Decision Trees

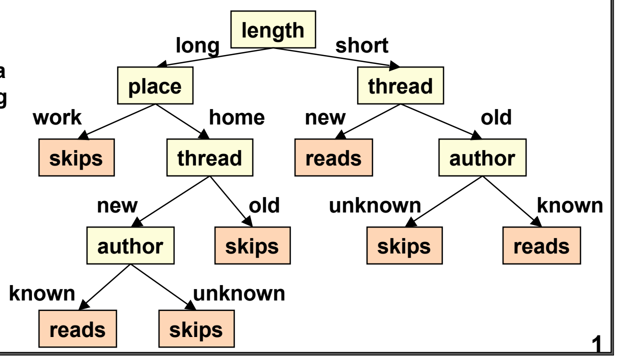
Decision trees are a popular method for classifying objects by means of a sequence of tests of feature values following a tree structure.

A decision tree is a tree where:
- the non-leaf nodes are labeled with attributes,
- all arcs out of a node labeled with attribute A are labelled with each of the possible values of the attribute A,
- the leafs of the tree are labeled with classifications

Example :

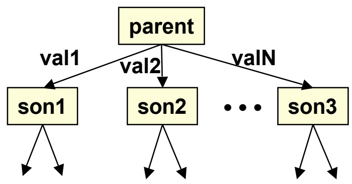
Decision tree for classifying a person as reading or skipping a book based on several attributes:

- author known or unknown
- thread new or old
- length long or short
- place at home or at work



Representing Decision Trees (1)

Decision trees can be represented by a nested IF-THEN-ELSE structure:



IF succ(parent) = { }
THEN return(parent) ELSE

IF value(parent, object) = val1
THEN <IF-THEN-ELSE structure for subtree1> ELSE

IF value(parent, object) = val2
THEN <IF-THEN-ELSE structure for subtree2> ELSE
• • •
IF value(parent, object) = valN
THEN <IF-THEN-ELSE structure for subtreeN> ELSE

Representing Decision Trees (2)

Decision trees can be represented by rules in a logic program.

Each leaf of the decision tree gives rise to a rule

leaf-value <- prop(Obj, $<att_1>$, $<val_1>$) $\wedge$ ... $\wedge$ prop(Obj, $<att_K>$, $<val_K>$)

where $<att_k>$ $<val_k>$ are all attribute-value pairs on the path from the leaf node to the root of the decision tree.

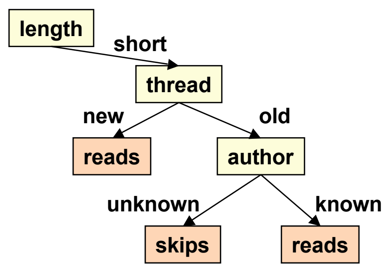
Example

The branch on the right gives rise to the rules:

reads <- prop(Obj, thread, new) $\wedge$
prop(Obj, length, short)

skips <- prop(Obj, author, unknown) $\wedge$
prop(Obj, thread, old) $\wedge$
prop(Obj, length, short)

reads <- prop(Obj, author, known) $\wedge$
prop(Obj, thread, old) $\wedge$
prop(Obj, length, short)



3

Learning Decision Trees

Decision trees can be learnt from examples.

Example	User Action	Author	Thread	Length	Where Read
e1	skips	known	new	long	home
e2	reads	unknown	new	short	work
e3	skips	unknown	old	long	work
e4	skips	known	old	long	home
e5	reads	known	new	short	home
e6	skips	known	old	long	work

Note that in the examples, the class is an attribute like other attributes. For learning a classifier from examples, one attribute has to be assigned the role of the goal attribute.

A decision tree can provide distinct leaf nodes for all combinations of attribute values.

➡ A correct decision exists for a given set of examples, if there are no examples which differ only in the goal attribute.

4

Learning Algorithm

Algorithm for learning a decision tree:

Given a set of examples, and a set of attributes and a goal attribute.

A Stop if all examples have the same classification.
Otherwise, choose an attribute to split on.

B For each value of this attribute, build a subtree for those examples with this attribute value and repeat A and B.

Note that the choice of an attribute in step A is not specified.

What attribute choices will give a "good" decision tree?

Quality measures for decision trees:

- Depth of tree
- Number of nodes
- Expected number of steps given a probability distribution of the attributes

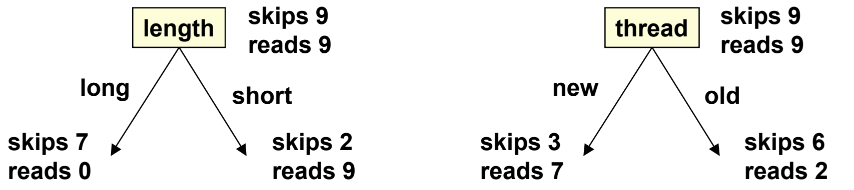
Extended Example Set

Example	User Action	Author	Thread	Length	Where Read
e1	skips	known	new	long	home
e2	reads	unknown	new	short	work
e3	skips	unknown	old	long	work
e4	skips	known	old	long	home
e5	reads	known	new	short	home
e6	skips	known	old	long	work
e7	skips	unknown	old	short	work
e8	reads	unknown	new	short	work
e9	skips	known	old	long	home
e10	skips	known	new	long	work
e11	skips	unknown	old	short	home
e12	skips	known	new	long	work
e13	reads	known	old	short	home
e14	reads	known	new	short	work
e15	reads	known	new	short	home
e16	reads	known	old	short	work
e17	reads	known	new	short	home
e18	reads	unknown	new	short	work

Effect of Attribute Choices

Each attribute effects a split of the example set according to the attribute values.

Example:
Splits effected by different choices of first attribute for the extended example set



Which attribute will give a shorter decision tree?

For a small decision tree, we want to choose an attribute where
- the example set is split into subsets as evenly as possible
- the classification within each subset is distributed as unevenly as possible

7

Maximizing Information Gain

Select attributes in the order of maximal information gain.

$H(G)$	entropy of source regarding goal attribute G with distribution according to example set
$H(G\|A=a_i)$	entropy of same source based on subset of examples where attribute A has value a_i
q_i	fraction of example set where attribute A has value a_i
IG	information gain by asking for the attribute value of A

$$\boxed{IG = H(G) - \sum_i q_i H(G \mid A = a_i)}$$

8

Splitting with Maximal Information Gain

Consider extended example set and empirical probability distributions.

H(G) = 1 — 9 skips and 9 reads examples

Test on author
author = known: [e1, e4, e5, e6, e9, e10, e12, e13, e14, e15, e16, e17]
author = unknown: [e2, e3, e7, e8, e11, e18]
IG = H(G) - 12/18 H(G|author = known) - 6/18 H(G|author = unknown) = 0

Test on thread
thread = new: [e1, e2, e5, e8, e10, e12, e14, e15, e17, e18]
thread = old: [e3, e4, e6, e7, e9, e11, e13, e16]
H(G|thread=new) = $- 3/10 \log_2 3/10 - 7/10 \log_2 7/10 = 0{,}881$
H(G|thread=old) = $- 6/8 \log_2 6/8 - 2/8 \log_2 2/8 = 0{,}811$
IG = H(G) - 10/18 H(G|thread=new) - 8/18 H(G|thread=old) = 0,150

Test on length
length = long: [e1, e3, e4, e6, e9, e10, e12]
length = short: [e2, e5, e7, e8, e11, e13, e14, e15, e16, e17, e18]
H(G|length=long) = 0
H(G|length=short) = $- 2/11 \log_2 2/11 - 9/11 \log_2 9/11 = 0{,}684$
IG = H(G) - 7/18 H(G|length=long) - 11/18 H(G|length=short) = 0,582

best choice!

9

Classification Errors

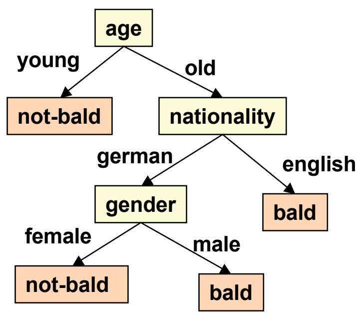
Decision trees may obtain zero error on a set of training examples, but may misclassify examples not contained in the training set.

Training error: probability of error for training examples
True error: probability of error for unrestricted examples

Overfitting of a decision tree to training data is an important source for errors.

Example:

male	young	german	not-bald
female	old	german	not-bald
male	old	english	bald
male	old	german	bald



Unfortunately, old english females will be classified as bald.

Overfitting is due to insufficient generalization of examples.

10

Inconsistent Example Sets

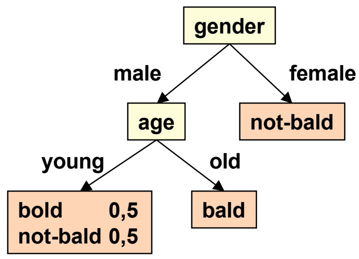
An example set is inconsistent if it contains examples which differ only by the value of the goal attribute.

Reasons for inconsistency:

- too few or irrelevant attributes
- noisy data

Decision trees may reflect inconsistent examples by assigning probability values to conflicting outcomes at leaf nodes.

e1	male	young	not-bald
e2	female	old	not-bald
e3	male	old	bald
e4	female	young	not-bald
e5	male	young	bald

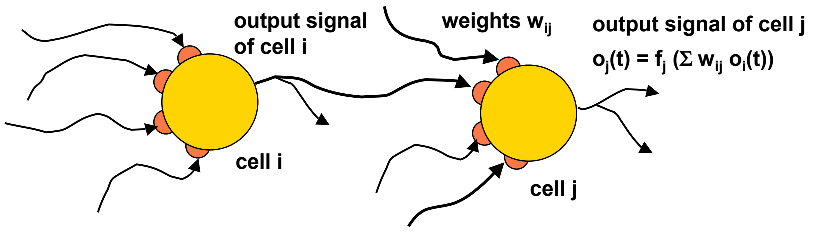


Summary of Decision Trees

- Decision trees represent classifiers which are easy to understand and to implement.
- A decision tree can represent any discrete function over an N-dimensional space of discrete-valued attributes.
- A consistent example set allows a decision with zero training error.
- A decision tree can be learnt from examples. There may be many possible decision trees for an example set.
- A good heuristic for obtaining a small decision tree is to select attributes in the order of maximal information gain.
- If the example set is small while the number of attributes is large, the classifier may cause a large true error due to overfitting.

Classification with Artificial Neural Networks

Artificial Neural Networks (NN) are composed of units which mimick the behaviour of natural neural networks.



- The output of each unit is a function f of the weighted sum of input signals
- Weights can be learnt from examples
- NNs can approximate any function

13

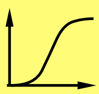
Multilayer Feed-forward Nets

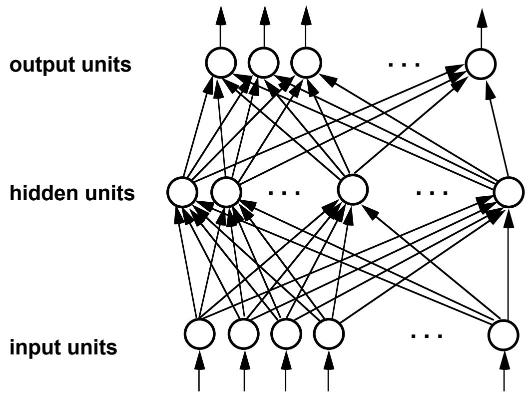
Classifiers are most frequently realized by multilayer feed-forward networks

Example:

3-layer net

- each unit of a layer is connected to each unit of the layer below
- units within a layer are not connected
- activation function f is differentiable (for learning)

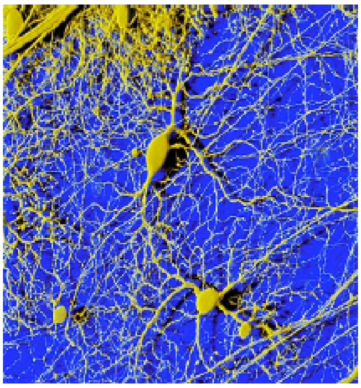




14

Natural Neural Networks

- ca. 10^{11} neurons in human brain
- ca. 10^{4} inputs for each neuron (average in humans)
- Spiked output
- Complex dynamical behaviour (e.g. cells fatigue)
- Various types of activation functions
- Several different cell types (e.g. multiplicative behaviour)
- Learning by mutual reinforcement



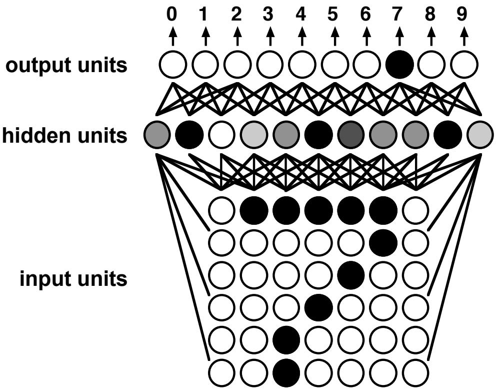
15

Example: Character Recognition with a Neural Net

Schematic drawing shows 3-layer feed-forward net:

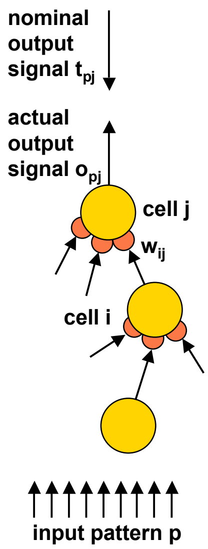
- input units are activated by sensors and feed hidden units
- hidden units feed output units
- each unit receives weighted sum of incoming signals (weights not shown)

How can a large number of weights be adjusted to achieve character recognition?



16

Learning by Backpropagation



Supervised learning procedure:
- **present example and determine output error signals**
- **adjust weights which contribute to errors**

<u>Adjusting weights</u>:

- **Error signal of output cell j for pattern p is**

  $\delta_{pj} = (t_{pj} - o_{pj})\, f_j'(net_{pj})$

  $f_j'()$ is the derivative of the activation function f()

- **Determine error signal δ_{pi} for internal cell i recursively from error signals of all cells k to which cell i contributes.**

  $\delta_{pi} = f_i'(net_{pi})\, \Sigma_k\, \delta_{pk} w_{ik}$

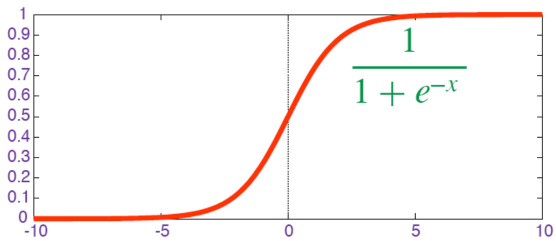
- **Modify all weights: $\Delta_p w_{ij} = \eta \delta_{pj} o_{pi}$ η is a positive constant**

The procedure must be repeated many times until the weights are "optimally" adjusted. There is no general convergence guarantee.

17

Activation Functions

An activation function must be differentiable for Backpropagation.
Typical activation function ("sigmoid"):



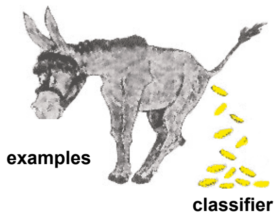
$$f(x) = \frac{1}{1+e^{-x}} \qquad f'(x) = f(x)(1-f(x))$$

18

Typical Applications for Neural Networks

NNs are cash cows (Goldesel) for engineers:

Feed examples and obtain classifier!



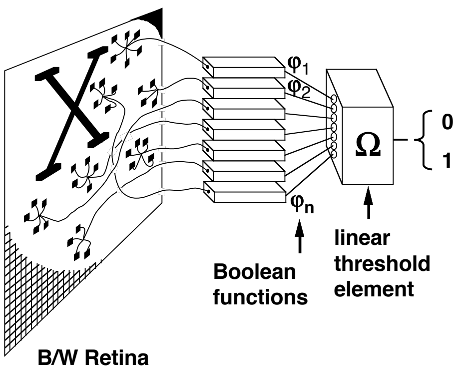
Useful primarily for applications which are difficult to analyze for humans:

- Speech recognition, e.g. determining the identity of a speaker
- Lipreading
- Image understanding, e.g. classifying x-rayed luggage as suspicious
- Event recognition, e.g. dangerous patterns in air traffic
- Predict which job an applicant is best suited for
- Diagnose diseases

Perceptrons (1)

A perceptron is a simple computational model (similar to NN) for combining local Boolean operations.

Investigation by Minsky and Papert (Perceptrons, 1969) showed that there are classification tasks which cannot be accomplished.



φ_i Boolean functions with local support in the retina:
- limited diameter
- limited number of cells

output is 0 or 1

Ω compares weighted sum of the φ_i with fixed threshold θ:

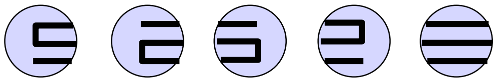
$$\Omega = \begin{cases} 1 & \text{if } \Sigma\, w_i \varphi_i > \theta \\ 0 & \text{otherwise} \end{cases}$$

Perceptrons (2)

A limited-diameter perceptron cannot determine connectedness

Assume perceptron with maximal diameter d for the support of each j_i.
Consider 4 shapes as below with a < d and b >> d.

Boolean operators may distinguish 5 local situations:



φ_1 φ_2 φ_3 φ_4 φ_5

φ_5 is clearly irrelevant for distinguishing between the 2 connected and the 2 disconnected shapes

For Ω to exist, we must have:

$w_1\, \varphi_1 + w_4\, \varphi_4 < \theta$
$w_2\, \varphi_2 + w_3\, \varphi_3 < \theta$
$\Sigma\, w_i\, \varphi_i < 2\theta$

$w_2\, \varphi_2 + w_4\, \varphi_4 > \theta$
$w_1\, \varphi_1 + w_3\, \varphi_3 > \theta$
$\Sigma\, w_i\, \varphi_i > 2\theta$

$\Rightarrow$ contradiction, hence Ω cannot exist

Summary of Artificial Neural Networks

- Artificial neural networks (NNs) can approximate continuous-valued functions of multiple continous-valued input variables.
- NNs are attractive because they can be taught by examples. Backpropagation is the basic learning scheme.
- Learning may require thousands of examples.
- Too many hidden units for too few examples may cause overfitting and hence bad performance.
- Powerful NNs may require several layers of hidden units. It is difficult to interpret the meaning of hidden units after learning.
- It is difficult to judge the capabilities and weaknesses of a NN except by testing examples.