THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

More Data Mining with Weka

Class 1 – Lesson 1

Introduction

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

More Data Mining with Weka

… **a practical course on how to**
use advanced facilities of Weka for data mining
(but not programming, just the interactive interfaces)

… **follows on from *Data Mining with Weka***

… will pick up some basic principles along the way

Ian H. Witten
University of Waikato, New Zealand

More Data Mining with Weka

- **This course assumes that you know about**
  - *What data mining is and why it's useful*
  - *The "simplicity-first" paradigm*
  - *Installing Weka and using the Explorer interface*
  - *Some popular classifier algorithms and filter methods*
  - *Using classifiers and filters in Weka …*
    and how to find out more about them
  - *Evaluating the result, including training/testing pitfalls*
  - *Interpret Weka's output and visualizing your data set*
  - *The overall data mining process*
- **See *Data Mining with Weka***
- **(Refresher: see videos on YouTube WekaMOOC channel)**

More Data Mining with Weka

- **As you know, a Weka is**
  - *a bird found only in New Zealand?*
  - ***Data mining workbench**:*
    Waikato Environment for Knowledge Analysis

Machine learning algorithms for data mining tasks

- 100+ algorithms for classification
- 75 for data preprocessing
- 25 to assist with feature selection
- 20 for clustering, finding association rules, etc

More Data Mining with Weka

What will you learn?

- *Experimenter, Knowledge Flow interface, Command Line interfaces*
- *Dealing with "big data"*
- *Text mining*
- *Supervised and unsupervised filters*
- *All about discretization, and sampling*
- *Attribute selection methods*
- *Meta-classifiers for attribute selection and filtering*
- *All about classification rules: rules vs. trees, producing rules*
- *Association rules and clustering*
- *Cost-sensitive evaluation and classification*

Use Weka on your own data … and understand what you're doing!

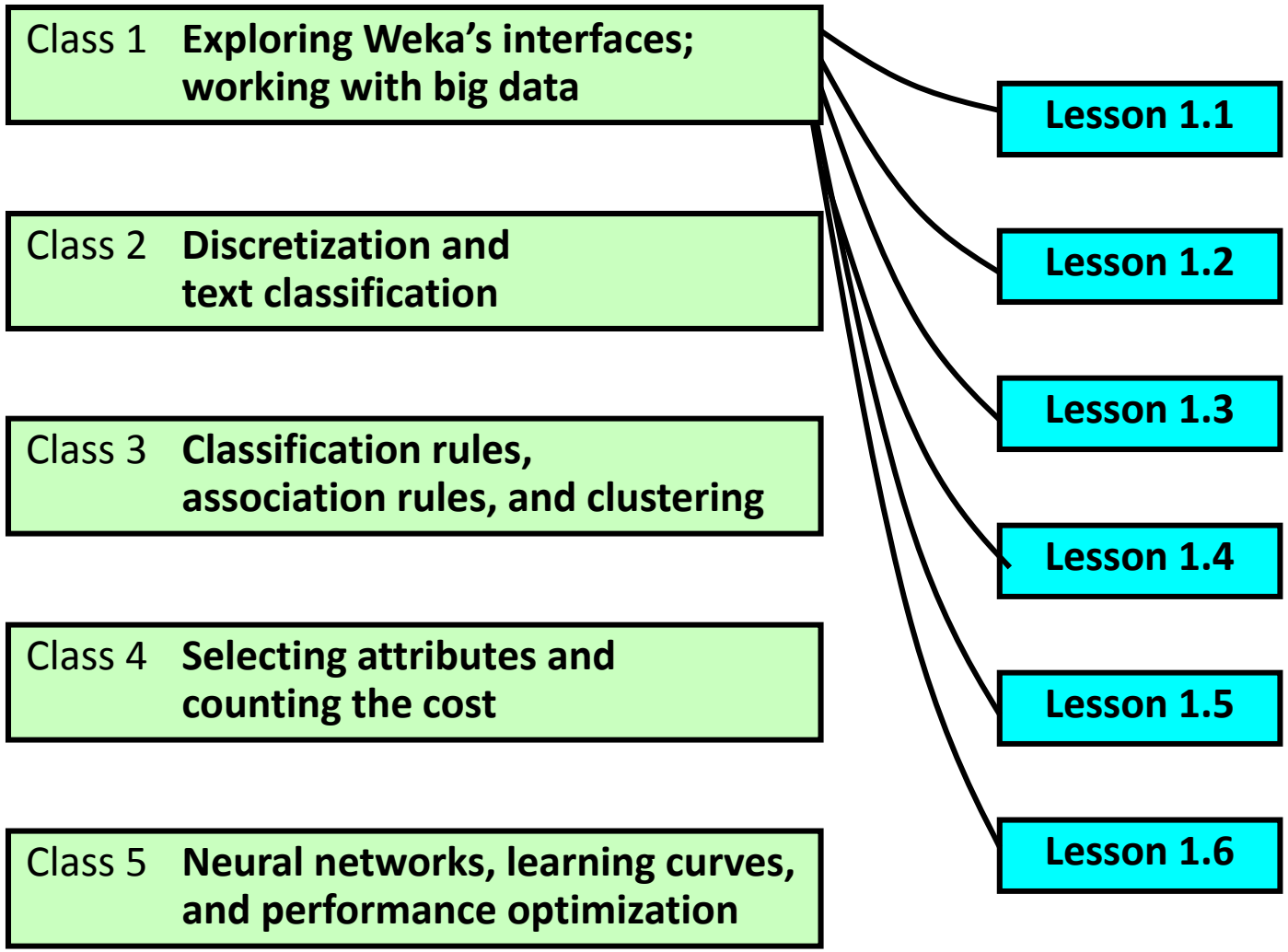
Class 1: Exploring Weka's interfaces, and working with big data

- Experimenter interface
- Using the Experimenter to compare classifiers
- Knowledge Flow interface
- Simple Command Line interface
- Working with big data
  - *Explorer: 1 million instances, 25 attributes*
  - *Command line interface: effectively unlimited*
  - *in the Activity you will process a multi-million-instance dataset*

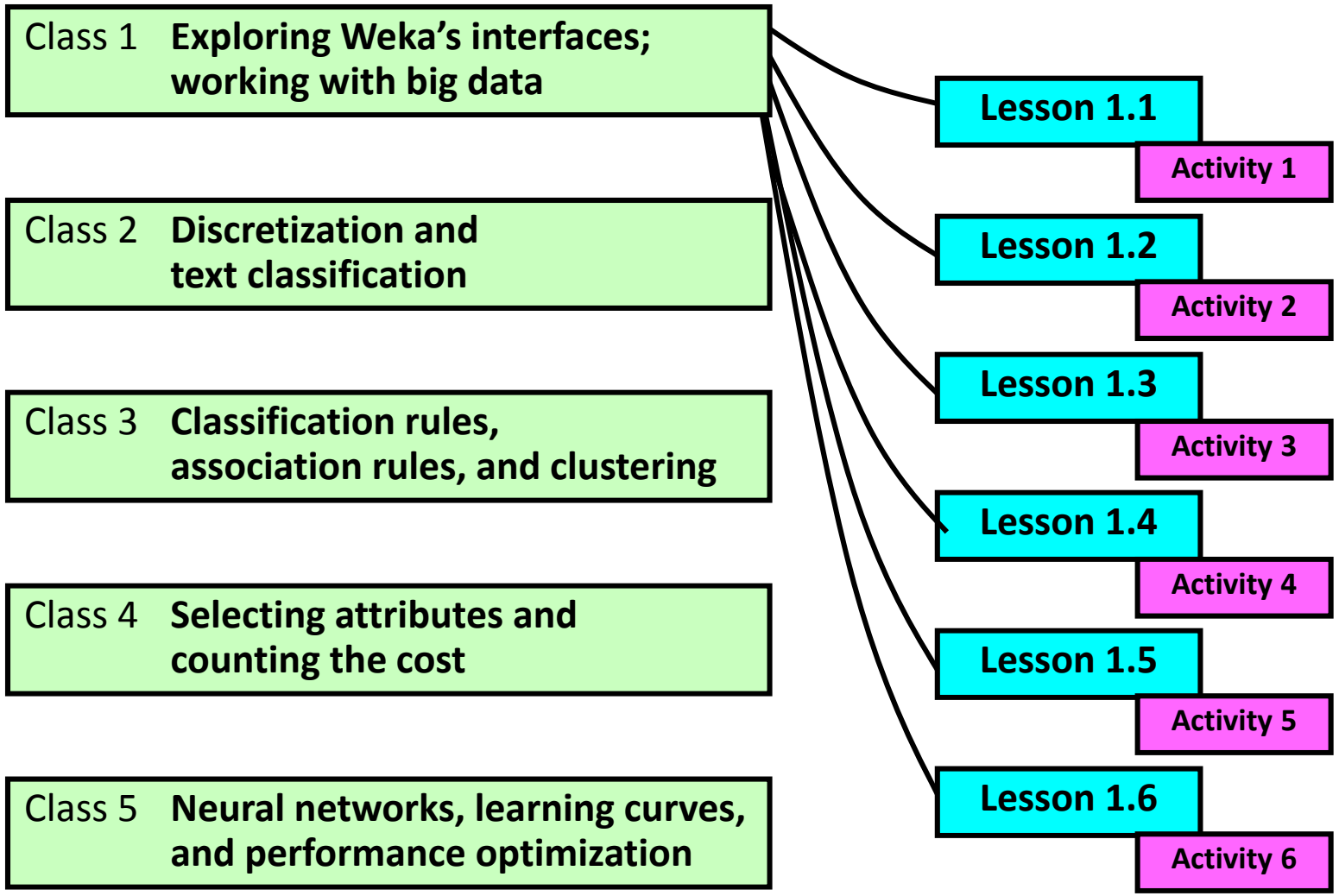
Course organization

Class 1	**Exploring Weka's interfaces; working with big data**
Class 2	**Discretization and text classification**
Class 3	**Classification rules, association rules, and clustering**
Class 4	**Selecting attributes and counting the cost**
Class 5	**Neural networks, learning curves, and performance optimization**

Course organization



Course organization



Course organization

Class 1	**Exploring Weka's interfaces; working with big data**
Class 2	**Discretization and text classification**
	→ **Mid-class assessment** 1/3
Class 3	**Classification rules, association rules, and clustering**
Class 4	**Selecting attributes and counting the cost**
Class 5	**Neural networks, learning curves, and performance optimization**
	→ **Post-class assessment** 2/3

Download Weka now!

Download from

http://www.cs.waikato.ac.nz/ml/weka

for Windows, Mac, Linux

Weka 3.6.11

the latest stable version of Weka

includes datasets for the course

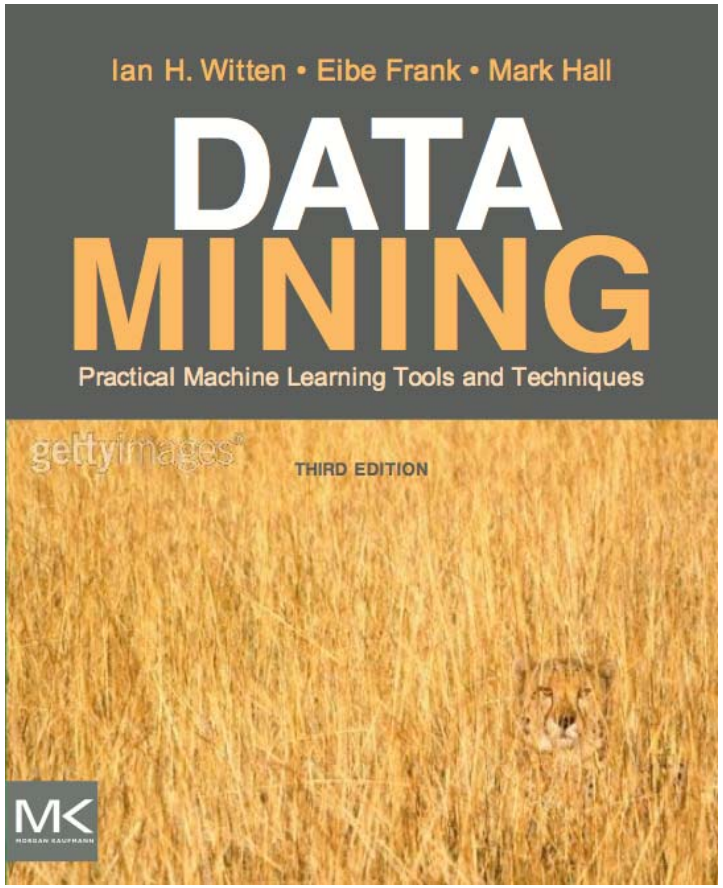
it's important to get the right version!

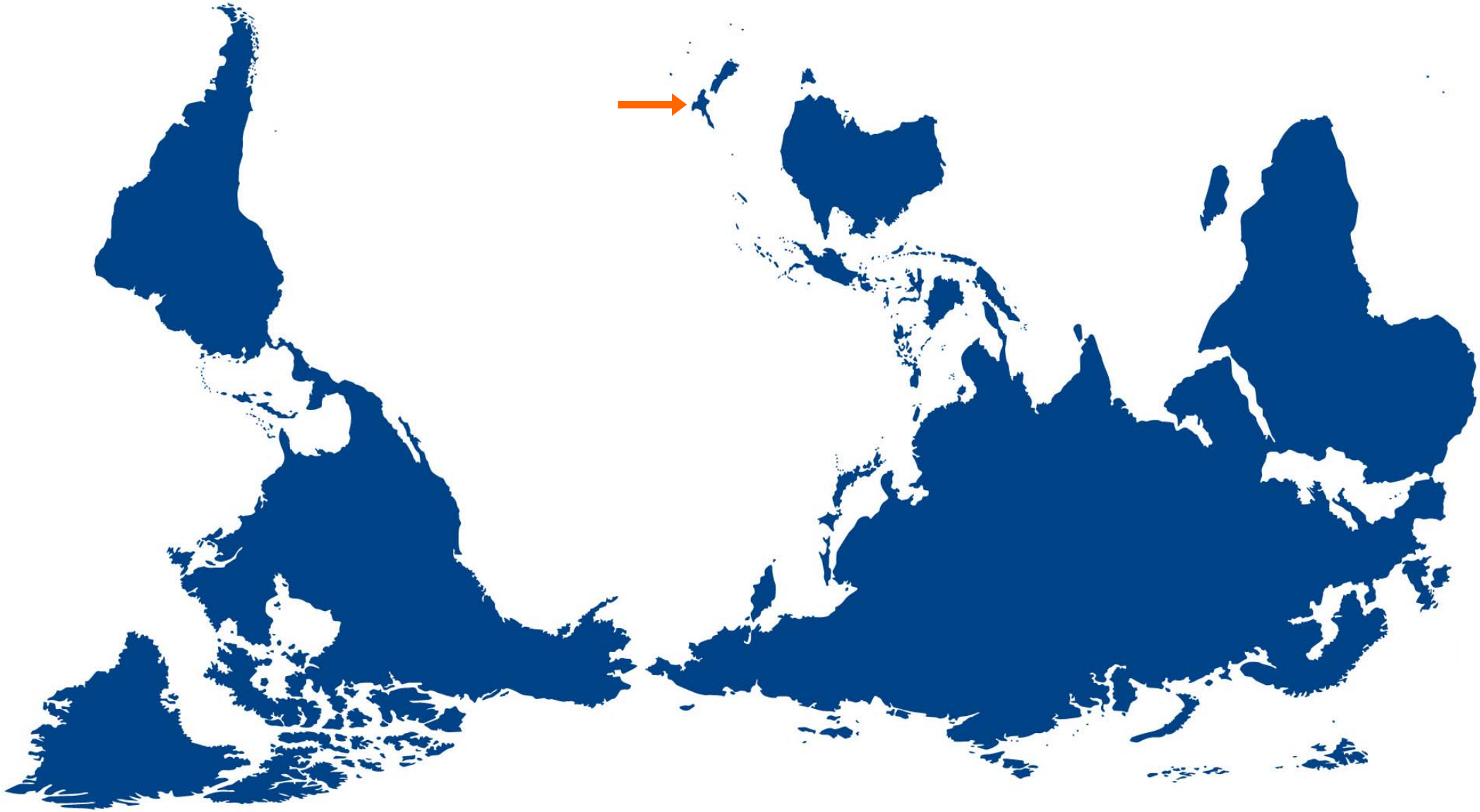
Textbook

This textbook discusses data mining, and Weka, in depth:

> *Data Mining: Practical machine learning tools and techniques*,
> by Ian H. Witten, Eibe Frank and Mark A. Hall. Morgan Kaufmann, 2011

The publisher has made available parts relevant to this course in ebook format.





THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

More Data Mining with Weka

Class 1 – Lesson 2

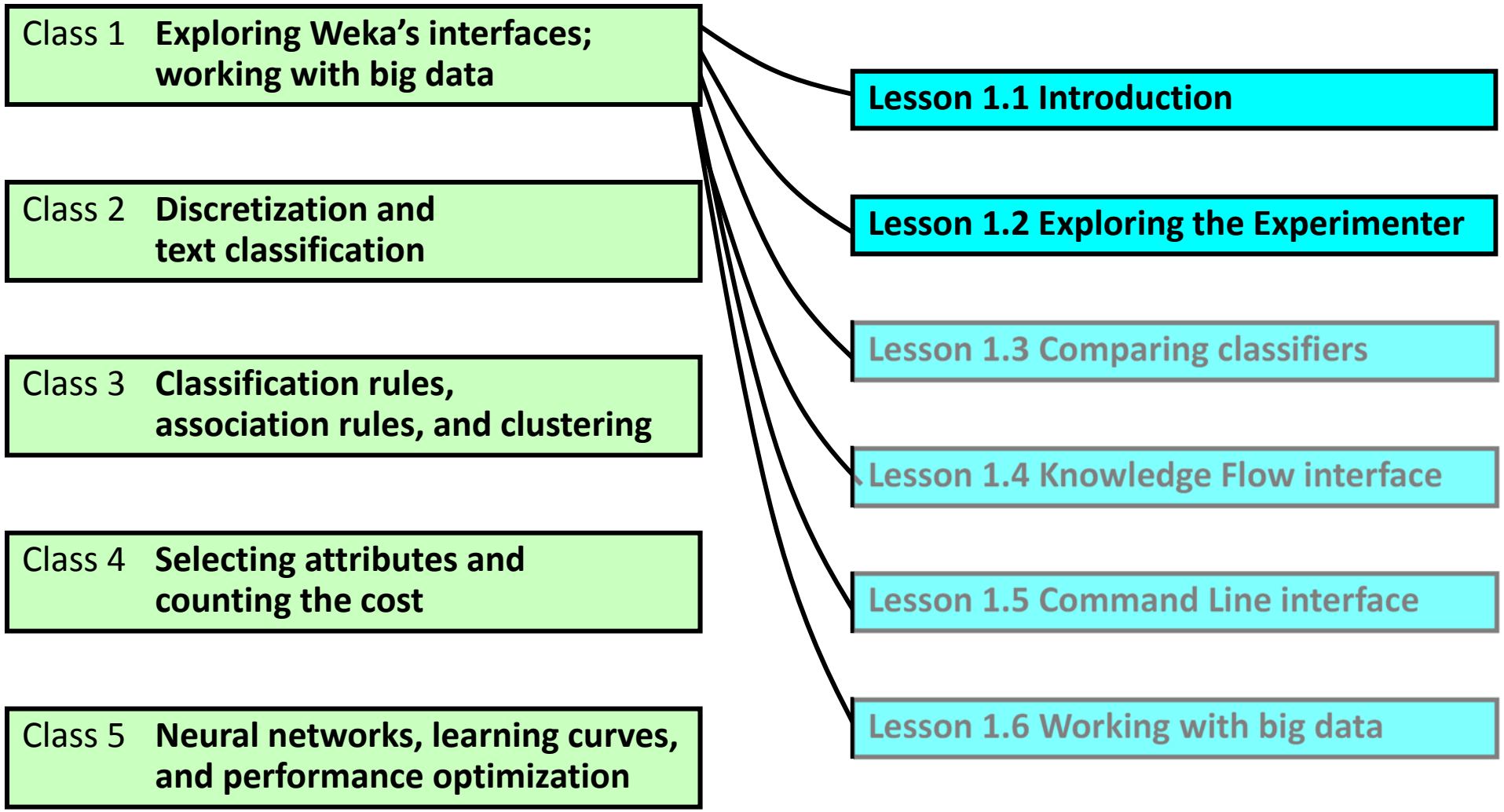
Exploring the Experimenter

Ian H. Witten

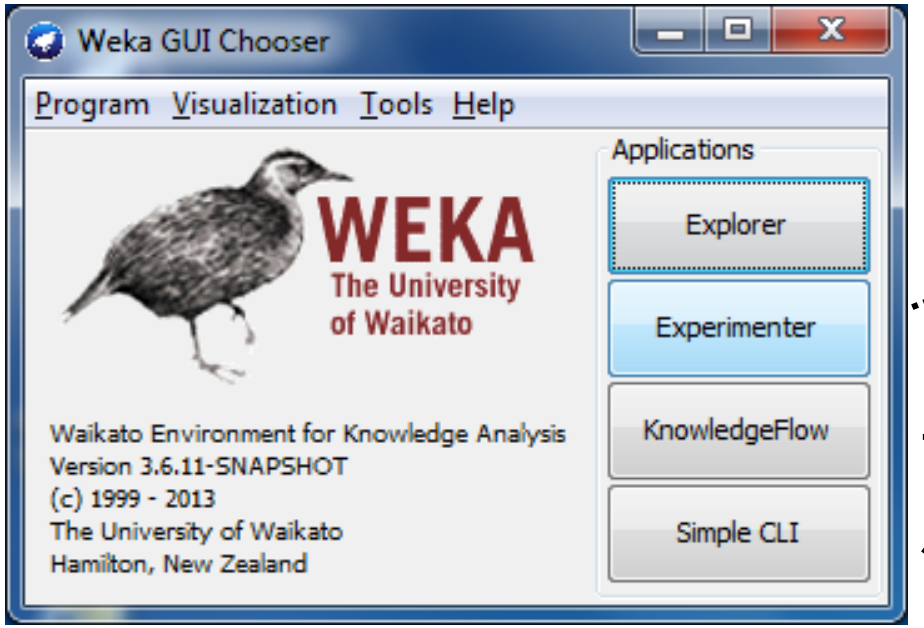
Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 1.2: Exploring the Experimenter



Lesson 1.2: Exploring the Experimenter



Trying out classifiers/filters

Performance comparisons

Graphical interface

Command-line interface

Lesson 1.2: Exploring the Experimenter

Use the Experimenter for …

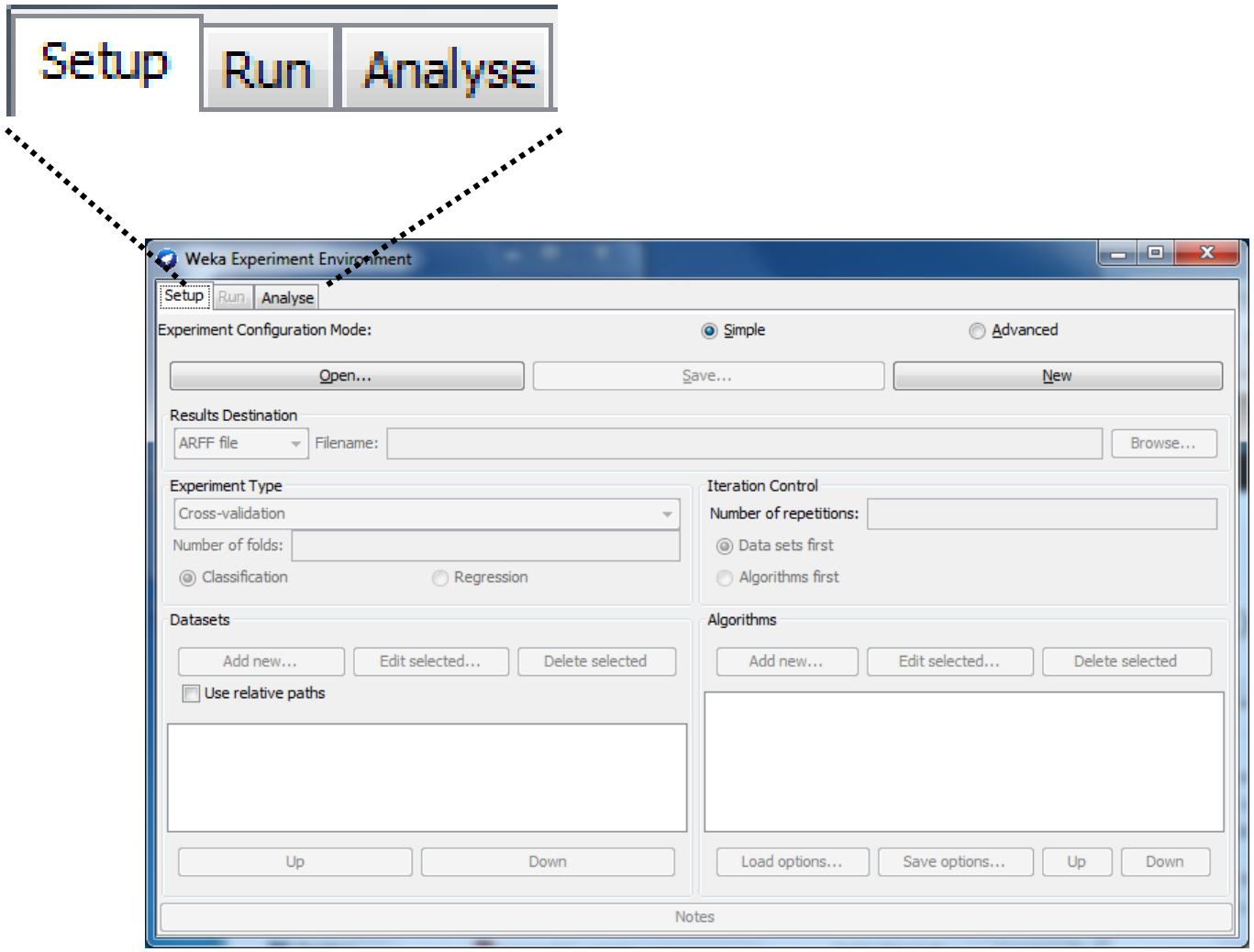
- determining mean and standard deviation performance of a classification algorithm on a dataset

  … or several algorithms on several datasets
- Is one classifier better than another on a particular dataset?

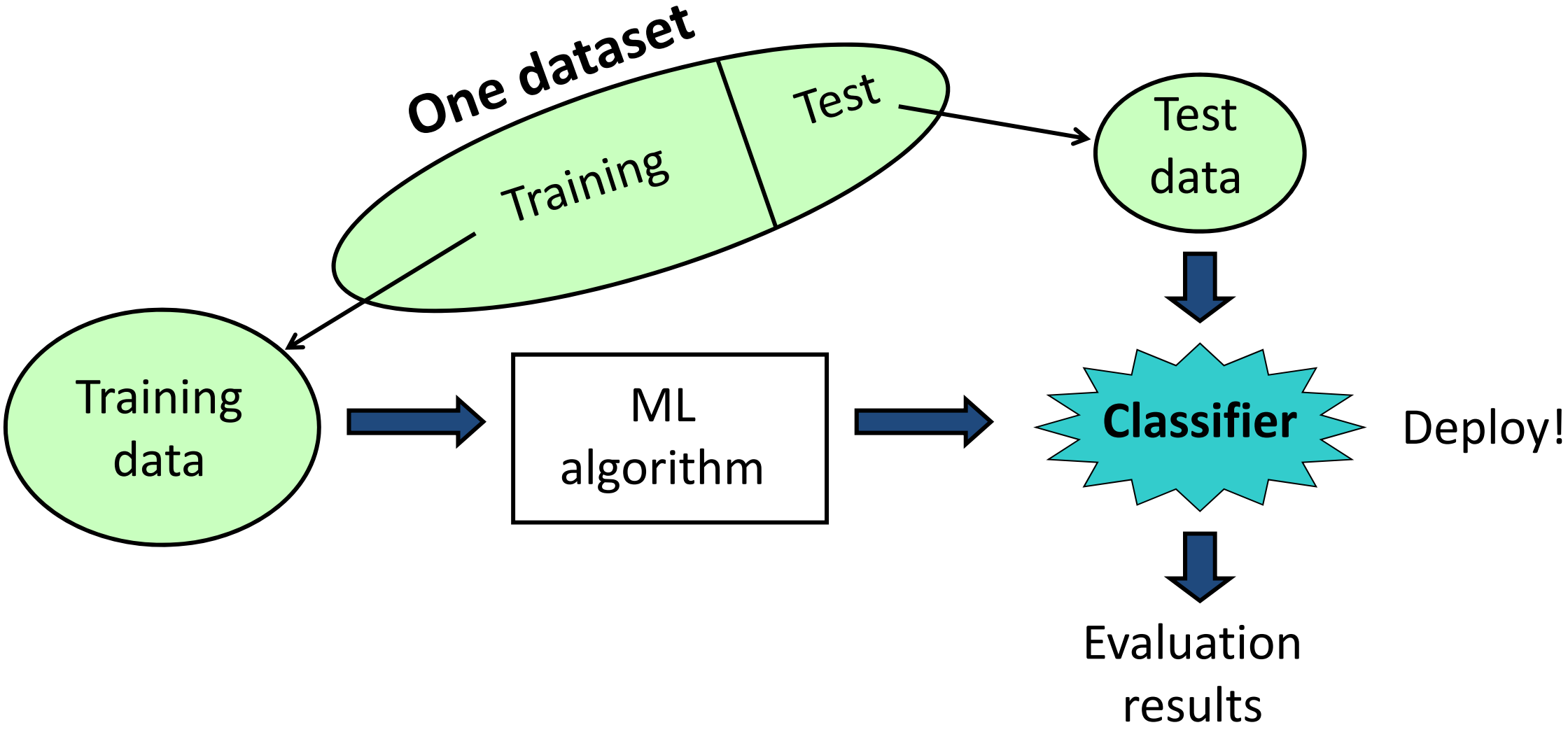
  … and is the difference statistically significant?
- Is one parameter setting for an algorithm better than another?
- The result of such tests can be expressed as an ARFF file
- Computation may take days or weeks

  … and can be distributed over several computers

Lesson 1.2: Exploring the Experimenter



Lesson 1.2: Exploring the Experimenter

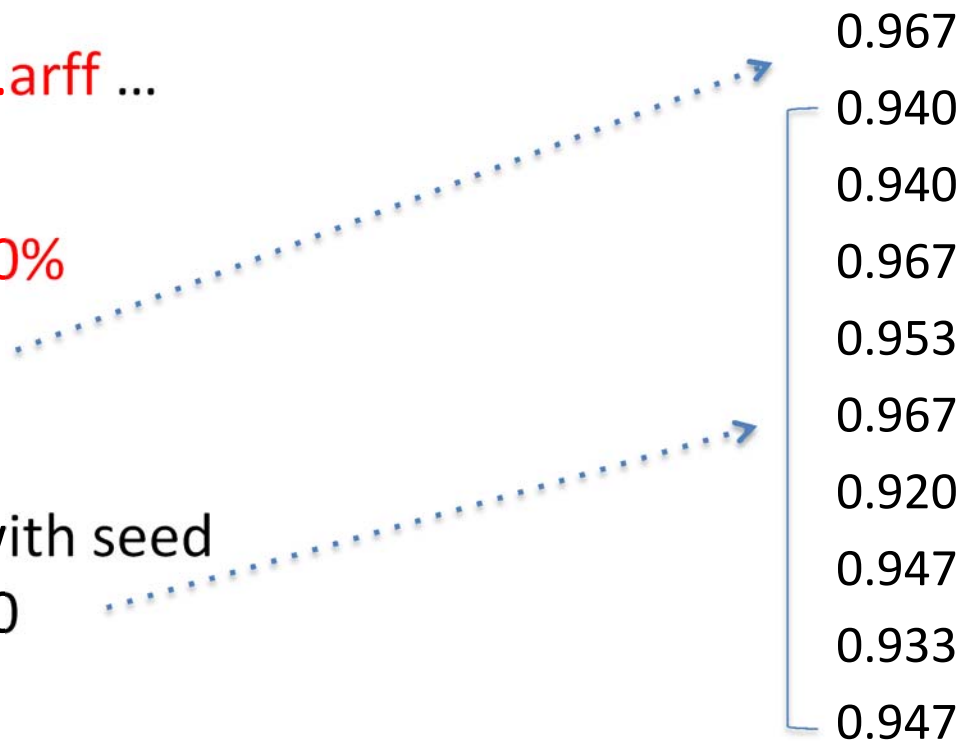


Basic assumption: training and test sets produced by independent sampling from an infinite population

Lesson 1.2: Exploring the Experimenter

Evaluate J48 on segment-challenge (*Data Mining with Weka,* Lesson 2.3)

- With segment-challenge.arff …
- and J48 (trees>J48)
- Set percentage split to 90%
- Run it: 96.7% accuracy
- Repeat
- [More options] Repeat with seed
  2, 3, 4, 5, 6, 7, 8, 9 10



Lesson 1.2: Exploring the Experimenter

Evaluate J48 on segment-challenge (*Data Mining with Weka,* Lesson 2.3)

Sample mean $\bar{x} = \frac{\sum x_i}{n}$

Variance $\sigma^2 = \frac{\sum (x_i - \bar{x})^2}{n - 1}$

Standard deviation σ

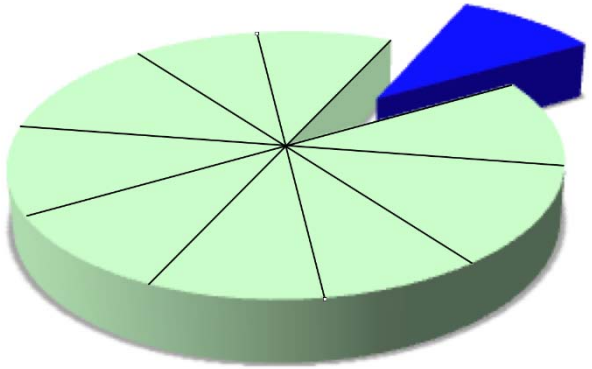
0.967
0.940
0.940
0.967
0.953
0.967
0.920
0.947
0.933
0.947

$\bar{x} = 0.949$, $\sigma = 0.018$

Lesson 1.2: Exploring the Experimenter

10-fold cross-validation (*Data Mining with Weka,* Lesson 2.5)

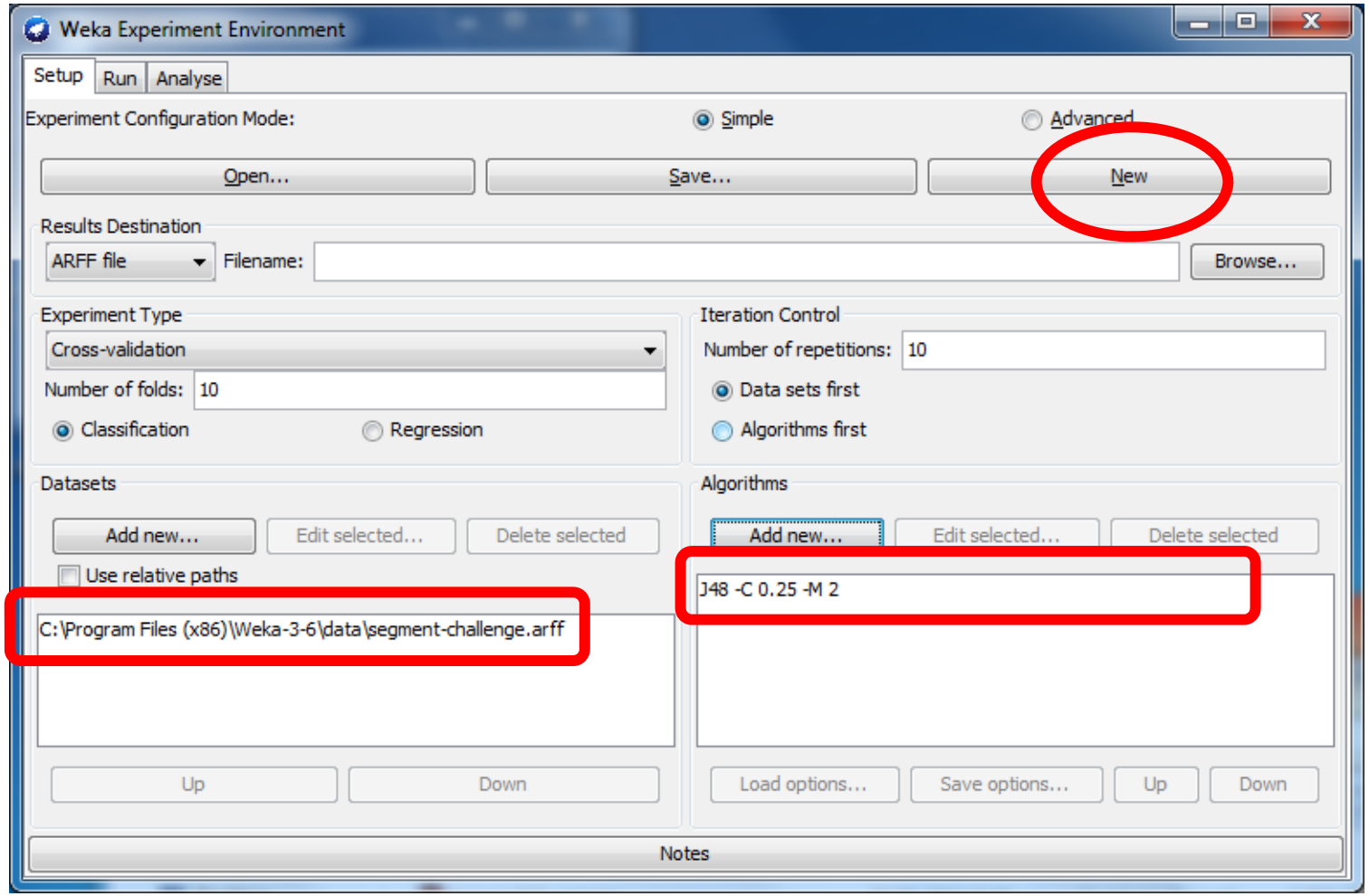
- Divide dataset into 10 parts (folds)
- Hold out each part in turn
- Average the results
- Each data point used once for testing, 9 times for training



Stratified cross-validation

- Ensure that each fold has the right proportion of each class value

Lesson 1.2: Exploring the Experimenter



***Setup* panel**

- click *New*
- note defaults
  - 10-fold cross-validation, repeat 10 times
- under *Datasets*, click *Add new*, open segment-challenge.arff
- under *Algorithms*, click *Add new*, open trees>J48

***Run* panel**

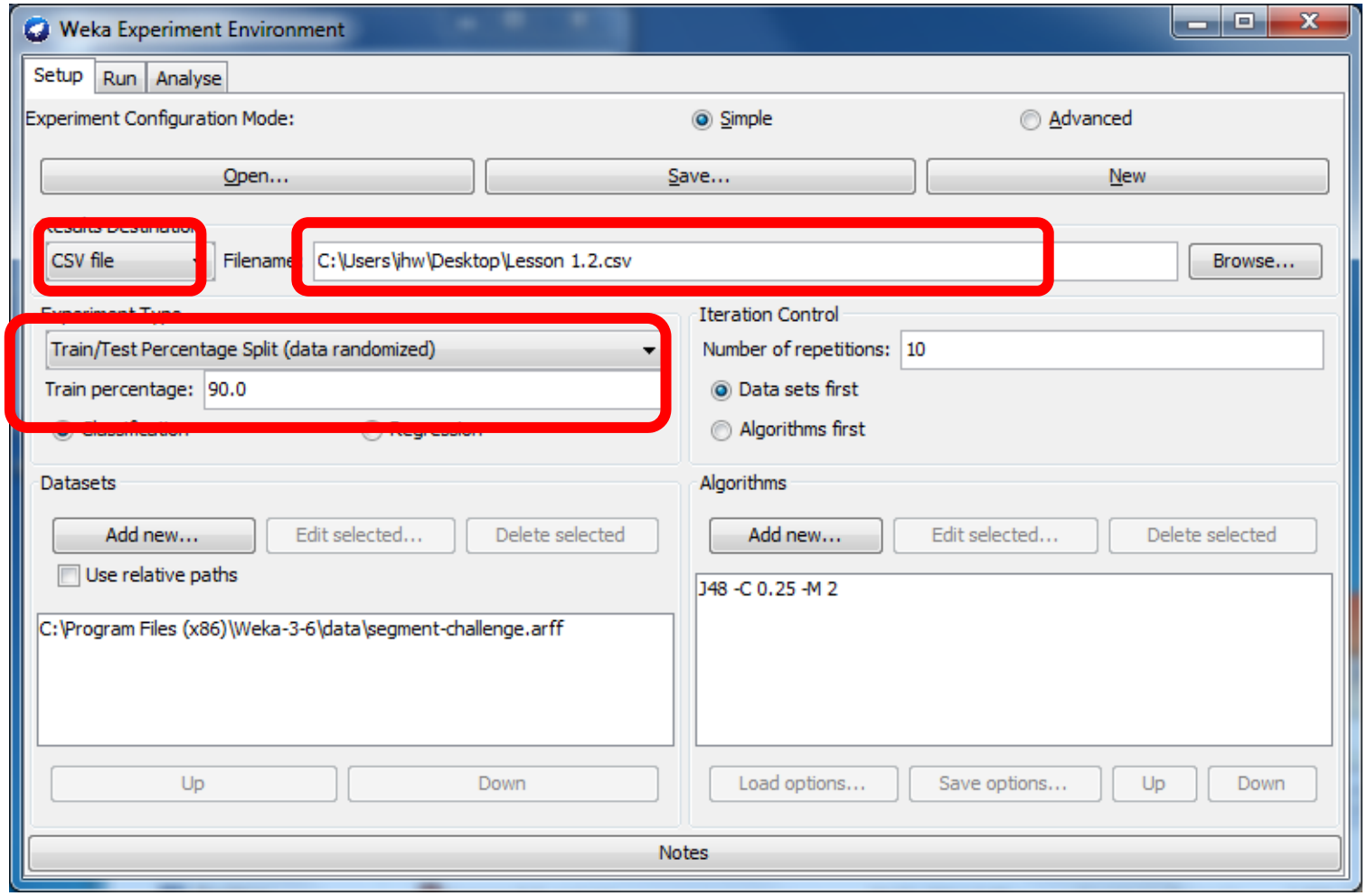
- click *Start*

***Analyse* panel**

- click *Experiment*
- Select *Show std. deviations*
- Click *Perform test*

$\overline{x} = 95.71\%, \ \sigma = 1.85\%$

Lesson 1.2: Exploring the Experimenter



To get detailed results

return to *Setup* panel

- select .csv file
- enter filename for results
- *Train/Test Split*; *90%*

Lesson 1.2: Exploring the Experimenter

switch to *Run* panel

- ❖ click *Start*
- ❖ Open results spreadsheet

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Key_Datas	ey_Ru	ey_Scheme	Key_Scheme	Key_Sche	Date_time	Number_of_	Number_	Number_	Number_	Number_	Percent_correct	cent_incorre	Percer	Kappa_s	Mean_a	Root_me
2	segment	1	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1350	150	145	5	0	96.7	3.3	0	0.9611	0.012	0.09709
3	segment	2	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1349	151	142	9	0	94.0	6.0	0	0.9304	0.019	0.12832
4	segment	3	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1350	150	142	8	0	94.7	5.3	0	0.9377	0.017	0.11916
5	segment	4	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1350	150	146	4	0	97.3	2.7	0	0.9689	0.009	0.07793
6	segment	5	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1350	150	143	7	0	95.3	4.7	0	0.9455	0.016	0.11343
7	segment	6	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1350	150	144	6	0	96.0	4.0	0	0.9533	0.014	0.10906
8	segment	7	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1349	151	143	8	0	94.7	5.3	0	0.9381	0.016	0.11822
9	segment	8	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1349	151	141	10	0	93.4	6.6	0	0.9227	0.02	0.13130
10	segment	9	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1349	151	144	7	0	95.4	4.6	0	0.9459	0.015	0.11158
11	segment	10	ers.trees.J48	'-C 0.25 -M 2	-2.2E+17	2.01E+07	1350	150	142	8	0	94.7	5.3	0	0.9377	0.016	0.11974
12																	
13																	

Lesson 1.2: Exploring the Experimenter

Re-run cross-validation experiment

- Open results spreadsheet

Lesson 1.2: Exploring the Experimenter

Setup panel

- Save/Load an experiment
- Save the results in Arff file … or in a database
- Preserve order in Train/Test split (can't do repetitions)
- Use several datasets, and several classifiers
- Advanced mode

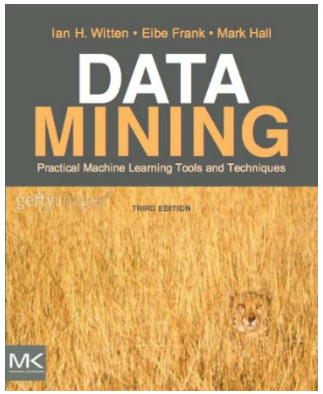
Run panel

Analyse panel

- Load results from .csv or Arff file … or from a database
- Many options

Lesson 1.2: Exploring the Experimenter

- Open Experimenter
- Setup, Run, Analyse panels
- Evaluate one classifier on one dataset
  - … using cross-validation, repeated 10 times
  - … using percentage split, repeated 10 times
- Examine spreadsheet output
- Analyse panel to get mean and standard deviation
- Other options on Setup and Run panels



Course text

- Chapter 13 *The Experimenter*

THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

More Data Mining with Weka

Class 1 – Lesson 3

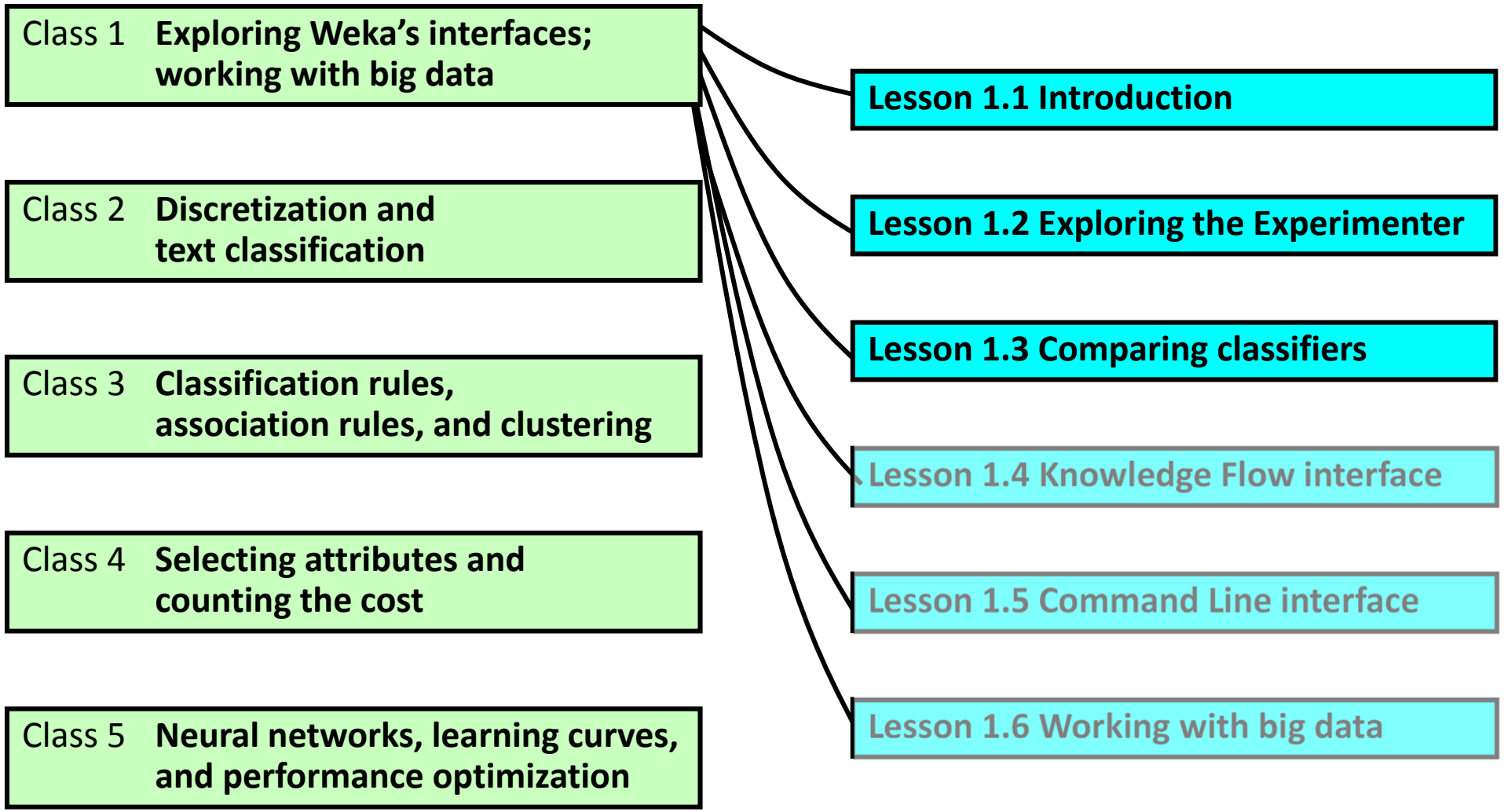
Comparing classifiers

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 1.3: Comparing classifiers



Lesson 1.3: Comparing classifiers

Is J48 better than (a) ZeroR and (b) OneR on the Iris data?

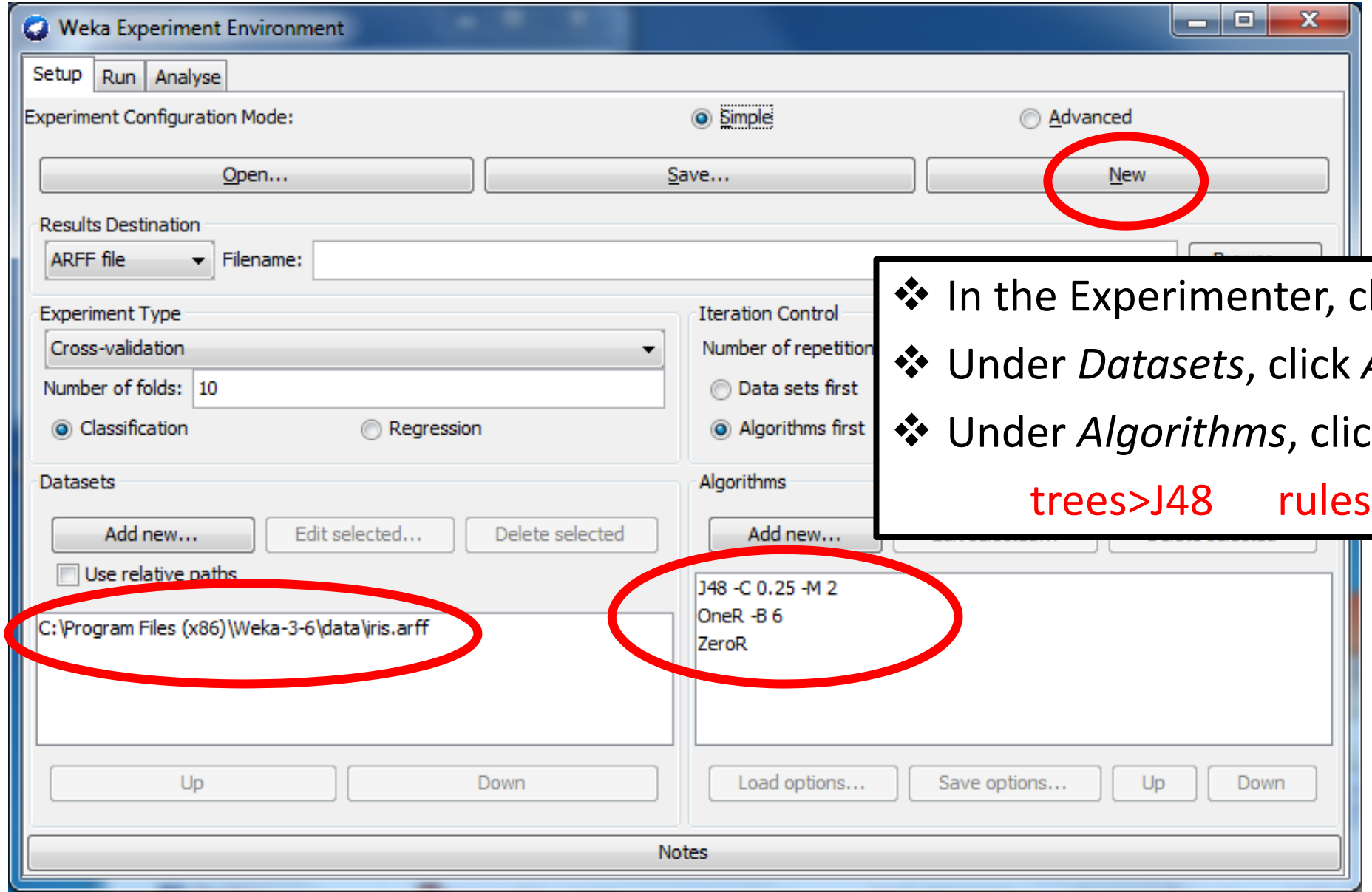
- In the Explorer, open iris.arff
- Using cross-validation, evaluate classification accuracy with …

ZeroR (rules>ZeroR)	33%
OneR (rules>OneR)	92%
J48 (trees>J48)	96%

But how reliable is this?

What would happen if you used a different random number seed??

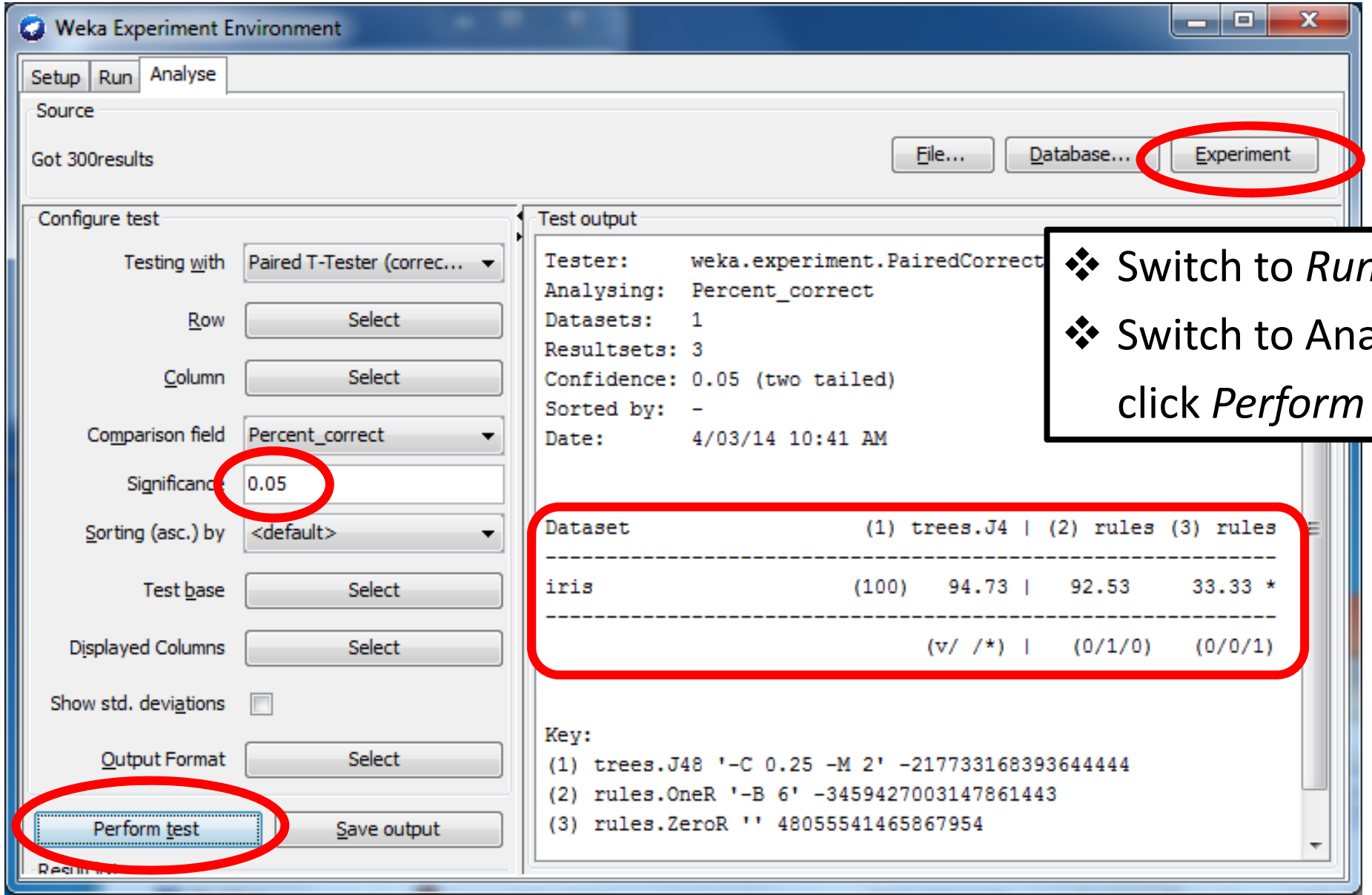
Lesson 1.3: Comparing classifiers



- In the Experimenter, click *New*
- Under *Datasets*, click *Add new*, open iris.arff
- Under *Algorithms*, click *Add new*, open

  trees>J48 rules>OneR rules>ZeroR

Lesson 1.3: Comparing classifiers



- Switch to *Run*; click *Start*
- Switch to Analyse, click *Experiment* click *Perform test*

Lesson 1.3: Comparing classifiers

```
Dataset                    (1) trees.J4 | (2) rules (3) rules
----------------------------------------------------------------
iris                      (100)   94.73 |   92.53     33.33 *
----------------------------------------------------------------
                                (v/ /*) |   (0/1/0)   (0/0/1)


Key:
(1) trees.J48
(2) rules.OneR
(3) rules.ZeroR
```

v significantly better

* significantly worse

- ZeroR (33.3%) is significantly worse than J48 (94.7%)
- Cannot be sure that OneR (92.5%) is significantly worse than J48
- … at the 5% level of statistical significance
- J48 seems better than ZeroR: pretty sure (5% level) that this is not due to chance
- … and better than OneR; but this may be due to chance (can't rule it out at 5% level)

Lesson 1.3: Comparing classifiers

Dataset	(1) trees.J4	(2) rules	(3) rules
iris	(100) 94.73	92.53	33.33 *
breast-cancer	(100) 74.28	66.91 *	70.30
german_credit	(100) 71.25	65.91 *	70.00
pima_diabetes	(100) 74.49	71.52	65.11 *
Glass	(100) 67.63	57.40 *	35.51 *
ionosphere	(100) 89.74	82.28 *	64.10 *
segment	(100) 95.71	64.35 *	15.73 *
	(v/ /*)	(0/2/5)	(0/2/5)

Key:
(1) trees.J48 '
(2) rules.OneR
(3) rules.ZeroR

J48 is significantly (5% level) **better than**

- both OneR and ZeroR on Glass, ionosphere, segment
- OneR on breast-cancer, german_credit
- ZeroR on iris, pima_diabetes

Lesson 1.3: Comparing classifiers

Dataset	(2) rules.On	(1) trees	(3) rules
iris	(100) 92.53	94.73	33.33 *
breast-cancer	(100) 66.91	74.28 v	70.30
german_credit	(100) 65.91	71.25 v	70.00 v
pima_diabetes	(100) 71.52	74.49	65.11 *
Glass	(100) 57.40	67.63 v	35.51 *
ionosphere	(100) 82.28	89.74 v	64.10 *
segment	(100) 64.35	95.71 v	15.73 *
	(v/ /*)	(5/2/0)	(1/1/5)

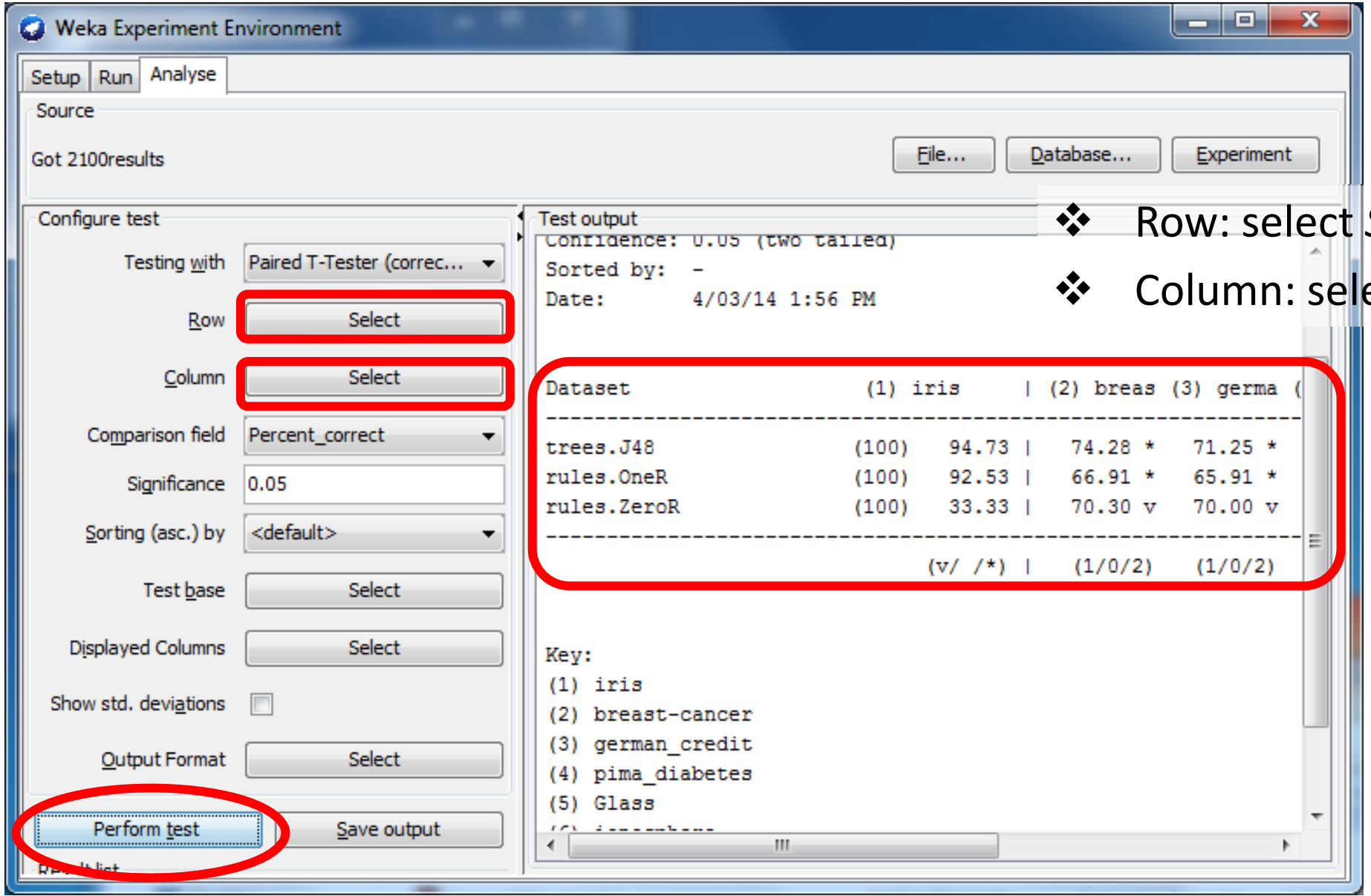
Key:
(1) trees.J48
(2) rules.OneR
(3) rules.ZeroR

Comparing OneR with ZeroR

Change “Test base” on Analyse panel

- significantly worse on german-credit
- about the same on breast-cancer
- significantly better on all the rest

Lesson 1.3: Comparing classifiers



- Row: select *Scheme* (not *Dataset*)
- Column: select Dataset (not *Scheme*)

Lesson 1.3: Comparing classifiers

- Statistical significance: the “null hypothesis”

  Classifier A’s performance is the same as B’s

- The observed result is highly unlikely if the null hypothesis is true

  “The null hypothesis can be rejected at the 5% level”
  [of statistical significance]

  “A performs significantly better than B at the 5% level”

- Can change the significance level (5% and 1% are common)
- Can change the comparison field (we have used % correct)
- Common to compare over a set of datasets

  “On these datasets, method A has xx wins and yy losses over method B”

- Multiple comparison problem

  if you make many tests, some will appear to be “significant” just by chance!

THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

More Data Mining with Weka

Class 1 – Lesson 4

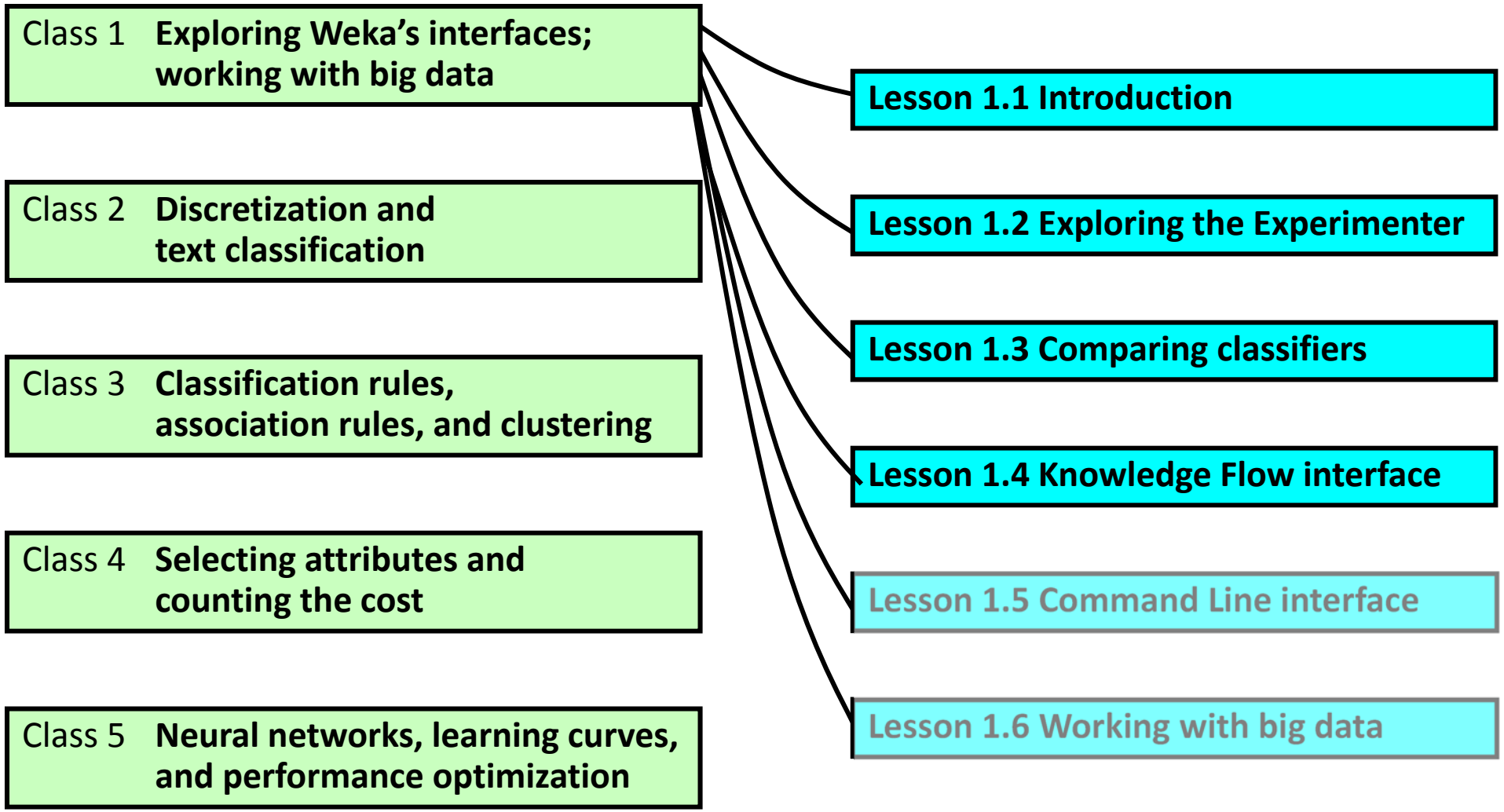
The Knowledge Flow interface

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 1.4: The Knowledge Flow interface



Lesson 1.4: The Knowledge Flow interface

The Knowledge Flow interface is an alternative to the Explorer

- Lay out filters, classifiers, evaluators interactively on a 2D canvas
- Components include data sources, data sinks, evaluation, visualization
- Different kinds of connections between the components
  - *Instance or dataset*
  - *test set, training set*
  - *classifier*
  - *output, text or chart*
- Can work incrementally, on potentially infinite data streams
- Can look inside cross-validation at the individual models produced

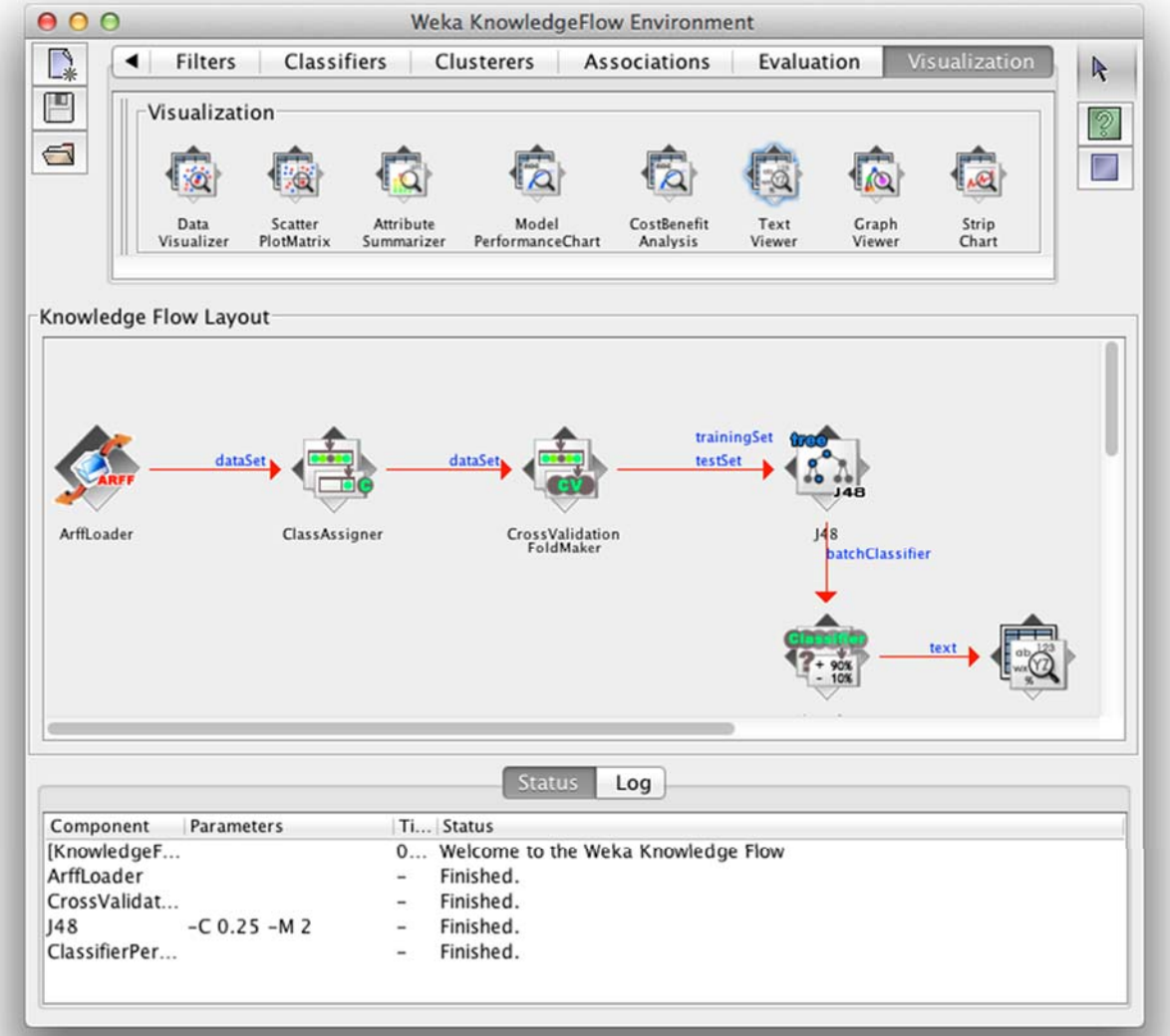
Lesson 1.4: The Knowledge Flow interface

Load an ARFF file, choose J48, evaluate using cross-validation

	Toolbar
❖ Choose an *ArffLoader*; Configure to set the file *iris.arff*	*DataSources*
❖ Connect up a *ClassAssigner* to select the class	*Evaluation*
❖ Connect the result to a *CrossValidationFoldMaker*	*Evaluation*
❖ Connect this to *J48*	*Classifiers*
❖ Make two connections, one for *trainingSet* and the other for *testSet*	
❖ Connect J48 to *ClassifierPerformanceEvaluator*	*Evaluation*
❖ Connect this to a *TextViewer*	*Visualization*

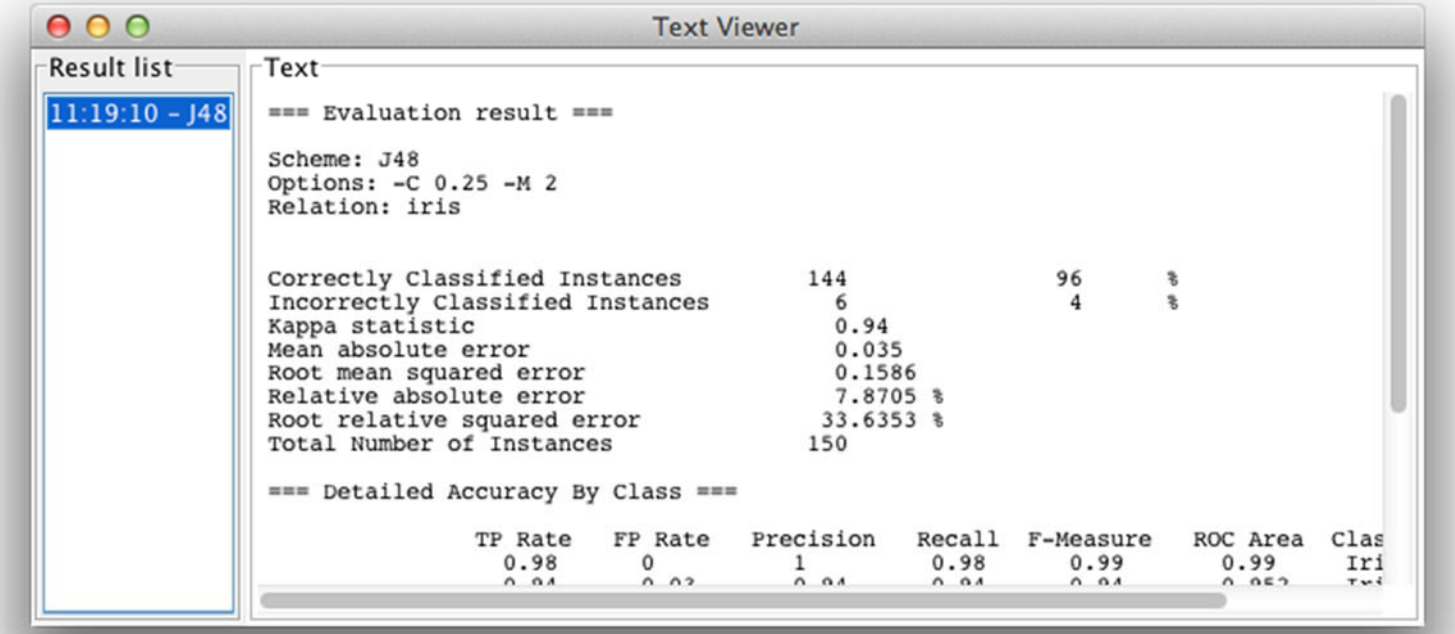
Then run it! (*ArffLoader*: Start loading)

Lesson 1.4: The Knowledge Flow interface



Lesson 1.4: The Knowledge Flow interface

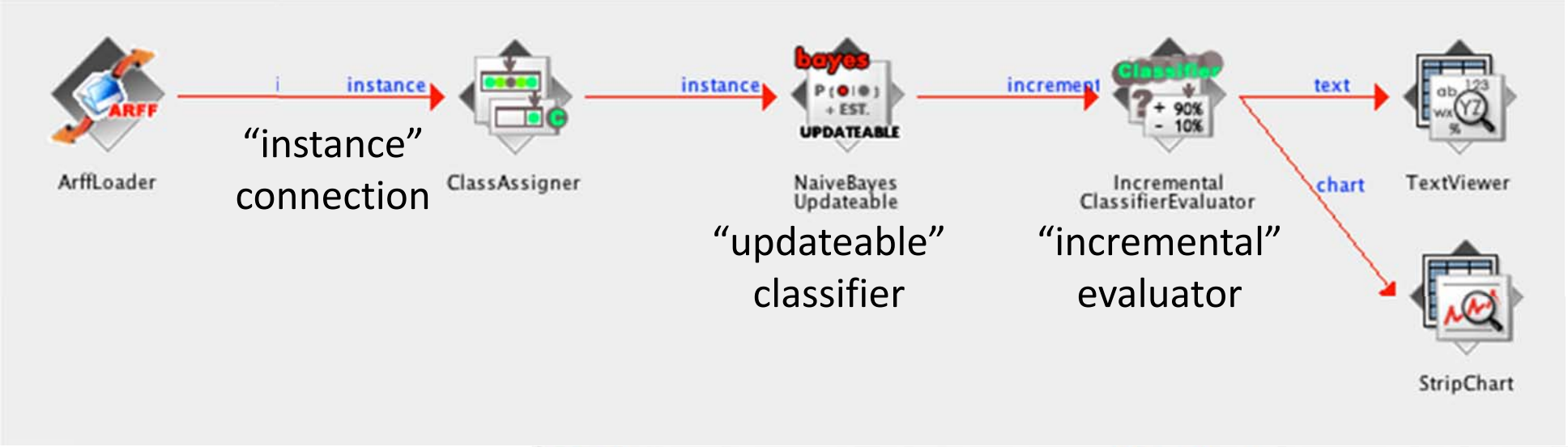
- *TextViewer*: Show results

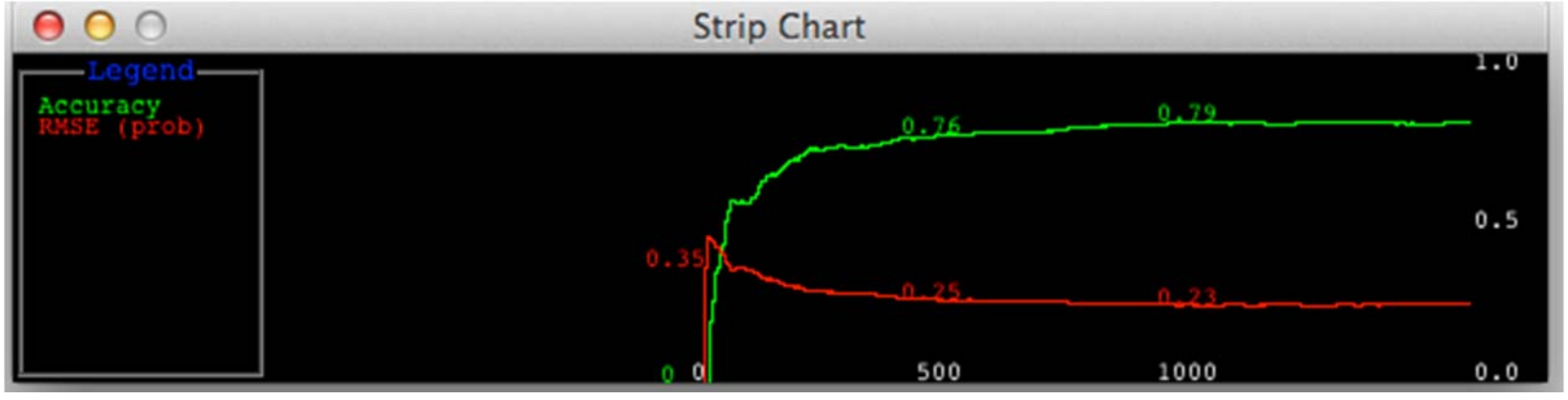


- Add a *ModelPerformanceChart*
- Connect the *visualizableError* output of *ClassifierPerformanceEvaluator* to it
- Show chart (need to run again)

Lesson 1.4: The Knowledge Flow interface

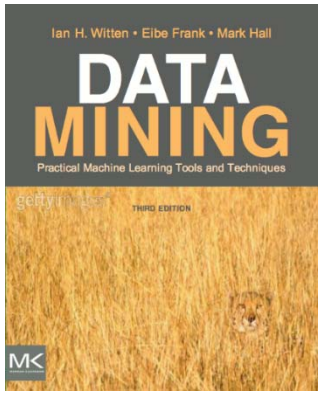
Working with stream data





Lesson 1.4: The Knowledge Flow interface

- Panels broadly similar to the Explorer's, except
  - *DataSources* are separate from *Filters*
  - Write data or models to files using *DataSinks*
  - *Evaluation* is a separate panel
- Facilities broadly similar too, except
  - Can deal incrementally with potentially infinite datasets
  - Can look inside cross-validation at the models for individual folds
- Some people like graphical interfaces



Course text

- Chapter 12 *The Knowledge Flow Interface*

THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

More Data Mining with Weka

Class 1 – Lesson 5

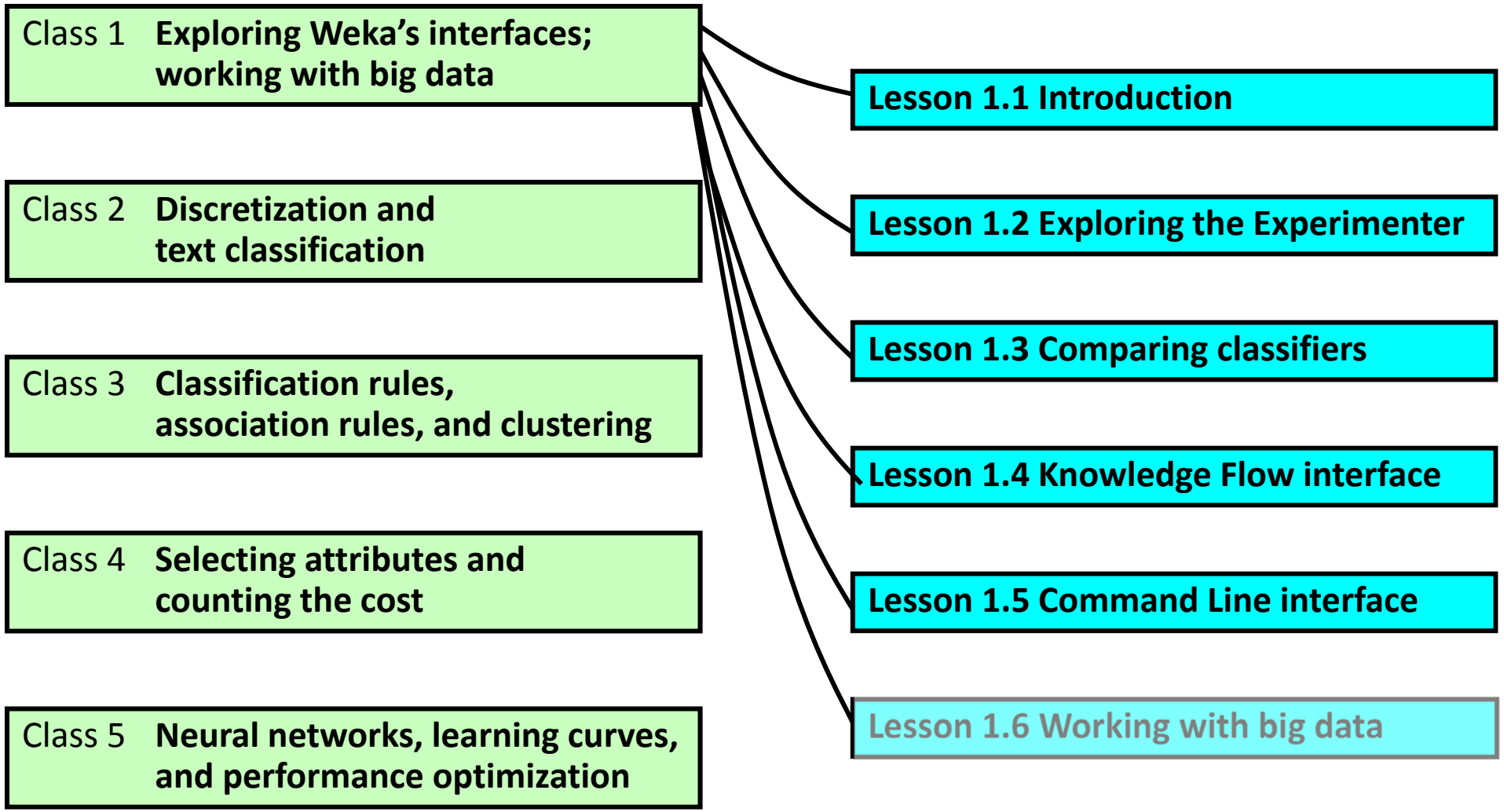
The Command Line interface

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 1.5: The Command Line interface



Lesson 1.5: The Command Line interface

Run a classifier from within the CLI

- Print options for J48:

  java weka.classifiers.trees.J48

- General options

  –h print help info
  –t <name of training file> [absolute path name …]
  –T <name of test file>

- Options specific to J48 (from Explorer configuration panel)
- Run J48:

  java weka.classifiers.trees.J48 –C 0.25 –M 2 ⟵ copy from Explorer
  –t "C:\Users\ihw\My Documents\Weka datasets\iris.arff" ⟵ training set

Lesson 1.5: The Command Line interface

Classes and packages

- J48 is a "class"
  - *a collection of variables, along with some "methods" that operate on them*
- "Package" is a directory containing related classes

weka.classifiers.trees.J48

- Javadoc: the definitive documentation for Weka

Weka-3-6\documentation.html

- … find J48 in the "All classes" list

Lesson 1.5: The Command Line interface

Using the Javadoc

"What's all this geeky stuff?" – *Forget it. Try to ignore things you don't understand!*

- Find the "converter" package

  weka.core.converters

- Find the "databaseLoader" class

  weka.core.converters.DatabaseLoader

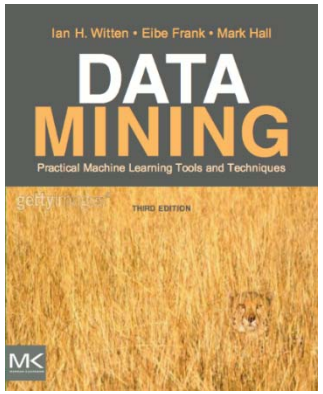
- Can load from any JDBC database

  specify URL, password, SQL query

- It's in the Explorer's Preprocess panel, but the documentation is here

Lesson 1.5: The Command Line interface

- Can do everything the Explorer does from the command line
- People often open a terminal window instead
  - then you can do scripting (if you know how)
  - … but you need to set up your environment properly
- Can copy and paste configured classifiers from the Explorer
- Advantage: more control over memory usage (next lesson)
- Javadoc is the definitive source of Weka documentation



Course text

- Chapter 14 *The Command-Line Interface*

THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

More Data Mining with Weka

Class 1 – Lesson 6

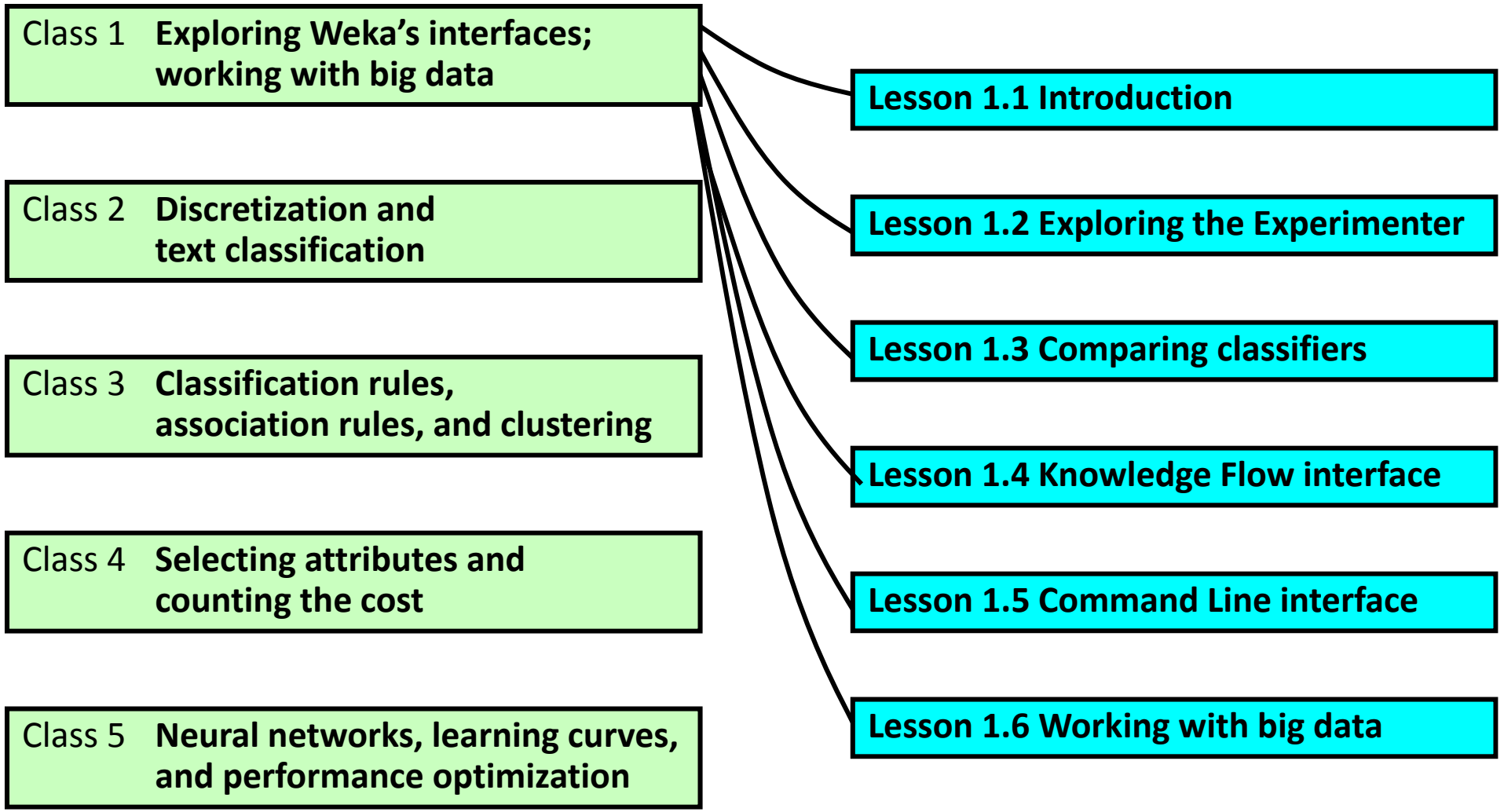
Working with big data

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 1.6: Working with big data

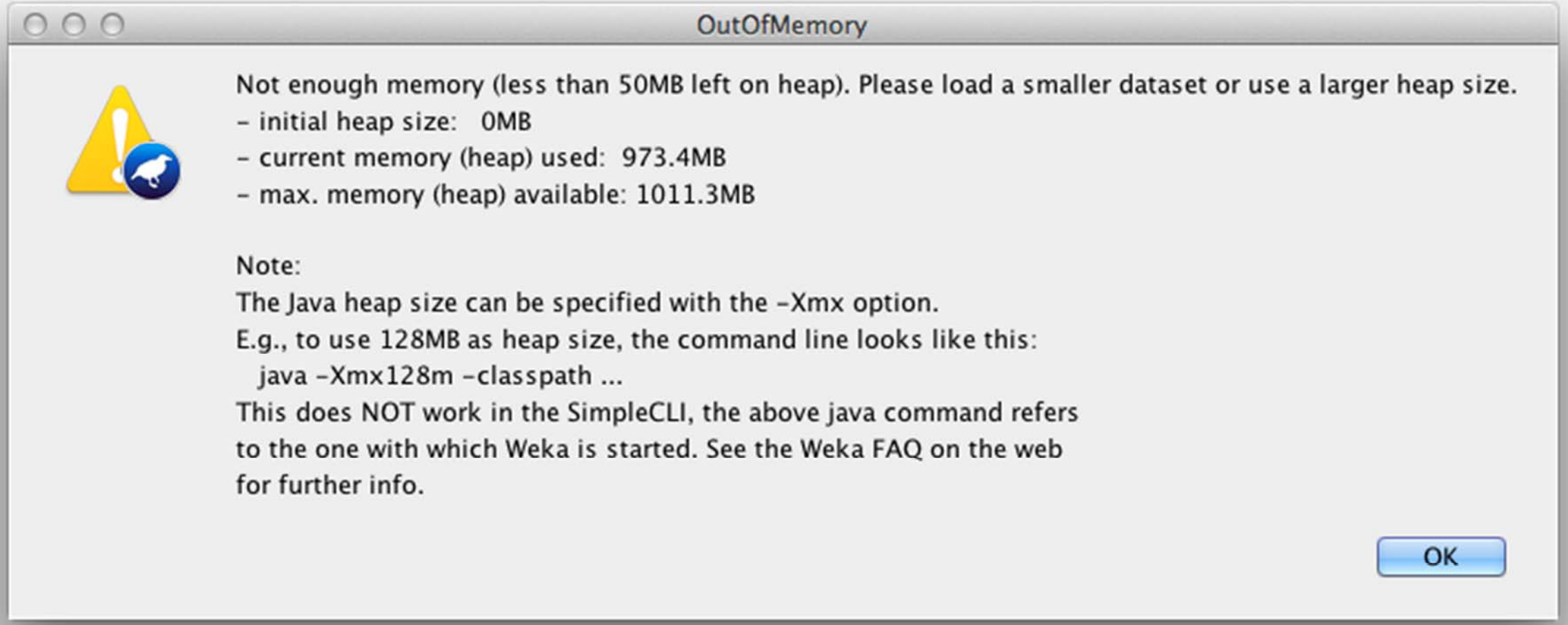


Lesson 1.6: Working with big data

How much can Explorer handle? (~ 1M instances, 25 attributes)

- Memory information: in Explorer, right-click on "Status"
  - *Free/total/max: 226,366,616 / 236,453,888 / 954,728,448 (bytes) [1 GB]*
  - *Meaning what? Geeks, check out Java's freeMemory(), totalMemory(), maxMemory() commands*
- Let's break it!
- Download a large dataset?
  - *"covertype" dataset used in the associated Activity*
  - *580,000 instances, 54 attributes (0.75 GB uncompressed)*
- Weka data generator
  - *Preprocess panel, Generate, choose LED24; show text: 100 instances, 25 attributes*
  - *100,000 examples (use % split!) NaiveBayes 74% J48 73%*
  - *1,000,000 examples NaiveBayes 74% J48 runs out of memory*
  - *2,000,000 examples Generate process grinds to a halt*
- (Run console version of Weka)

Lesson 1.6: Working with big data



Lesson 1.6: Working with big data

"Updateable" classifiers

- Incremental classification models: process one instance at a time
  - *AODE, AODEsr, DMNBtext, IB1, IBk, KStar, LWL, NaiveBayesMultinomialUpdateable, NaiveBayesUpdateable, NNge, RacedIncrementalLogitBoost, SPegasos, Winnow*
- *NaiveBayesUpdateable*: same as *NaiveBayes*
- *NaiveBayesMultinomialUpdateable*: see lessons on Text Mining
- *IB1*, *IBk* (but testing can be very slow)
- *KStar*, *LWL* (locally weighted learning): instance-based
- *SPegasos* (in *functions*)
  - *builds a linear classifier, SVM-style (restricted to numeric or binary class)*
- *RacedIncrementalLogitBoost*: a kind of boosting

Lesson 1.6: Working with big data

How much can Weka (Simple CLI) handle? – unlimited (conditions apply)

- Create a huge dataset

  java weka.datagenerators.classifiers.classification.LED24 –n 100000 –o C:\Users\ihw\test.arff

  - *Test file with 100 K instances, 5 MB*

  java weka.datagenerators.classifiers.classification.LED24 –n 10000000 –o C:\Users\ihw\train.arff

  - *Training file with 10 M instances; 0.5 GB*
- Use NaiveBayesUpdateable

  java weka.classifiers.bayes.NaiveBayesUpdateable –t …train.arff –T …test.arff

  - *74%; 4 mins*
  - *Note: if no test file specified, will do cross-validation, which will fail (non-incremental)*
- Try with 100 M examples (5 GB training file) – no problem (40 mins)

Lesson 1.6: Working with big data

- *Explorer* can handle ~ 1M instances, 25 attributes (50 MB file)
- *Simple CLI* works incrementally wherever it can
- Some classifier implementations are “Updateable”
  - *find them with Javadoc; see Lesson 1.5 Activity*
- Updateable classifiers deal with arbitrarily large files (multi GB)
  - *but don’t attempt cross-validation*
- Working with big data can be difficult and frustrating
  - *see the associated Activity*

More Data Mining with Weka

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz