Lab 5

Decision trees

Useful information

Alex Becheru

irlab.becheru.net

irlab@becheru.net

Overfitting

An FBI forensic lab can determine if a person is european or asian just by looking at their hair attributes.

A lot of factors can influence hair attributes, like olive oil. It is known that people which use olive oil have healthier hair.

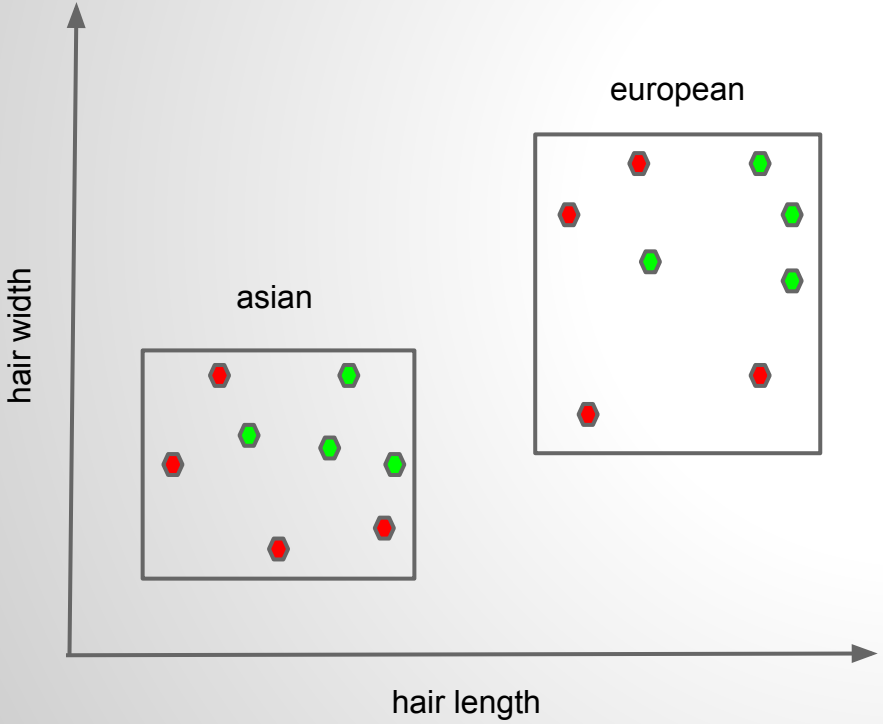
Imagine that the forensic lab hired a team of IR experts to build them an automatic classifier.

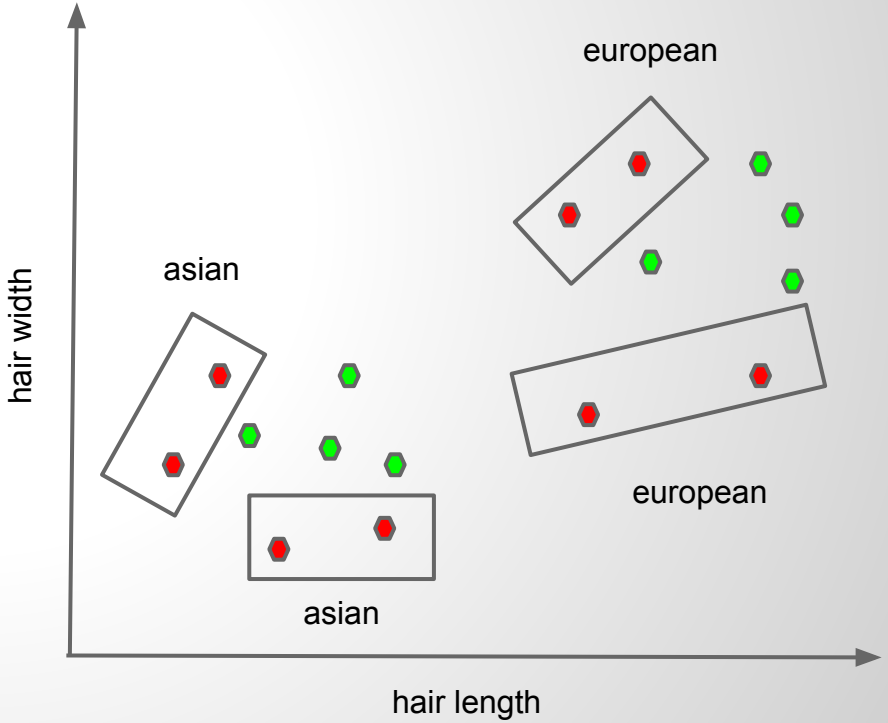
The team of experts built two classifiers based on people that do not use frequently olive oil. Which classifier do you think is better?

hair without olive oil

hair with olive oil

The classifier on the left is better as it successfully classifies even people that use olive oil.

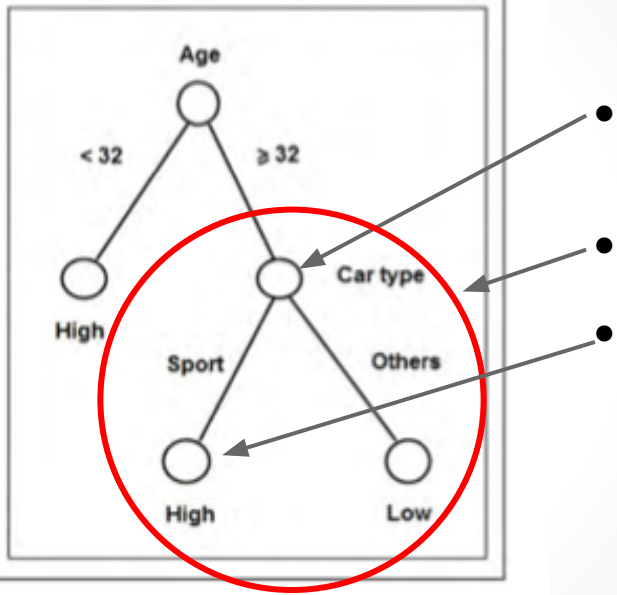




Decisions tree

Decisions trees are used to predict the membership of objects to different categories

Age	Car type	Accident risk
20	Sport	High
18	Sport	High
40	Minivan	Low
50	Premium	Low
35	Compact	Low
30	Sport	High
32	Sport	High
40	Full size Van	Low
33	Mini	High
39	Convertible	Low



- Each internal node expresses the testing made on a certain attribute
- Each branch expresses a test result
- Each terminal node represents the decisions classes

Basic algorithm (Hunt's)

1. at node t we have Ot set of objects
2. If Ot has objects of the same class then Ot is a leaf node.
3. If Ot has objects of different classes then we use an attribute to further split the node t. In this case node t is an internal node

Question 1?

How do we split a node?

A node must be split by an attribute that gives the most pure terminal nodes.

Driver's Age	Car Type	Accident (class attribute)
<30	Sport	Yes
<30	Family	Yes
>30	Sport	No

If we chose the attribute "Car Type" to split then:

- the terminal node "Car Type=Family" will have all objects classified "Accident=Yes". This means that all the people that have family cars are likely to make an accident
- the terminal node "Car Type=Sport" will 50% of objects classified with a risk of accident and the rest without a risk of accident.

1/2 pure node

If we chose the attribute "Driver's age" to split then:

- the terminal node "Driver's age<30" will have all objects classified "Accident=Yes". This means that all the people that are under 30 are very likely to have an accident
- the terminal node "Driver's age>30" will have also all the nodes classified with "Accident=No".

2/2 pure nodes

Impurity indices

Split indices or impurity indices measures a node impurity. The are several types of indices:

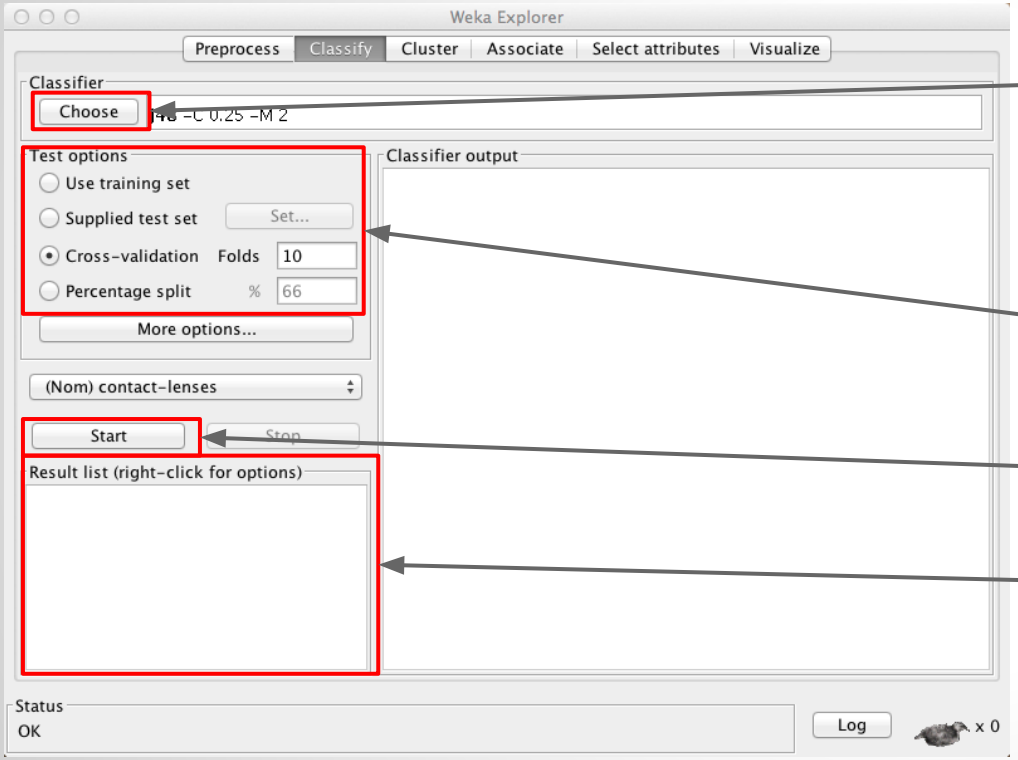
- Gini
- Entropy
- Misclassification
- Chi-square measure

Things to know

The entropy of a tree is a measure that characterizes a tree.

The information gain of an attribute determines by what degree the attribute creates pure nodes.

Things to know about the classification process in Weka



Choose an algorithm

Choose a testing method

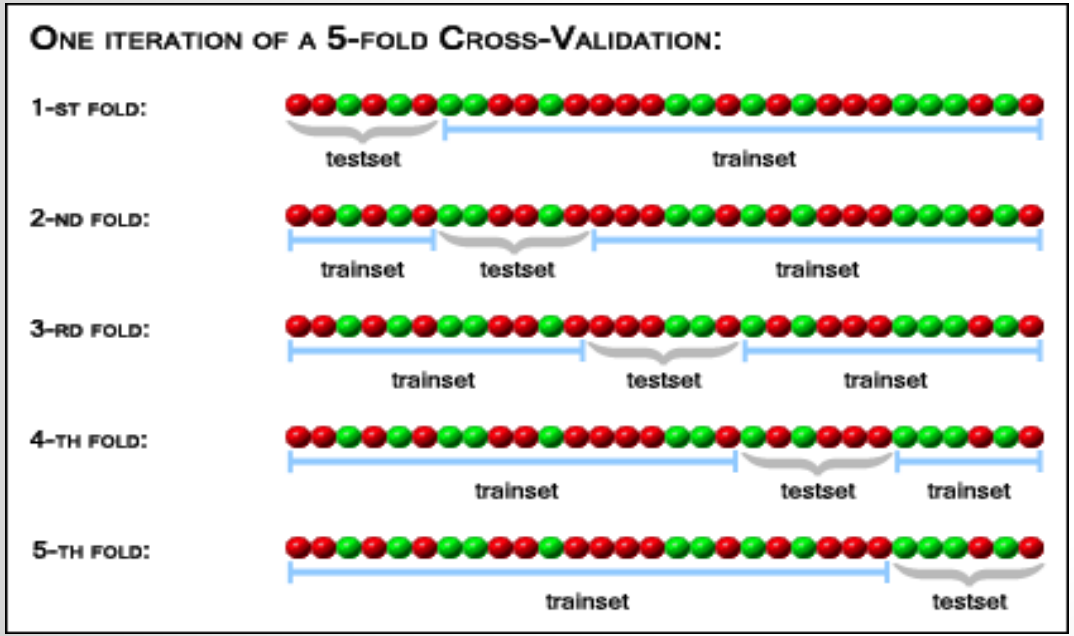
Start the algorithm

Results list

Testing methods

- use training set. Uses the same file for learning and testing.
- supplied test set. Use another file to test the classification that was done on the current file
- percentage split. Split the current file in 2 separate subfiles. One subfile used for learning and another for testing

Cross Validation



It is a repeated percentage split test.

number of folds = how many times the percentage split is done

Each fold a different subset for testing is used.

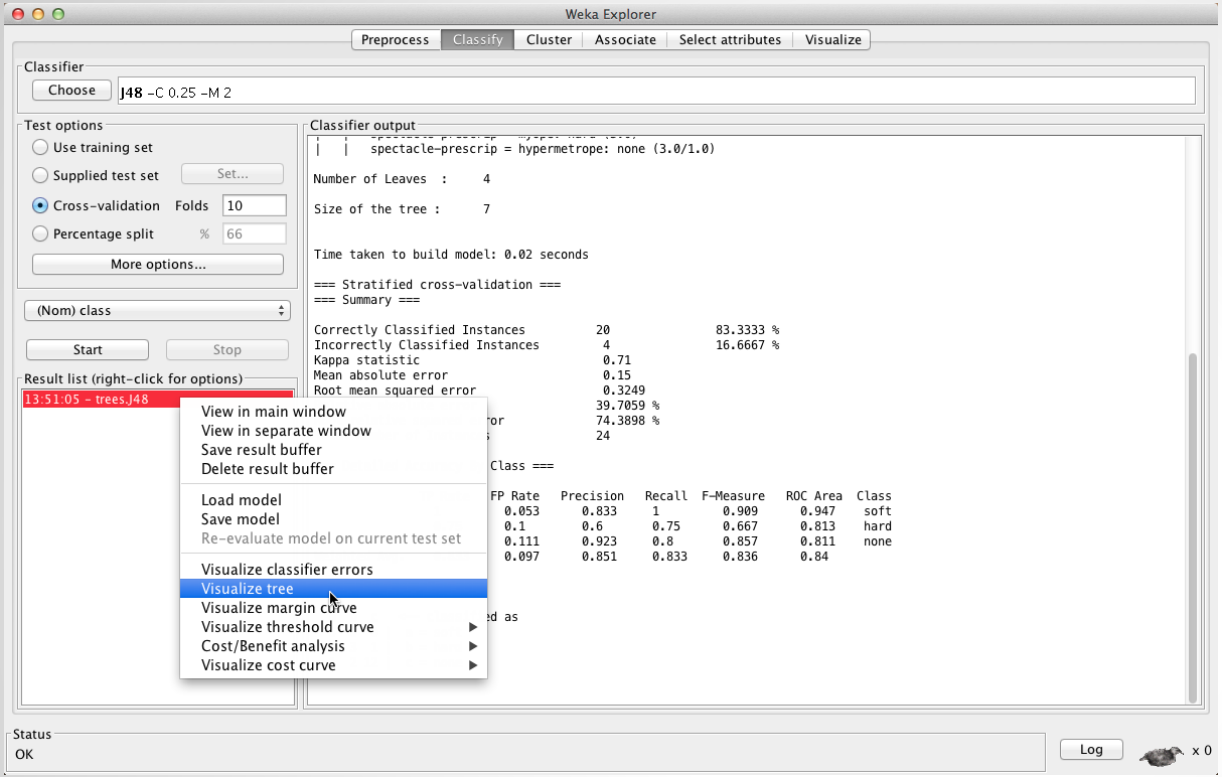
How to read Weka classification results

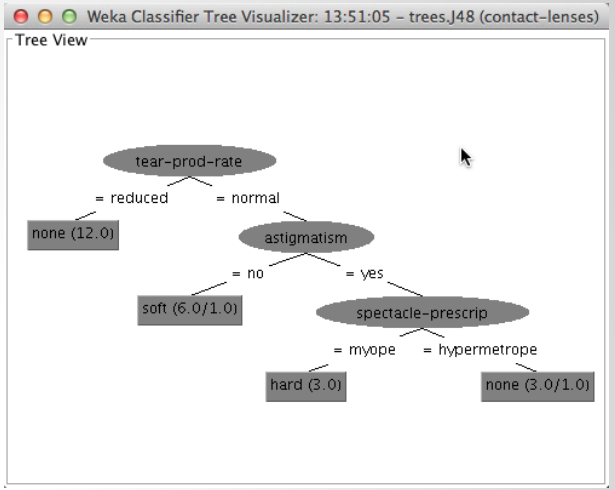
```
Number of Leaves  :     4

Size of the tree :      7


Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          20               83.3333 %
Incorrectly Classified Instances         4               16.6667 %
Kappa statistic                          0.71
Mean absolute error                      0.15
Root mean squared error                  0.3249
Relative absolute error                 39.7059 %
Root relative squared error             74.3898 %
Total Number of Instances               24

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0.053      0.833     1         0.909      0.947    soft
                 0.75      0.1        0.6       0.75      0.667      0.813    hard
                 0.8       0.111      0.923     0.8       0.857      0.811    none
Weighted Avg.    0.833     0.097      0.851     0.833     0.836      0.84

=== Confusion Matrix ===

 a  b  c   <-- classified as
 5  0  0 |  a = soft
 0  3  1 |  b = hard
 1  2 12 |  c = none
```

Recall is the number of correct results divided by the number of results that should have been returned

Confusion matrix

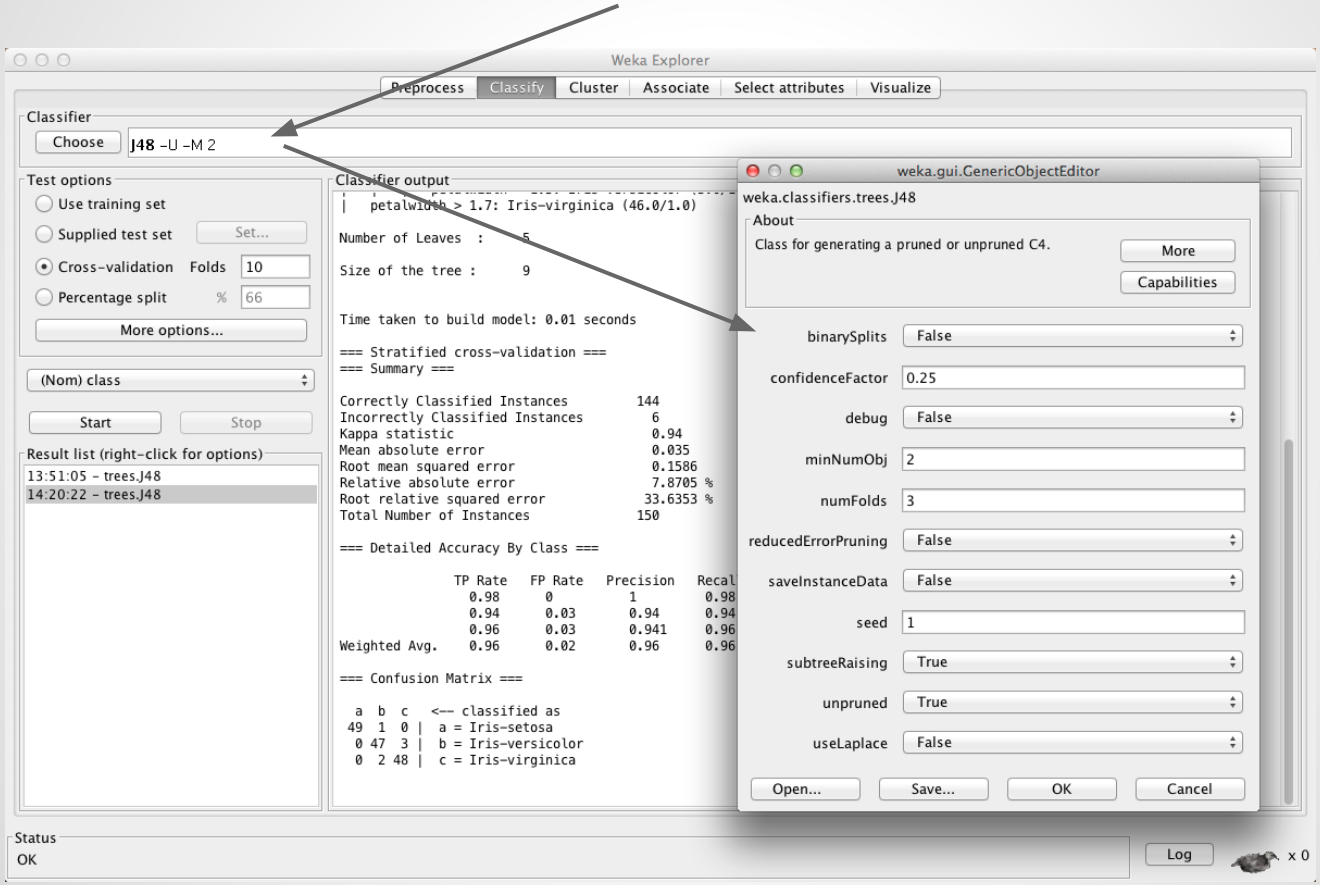
a	b	c		
5	0	0	a	all the instances in class a are classified as class a
0	3	1	b	3 of the instances in class b are classified as b and 1 is classified as c
1	2	12	c	1 in C, 2 in b, and 12 in C





You can visualise the tree by right clicking on a result and choosing “visualise tree”

You can set up the algorithms parameters by clicking on it's name



J48 algorithm

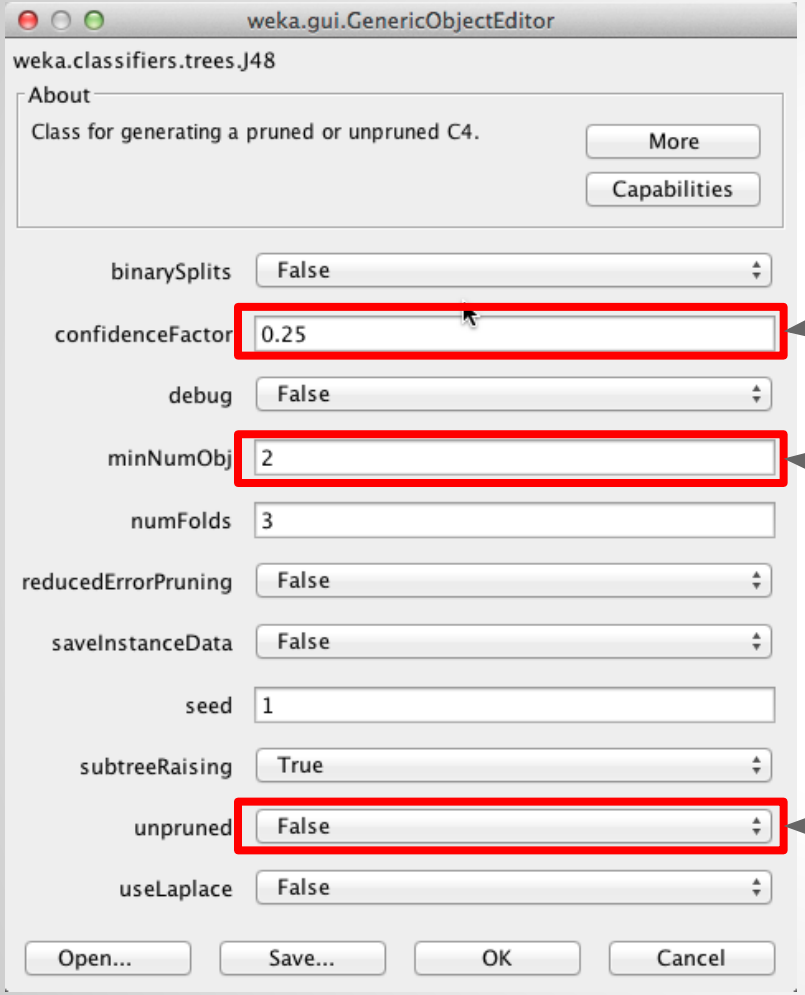
Actually called C4.5

Based on Entropy

This algorithm has a few base cases.

- All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class.
- None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class.
- Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.

1. Check for base cases
2. For each attribute a
3. Find the normalized information gain from splitting on a
4. Let a_best be the attribute with the highest normalized information gain
5. Create a decision node that splits on a_best
6. Recurse on the sublists obtained by splitting on a_best, and add those nodes as children of node



The amount of pruning

Minimum number of objects
that can form a terminal leaf

False = apply pruning
True = no pruning

Pruning

It’s a method to avoid overfitting of data.

It aggregates leaf nodes that are specific to the training set into larger leafs. Especially the leaves that contain too few objects.

Exercise 1

- open iris.arff
- apply J48 with cross validation=10
- apply J48 with percentage split=10%
- observe the differences

Exercise 2

- open labor.arff
- apply J48 with percentage split=10%
- look at the confusion matrix say which class is not identified
- apply J48 with cross validation=10
- observe the difference

Exercise 3

- open labor.arff
- determine which are the best attributes to use with J48

Exercise 4

- open glass.arff
- apply J48 with cross validation=10
- apply J48 with pruning
- observe the difference

Exercise 5

- open glass.arff
- apply J48 with pruning and the minimal number of objects =10
- compare with results from exercise 4

Exercise 6

- compare J48 with Decision Stump
- use iris.arff, vote.arff

Homework

Learn to calculate a tree Entropy and information gain for attributes

Questions?