Data Mining with Weka

Class 3 – Lesson 1

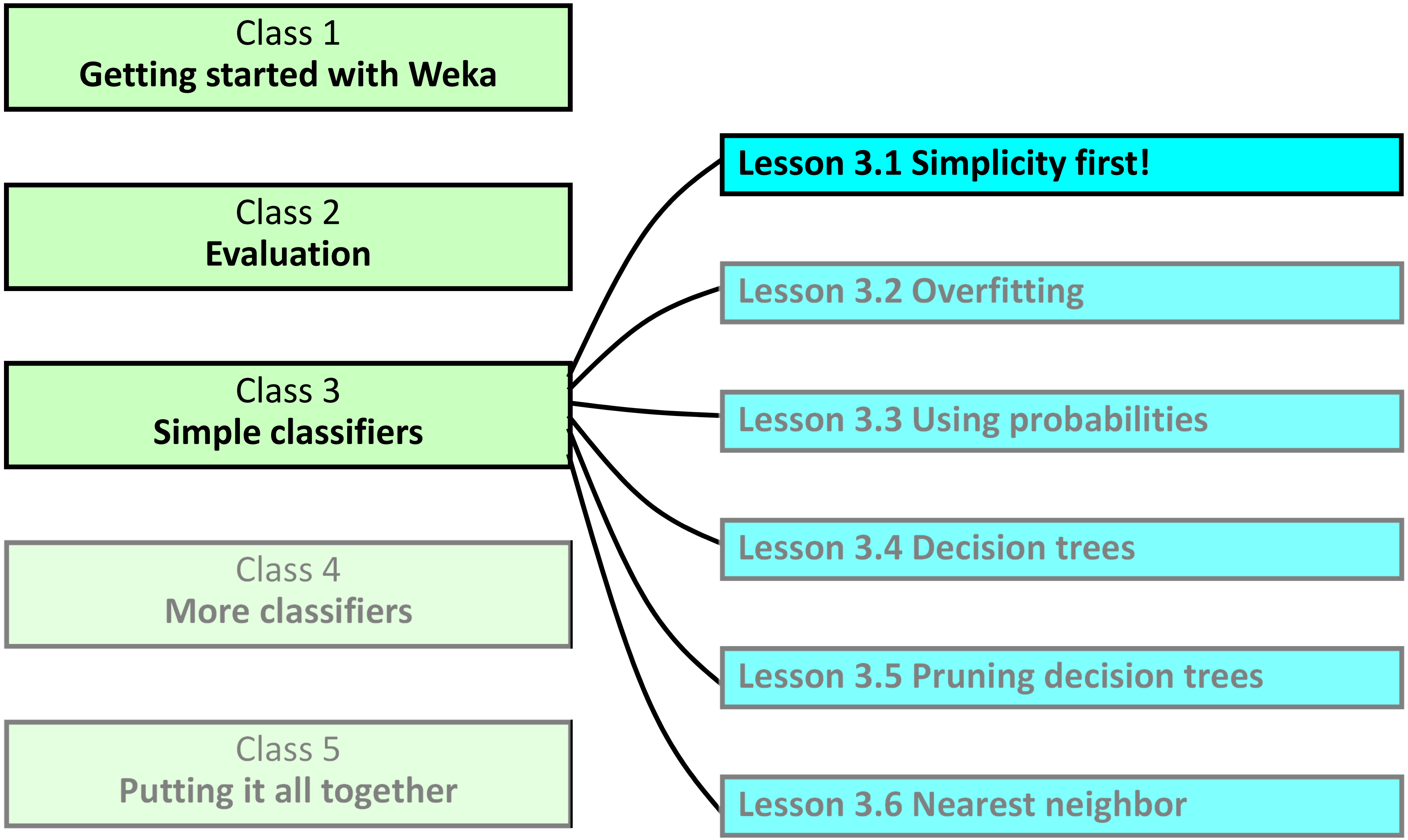
Simplicity first!

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 3.1 Simplicity first!



Lesson 3.1 Simplicity first!

Simple algorithms often work very well!

- There are many kinds of simple structure, eg:
  - *One attribute does all the work* — *Lessons 3.1, 3.2*
  - *Attributes contribute equally and independently* — *Lesson 3.3*
  - *A decision tree that tests a few attributes* — *Lessons 3.4, 3.5*
  - *Calculate distance from training instances* — *Lesson 3.6*
  - *Result depends on a linear combination of attributes* — *Class 4*

- Success of method depends on the domain
  - *Data mining is an experimental science*

Lesson 3.1 Simplicity first!

OneR: One attribute does all the work

- Learn a 1-level "decision tree"
  - *i.e., rules that all test one particular attribute*

- Basic version
  - *One branch for each value*
  - *Each branch assigns most frequent class*
  - *Error rate: proportion of instances that don't belong to the majority class of their corresponding branch*
  - *Choose attribute with smallest error rate*

Lesson 3.1 Simplicity first!

```
For each attribute,
    For each value of the attribute,
    make a rule as follows:
        count how often each class appears
        find the most frequent class
        make the rule assign that class
          to this attribute-value
    Calculate the error rate of this attribute's rules
Choose the attribute with the smallest error rate
```

Lesson 3.1 Simplicity first!

Outlook	Temp	Humidity	Wind	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

Attribute	Rules	Errors	Total errors
Outlook	Sunny → No	2/5	4/14
	Overcast → Yes	0/4	
	Rainy → Yes	2/5	
Temp	Hot → No*	2/4	5/14
	Mild → Yes	2/6	
	Cool → Yes	1/4	
Humidity	High → No	3/7	4/14
	Normal → Yes	1/7	
Wind	False → Yes	2/8	5/14
	True → No*	3/6	

* indicates a tie

Lesson 3.1 Simplicity first!

Use OneR

- Open file **weather.nominal.arff**
- Choose OneR rule learner (**rules>OneR**)
- Look at the rule (*note: Weka runs OneR 11 times*)

Lesson 3.1 Simplicity first!

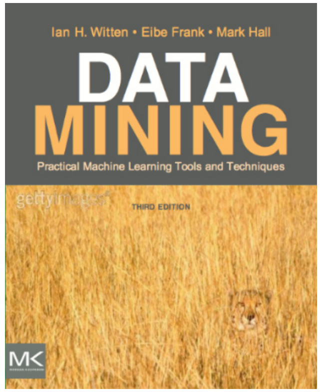
OneR: One attribute does all the work

- Incredibly simple method, described in 1993

  "Very Simple Classification Rules Perform Well on Most Commonly Used Datasets"

  - Experimental evaluation on 16 datasets
  - Used cross-validation
  - Simple rules often outperformed far more complex methods

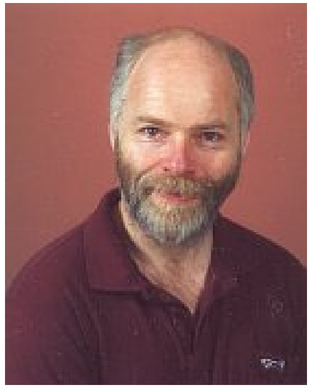
- How can it work so well?
  - some datasets really are simple
  - some are so small/noisy/complex that nothing can be learned from them!



Course text

- Section 4.1 *Inferring rudimentary rules*

Rob Holte,
Alberta, Canada



Data Mining with Weka

Class 3 – Lesson 2

Overfitting

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 3.2 Overfitting

Class 1
Getting started with Weka

Class 2
Evaluation

Class 3
Simple classifiers

- Lesson 3.1 Simplicity first!
- Lesson 3.2 Overfitting
- Lesson 3.3 Using probabilities
- Lesson 3.4 Decision trees
- Lesson 3.5 Pruning decision trees
- Lesson 3.6 Nearest neighbor

Class 4
More classifiers

Class 5
Putting it all together

Lesson 3.2 Overfitting

- Any machine learning method may “overfit” the training data …

  … by producing a classifier that fits the training data too tightly
- Works well on training data but not on independent test data

- Remember the “User classifier”? Imagine tediously putting a tiny circle around every single training data point

- Overfitting is a general problem
- … we illustrate it with OneR

Lesson 3.2 Overfitting

Numeric attributes

Outlook	Temp	Humidity	Wind	Play
Sunny	85	85	False	No
Sunny	80	90	True	No
Overcast	83	86	False	Yes
Rainy	75	80	False	Yes
…	…	…	…	…

Attribute	Rules	Errors	Total errors
Temp	85 → No	0/1	0/14
	80 → Yes	0/1	
	83 → Yes	0/1	
	75 → No	0/1	
	…	…	

- ❖ OneR has a parameter that limits the complexity of such rules
- ❖ How exactly does it work? Not so important …

Lesson 3.2 Overfitting

Experiment with OneR

- Open file weather.numeric.arff
- Choose OneR rule learner (rules>OneR)
- Resulting rule is based on outlook attribute, so remove outlook
- Rule is based on humidity attribute

```
humidity:   < 82.5   -> yes
            >= 82.5  -> no
(10/14 instances correct)
```

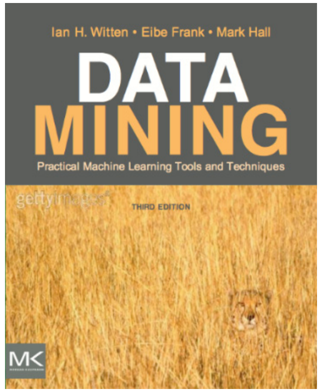
Lesson 3.2 Overfitting

Experiment with diabetes dataset

- Open file diabetes.arff
- Choose ZeroR rule learner (rules>ZeroR)
- Use cross-validation: 65.1%
- Choose OneR rule learner (rules>OneR)
- Use cross-validation: 72.1%
- Look at the rule (plas = plasma glucose concentration)
- Change minBucketSize parameter to 1: 54.9%
- Evaluate on training set: 86.6%
- Look at rule again

Lesson 3.2 Overfitting

- ❖ Overfitting is a general phenomenon that plagues all ML methods
- ❖ One reason why you must never evaluate on the training set
- ❖ Overfitting can occur more generally
- ❖ E.g try many ML methods, choose the best for your data
  - – you cannot expect to get the same performance on new test data
- ❖ Divide data into training, test, validation sets?



Course text

- ❖ Section 4.1 *Inferring rudimentary rules*

Data Mining with Weka

Class 3 – Lesson 3

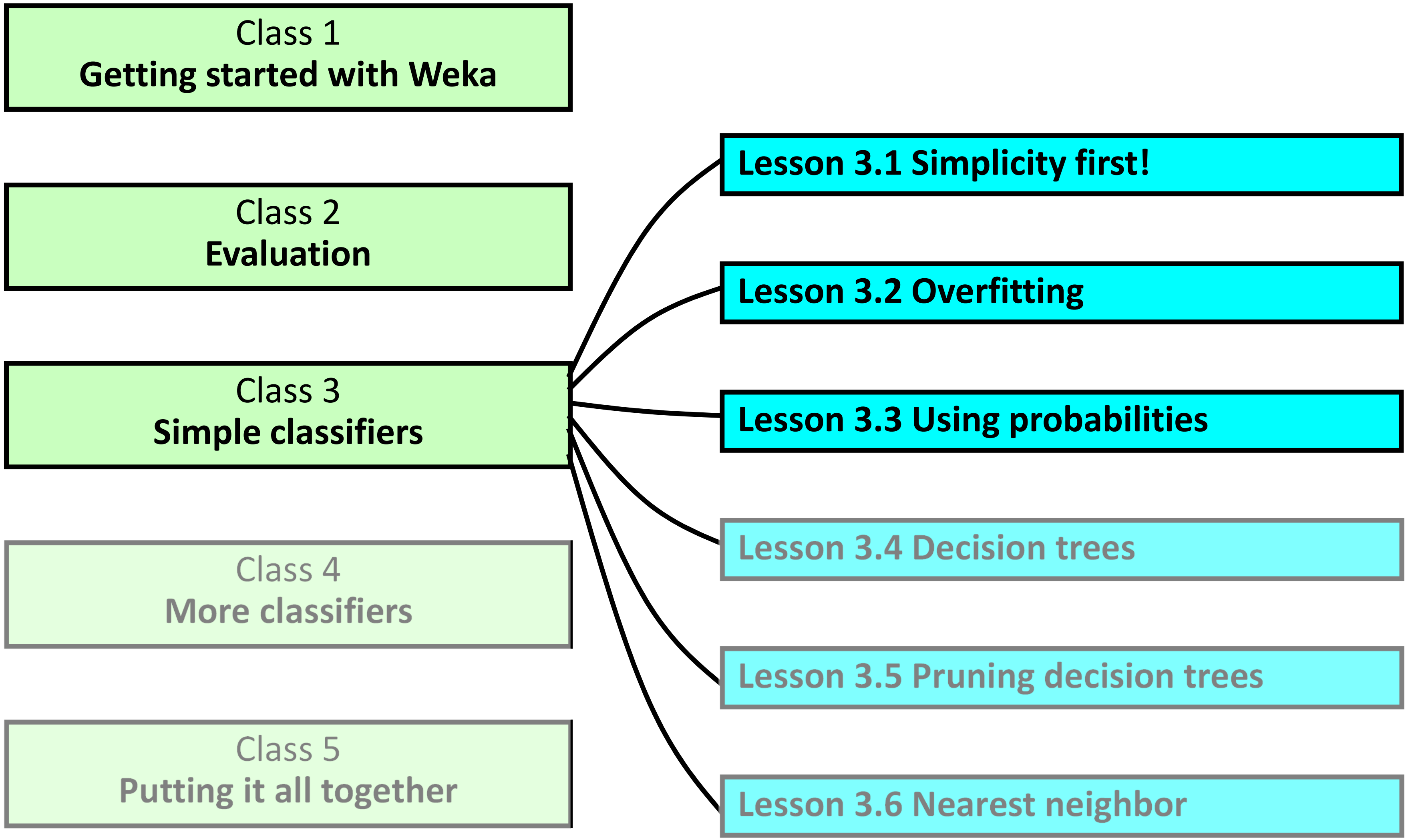
Using probabilities

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 3.3 Using probabilities



Lesson 3.3 Using probabilities

(OneR: One attribute does all the work)

Opposite strategy: use *all* the attributes

"Naïve Bayes" method

- Two assumptions: Attributes are
  - *equally important a priori*
  - *statistically independent (given the class value)*

    i.e., knowing the value of one attribute says nothing about the value of another *(if the class is known)*
- Independence assumption is never correct!
- But … often works well in practice

Lesson 3.3 Using probabilities

Probability of event *H* given evidence *E*

$$\Pr[H \mid E] = \frac{\Pr[E \mid H]\Pr[H]}{\Pr[E]}$$

class

instance

- Pr[*H*] is *a priori* probability of *H*
  - *Probability of event before evidence is seen*
- Pr[*H* | *E*] is *a posteriori* probability of *H*
  - *Probability of event after evidence is seen*
- "Naïve" assumption:
  - *Evidence splits into parts that are independent*

$$\Pr[H \mid E] = \frac{\Pr[E_1 \mid H]\Pr[E_2 \mid H]\ldots\Pr[E_n \mid H]\Pr[H]}{\Pr[E]}$$

Thomas Bayes, British mathematician, 1702 –1761

Lesson 3.3 Using probabilities

Outlook			Temperature			Humidity			Wind			Play	
	Yes	*No*		*Yes*	*No*		*Yes*	*No*		*Yes*	*No*	*Yes*	*No*
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								

Outlook	Temp	Humidity	Wind	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

$$\Pr[H \mid E] = \frac{\Pr[E_1 \mid H]\Pr[E_2 \mid H]\ldots\Pr[E_n \mid H]\Pr[H]}{\Pr[E]}$$

Lesson 3.3 Using probabilities

Outlook			Temperature			Humidity			Wind			Play	
	Yes	*No*		*Yes*	*No*		*Yes*	*No*		*Yes*	*No*	*Yes*	*No*
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								

A new day:

Outlook	Temp.	Humidity	Wind	Play
Sunny	Cool	High	True	?

$$\Pr[H \mid E] = \frac{\Pr[E_1 \mid H]\Pr[E_2 \mid H]\ldots\Pr[E_n \mid H]\Pr[H]}{\Pr[E]}$$

Likelihood of the two classes

For "yes" = 2/9 × 3/9 × 3/9 × 3/9 × 9/14 = 0.0053

For "no" = 3/5 × 1/ × 4/5 × 3/5 × 5/14 = 0.0206

Conversion into a probability by normalization:

P("yes") = 0.0053 / (0.0053 + 0.0206) = 0.205

P("no") = 0.0206 / (0.0053 + 0.0206) = 0.795

Lesson 3.3 Using probabilities

Outlook	Temp.	Humidity	Wind	Play
Sunny	Cool	High	True	?

Evidence E

Probability of class "yes"

$$\Pr[yes \mid E] = \Pr[Outlook = Sunny \mid yes]$$
$$\times \Pr[Temperature = Cool \mid yes]$$
$$\times \Pr[Humidity = High \mid yes]$$
$$\times \Pr[Windy = True \mid yes]$$
$$\times \frac{\Pr[yes]}{\Pr[E]}$$

$$= \frac{\frac{2}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{9}{14}}{\Pr[E]}$$

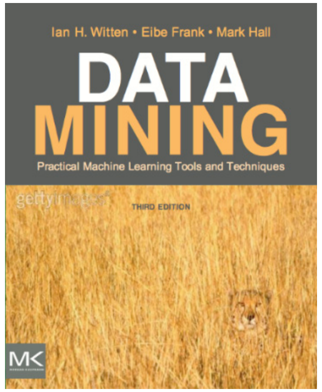
Lesson 3.3 Using probabilities

Use Naïve Bayes

- Open file **weather.nominal.arff**
- Choose Naïve Bayes method (**bayes>NaiveBayes**)
- Look at the classifier
- Avoid zero frequencies: start all counts at 1

Lesson 3.3 Using probabilities

- “Naïve Bayes”: all attributes contribute equally and independently
- Works surprisingly well
  - even if independence assumption is clearly violated
- Why?
  - classification doesn’t need accurate probability estimates
    so long as the greatest probability is assigned to the correct class
- Adding redundant attributes causes problems
  (e.g. identical attributes) → *attribute selection*



Course text

- Section 4.2 *Statistical modeling*

Data Mining with Weka

Class 3 – Lesson 4

Decision trees

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

Lesson 3.4 Decision trees

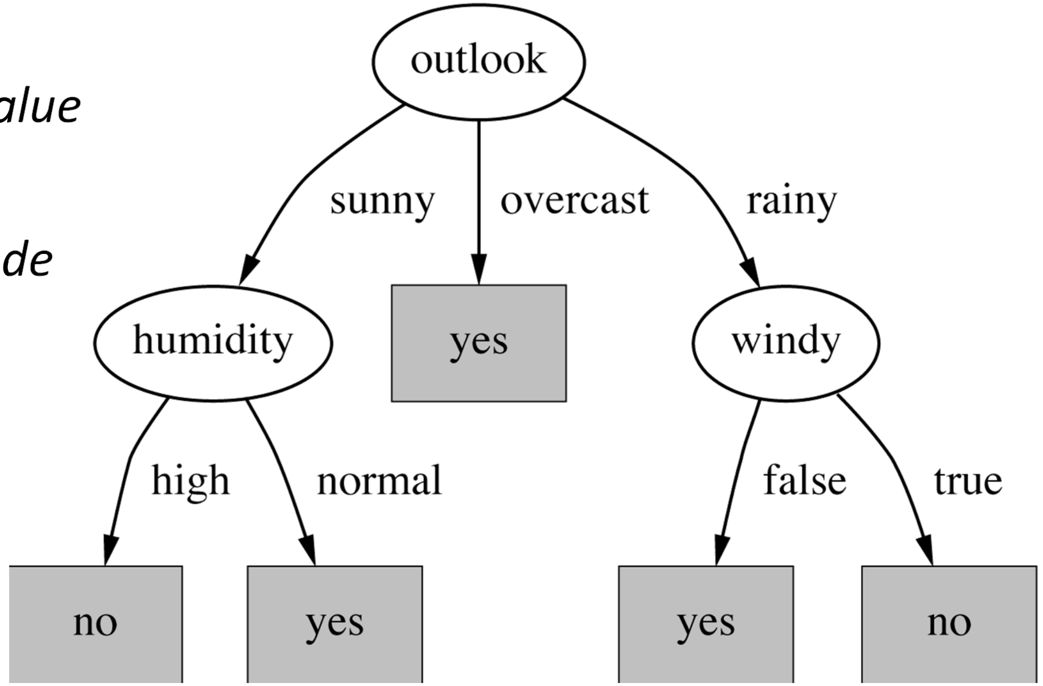
- Class 1 **Getting started with Weka**
- Class 2 **Evaluation**
- Class 3 **Simple classifiers**
- Class 4 **More classifiers**
- Class 5 **Putting it all together**

- **Lesson 3.1 Simplicity first!**
- **Lesson 3.2 Overfitting**
- **Lesson 3.3 Using probabilities**
- **Lesson 3.4 Decision trees**
- **Lesson 3.5 Pruning decision trees**
- **Lesson 3.6 Nearest neighbor**

Lesson 3.4 Decision trees

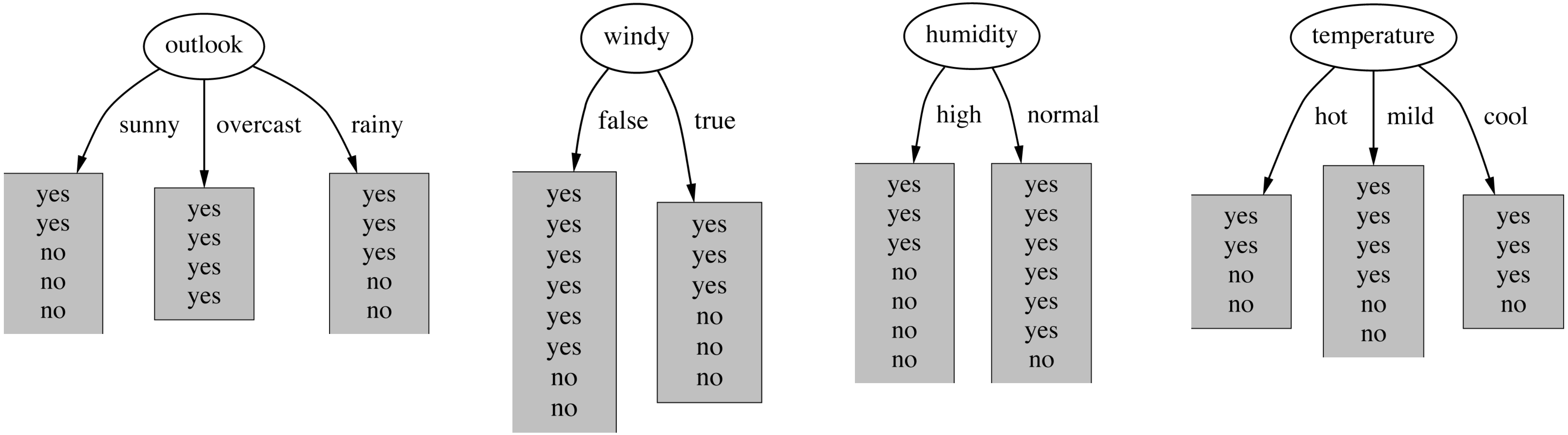
Top-down: recursive *divide-and-conquer*

- **Select** attribute for root node
  - *Create branch for each possible attribute value*
- **Split** instances into subsets
  - *One for each branch extending from the node*
- **Repeat** recursively for each branch
  - *using only instances that reach the branch*
- **Stop**
  - *if all instances have the same class*



Lesson 3.4 Decision trees

Which attribute to select?



Lesson 3.4 Decision trees

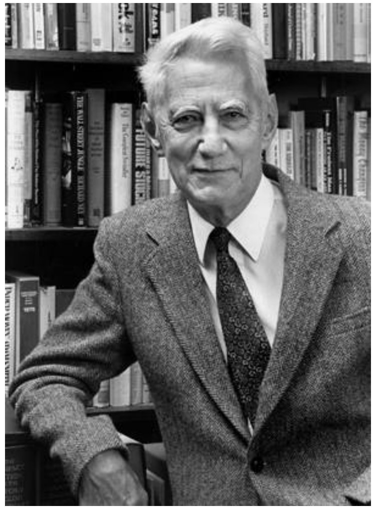
Which is the best attribute?

- Aim: to get the smallest tree
- Heuristic
  - *choose the attribute that produces the "purest" nodes*
  - *I.e. the greatest information gain*
- Information theory: measure information in bits

$$\text{entropy}(p_1, p_2, \ldots, p_n) = -p_1 \log p_1 - p_2 \log p_2 \ldots - p_n \log p_n$$

Information gain

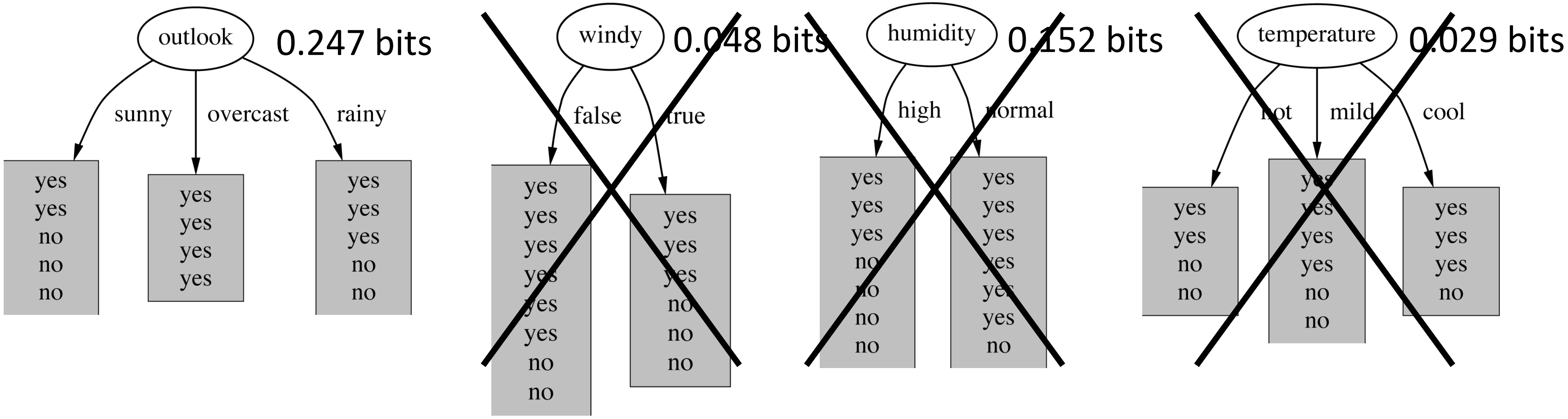
- Amount of information gained by knowing the value of the attribute
- (Entropy of distribution before the split) – (entropy of distribution after it)



Claude Shannon, American mathematician and scientist 1916–2001

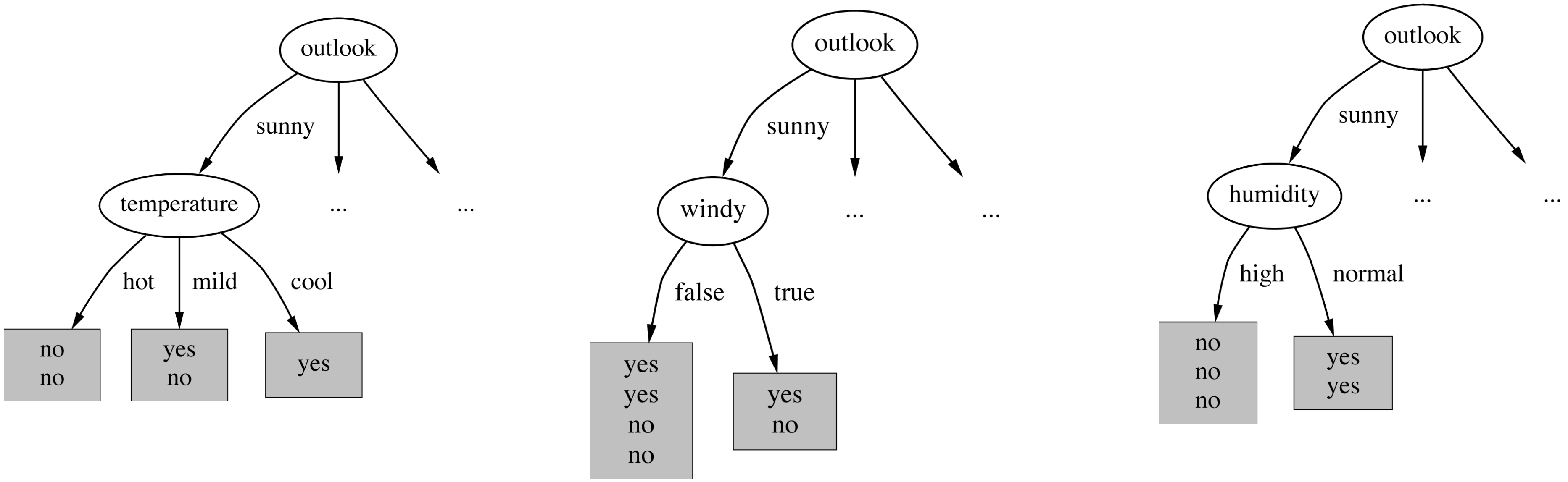
Lesson 3.4 Decision trees

Which attribute to select?



Lesson 3.4 Decision trees

Continue to split …



gain(*temperature*) = 0.571 bits

gain(*windy*) = 0.020 bits

gain(*humidity*) = 0.971 bits

Lesson 3.4 Decision trees

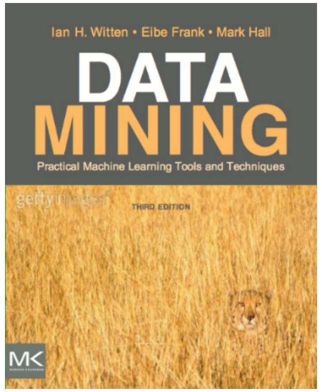
Use J48 on the weather data

- Open file **weather.nominal.arff**
- Choose J48 decision tree learner (**trees>J48**)
- Look at the tree
- Use right-click menu to visualize the tree

Lesson 3.4 Decision trees

- **J48: “top-down induction of decision trees”**
- Soundly based in information theory
- Produces a tree that people can understand
- Many different criteria for attribute selection
  - – rarely make a large difference
- Needs further modification to be useful in practice

  (next lesson)



Course text

- Section 4.3 *Divide-and-conquer: Constructing decision trees*

Data Mining with Weka

Class 3 – Lesson 5

Pruning decision trees

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 3.5 Pruning decision trees

Class 1
Getting started with Weka

Class 2
Evaluation

Class 3
Simple classifiers

Class 4
More classifiers

Class 5
Putting it all together

Lesson 3.1 Simplicity first!

Lesson 3.2 Overfitting

Lesson 3.3 Using probabilities

Lesson 3.4 Decision trees

Lesson 3.5 Pruning decision trees

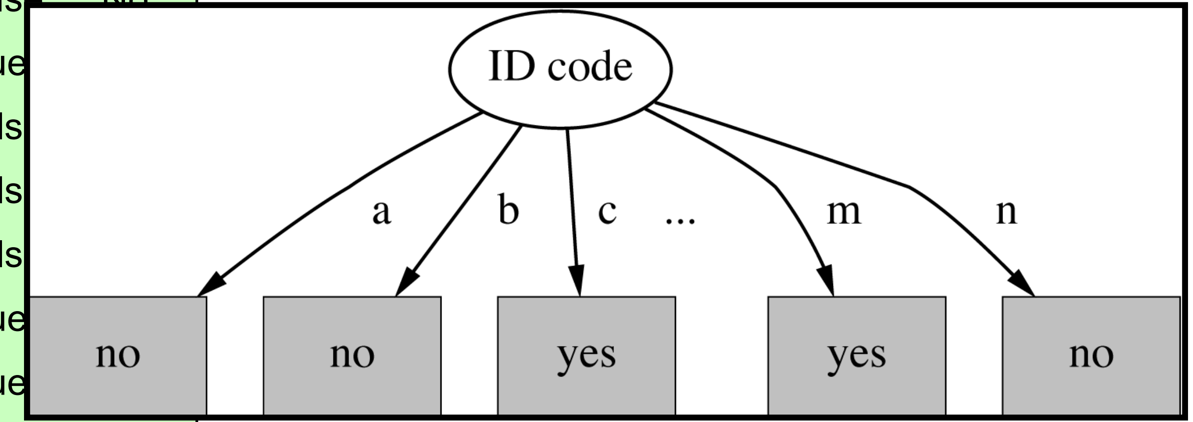
Lesson 3.6 Nearest neighbor

Lesson 3.5 Pruning decision trees

Lesson 3.5 Pruning decision trees

Highly branching attributes — Extreme case: ID code

ID code	Outlook	Temp	Humidity	Wind	Play
a	Sunny	Hot	High	False	No
b	Sunny	Hot	High	True	
c	Overcast	Hot	High	Fals	
d	Rainy	Mild	High	Fals	
e	Rainy	Cool	Normal	Fals	
f	Rainy	Cool	Normal	True	
g	Overcast	Cool	Normal	True	
h	Sunny	Mild	High	False	No
i	Sunny	Cool	Normal	False	Yes
j	Rainy	Mild	Normal	False	Yes
k	Sunny	Mild	Normal	True	Yes
l	Overcast	Mild	High	True	Yes
m	Overcast	Hot	Normal	False	Yes
n	Rainy	Mild	High	True	No



Information gain is maximal (0.940 bits)

Lesson 3.5 Pruning decision trees

How to prune?

- Don't continue splitting if the nodes get very small
  (J48 minNumObj parameter, default value 2)
- Build full tree and then work back from the leaves, applying a statistical test at each stage
  (confidenceFactor parameter, default value 0.25)
- Sometimes it's good to prune an interior node, raising the subtree beneath it up one level
  (subtreeRaising, default *true*)
- Messy … complicated … not particularly illuminating

Lesson 3.5 Pruning decision trees

Over-fitting (again!)

Sometimes simplifying a decision tree gives better results

- Open file **diabetes.arff**
- Choose J48 decision tree learner (**trees>J48**)

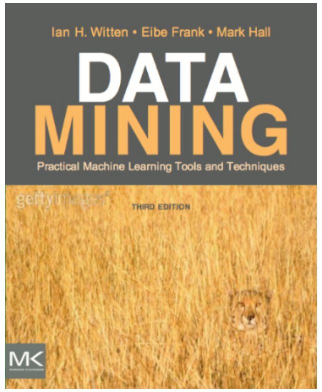
Prunes by default:	73.8% accuracy, tree has	20 leaves, 39 nodes
Turn off pruning:	72.7%	22 leaves, 43 nodes

- Extreme example: **breast-cancer.arff**

Default (pruned):	75.5% accuracy, tree has	4 leaves, 6 nodes
Unpruned:	69.6%	152 leaves, 179 nodes

Lesson 3.5 Pruning decision trees

- C4.5/J48 is a popular early machine learning method
- Many different pruning methods
  - mainly change the size of the pruned tree
- Pruning is a general technique that can apply to structures other than trees (e.g. decision rules)
- Univariate vs. multivariate decision trees
  - Single vs. compound tests at the nodes
- From C4.5 to J48 (recall Lesson 1.4)



Course text

- Section 6.1 *Decision trees*

Ross Quinlan,
Australian computer scientist

Data Mining with Weka

Class 3 – Lesson 6

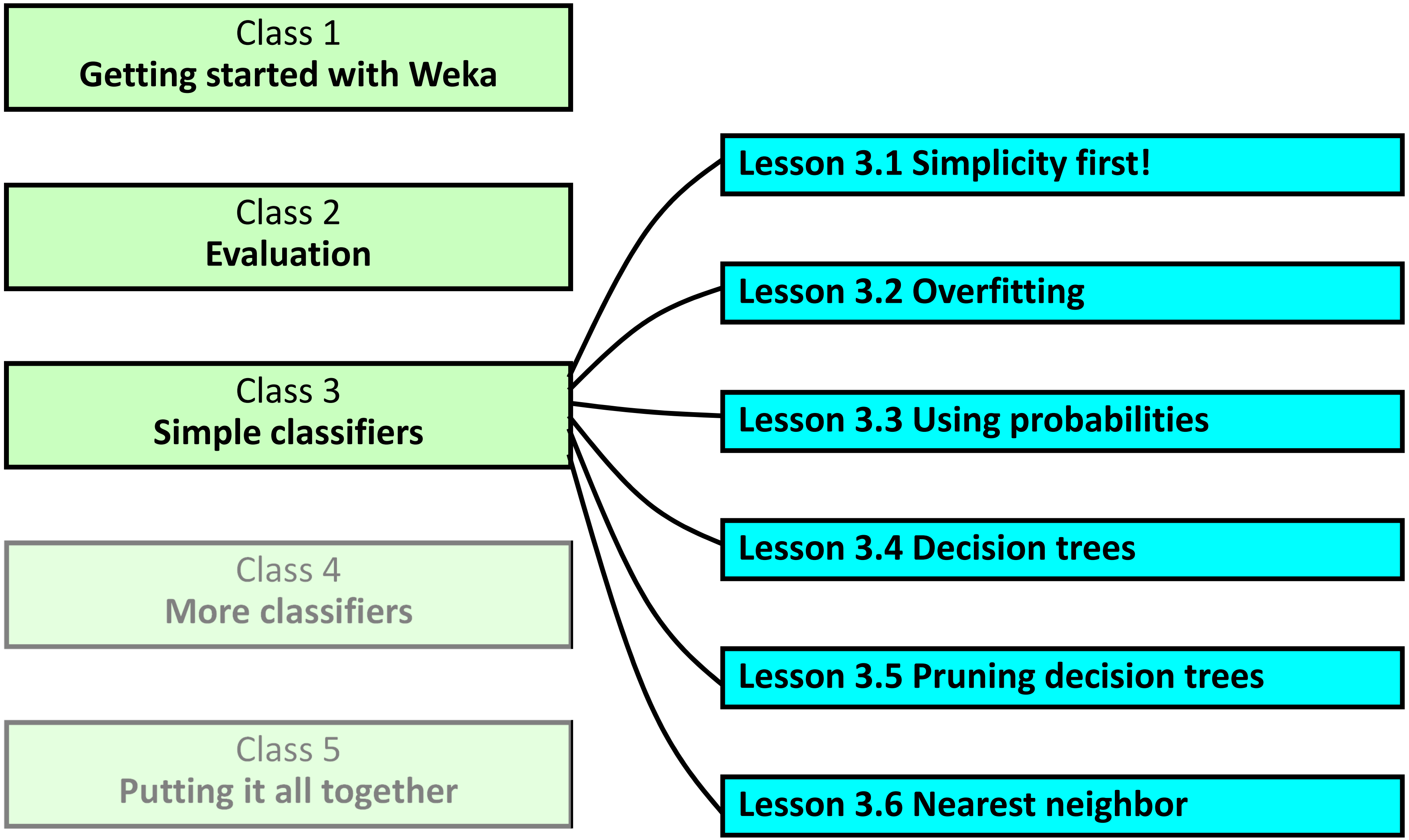
Nearest neighbor

Ian H. Witten

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz

Lesson 3.6 Nearest neighbor

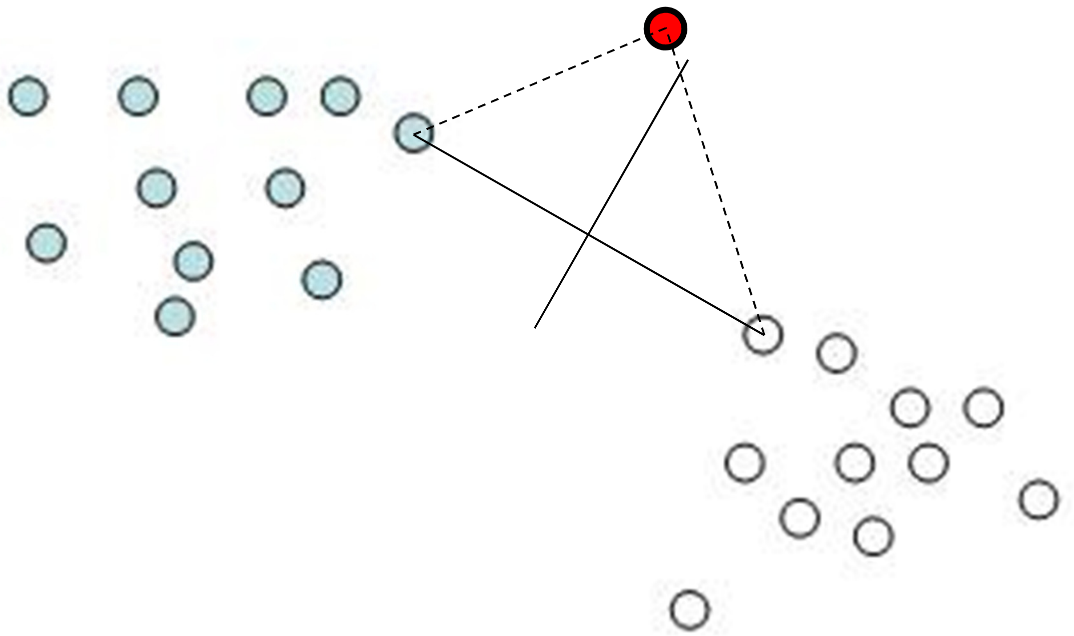


Lesson 3.6 Nearest neighbor

"Rote learning": simplest form of learning

- To classify a new instance, search training set for one that's "most like" it
  - *the instances themselves represent the "knowledge"*
  - *lazy learning: do nothing until you have to make predictions*
- "Instance-based" learning = "nearest-neighbor" learning

Lesson 3.6 Nearest neighbor



Lesson 3.6 Nearest neighbor

Search training set for one that's "most like" it

- Need a similarity function
  - *Regular ("Euclidean") distance? (sum of squares of differences)*
  - *Manhattan ("city-block") distance? (sum of absolute differences)*
  - *Nominal attributes? Distance = 1 if different, 0 if same*
  - *Normalize the attributes to lie between 0 and 1?*

Lesson 3.6 Nearest neighbor

What about noisy instances?

- Nearest-neighbor
- *k*-nearest-neighbors
  - *choose majority class among several neighbors (k of them)*
- In Weka,

  lazy>IBk (instance-based learning)

Lesson 3.6 Nearest neighbor

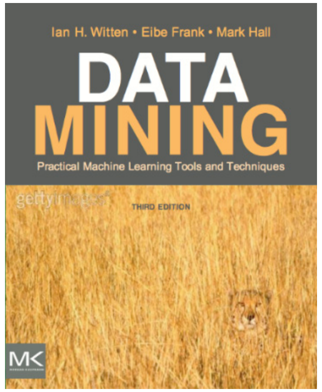
Investigate effect of changing *k*

- Glass dataset
- lazy > IBk, *k* = 1, 5, 20
- 10-fold cross-validation

k = 1	*k* = 5	*k* = 20
70.6%	67.8%	65.4%

Lesson 3.6 Nearest neighbor

- Often very accurate … but slow:
  - scan entire training data to make each prediction?
  - sophisticated data structures can make this faster
- Assumes all attributes equally important
  - Remedy: attribute selection or weights
- Remedies against noisy instances:
  - Majority vote over the k nearest neighbors
  - Weight instances according to prediction accuracy
  - Identify reliable "prototypes" for each class
- Statisticians have used k-NN since 1950s
  - If training set size $n \to \infty$ and $k \to \infty$ and $k/n \to 0$, error approaches minimum



Course text

- Section 4.7 *Instance-based learning*

Data Mining with Weka

Department of Computer Science
University of Waikato
New Zealand

weka.waikato.ac.nz