

Weka & Rapid Miner Tutorial

Original version by Chibuike Muoh
Modified by DBDMG

WEKA:: Installation

- Download software from <http://www.cs.waikato.ac.nz/ml/weka/>
 - If you are interested in modifying/extending Weka there is a developer version that includes the source code
- Set the Weka environment variable for java
 - `setenv WEKAHOME /usr/local/weka/weka-3-0-2`
 - `setenv CLASSPATH $WEKAHOME/weka.jar:$CLASSPATH`
- Download some data from <http://mlearn.ics.uci.edu/MLRepository.html>

WEKA:: Introduction

- A collection of open source ML algorithms
 - pre-processing
 - classifiers
 - clustering
 - association rule
- Created by researchers at the University of Waikato in New Zealand
- Java based

WEKA:: Introduction .contd

- Routines are implemented as classes and logically arranged in packages
- Comes with an extensive GUI interface
- Weka routines can be used stand alone via the command line
- Weka routines can be used in your own Java code

WEKA:: Data format

- Uses flat text files to describe the data
- Can work with a wide variety of data files including its own “.arff” format and C4.5 file formats
- Data can be imported from a file in various formats:
 - ARFF, CSV, C4.5, binary
- Data can also be read from a URL or from an SQL database (using JDBC)

WEKA:: ARRF file format

```
@relation heart-disease-simplified

@attribute age numeric
@attribute sex { female, male}
@attribute chest_pain_type { typ_angina, asympt, non_anginal, atyp_angina}
@attribute cholesterol numeric
@attribute exercise_induced_angina { no, yes}
@attribute class { present, not_present}

@data
63,male,typ_angina,233,no,not_present
67,male,asympt,286,yes,present
67,male,asympt,229,yes,present
38,female,non_anginal,?,no,not_present
...
```

A more thorough description is available here
<http://www.cs.waikato.ac.nz/~ml/weka/arff.html>

WEKA:: Example

- Using weka from command line
- Example
 - Decision tree: J48
 - Training data: iris.arff
 - Cross-validation (which is the default option)

WEKA:: Example

- Using weka from command line
- The command is

```
java -classpath weka.jar
  weka.classifiers.trees.J48 -t iris.arff
```
- The output is
 - the generated model
 - the cross-validation results

WEKA:: Example

- The model

```

petalwidth <= 0.6: Iris-setosa (50.0)
  petalwidth > 0.6
    | petalwidth <= 1.7
    | | petallength <= 4.9: Iris-versicolor (48.0/1.0)
    | | petallength > 4.9
    | | | petalwidth <= 1.5: Iris-virginica (3.0)
    | | | petalwidth > 1.5: Iris-versicolor (3.0/1.0)
    | petalwidth > 1.7: Iris-virginica (46.0/1.0)
  
```

WEKA:: Example

- The results

```

=== Stratified cross-validation ===
Correctly Classified Instances      144  96%
Incorrectly Classified Instances     6    4%
.....
=== Confusion Matrix ===
 a  b  c  <-- classified as
49  1  0  | a = Iris-setosa
 0 47  3  | b = Iris-versicolor
 0  2 48  | c = Iris-virginica
  
```

WEKA:: Interface

- Simple **CLI** provides a command line interface to weka's routines
- **Explorer** interface provides a graphical front end to weka's routines and components
- **Experimenter** allows you to build classification experiments
- **KnowledgeFlow** provides an alternative to the Explorer as a graphical front end to Weka's core algorithms with drag and drop support.

WEKA:: Interface

The screenshot displays the 'Weka GUI Chooser' window. At the top, it reads 'Waikato Environment for Knowledge Analysis (c) 1999 - 2003 University of Waikato New Zealand' and features a central image of a kiwi bird. Below the image, four buttons are arranged in a 2x2 grid: 'Simple CLI', 'Explorer', 'Experimenter', and 'KnowledgeFlow'. A label 'GUI' is positioned above these buttons. Four blue arrows point from the 'Simple CLI' and 'Explorer' buttons to the corresponding windows shown in the background. The 'Simple CLI' window shows a terminal with instructions. The 'Explorer' window shows a data table with columns for 'Name', 'Age', 'Sex', 'Length', 'Wing', 'Tail', and 'Beak'. The 'Experimenter' window shows a complex workflow diagram with nodes for 'DataLoader', 'AttributeSelector', 'Classifier', and 'ClassifierEvaluator'. The 'KnowledgeFlow' window shows a similar workflow diagram with nodes for 'DataLoader', 'AttributeSelector', 'Classifier', and 'ClassifierEvaluator'.

WEKA:: Explorer: Preprocessing

- Pre-processing tools in WEKA are called “filters”
- WEKA contains filters for:
 - Discretization, normalization, resampling, **attribute selection**, transforming, combining attributes, etc
- Open a file
- Choose the filter
- Select “Apply”

Weka Knowledge Explorer

Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Open file... Open URL... Open DB... Undo Save...

Filter: Choose None Apply

Current relation: Relation: iris Instances: 150 Attributes: 5

No.	Name
1	sepalength
2	sepalwidth
3	petalength
4	petalwidth
5	class

Selected attribute: Name: sepalength Type: Numeric Missing: 0 (0%) Distinct: 35 Unique: 9 (6%)

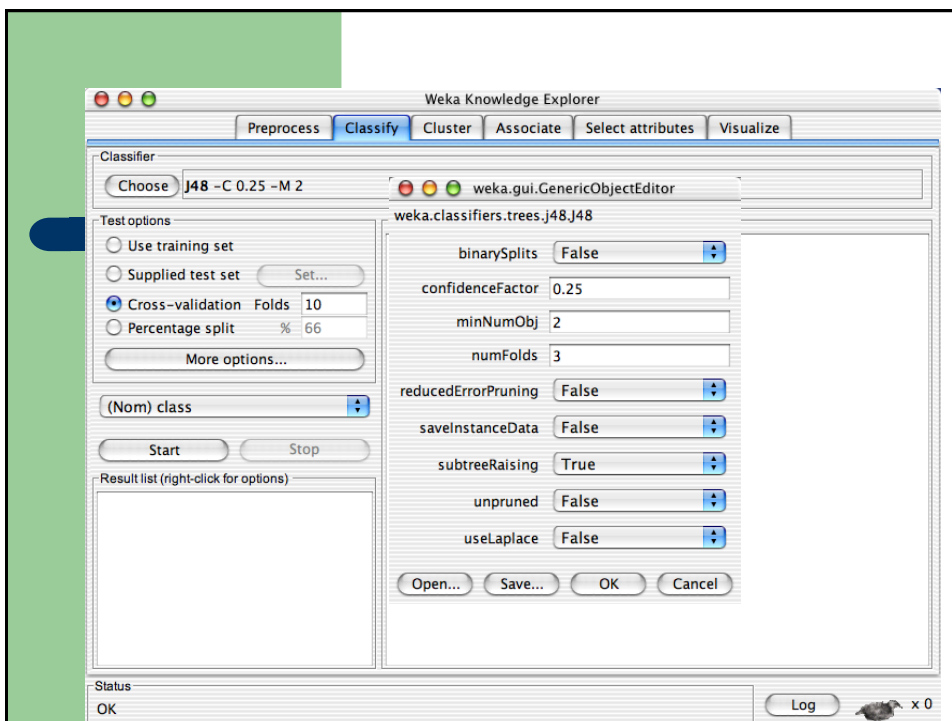
Statistic	Value
Minimum	4.3
Maximum	7.9
Mean	5.843
StdDev	0.828

Colour: class (Nom) Visualize All

Status: OK Log x 0

WEKA:: Explorer: building “classifiers”

- Classifiers in WEKA are models for predicting nominal or numeric quantities
- Implemented learning schemes include:
 - Decision trees and lists, instance-based classifiers, support vector machines, multi-layer perceptrons, logistic regression, Bayes’ nets, ...
- “Meta”-classifiers include:
 - Bagging, boosting, stacking, error-correcting output codes, locally weighted learning, ...



WEKA:: Using Weka in your Java code

- Main steps
 1. Creating an **Instances** object to store the dataset
 2. Possible data discretization
 3. Data mining algorithm application
 4. Visualize and save the results

WEKA:: Explorer

- Example showing simple analysis on the Iris dataset

RapidMiner:: Introduction

- A very comprehensive open-source software implementing tools for
 - intelligent data analysis, data mining, knowledge discovery, machine learning, predictive analytics, forecasting, and analytics in business intelligence (BI).
- Is implemented in Java and available under GPL among other licenses
- Available from <http://rapid-i.com>

RapidMiner:: Intro. Contd.

- Is similar in spirit to Weka's Knowledge flow
- Data mining processes/routines are views as sequential operators
 - Knowledge discovery process are modeled as operator chains/trees
 - Operators define their expected inputs and delivered outputs as well as their parameters
- Has over 400 data mining operators

RapidMiner:: Intro. Contd.

- Uses XML for describing operator trees in the KD process
- Alternatively can be started through the command line and passed the XML process file

RapidMiner:: Screenshot

