAR=9
BINARY=0
VARBINARY=0
LONGVARBINARY=9
BIT=1
BLOB=9
DATE=8
TIME=8
DATETIME=8
TIMESTAMP=8
# other options
CREATE_DOUBLE=DOUBLE
CREATE_STRING=TEXT
CREATE_INT=INT
checkUpperCaseNames=false
checkLowerCaseNames=false
checkForTable=true
```

7.5.3.DatabaseUtils.props.oracle10g_ok

```
# Database settings for Oracle 10g Express Edition
#
# url:     http://www.oracle.com/
# jdbc:
http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/
# author:  Fracpete (fracpete at waikato dot ac dot nz)
# version: $Revision: 1.3 $
# JDBC driver (comma-separated list)
jdbcDriver=oracle.jdbc.driver.OracleDriver
# database URL
jdbcURL=jdbc:oracle:thin:@server_name:1521:database_name
# specific data types
 string, getString() = 0;    --> nominal
 boolean, getBoolean() = 1;  --> nominal
 double, getDouble() = 2;    --> numeric
 byte, getByte() = 3;        --> numeric
 short, getByte()= 4;        --> numeric
 int, getInteger() = 5;      --> numeric
 long, getLong() = 6;        --> numeric
 gloat, getFloat() = 7;      --> numeric
 date, getDate() = 8;        --> date
 text, getString() = 9;      --> string
CHAR=0
NCHAR=0
VARCHAR2=0
NVARCHAR2=0
RAW=9
NUMBER=2
BINARY_FLOAT=2
DATE=8
TIMESTAMP=8
ROWID=9
DOUBLE_PRECISION=2
# other options
CREATE_INT=INTEGER
CREATE_STRING=VARCHAR2(4000)
CREATE_DOUBLE=NUMBER
checkUpperCaseNames=true
checkForTable=true
```

8.WEKA 环境构建

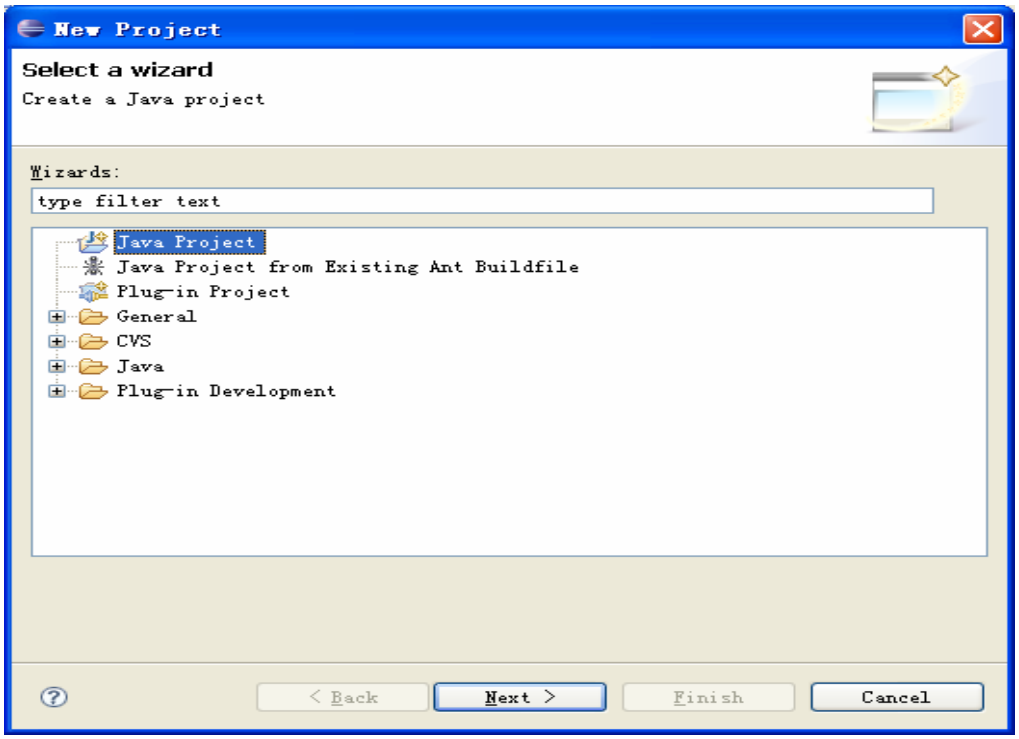
8.1.在 Eclipse 中配置 Weka

首先讲讲在 eclipse 中如何配置weka 吧！先配置好java 环境，设置好路径，安装eclipse, 这里我用的是jdk1.5, Eclipse3.2。感谢bbs.wekacn.org。

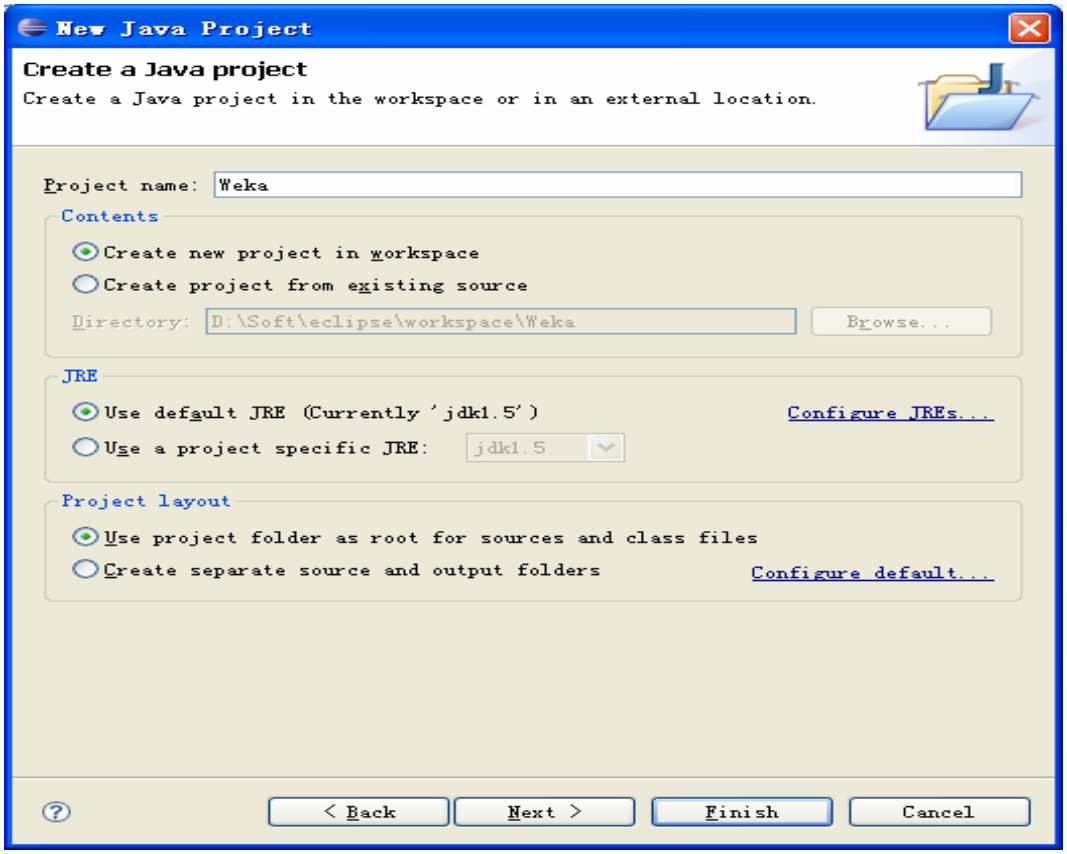
先在 WEKA 的官方网页上下载WEKAhttp://www.cs.waikato.ac.nz/ml/weka/ 的安装程序，为了和书一致 (Ian H. Witten and Eibe Frank (2005) "Data Mining: Practical machine learningtools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco, 2005. 国内有翻译的版本) ，请下载book version, 3.4.10.

安装后，在安装目录下有个weka-src.jar 包，里面是源代码，用jar –vxf weak-src.jar 解压缩，或者用winzip, winrar 也可以解开。

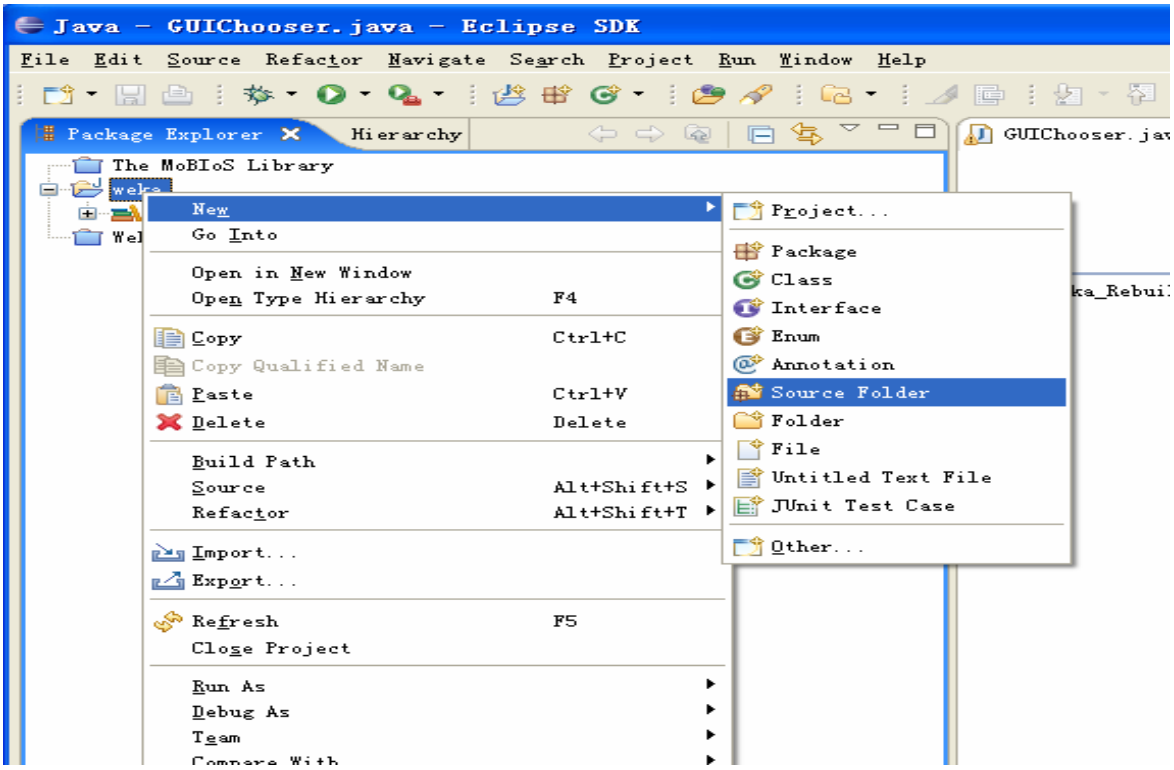
打开 Eclipse, 然后新建一个java 工程， 点下一步.

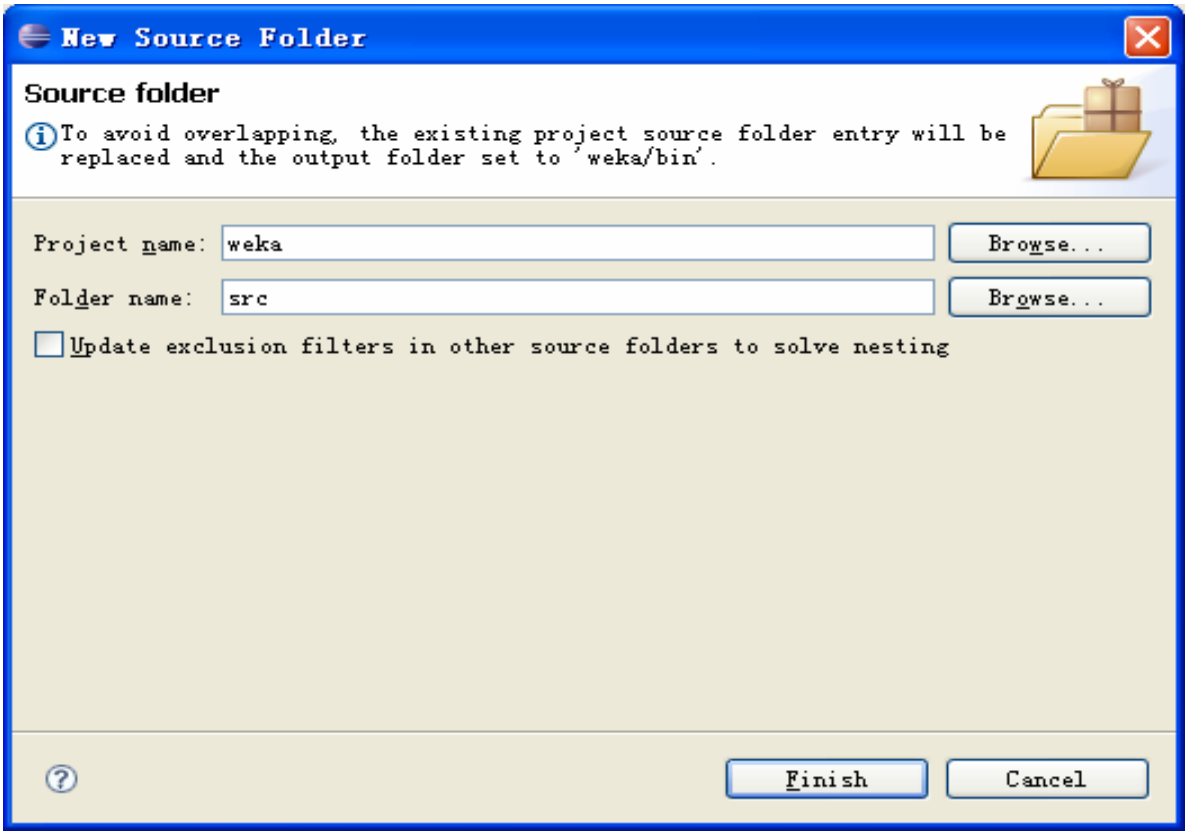


然后在工程文件的名字上随便填写 Weka, 然后配置一下jre（如果有需要的话），点击完成。

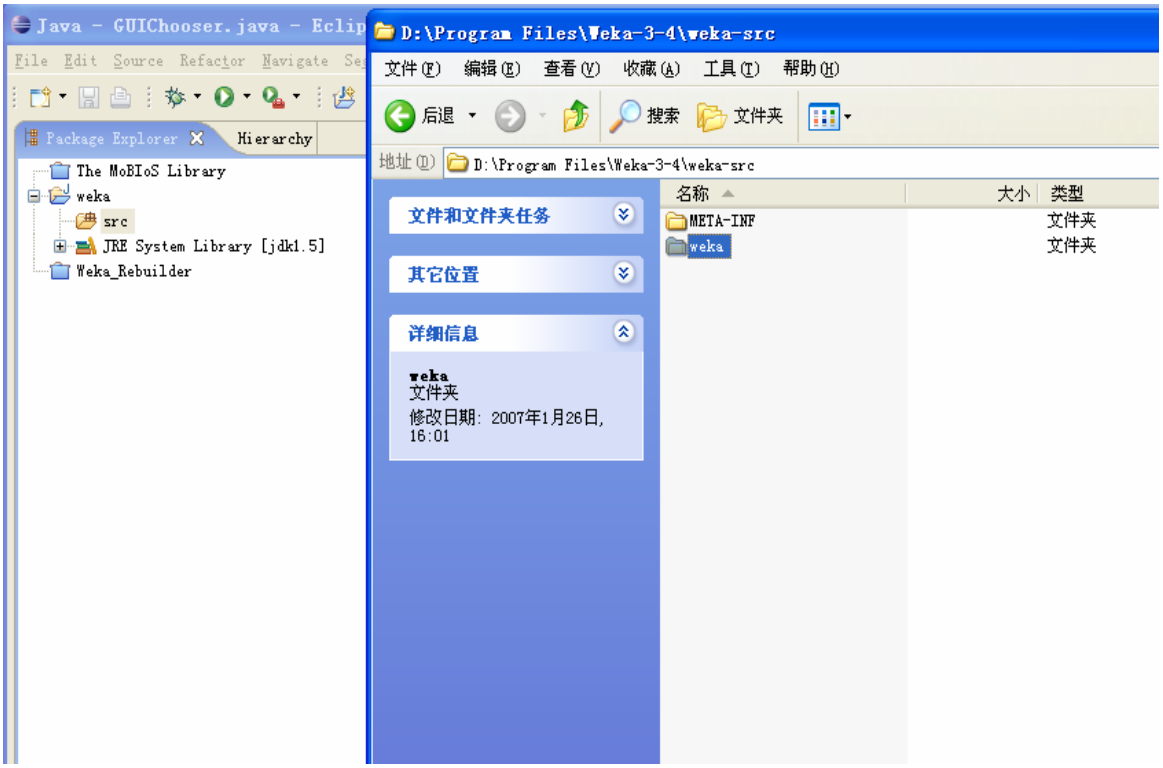


然后选中刚才建立的 Weka 项目，按鼠标右键，新建一个”source folder” 文件夹src 用来存放源代码。

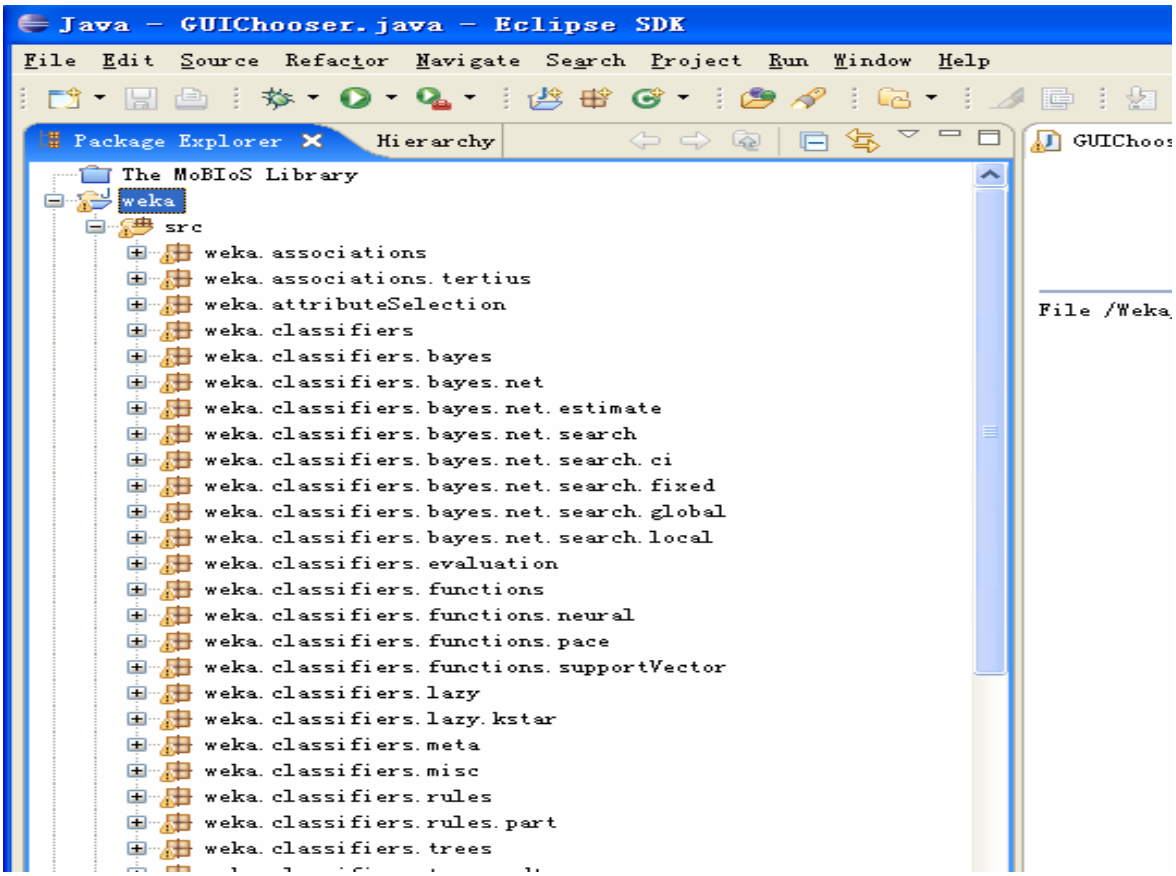




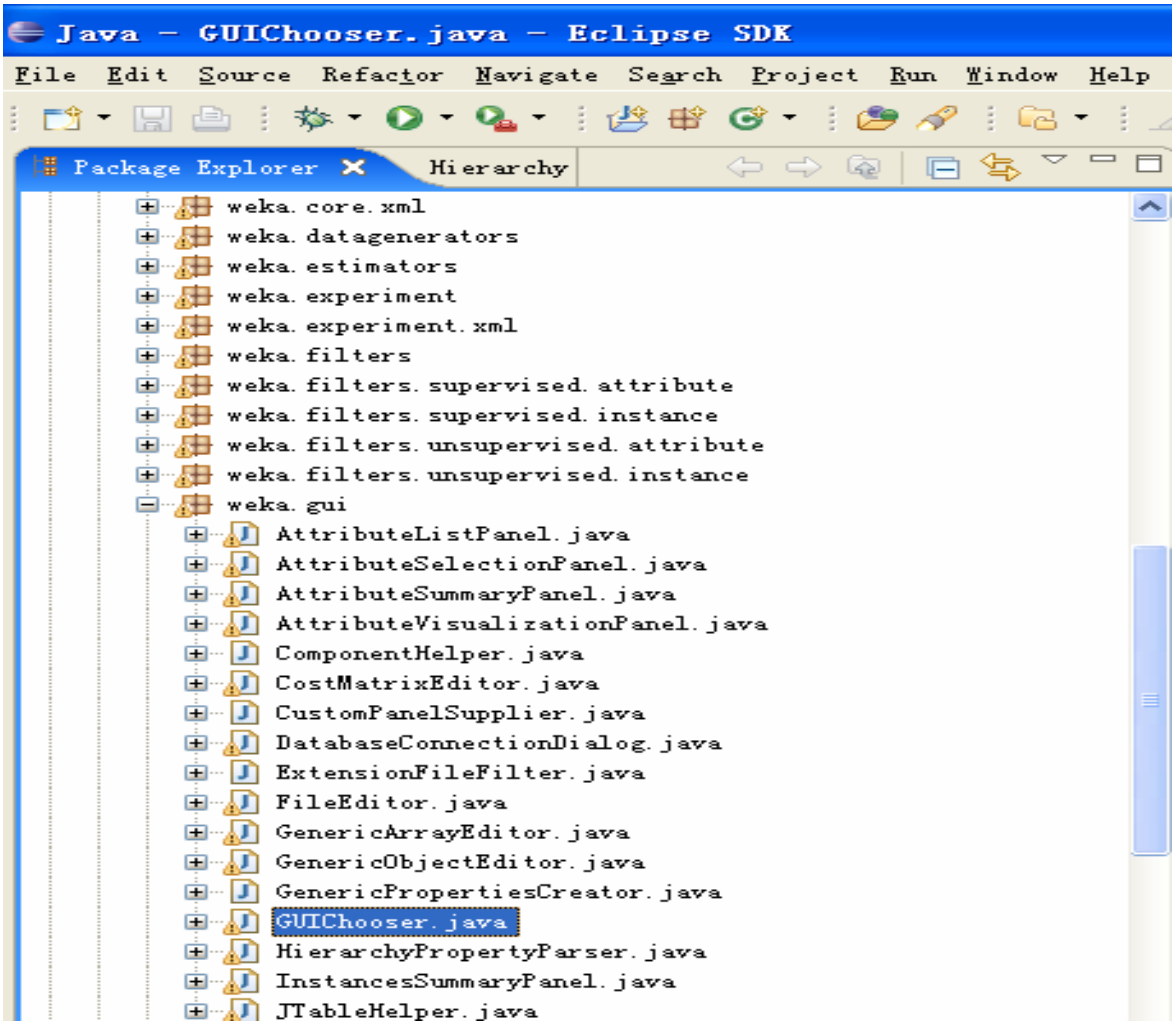
然后把刚才解开压缩的源文件所在的目录，即 weka 目录，在资源管理器中拖入到src 目录中， 如下图：



拖入完毕后，eclipse 开始拷贝源代码到工作目录下，等待一会，完成后，如图所示：



里面有很多 warning, 不要管他，如果要在eclipse 中运行weka, 那么请打开weka.gui 包，找到GUIChooser， 这个文件中含有main.。



然后按鼠标右键，选择 run as java application 即可。很简单吧！

8.1.1.在 eclipse 下使用 java 调用 weka

要使用程序方式使用 weka，步骤如下：

一、在 eclipse 里新建一个 java project：

1. 建立工程：单击菜单中 file->new->java project，在弹出对话框的 project name 中起任意一个名字，此处假设是 wekaTest。单击 Finish 按钮（在对话框底部）。

2. 建立 package：在 package Explorer 中找到刚才新建的工程，在其上右键->New->package。在 Name 文本框里面输入名称，此处假设为 Test。单击 Finish 按钮。

3. 建立程序文件:在刚才新建的 package 上面右键->New->class,选中 public static void main(String[] args)多选框，单击 Finish。

二、在该工程中添加 weka 的引用：

1. package Explorer 中工程名上右键，选择弹出菜单最后一项 properties->在左面选中 java Build Path->在右面的 Library 页面->单击 Add External JARs...->浏览 weka 所在目录，将 weka.jar 添加进来，然后单击 ok。

2. 在 package Explorer 中在双击 Test 文件，然后在 package wekaTest;一句下面添加四句代码：

```
import java.io.File;
```

```
import weka.classifiers.Classifier;
import weka.classifiers.trees.J48;
import weka.core.Instances;
import weka.core.converters.ArffLoader;
```

三、在程序中添加 weka 调用代码：

将以下代码添加到 Main 函数中（在// TODO Auto-generated method stub 下面）：

```
    Classifier m_classifier = new J48();
        File inputFile = new File("D:\\Program Files\\Weka-3-6\\data\\cpu.with.vendor.arff");//训练语料文件
        ArffLoader atf = new ArffLoader();
        atf.setFile(inputFile);
        Instances instancesTrain = atf.getDataSet(); // 读入训练文件
        inputFile = new File("D:\\Program Files\\Weka-3-6\\data\\cpu.with.vendor.arff");//测试语料文件
        atf.setFile(inputFile);
        Instances instancesTest = atf.getDataSet(); // 读入测试文件
        instancesTest.setClassIndex(0);//设置分类属性所在行号（第一行为 0 号），instancesTest.numAttributes()可以取得属性总数
        double sum = instancesTest.numInstances(),//测试语料实例数
        right = 0.0f;
        instancesTrain.setClassIndex(0);
         m_classifier.buildClassifier(instancesTrain); //训练
        for(int  i = 0;i<sum;i++)//测试分类结果
        {

if(m_classifier.classifyInstance(instancesTest.instance(i))==instancesTest.instance(i).classValue())// 如果预测值和答案值相等（测试语料中的分类列提供的须为正确答案，结果才有意义）
            {
              right++;//正确值加 1
            }
        }
        System.out.println("J48 classification precision:"+(right/sum));
```

四、运行一下试试。

8.2.在 windows_JCreator 中建立 weka 开发环境的建立

说明：1、本文假设你已经安装了 JDK，并获得了 weka 的安装程序。

2、关于 weka 源码：安装目录下有个 weka-src.jar 解压后就是源码。

3、DMman 对 JCreator 的若干细节不甚了解，只是勉强在其下运行起来了 weka。

WEKA 开发环境的建立(Windows+JCreator 版)步骤：

1、将 weka-src.jar 解压后得到源码，位于 weka 目录下。在本地建立目录，我们假设建立了 C：\myweka；在其下建立文件夹 src、classes，将含有源码的 weka 目录放到 src 下。

2、打开 JCreator，新建工程，选择 Basic Java Application，然后出现选择目录。请注意 这里很重要,否则得不到原来的层次架构。Name myweka

Location c:\myweka\

SourcePath c:\myweka\src

OutPut c:\myweka\classes

选择 建立一个新的工作区（不选择 添加到原来的工作区）然后 finish 即可。

3、在项目上点右键-编译项目。编译可以通过。这时 JCreator 自己生产了 2 个文件 myweka.java mywekaFrame.java

如果你此时使用“运行项目”命令的话，将出现他们的界面（不知如何解决）。

可以在 JCreator 窗口目录下找到 src-weka-gui-GUIChooser.java 选中它,然后 使用 “运行文件”命令运行它，这时会弹出很多异常，不用惧。

这是因为你自己编译的 classes 目录下少些东西，不是.class 的文件。请根据异常的提示对照一个 weka 安装目录下 weka.jar 解压后的 classes 目录，看看少些什么，拷贝到你自己的 classes 下就可以了。 比如 weka.gui 下有个文件夹 images，以及几个图片和几个.pros 文件；weka.gui.beans 文件夹下有个 icons 文件夹；weka.core 下面有个 vesion.txt 等等。

这时，你再去运行文件 weka.gui.GUIChooser.java就可以了！同时你也可以运行 weka.gui.SimpleCLI.java，weka。gui.Exporer.Exporer,java等分别看到几个主要的界面。

参考文献

http://bbs.wekacn.org/viewtopic.php?t=43&sid=8b7b6e0884af8cb0f4c730d3b51bfab8。

9.WEKA 二次开发

9.1.如何向 WEKA 中加入新算法

编写新算法，所编写的新算法必须符合 Weka 的接口标准。在此以从 Weka 中文站上下载的一个算法（模糊 C 均值聚类算法：FuzzyCMeans）的添加为例说明其具体过程。

2. 由于 FuzzyCMeans 是聚类算法，所以直接将 FuzzyCMeans.java 源程序考到 weka.clusterers 包下

3.再修改 weka.gui.GenericObjectEditor.props ，在#Lists the Clusterers I want to choose from 的 weka.clusterers.Clusterer=\下加入：weka.clusterers.FuzzyCMeans

4.相应的修改 weka.gui.GenericPropertiesCreator.props ，此去不用修改，因为包 weka.clusterers 已经存在，若加入新的包时则必须修改这里，加入新的包

这样加入之后，重新编译，运行后，可以在 weka 的 Explorer 界面上的 Cluster 选项卡中的聚类算法中找到刚刚新添加的 FuzzyCMeans 算法。

添加过程简单吧！关键问题是要弄清楚 Weka 的内核以及其接口标准，然后编写出符合此规范的新算法。

10.附录

10.1.WEKA 常见问题解答(FAQ)

1:WEKA 去哪里下载？
官方的下载网址是 http://www.cs.waikato.ac.nz/ml/weka/ind ... ading.html
需要你根据自己的操作系统和 JRE 条件选择合适的下载。如果你用 Windows 并且不知道什么叫 JRE，那么下载那个"a self-extracting executable that includes Java VM 1.4"。
本站提供镜像，但不保证是最新的版本。http://www.wekacn.org/download/

2:怎么我下载的 WEKA 不能用？
下载后不能用可能有两个原因。
一个可能是直接在下载软件里下载了那个网址以"prdownloads"开头的 exe 文件。其实这是一个网页文件，你在 IE 中打开它可以看到很多下载点，选一个才可以正常下载。
如果你下载的文件有若干 M 却不能运行，很可能是因为你下载了不含 JRE 的版本而你的机器上没有 JRE。去下那个"includes Java VM"的吧。
运行出现问题可参考问题 7。

3:WEKA 作者写的那本书哪里可以找到？
买！China-pub 有影印版以及中文版。
嗯。。。买不起的话，在论坛里找找看看。

4:调整 weka 的可用内存数量：
需要调整 java 的可用内存（Memery，记忆体）。方法如下。
如果你使用较新版本的 WEKA，则打开 WEKA 安装目录下的 Runweka.ini 文件，找到这一行：
maxheap=128m
把 128 修改为你想要的 Java 可用内存，但最好不要超过机器的物理内存大小。
如果使用比较老的版本，则安装目录下没有 Runweka.ini。这时打开 Runweka.bat 文件，找到
java -Xmx128m
把 128 改成你想要的内存大小。
如果在 weka 下面没有找到所说的 runweka.ini 或者 runweka.bat 怎么办？
确定是 weka 安装目录下？
确定是 windows 系统？
确定明白文件名中扩展名的概念？
还是不行就换个更高版本

5:WEKA 怎么做关联规则/购物栏分析/Association Rules？
一般之所以会有这个问题是因为 WEKA 提示内存不足。把不关注的事件（没有购买那件商品），表示为一个缺失值，有助于减少内存。例如下面这个 ARFF 文件
代码:
@relation store
@attribute milk {no,yes}
@attribute egg {no,yes}

....
@data
yes,no,yes,no,yes,no,no.....no
.....
建议修改为
代码:
@relation store
@attribute 22 {yes}
@attribute 23 {yes}
...
@data
yes,?,yes,?,yes,?,?,...,?
.....
还可以考虑把数据用稀疏格式表示。

6:Simple CLI 为什么不能输入命令？
在最下方的文本框中输入。

7:运行 WEKA 为什么提示 Could not find the main class ？
这种情况一般发生在自带 JVM 的 WEKA 版本上，如果你自带的 JRE 是 1.4 版的也可能出现这种情况。原因是较新版本 WEKA 的 class 文件是在 1.5（5.0）下编译的，与 1.4 的 JRE 不兼容。
解决方案参考
viewtopic.php?t=42

8:哪里可以找到 Weka 中 XXX 算法的有关资料？
打开该算法的配置面板，点击"More"按钮，在这里一般有这个算法的参考文献。

9:算法的源码在什么地方能看到 要是对算法进行修改 该如何操作？
在 weka-src.jar 中能找到（在 weka 的安装目录下），把 weka-src.jar 包解压后装载到 myeclipse 下，就可以看到源码，并且可以对算法进行修改。

10:大数据集与商业应用问题？
问题：WEKA 在处理大数据集方面如何？比如 G 级的数据？
它足够成熟到进行实际商业应用吗？
答：weka 中很多算法是基于内存的运算，G 级的数据会有问题。而它的 KnowledgeFlow 模块有一些支持增量计算的功能，也许可以满足你的需求。

11:请问怎样用 weka 对数据进行神经网络训练？
请问怎样用 weka 对数据进行神经网络训练，神经网络是有反馈结果的（有监督的学习，不同于 k-means）。我刚用 weka，找了好久未找到，求高手帮助。
答：我也是一直在寻找 weka 中的神经网络训练，看到一本书里写到，在分类器中可以找到 multilayerperceptron，这个事神经网络的反向传播算法。我在书上看到了，你可以看看，应该有！

12：用 Weka J48 生成的决策树如何修剪？
在 Weka 中，用 J48 算法得到了分类树，可是分类树太过复杂，如何进行修剪，还是这些规则要

全部使用啊！
答：可以通过设置 J48 的 confidenceFactor 参数来决定生成的决策树的复杂程度。

13：weka 中怎样做 jakknief 检验？将 Cross-Validation folds n=文件中的 instance 数目么？
答：设置 WEKA 的 test options 里面的 CV 数目，并不能用来衡量预测的误差，只能算是验证误差，因为我们是根据 CV 后跑的结果来优化参数。所以我做检验的时候还是自己分组的数据。

14: 分享一下 Jason Brownlee 先生为 WEKA 写的"神经网络"插件
这个是 Jason Brownlee 先生写的包括神经网络算法，其中包含 lvq、单层感知器、多层感知器等分类算法. 如果还没有 WEKA 工具的朋友，请到:http://sourceforge.net/projects/weka 或者官网:http://www.cs.waikato.ac.nz/ml/weka/下载，然后安装到指定的目录，安装完毕后到其目录下找到 weka-src.jar 文件，解压，然后按照 http://bbs.wekacn.org/viewtopic.php?t=148 此帖把 WEKA 开发环境搭建好.
接下来我们把附件里的文件解压，然后把解压后的文件里的 wekaclassalgos-src.zip 解压，解压得的文件拖到 Eclipse 建立好的 WEKA 环境的 src 源文件包下，如果有提示要覆盖的点"是/yes"，这样就完整的，成功的把神经网络的插件源代码成功导入.
最后，我们找到 weka.gui 包下的 GUIChooser.java 文件，运行它即可.
如果不想搭建源代码工程的朋友，只是想用在已安装好的 WEKA 上运行该插件，请根据解压附件后里边的 wekaclassalgos_readme.txt 文件进行安装插件.
作者会不断的更新插件，请到 http://sourceforge.net/projects/wekaclassalgos 查看

15：一个 Weka 源码分析的博客
http://blogger.org.cn/blog/blog.asp?subjectid=2725&name=DMman

10.2.Weka 网络资源

WEKA 连接数据库指南（mysql 版）

http://bbs2.wekacn.org/viewtopic.php?f=2&t=216&sid=13cdd0c42079a4b719d5d54b83855780

weka 连接 mysql 数据库(windows xp 版),无需解压缩,无需配置 DatabaseUtils.props 文件

http://bbs2.wekacn.org/viewtopic.php?f=2&t=293&sid=13cdd0c42079a4b719d5d54b83855780

ps:小弟刚开始就是使用上面的方法(由于需要对%WEKA_HOME%\RunWeka.bat 进行修改,一味的追求连上数据库,破坏了源程序的完整性.就算配好了,更换数据库时便需要再度对其修改.因此不推荐大家使用),运气不好,没整出来!~

数据挖掘青年(DMman)

Weka 如何连接数据库

http://blogger.org.cn/blog/more.asp?name=DMman&id=24991