Fundamental Theories and Applications of Neural Networks

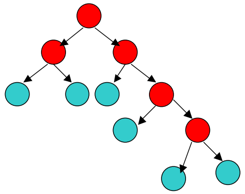
Neural Network Trees

Content of this lecture

- What is a decision tree?
- How to induce a decision tree?
- What are the drawbacks of decision trees?
- What is an NNTree?
- What are the advantages of NNTrees ?
- How to induce NNTrees efficiently?

What is a decision tree?

- Decision tree (DT) is a tree structure.
- In a DT, there are two kinds of nodes
  - Internal nodes: Used to make local decisions based on the local information they possess.
  - Terminal nodes: Used to make the final decision.



Local decision making in a DT

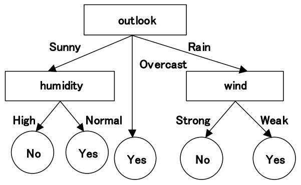
- In each internal node, a decision function $f(x)$ is used to make local decisions.
- For binary decision trees, an input pattern is classified to the left child node if $f(\mathbf{x})<0$; and to the right child node otherwise.
- The decision function in a standard DT is $f(\mathbf{x})= x_i-a_{i.}$
- That is, only one of the features is used for making the decisions.

Final decision making in a DT

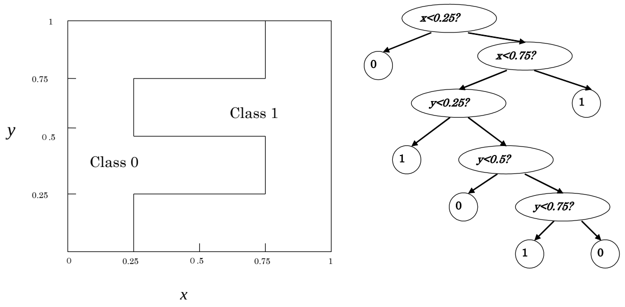
- In each terminal node, the distribution of data assigned to the node by the tree is used for making the final decisions.
- If the majority of data belong to the *i*th class, the terminal node is assigned with the class label *i*.
- All test data assigned to this node by the tree are assigned with the label *i*.

Example 1: Shall I play tennis today ?

- Play tennis if (outlook is sunny & humidity is normal).
- Play tennis if (outlook is overcast).
- Play tennis if (outlook is rain & wind is weak).
- Otherwise not play.



Example 2: A binary decision tree



How to induce (design) decision trees?

- One of the most popular tools for inducing DT is C4.5.
- C4.5 was proposed by Quinlan.
- C4.5 is often used as a standard for comparison with newly developed algorithms.
- The source code of C4.5 can be down loaded from the following web site:
  - http://www.rulequest.com/Personal/
- There are many other programs for inducing DTs.

Basic steps for inducing a DT

Step 1: If all data assigned to the current node belong to the same class, the current node is terminal. Define the class label and return;

Step 2: Otherwise, find the best decision function *f*(*x*) based on some criterion (Information Gain Ratio is used in C4.5);

Step 3: Split the data assigned to the current node to *N*-groups using *f*(*x*);

Step 4: For each group, if it is not empty, make a new node, set the current node as this new node, and call the same sub-routine recursively.

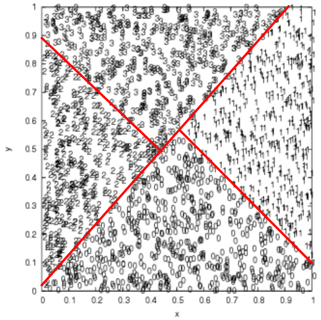
Pros and cons of DTs

- Pros:
  - Comprehensible.
  - Easy to design.
  - Easy to implement.
  - Good for structural learning.
- Cons
  - May become very large for complex problems.
  - Difficult to know the true concept.
  - Too many rules to be understood by human users.

Why DTs become large ?

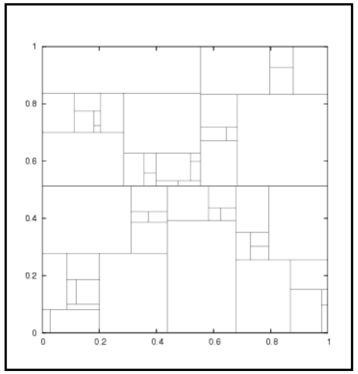
- The decision boundary corresponding to $\boldsymbol{f}(\boldsymbol{x})= \boldsymbol{x}_i - \boldsymbol{a}_i$ is an axis-parallel hyperplane.
- The main reason that standard DTs become every large is that only axis-parallel hyperplanes are used.
- Standard DTs are also called axis-parallel decision trees (APDTs).
- For complex problems, many hyperplanes are required.

A Simple Example



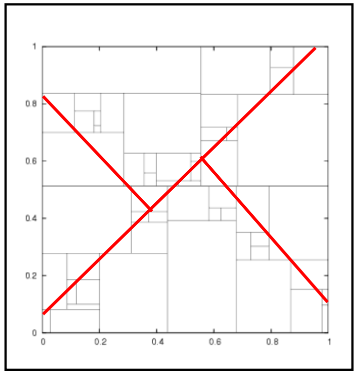
- 2,000 points plotted at random in the square $[0, 1]^2$
- Theoretic decision boundaries:
  - L_1: y = 1.1 x
  - L_2: y = - 0.91 x + 1.0
  - L_3: y = - 0.91 x + 0.91

APDT for the Simple Example



What are the concepts hidden in the decision boundaries?

APDT for the Simple Example



What are the concepts hidden in the decision boundaries?

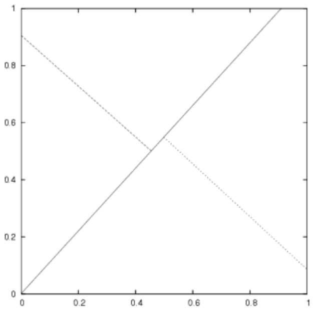
The oblique decision tree

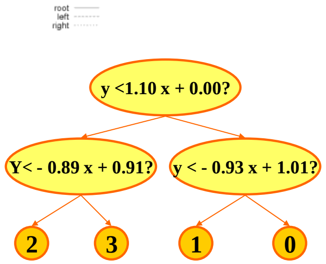
- One way to reduce the tree size is to use multivariate decision functions.
- **Oblique decision tree** (ODT) is the simplest MDT.
  - Linear combination of features is used as the decision function

$$F(x) = \sum_{i=1}^{d} w_i x_i$$

  - If $F(\mathbf{x})<0$, visit the left child; otherwise, visit the right child.

An ODT for the Simple Example





y <1.10 x + 0.00?

Y< - 0.89 x + 0.91?

y < - 0.93 x + 1.01?

2

3

1

0

Pros and cons of MDT

- Pros
  - A multivariate decision function is more powerful, and thus less nodes or less steps are required for making a decision.
  - Better generalization ability can be expected.
- Cons
  - In general, the MDT are not comprehensible.
  - To get the best decision function is very difficult.

How to make MDT comprehensible?

- One way to make the MDT more comprehensible is to use a small neural network (NN) in each internal node as the decision function.
- This is a hybrid learning model that combines DT and NN
- We call this model the neural network decision tree (NNTree).
- One way to make NNTree comprehensible is to use each NN to learn some intermediate concept, and assign a label for each internal node.
- The tree can be transformed to a very compact rule set
- Note that each intermediate concept is still represented in a black-box.

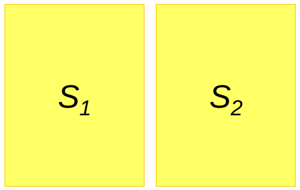
How to induce MDT more efficiently?

- Instead of generating many decision functions, we propose to generate only one decision function through supervised learning.
- The teacher signal g(x) of a data is called the group label.
- If g(x) =i, x is assigned to the i-th child of the current node.

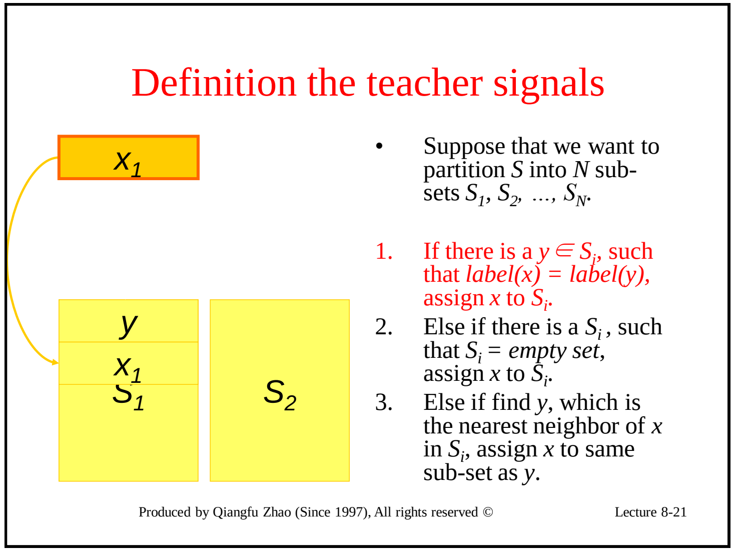
> ➢ **Put all data with the same class label to the same group**
> ➢ **Put data that are close to each other to the same group**

Definition the teacher signals

- Suppose that we want to partition S into N sub-sets $S_1, S_2, \ldots, S_N$.



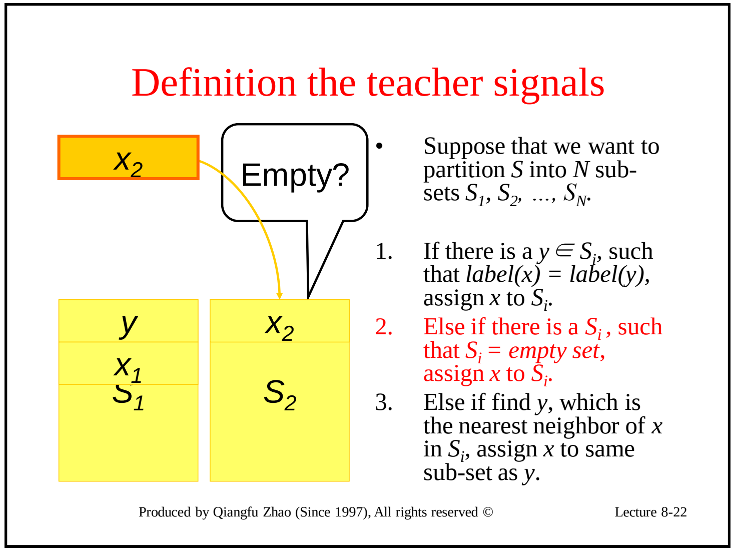
1. If there is a $y \in S_i$, such that $label(x) = label(y)$, assign x to S_i.
2. Else if there is a S_i, such that S_i = *empty set*, assign x to S_i.
3. Else if find y, which is the nearest neighbor of x in S_i, assign x to same sub-set as y.



Definition the teacher signals

- Suppose that we want to partition S into N sub-sets S_1, S_2, …, S_N.

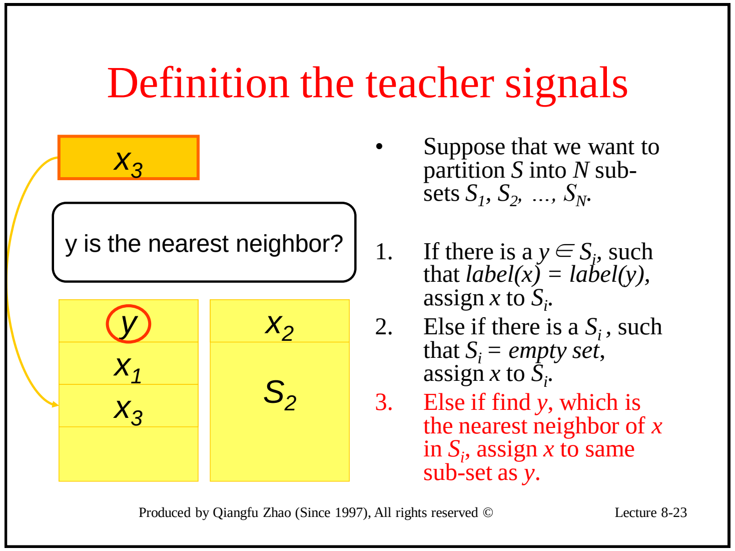
1. If there is a $y \in S_i$, such that *label(x) = label(y)*, assign x to S_i.
2. Else if there is a S_i , such that S_i = *empty set*, assign x to S_i.
3. Else if find y, which is the nearest neighbor of x in S_i, assign x to same sub-set as y.



Definition the teacher signals

- Suppose that we want to partition S into N sub-sets S_1, S_2, …, S_N.

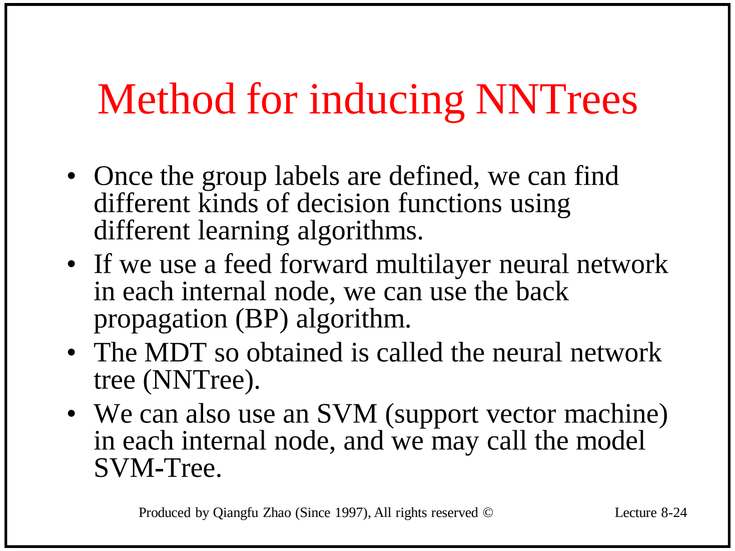
1. If there is a $y \in S_i$, such that *label(x) = label(y)*, assign x to S_i.
2. Else if there is a S_i , such that S_i = *empty set*, assign x to S_i.
3. Else if find y, which is the nearest neighbor of x in S_i, assign x to same sub-set as y.



Definition the teacher signals

- Suppose that we want to partition S into N sub-sets S_1, S_2, …, S_N.

1. If there is a $y \in S_i$, such that *label(x) = label(y)*, assign x to S_i.
2. Else if there is a S_i , such that S_i = *empty set*, assign x to S_i.
3. Else if find y, which is the nearest neighbor of x in S_i, assign x to same sub-set as y.



Method for inducing NNTrees

- Once the group labels are defined, we can find different kinds of decision functions using different learning algorithms.
- If we use a feed forward multilayer neural network in each internal node, we can use the back propagation (BP) algorithm.
- The MDT so obtained is called the neural network tree (NNTree).
- We can also use an SVM (support vector machine) in each internal node, and we may call the model SVM-Tree.

Advantages of NNTrees

- Compared with the APDT, experimental results show that NNTrees have better generalization ability.
- Compared with single model, fully connected neural networks, NNTrees are better for structural learning, and can make decisions more quickly.

Team Project VII

- Down load the program from the web site of this course, and try to understand the basic idea of the program.
- Down load 2 databases from the UC Irvine Machine Learning Repository, and try to induce an NNTree for this database.
- Solve the same problem using the BP based multilayer neural, and compare the results with those of the NNTree.