Advanced Data Mining with Weka

Class 4 – Lesson 1

What is distributed Weka?

Mark Hall

Pentaho
Lesson 4.1: What is distributed Weka?

Class 1 Time series forecasting

Class 2 Data stream mining in Weka and MOA

Class 3 Interfacing to R and other data mining packages

Class 4 Distributed processing with Apache Spark

Class 5 Scripting Weka in Python

Lesson 4.1 What is distributed Weka?

Lesson 4.2 Installing with Apache Spark

Lesson 4.3 Using Naive Bayes and JRip

Lesson 4.4 Map tasks and Reduce tasks

Lesson 4.5 Miscellaneous capabilities

Lesson 4.6 Application: Image classification
Lesson 4.1: What is distributed Weka?

- A plugin that allows Weka algorithms to run on a cluster of machines
- Use when a dataset is too large to load into RAM on your desktop, OR
- Processing would take too long on a single machine
Lesson 4.1: What is distributed Weka?

- Class 2 covered data stream mining
  - sequential online algorithms for handling large datasets
- Distributed Weka works with distributed processing frameworks that use map-reduce
  - Suited to large offline batch-based processing
- Divide (the data) and conquer over multiple processing machines
- More on map-reduce shortly...
Lesson 4.1: What is distributed Weka?

- Two packages are needed:
  - distributedWekaBase
    - General map-reduce tasks for machine learning that are not tied to any particular map-reduce framework implementation
    - Tasks for training classifiers and clusterers, and computing summary statistics and correlations
  - distributedWekaSpark
    - A wrapper for the base tasks that works on the Spark platform
    - There is also a package (several actually) that works with Hadoop
Lesson 4.1: What is distributed Weka?

Map-reduce programs involve a “map” and “reduce” phase

Dataset → Map tasks → Reduce task(s)

- Dataset
- Data split

Map tasks
- Processing:
  - E.g. sorting, filtering, computing partial results

Reduce task(s)
- <key, result>
- Summarize:
  - E.g. counting, adding, averaging

Map-reduce frameworks provide orchestration, redundancy and fault-tolerance
Lesson 4.1: What is distributed Weka?

- **Goals of distributed Weka**
  - Provide a similar experience to that of using desktop Weka
  - Use any classification or regression learner
  - Generate output (including evaluation) that looks just like that produced by desktop Weka
  - Produce models that are normal Weka models (some caveats apply)

- **Not a goal (initially at least)**
  - Providing distributed implementations of every learning algorithm in Weka
    - One exception: k-means clustering
  - We’ll see how distributed Weka handles building models later...
Lesson 4.1: What is distributed Weka?

- What distributed Weka is
- When you would want to use it
- What map-reduce is
- Basic goals in the design of distributed Weka
Advanced Data Mining with Weka

Class 4 – Lesson 2

Installing with Apache Spark

Mark Hall

Pentaho
Lesson 4.2: Installing with Apache Spark

Class 1: Time series forecasting

Class 2: Data stream mining in Weka and MOA

Class 3: Interfacing to R and other data mining packages

Class 4: Distributed processing with Apache Spark

Class 5: Scripting Weka in Python

Lesson 4.1 What is distributed Weka?

Lesson 4.2 Installing with Apache Spark

Lesson 4.3 Using Naive Bayes and JRip

Lesson 4.4 Map tasks and Reduce tasks

Lesson 4.5 Miscellaneous capabilities

Lesson 4.6 Application: Image classification
Lesson 4.2: Installing with Apache Spark

- Install **distributedWekaSpark** via the package manager
  - This automatically installs the general framework-independent **distributedWekaBase** package as well
- Restart Weka
- Check that the package has installed and loaded properly by starting the Knowledge Flow UI
The hypothyroid data

- A benchmark dataset from the UCI machine learning repository
- Predict the type of thyroid disease a patient has
  - Input attributes: demographic and medical information
- 3772 instances with 30 attributes
- A version of this data, in CSV format without a header row, can be found in
  ${user.home}\wekafiles\packages\distributedWekaSpark\sample_data
Lesson 4.2: Installing with Apache Spark

Why CSV without a header rather than ARFF?

- Hadoop and Spark split data files up into blocks
  - Distributed storage
  - Data local processing

- There are “readers” for text files and various structured binary files
  - Maintain the integrity of individual records

- ARFF would require a special reader, due to the ARFF header only being present in one block of the data
Lesson 4.2: Installing with Apache Spark

- Getting distributed Weka installed
- Our test dataset: the hypothyroid data
- Data format processed by distributed Weka
- Distributed Weka job to generate summary statistics
Advanced Data Mining with Weka

Class 4 – Lesson 3

Using Naive Bayes and JRip

Mark Hall

Pentaho

weka.waikato.ac.nz
Lesson 4.3: Using Naive Bayes and JRip

Class 1  Time series forecasting

Class 2  Data stream mining in Weka and MOA

Class 3  Interfacing to R and other data mining packages

Class 4  Distributed processing with Apache Spark

Class 5  Scripting Weka in Python

Lesson 4.1 What is distributed Weka?

Lesson 4.2 Installing with Apache Spark

Lesson 4.3 Using Naive Bayes and JRip

Lesson 4.4 Map tasks and Reduce tasks

Lesson 4.5 Miscellaneous capabilities

Lesson 4.6 Application: Image classification
No slides for Lesson 4.3
Advanced Data Mining with Weka

Class 4 – Lesson 4

Map tasks and Reduce tasks

Mark Hall

Pentaho

weka.waikato.ac.nz
Lesson 4.4: Map tasks and Reduce tasks

Class 1  Time series forecasting

Class 2  Data stream mining in Weka and MOA

Class 3  Interfacing to R and other data mining packages

Class 4  Distributed processing with Apache Spark

Class 5  Scripting Weka in Python

Lesson 4.1 What is distributed Weka?

Lesson 4.2 Installing with Apache Spark

Lesson 4.3 Using Naive Bayes and JRip

Lesson 4.4 Map tasks and Reduce tasks

Lesson 4.5 Miscellaneous capabilities

Lesson 4.6 Application: Image classification
How is a classifier learned in Spark?

**Dataset**
- Data split
- Data split

**Map tasks**
- Learn a model
- Learn a model

**Reduce task**
- Either:
  1. Aggregate models to form one final model of the same type OR
  2. Make an ensemble classifier using all the individual models

**Results**
Lesson 4.4: Map tasks and Reduce tasks

Cross validation in Spark

- Implemented with two phases (passes over the data):
  1. Phase one: model construction
  2. Phase two: model evaluation
Lesson 4.4: Map tasks and Reduce tasks

Cross-validation in Spark phase 1: model construction

Map tasks: build partial models on parts of folds
- M1: fold 2 + 3
- M2: fold 1 + 3
- M3: fold 1 + 2

Reduce tasks: Aggregate the partial models for each fold
- M1: fold 2 + 3
- M2: fold 1 + 3
- M3: fold 1 + 2

Results
Lesson 4.4: Map tasks and Reduce tasks

Cross-validation in Spark phase 2: model evaluation

Dataset

Map tasks: evaluate fold models

Reduce task

Aggregate all partial evaluation results

Results
Lesson 4.3 & 4.4: Exploring the Knowledge Flow templates

- Creating ARFF metadata and summary statistics for a dataset
- How distributed Weka builds models
- Distributed cross-validation
Advanced Data Mining with Weka

Class 4 – Lesson 5

Miscellaneous capabilities

Mark Hall

Pentaho

weka.waikato.ac.nz
Lesson 4.5: Miscellaneous capabilities

Class 1  Time series forecasting

Class 2  Data stream mining in Weka and MOA

Class 3  Interfacing to R and other data mining packages

Class 4  Distributed processing with Apache Spark

Class 5  Scripting Weka in Python

Lesson 4.1 What is distributed Weka?

Lesson 4.2 Installing for Apache Spark

Lesson 4.3 Using Naive Bayes and JRip

Lesson 4.4 Map tasks and Reduce tasks

Lesson 4.5 Miscellaneous capabilities

Lesson 4.6 Application: Image classification
Lesson 4.5: Miscellaneous capabilities

- Computing a correlation matrix in Spark and using it as input to PCA
- Running k-means clustering in Spark
- Where to go for information on setting up Spark clusters
Lesson 4.5: Miscellaneous capabilities

Further reading

- Distributed Weka for Spark
  - http://markahall.blogspot.co.nz/2015/03/weka-and-spark.html
- Distributed Weka for Hadoop
- K-means|| clustering in distributed Weka
- Apache Spark documentation
  - http://spark.apache.org/docs/latest/
- Setting up a simple stand-alone cluster
Advanced Data Mining with Weka

Class 4 – Lesson 6

Application: Image classification

Michael Mayo

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz
Lesson 4.6: Application: Image classification
Lesson 4.6: Application: Image classification

- **Image features** are a way of describing an image using numbers.
- For example:
  - How bright is the image \( (f_1) \)?
  - How much yellow is in the image \( (f_2) \)?
  - How much green is in the image \( (f_3) \)?
  - How symmetrical is the image \( (f_4) \)?

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>50%</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>50%</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>10%</td>
</tr>
<tr>
<td>( f_4 )</td>
<td>100%</td>
</tr>
</tbody>
</table>
Lesson 4.6: Application: Image classification

- **Image filters** extract the same features for a set of images

```
@relation butterfly_vs_owl
@attribute filename string
@attribute class {BUTTERFLY, OWL}
@data
mno001.jpg, BUTTERFLY
mno002.jpg, BUTTERFLY
mno003.jpg, BUTTERFLY
mno004.jpg, BUTTERFLY
owl001.jpg, OWL
owl002.jpg, OWL
owl003.jpg, OWL
owl004.jpg, OWL
```

```
@relation butterfly_vs_owl
@attribute filename string
@attribute f1 numeric
@attribute f2 numeric
@attribute f3 numeric
@attribute class {BUTTERFLY, OWL}
@data
mno001.jpg, 3, 7, 0, BUTTERFLY
mno002.jpg, 1, 2, 0, BUTTERFLY
mno003.jpg, 3, 4, 0, BUTTERFLY
mno004.jpg, 6, 3, 0, BUTTERFLY
owl001.jpg, 3, 5, 0, OWL
owl002.jpg, 7, 3, 0, OWL
owl003.jpg, 3, 5, 0, OWL
owl004.jpg, 7, 5, 1, OWL
```
Lesson 4.6: Application: Image classification

1. Install `imageFilters` package using the Package Manager
2. Create your own ARFF file or use the example at %HOMEPATH%/wekafiles/packages/imageFilters/data
3. Open the ARFF file in the WEKA Explorer
4. Select an image filter from `filters/unsupervised/instance/imagefilter`
5. Set the filter’s `imageDirectory` option to the correct directory
6. Click the Apply button
7. Repeat 5-7 if you wish to apply more than one filter
8. (Optional) Remove the first filename attribute
9. Select a classifier and perform some experiments
Lesson 4.6: Application: Image classification

- Summary
  - Image features are mathematical properties of images
  - Image filters can be applied to calculate image features for an entire dataset of images
  - Different features measure different properties of the image
  - Experimenting with WEKA can help you identify the best combination of image feature and classifier for your data
Lesson 4.6: Application: Image classification

References


Advanced Data Mining with Weka

Department of Computer Science
University of Waikato
New Zealand

Creative Commons Attribution 3.0 Unported License
creativecommons.org/licenses/by/3.0/

weka.waikato.ac.nz