Learning from Data for Linguists
Lecture 3: Decision Trees and related issues

Malvina Nissim and Johannes Bjerva
m.nissim@rug.nl, j.bjerva@rug.nl

ESSLLI, 24 August 2016

Getting the updated code

START AS SOON AS YOU ENTER THE ROOM! (PLEASE)

- Approach 1: Use *git* (updateable, recommended if you have *git*)
  1. In your terminal, type: 'git clone https://github.com/bjerva/esslli-learning-from-data-students.git'
  2. Followed by 'cd esslli-learning-from-data-students'
  3. Whenever the code is updated, type: 'git pull'
- Approach 2: Download a zip (static)
  1. Download the zip archive from: `https://github.com/bjerva/esslli-learning-from-data-students/archive/master.zip`
  2. Whenever the code is updated, download the archive again.

Running an experiment

1. Navigate to your 'esslli-learning-from-data-students' (using cd in the terminal)
2. To extract features and learn model:
   python run_experiment.py −−csv data/trainset-sentiment-extra.csv −−nwords 1 −−algorithms nb

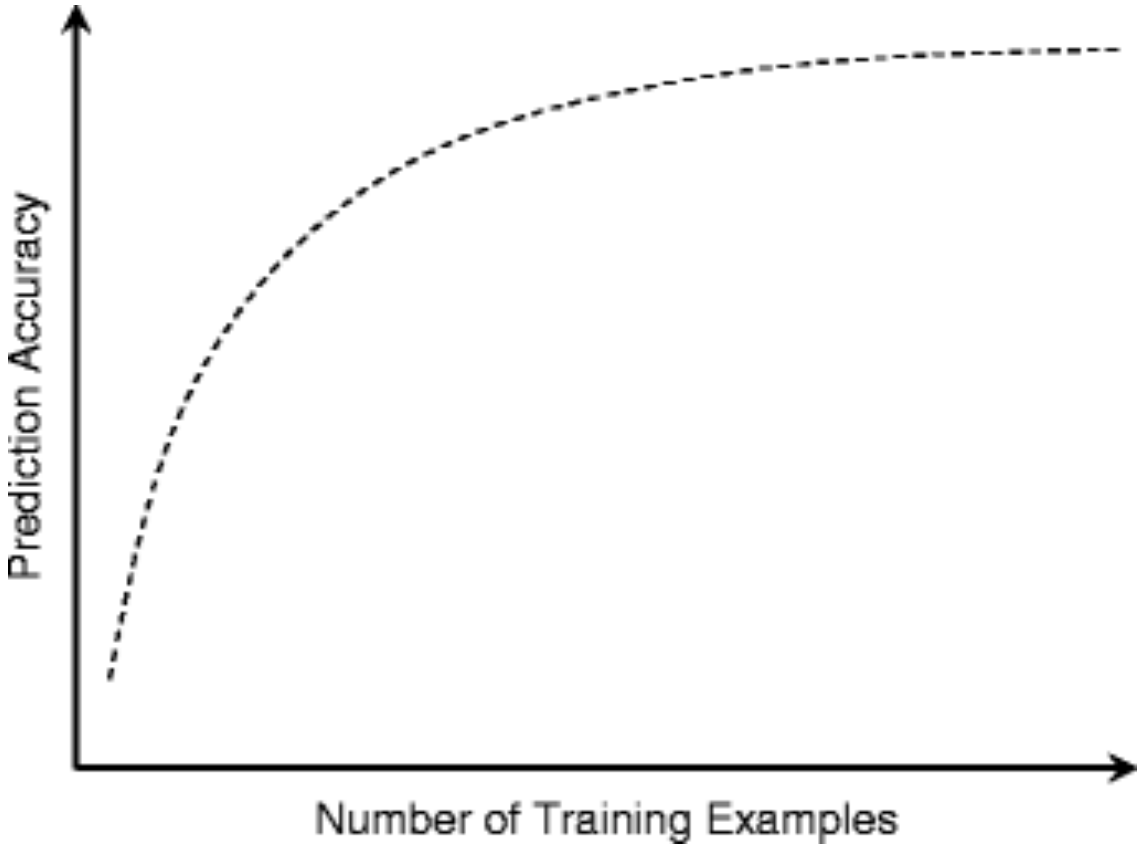
Learning Curves

what is a learning curve?

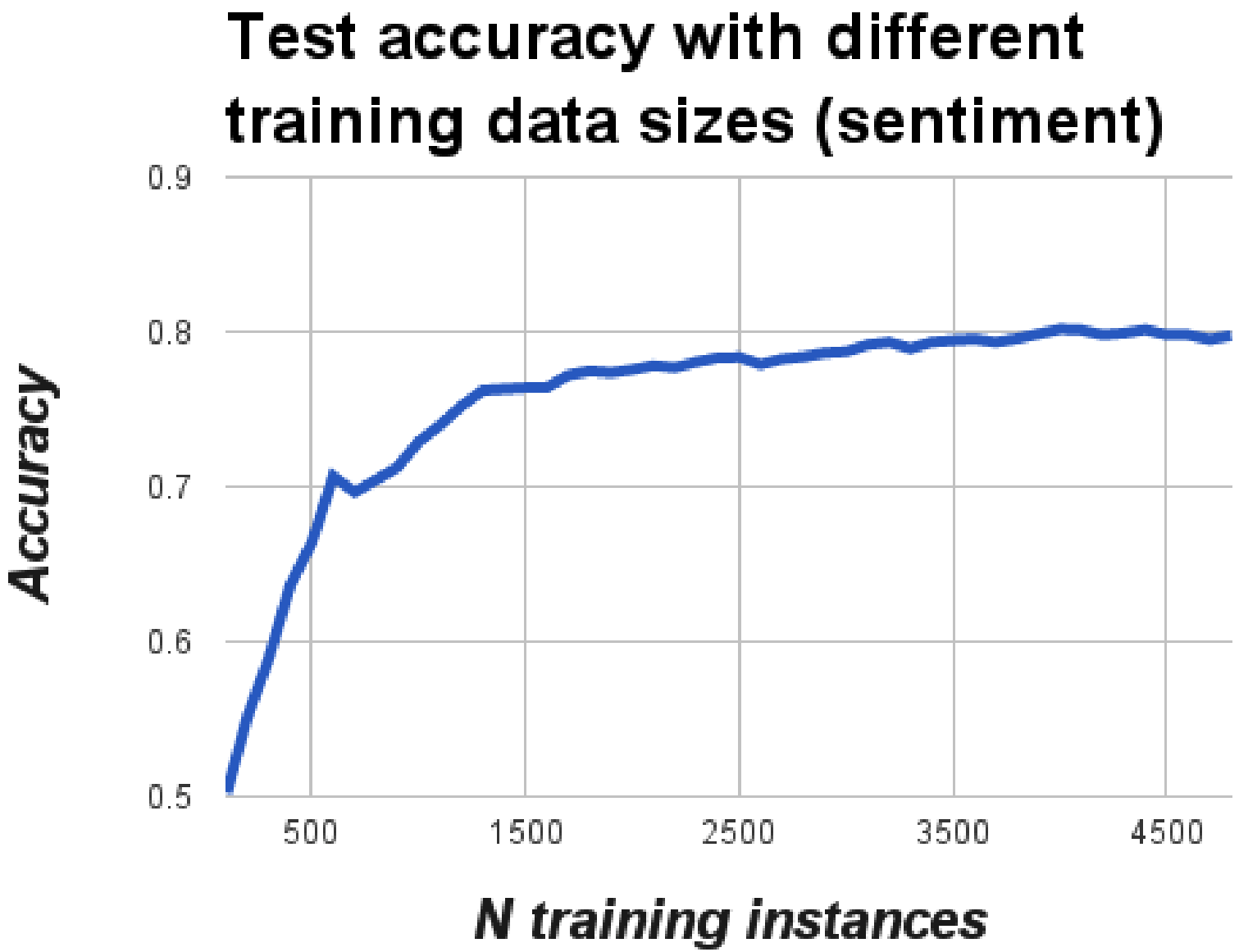
Learning Curves

what is a learning curve?

we can plot accuracy (or error rate) on one axis (y) and **training size** on the other (x)

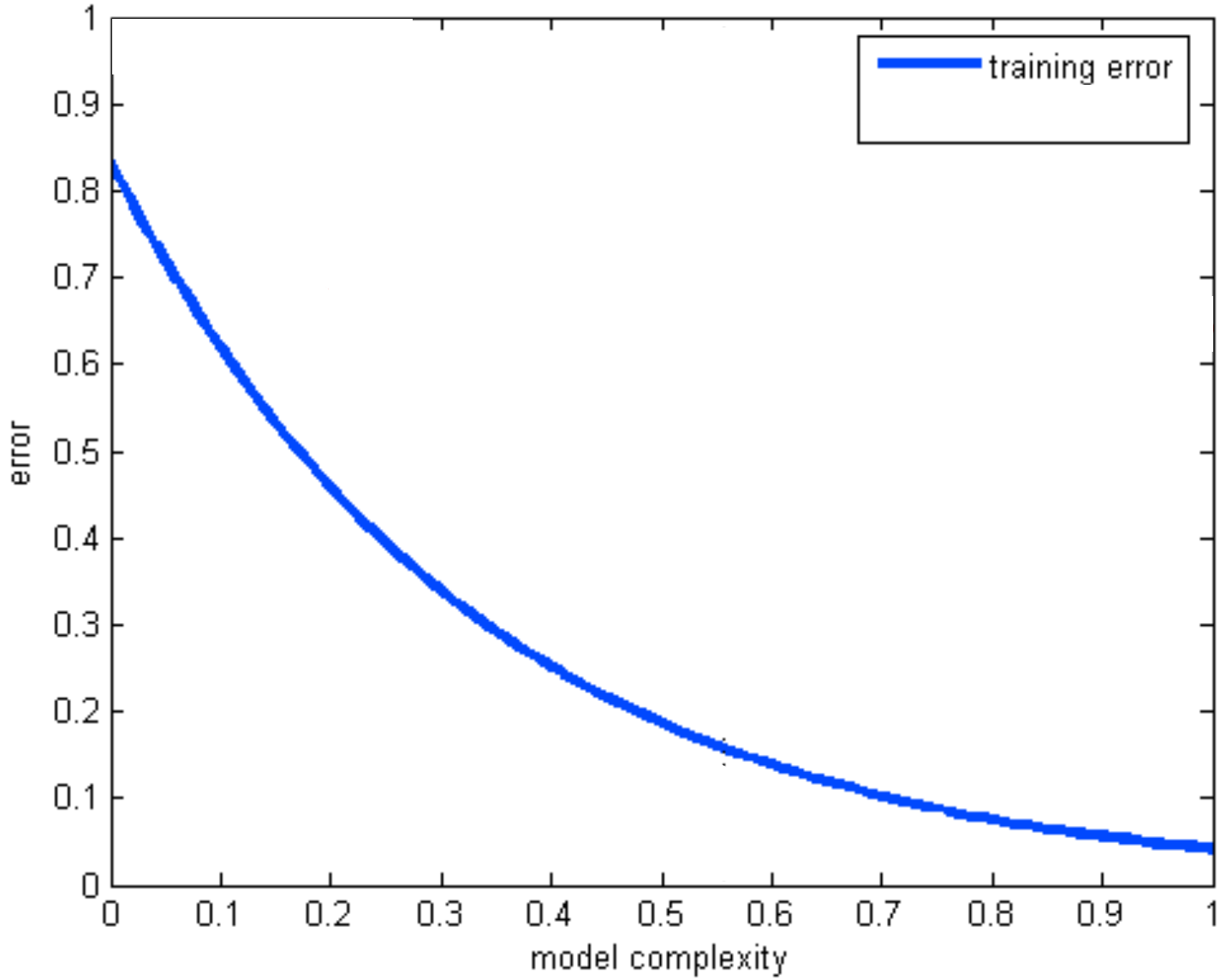


Size and Learning curve

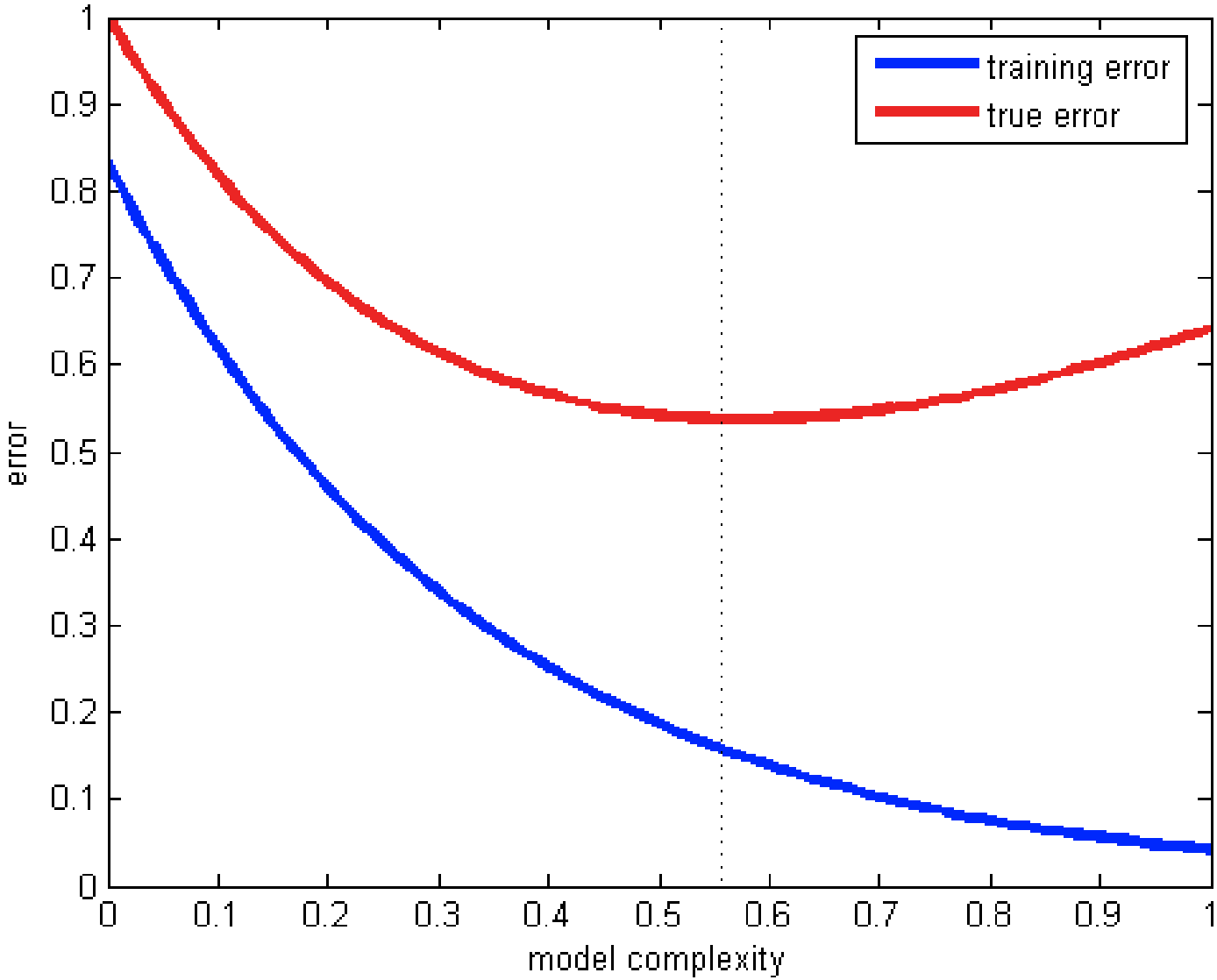


where do such scores come from?

Another curve (two, actually)



Another curve (two, actually)



Overfitting and Underfitting

$\rightarrow$ a model is overfitting when it fits the data too tightly, and it's modelling noise or error instead of generalising well

$\rightarrow$ how can we find this out?

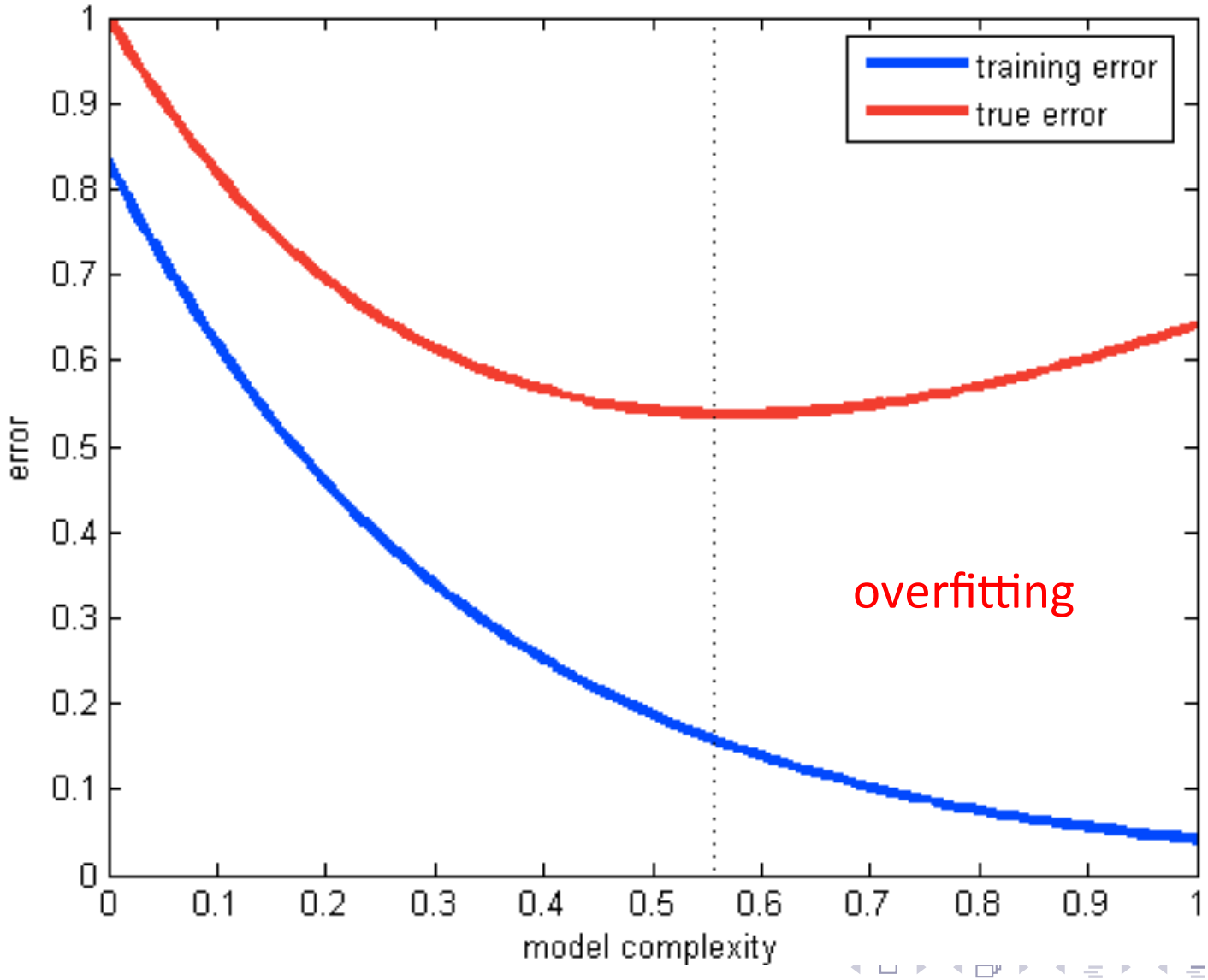
Overfitting and Underfitting

$\rightarrow$ a model is overfitting when it fits the data too tightly, and it's modelling noise or error instead of generalising well

$\rightarrow$ how can we find this out?

observe error: learning curve on training vs test data

Overfitting and Underfitting



poll

Overfitting and Underfitting

- What is a good metaphor for an overfitted model?
  - a lazy botanist: everything green is a tree
  - **a picky botanist: nothing he hasn't seen yet is a tree**
- What is a good metaphor for an underfitted model?
  - **a lazy botanist: everything green is a tree**
  - a picky botanist: nothing he hasn't seen yet is a tree

Decision Trees

(slides by Barbara Rosario)

Decision Trees

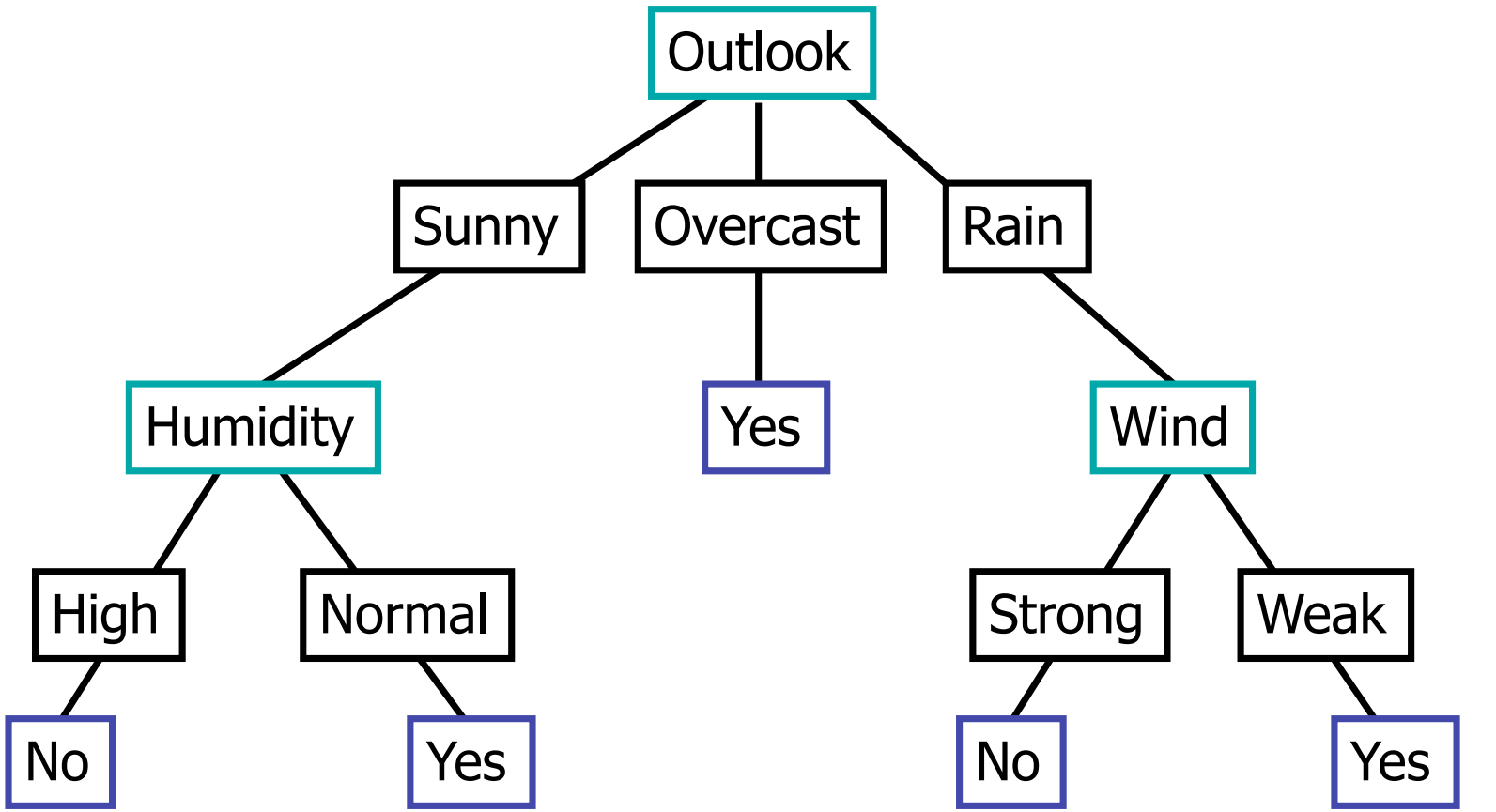
- *Decision tree* is a classifier in the form of a tree structure, where each node is either:
  - *Decision node* - specifies some test to be carried out on a single attribute-value, with one branch and sub-tree for each possible outcome of the test
  - *Leaf node* - indicates the value of the target attribute (class) of examples
- A decision tree can be used to classify an example by starting at the root of the tree and moving through it until a leaf node, which provides the classification of the instance.

Training Examples

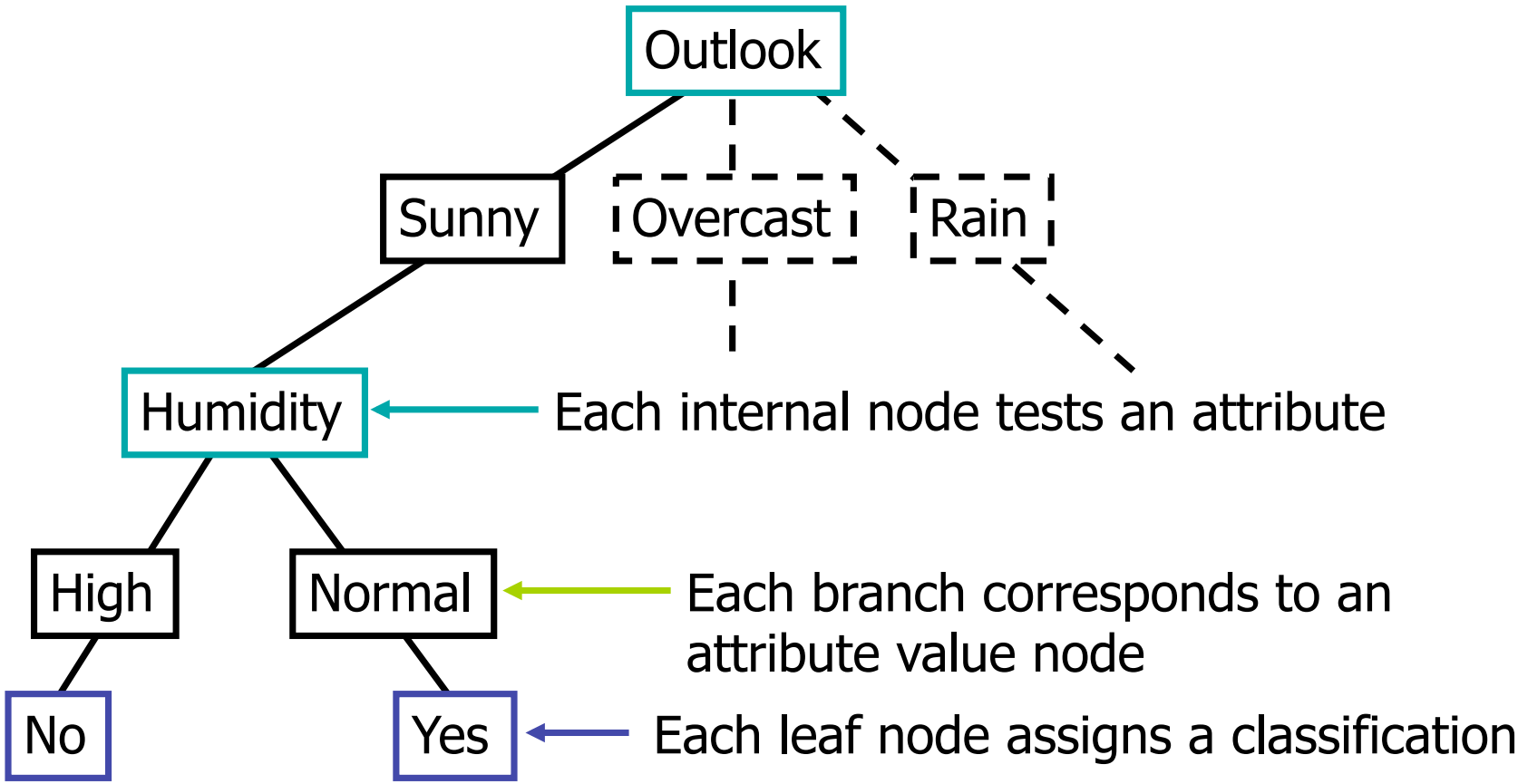
Goal: learn when we can play Tennis and when we cannot

Day	Outlook	Temp.	Humidity	Wind	**Play Tennis**
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

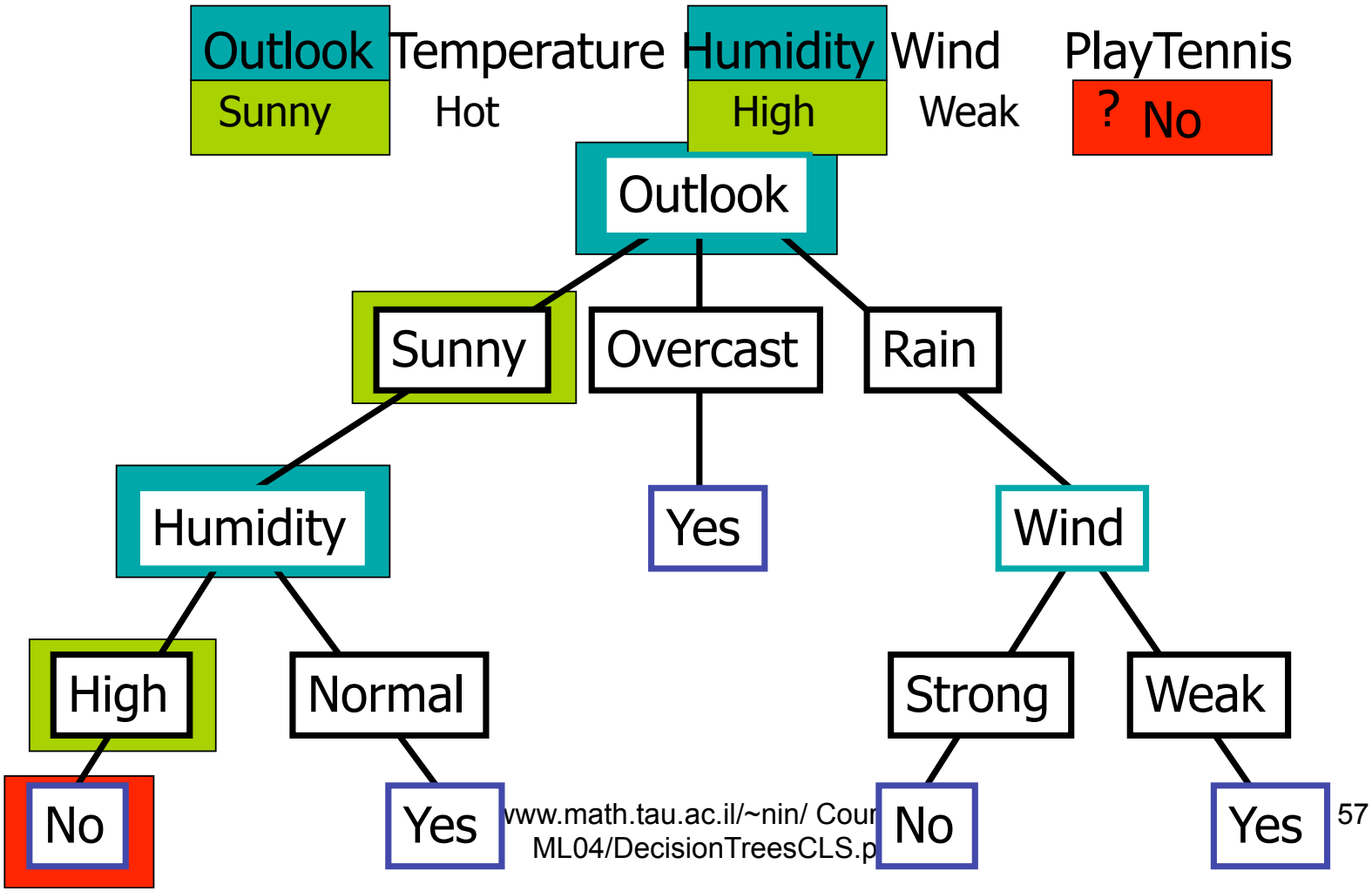
Decision Tree for PlayTennis



Decision Tree for PlayTennis



Decision Tree for PlayTennis



Building Decision Trees

- Once we have a decision tree, it is straightforward to use it to assign labels to new input values.
- How we can build a decision tree that models a given training set?

Building Decision Trees

- The central focus of the decision tree growing algorithm is selecting which attribute to test at each node in the tree. The goal is to select the attribute that is most useful for classifying examples.
- Top-down, greedy search through the space of possible decision trees.
  - That is, it picks the best attribute and never looks back to reconsider earlier choices.

Building Decision Trees

- Splitting criterion
  - Finding the features and the values to split on
    - for example, why test first “cts” and not “vs”?
    - Why test on “cts < 2” and not “cts < 5” ?
  - Split that gives us the *maximum information gain* (or the *maximum reduction of uncertainty*)
- Stopping criterion
  - When all the elements at one node have the same class, no need to split further
- In practice, one first builds a large tree and then one prunes it back (to avoid overfitting)

- See *Foundations of Statistical Natural Language Processing*, Manning and Schuetze for a good introduction

what did we say was the **best split**?

what did we say was the **best split**?

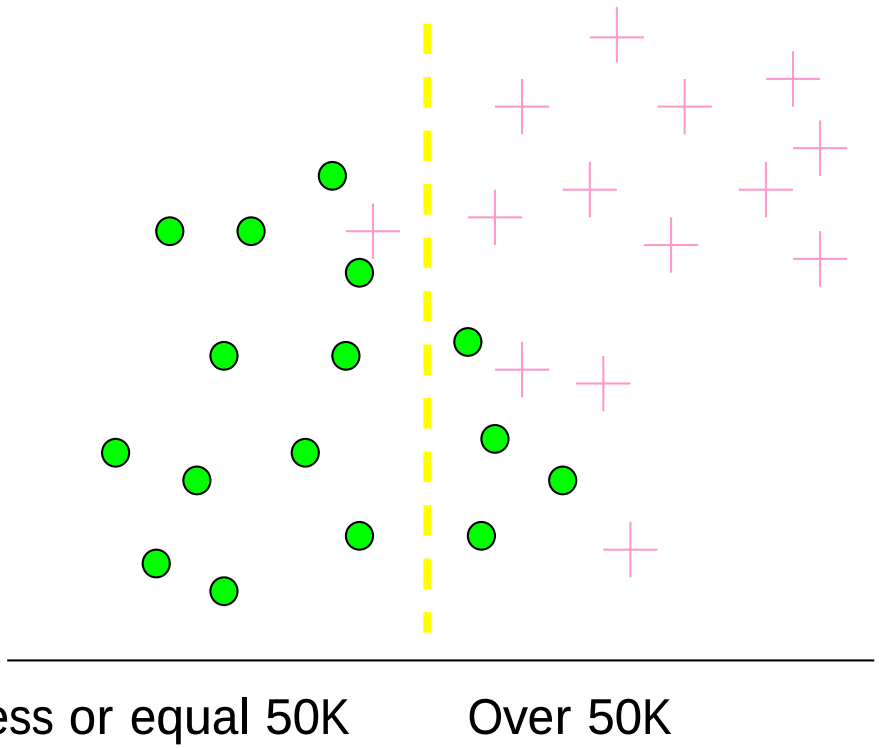
- Information Gain is the **mutual information** between input attribute A and target variable Y
- Information Gain is the **expected reduction in entropy** of target variable Y for data sample S, due to sorting on variable A

- (feature) "cheerful" is found in
  - 70% of POSITIVE tweets
  - 25% of NEGATIVE tweets
- (feature) "emotion" is found in
  - 40% of POSITIVE tweets
  - 30% of NEGATIVE tweets

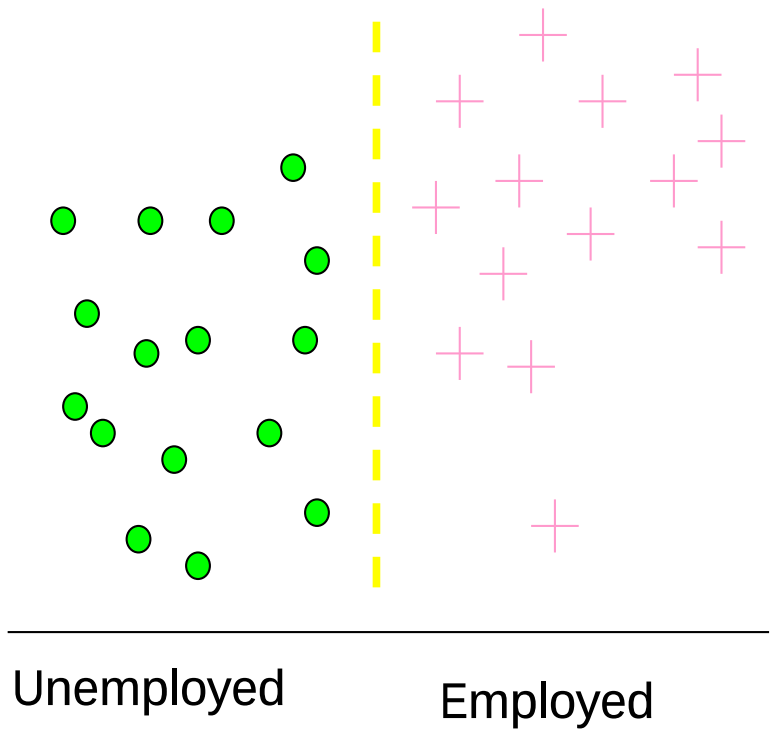
Information Gain

Which test is more informative?

Split over whether Balance exceeds 50K



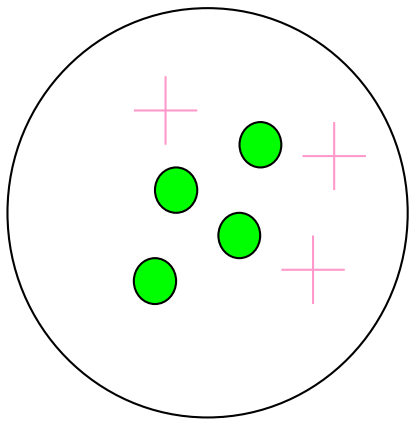
Split over whether applicant is employed

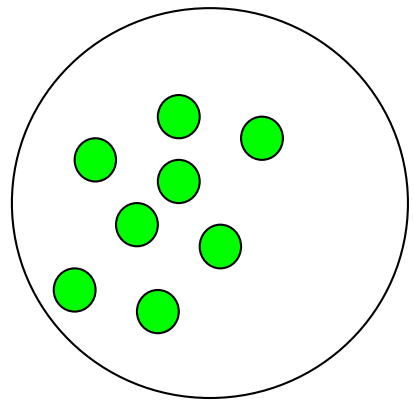


Information Gain

Impurity/Entropy (informal)

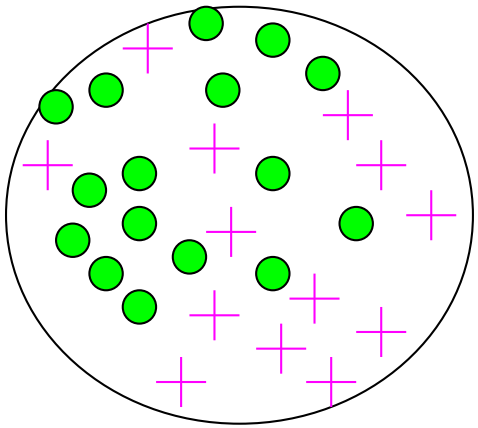
- Measures the level of **impurity** in a group of examples



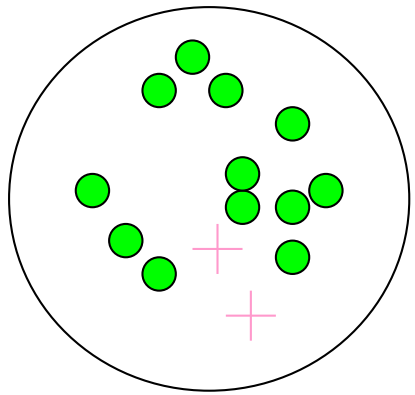


Impurity

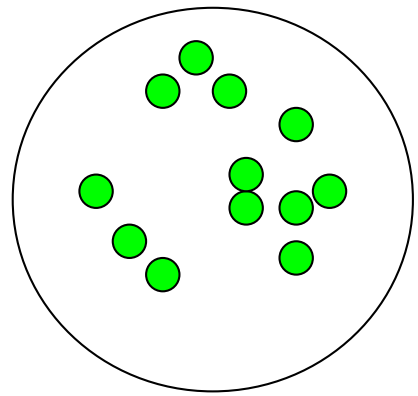
Very impure group



Less impure

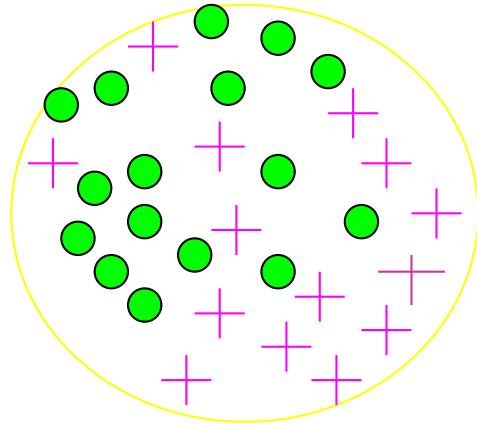


Minimum impurity



Entropy: a common way to measure impurity

- Entropy = $\sum_i - p_i \log_2 p_i$



p_i is the probability of class i

Compute it as the proportion of class i in the set.

16/30 are green circles; 14/30 are pink crosses
$\log_2(16/30) = -.9$; $\log_2(14/30) = -1.1$
Entropy = -(16/30)(-.9) –(14/30)(-1.1) = .99

- Entropy comes from information theory. The higher the entropy the more the information content.

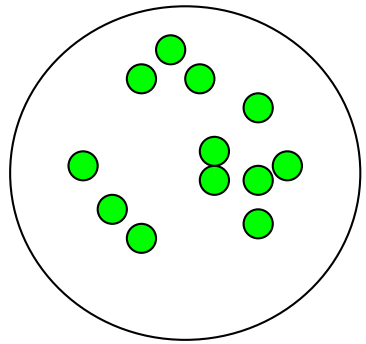
What does that mean for learning from examples?

2-Class Cases:

- What is the entropy of a group in which all examples belong to the same class?
  - entropy = $- 1 \log_2 1 = 0$

  not a good training set for learning

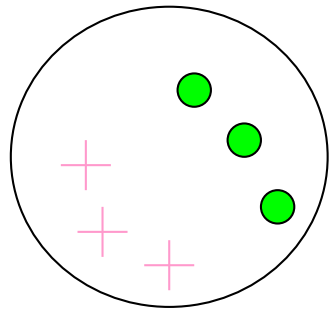
Minimum impurity



- What is the entropy of a group with 50% in either class?
  - entropy = $-0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

  good training set for learning

Maximum impurity



Information Gain

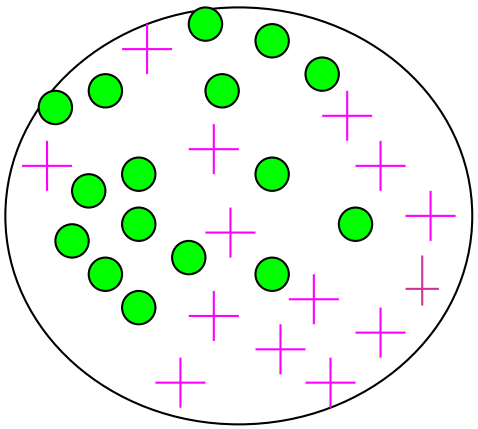
- We want to determine which attribute in a given set of training feature vectors is most useful for discriminating between the classes to be learned.
- Information gain tells us how important a given attribute of the feature vectors is.
- We will use it to decide the ordering of attributes in the nodes of a decision tree.

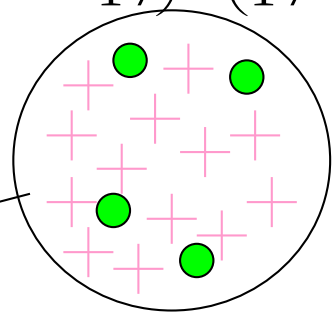
Calculating Information Gain

Information Gain = entropy(parent) – [average entropy(children)]

child entropy $-\left(\frac{13}{17}\cdot\log_2\frac{13}{17}\right)-\left(\frac{4}{17}\cdot\log_2\frac{4}{17}\right)=0.787$

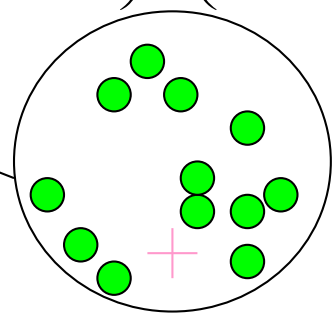
Entire population (30 instances)





17 instances

child entropy $-\left(\frac{1}{13}\cdot\log_2\frac{1}{13}\right)-\left(\frac{12}{13}\cdot\log_2\frac{12}{13}\right)=0.391$



13 instances

parent entropy $-\left(\frac{14}{30}\cdot\log_2\frac{14}{30}\right)-\left(\frac{16}{30}\cdot\log_2\frac{16}{30}\right)=0.996$

(Weighted) Average Entropy of Children = $\left(\frac{17}{30}\cdot 0.787\right)+\left(\frac{13}{30}\cdot 0.391\right)=0.615$

Information Gain= 0.996 - 0.615 = 0.38 for this split

Simple Example

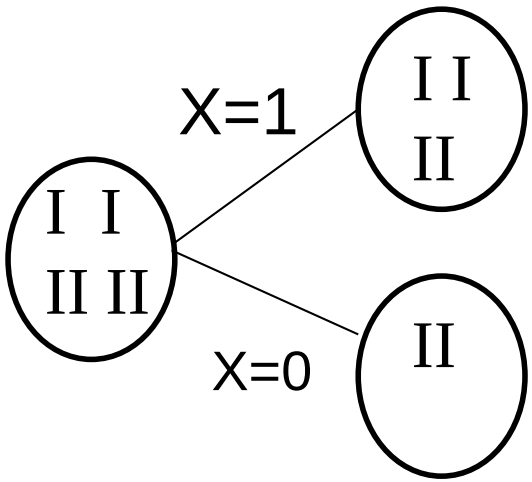
Training Set: 3 features and 2 classes

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

How would you distinguish class I from class II?

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Split on attribute X



If X is the best attribute,
this node would be further split.

$E_{child1} = -(1/3)\log_2(1/3)-(2/3)\log_2(2/3)$
$= .5284 + .39$
$= .9184$

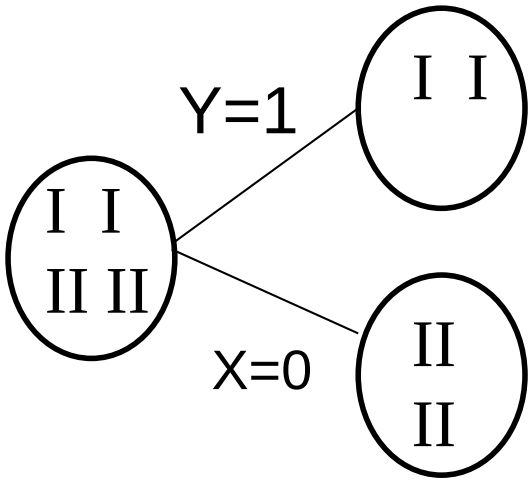
$E_{child2} = 0$

$E_{parent} = 1$

GAIN = 1 – (3/4)(.9184) – (1/4)(0) = .3112

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Split on attribute Y



$E_{child1} = 0$

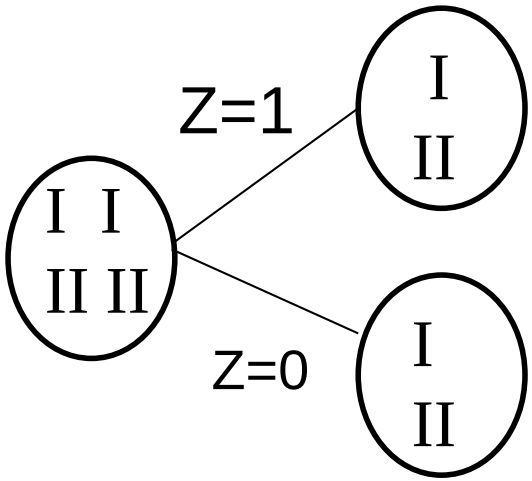
$E_{child2} = 0$

$E_{parent} = 1$

GAIN = 1 –(1/2) 0 – (1/2)0 = 1; BEST ONE

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Split on attribute Z



$E_{child1} = 1$

$E_{child2} = 1$

$E_{parent} = 1$

GAIN = 1 – (1/2)(1) – (1/2)(1) = 0 ie. NO GAIN; WORST

Building Decision Trees

- Once we have a decision tree, it is straightforward to use it to assign labels to new input values.
- How we can build a decision tree that models a given training set?

Pruning

- grow decision tree to its entirety
- trim the nodes of the decision tree in a bottom-up fashion
- if generalisation error improves after trimming, replace sub-tree by a leaf node
- class label of leaf node is determined from majority class of instances in the sub-tree

Pruning

- grow decision tree to its entirety
- trim the nodes of the decision tree in a bottom-up fashion
- if generalisation error improves after trimming, replace sub-tree by a leaf node
- class label of leaf node is determined from majority class of instances in the sub-tree

For the experiments you can manipulate the following parameters:

- maximum number of leaves
- minimum number of samples per leaf
- features (characters)

Practice!

choose the best settings for a decision tree model in a language identification task

(let's see a tree first)

Practice with Decision Trees

options to play with:

- −−max-nodes N (only for dt algorithm, default = NONE)
- −−min-samples N (only for dt algorithm, default = 1)
- −−nchars N

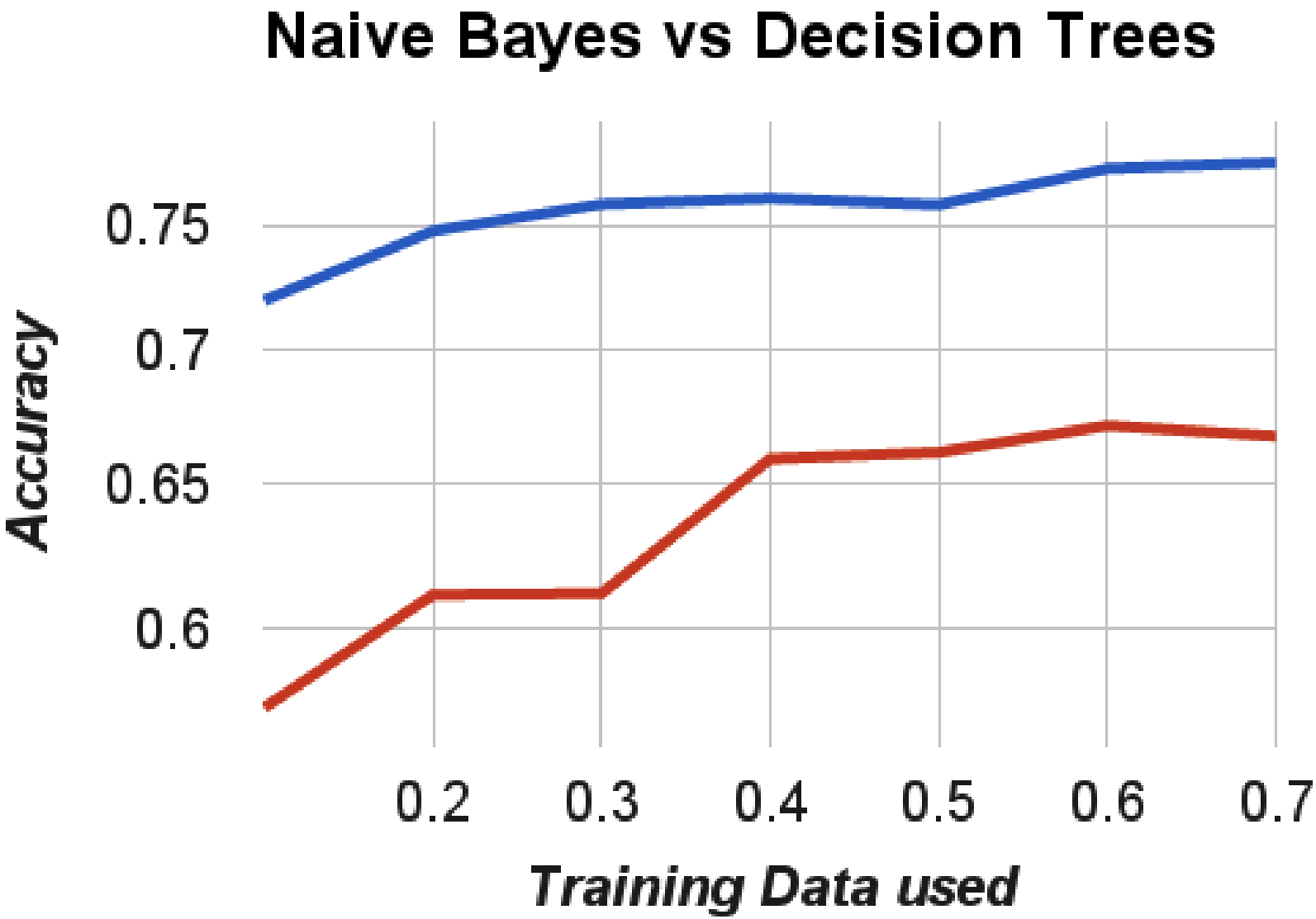
visualisation options:

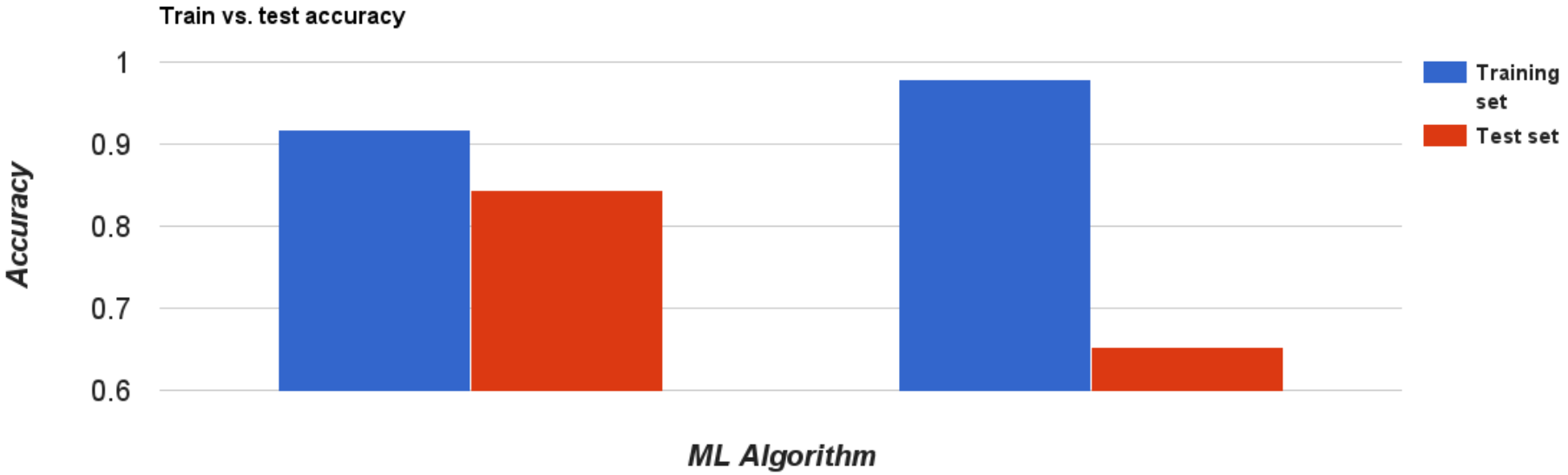
- −−cm (print confusion matrix + classification report)
- −−plot (shows CM)

example run:

```
python run_experiment.py --csv data/langident.csv --nchars
1 --algorithms dt --cm
```

Reflections and Concepts


Naive Bayes vs Decision Trees
Accuracy
0.75
0.7
0.65
0.6
0.2
0.3
0.4
0.5
0.6
0.7
Training Data used


Train vs. test accuracy
1
0.9
0.8
0.7
0.6
Accuracy
ML Algorithm
Training set
Test set