Информациски системи и големи податоци

Decision Tree

Introduction

- Decision tree learning is one of the most widely used techniques for classification.
  - Its classification accuracy is competitive with other methods, and
  - it is very efficient.
- The classification model is a tree, called **decision tree**.
- **C4.5** by Ross Quinlan is perhaps the best known algorithm

What symptom tells you most about the disease?

S1	S2	S3	D
y	n	n	y
n	y	y	y
n	y	n	n
n	n	n	n
y	y	n	y

A) S1
B) S2
C) S3

Why?

What symptom tells you most about the disease?

S1/D	y	n
y	2	0
n	1	2

S2/D	y	n
y	2	1
n	1	1

s3/D	y	n
y	1	0
n	2	2

Confusion matrix
(evaluation metric)

A) S1
B) S2
C) S3

Why?

S1	S2	S3	D
y	n	n	y
n	y	y	y
n	y	n	n
n	n	n	n
y	y	n	y

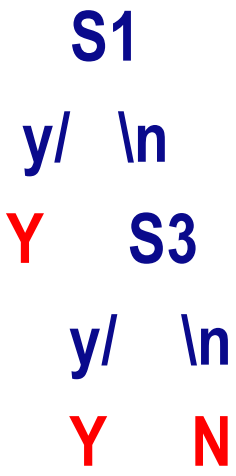
If you know S1=n, what symptom tells you most about the disease?

S1	S2	S3	D
y	n	n	y
n	y	y	y
n	y	n	n
n	n	n	n
y	y	n	y

A) S1
B) S2
C) S3

Why?

Resulting decision tree



The key question: what criterion to use do decide which question to ask?

Entropy and Information Gain

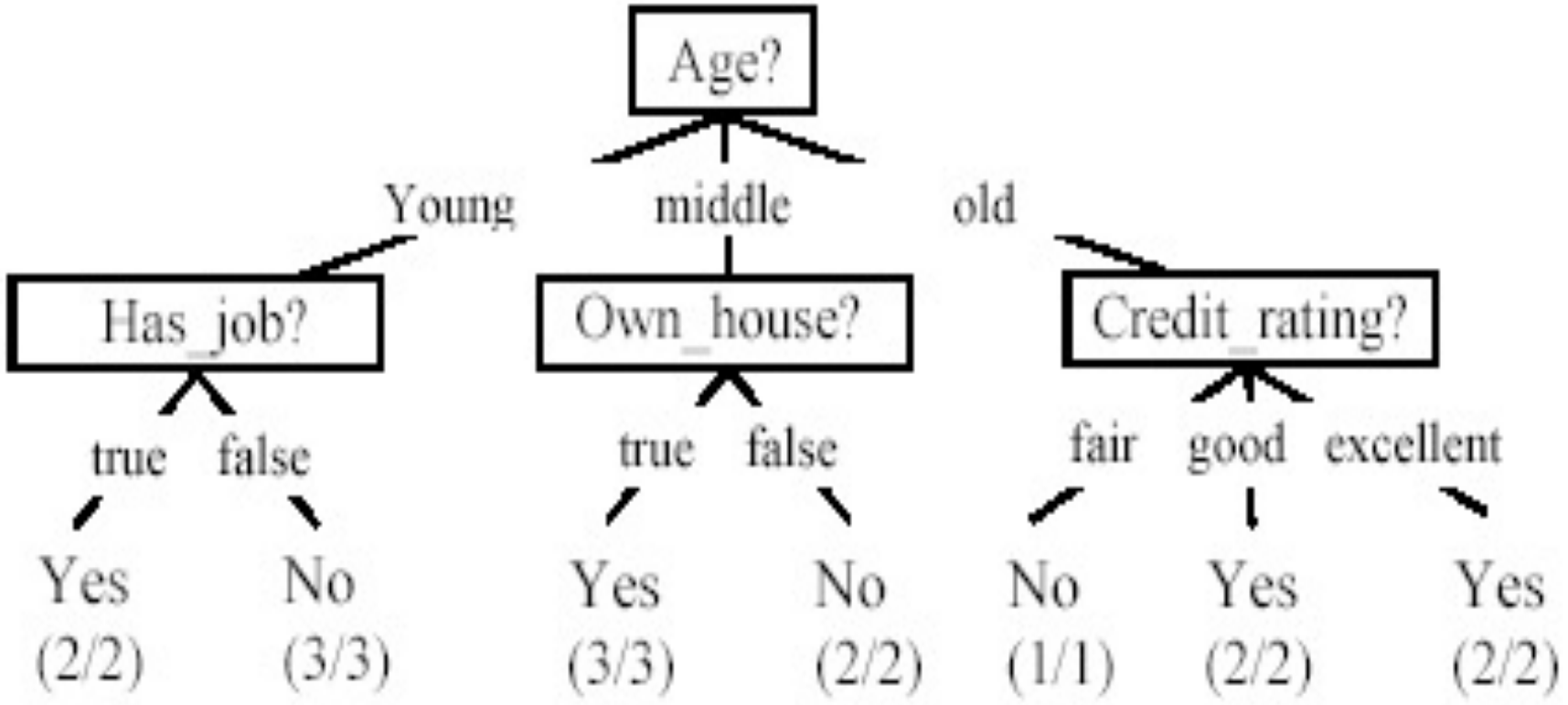
The loan data

Approved or not

ID	Age	Has_Job	Own_House	Credit_Rating	Class
1	young	false	false	fair	**No**
2	young	false	false	good	**No**
3	young	true	false	good	**Yes**
4	young	true	true	fair	**Yes**
5	young	false	false	fair	**No**
6	middle	false	false	fair	**No**
7	middle	false	false	good	**No**
8	middle	true	true	good	**Yes**
9	middle	false	true	excellent	**Yes**
10	middle	false	true	excellent	**Yes**
11	old	false	true	excellent	**Yes**
12	old	false	true	good	**Yes**
13	old	true	false	good	**Yes**
14	old	true	false	excellent	**Yes**
15	old	false	false	fair	**No**

A decision tree from the loan data

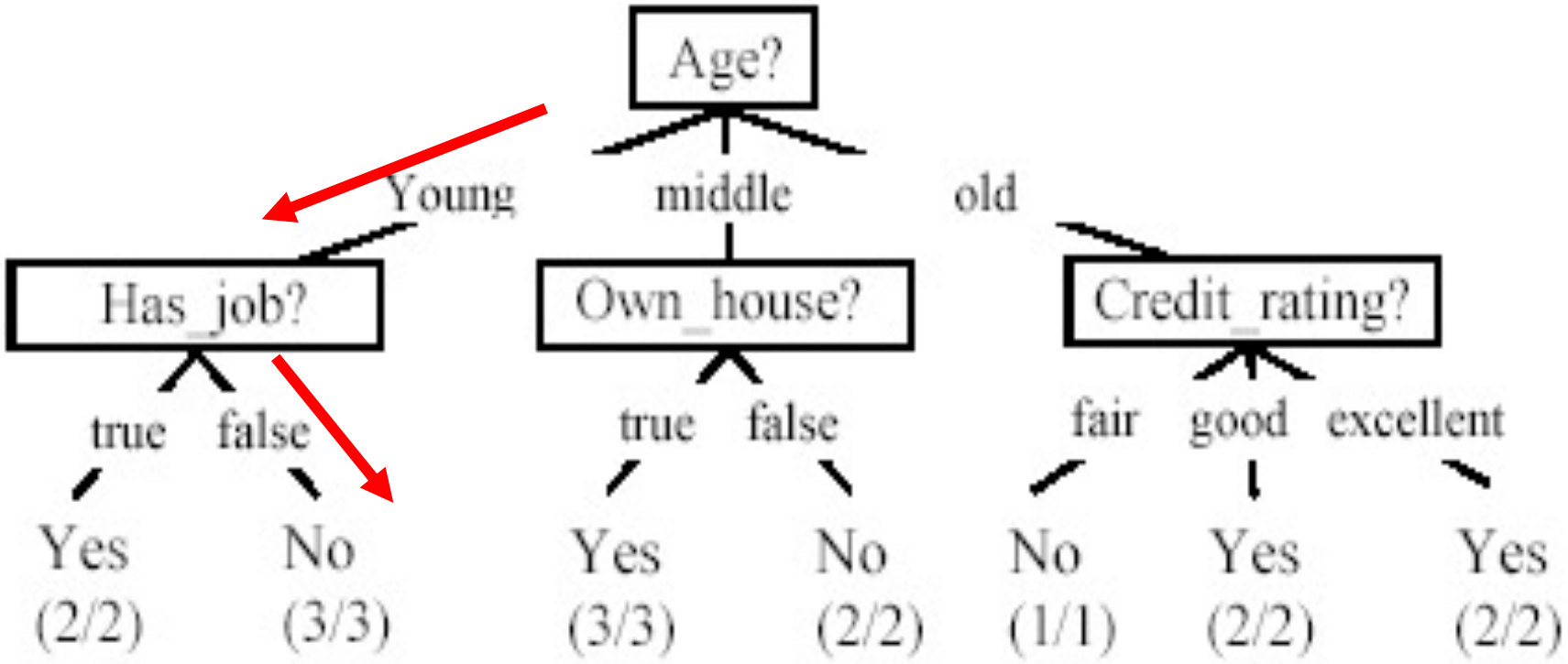
- Decision nodes and leaf nodes (classes)



Use the decision tree

Age	Has_Job	Own_house	Credit-Rating	Class
young	false	false	good	?

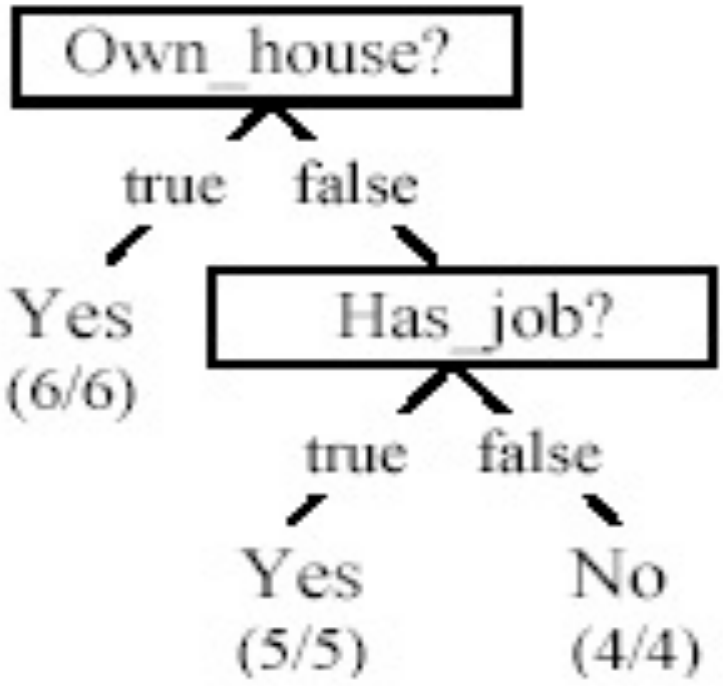
No



Is the decision tree unique?

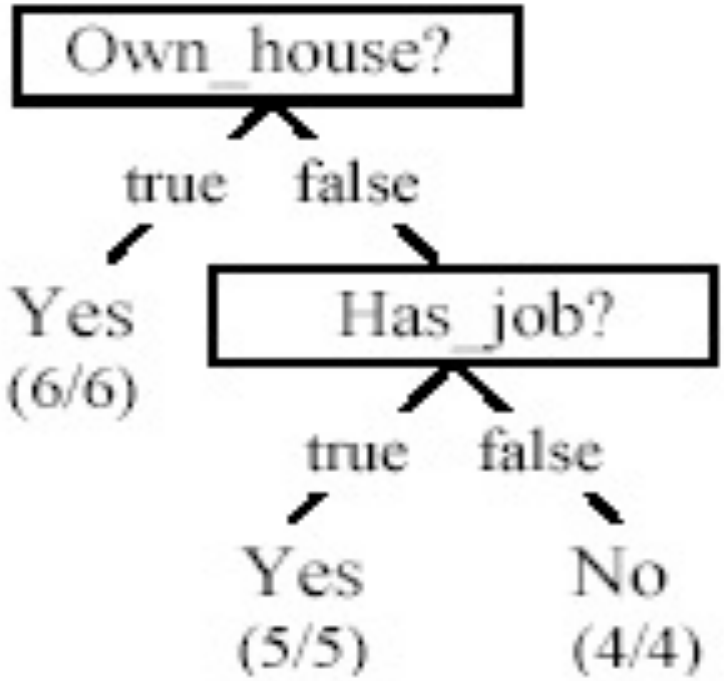
- No. Here is a simpler tree.
- We want smaller tree and accurate tree.
  - Easy to understand and perform better.

- Finding the best tree is NP-hard.
- All current tree building algorithms are heuristic algorithms



From a decision tree to a set of rules

- A decision tree can be converted to a set of rules
- Each path from the root to a leaf is a rule.



Own_house = true → Class =Yes [sup=6/15, conf=6/6]
Own_house = false, Has_job = true → Class = Yes [sup=5/15, conf=5/5]
Own_house = false, Has_job = false → Class = No [sup=4/15, conf=4/4]

Algorithm for decision tree learning

- Basic algorithm (a greedy **divide-and-conquer** algorithm)
  - Assume attributes are categorical now (continuous attributes can be handled too)
  - Tree is constructed in a top-down recursive manner
  - At start, all the training examples are at the root
  - Examples are partitioned recursively based on selected attributes
  - Attributes are selected on the basis of an impurity function (e.g., information gain)
- Conditions for stopping partitioning
  - All examples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority class is the leaf
  - There are no examples left

Decision tree learning algorithm

```
. Algorithm decisionTree(D, A, T)
1    if D contains only training examples of the same class c_j ∈ C then
2        make T a leaf node labeled with class c_j;
3    elseif A = ∅ then
4        make T a leaf node labeled with c_j, which is the most frequent class in D
5    else  // D contains examples belonging to a mixture of classes. We select a single
6          // attribute to partition D into subsets so that each subset is purer
7        p_0 = impurityEval-1(D);
8        for each attribute A_i ∈ {A_1, A_2, ..., A_k} do
9            p_i = impurityEval-2(A_i, D)
10       end
11       Select A_g ∈ {A_1, A_2, ..., A_k} that gives the biggest impurity reduction,
             computed using p_0 − p_i;
12       if p_0 − p_g < threshold then    // A_g does not significantly reduce impurity p_0
13         make T a leaf node labeled with c_j, the most frequent class in D.
14       else                             // A_g is able to reduce impurity p_0
15           Make T a decision node on A_g;
16           Let the possible values of A_g be v_1, v_2, ..., v_m. Partition D into m
                 disjoint subsets D_1, D_2, ..., D_m based on the m values of A_g.
17           for each D_j in {D_1, D_2, ..., D_m} do
18               if D_j ≠ ∅ then
19                   create a branch (edge) node T_j for v_j as a child node of T;
20                   decisionTree(D_j, A-{A_g}, T_j)// A_g is removed
21               end
22           end
23       end
24   end
```

Choose an attribute to partition data

- The *key* to building a decision tree - which attribute to choose in order to branch.
- The objective is to reduce impurity or uncertainty in data as much as possible.
  - A subset of data is pure if all instances belong to the same class.
- The *heuristic* in C4.5 is to choose the attribute with the maximum Information Gain or Gain Ratio based on information theory.

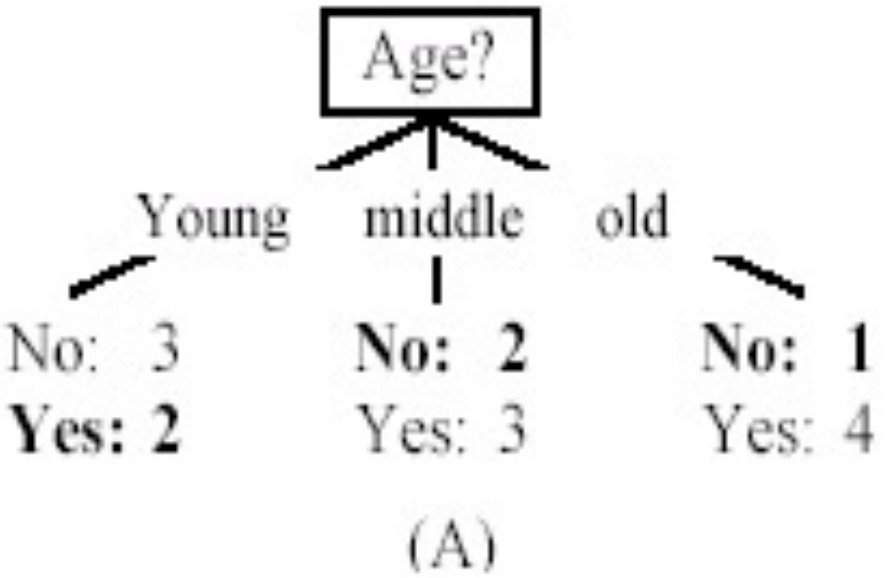
The loan data (reproduced)

Approved or not

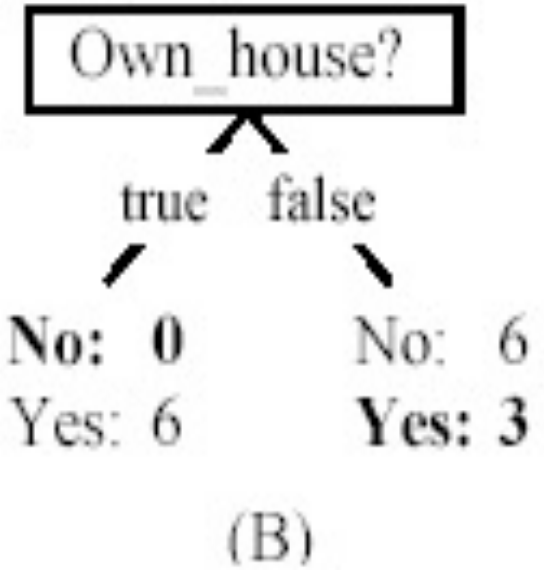
ID	Age	Has_Job	Own_House	Credit_Rating	Class
1	young	false	false	fair	No
2	young	false	false	good	No
3	young	true	false	good	Yes
4	young	true	true	fair	Yes
5	young	false	false	fair	No
6	middle	false	false	fair	No
7	middle	false	false	good	No
8	middle	true	true	good	Yes
9	middle	false	true	excellent	Yes
10	middle	false	true	excellent	Yes
11	old	false	true	excellent	Yes
12	old	false	true	good	Yes
13	old	true	false	good	Yes
14	old	true	false	excellent	Yes
15	old	false	false	fair	No

Two possible roots, which is better?

Bold are the errors



(A)



(B)

Information theory

- Information theory provides a mathematical basis for measuring the information content.
- To understand the notion of information, think about it as providing the answer to a question, for example, whether a coin will come up heads.
  - If one already has a good guess about the answer, then the actual answer is less informative.
  - If one already knows that the coin is rigged so that it will come with heads with probability 0.99, then a message (advanced information) about the actual outcome of a flip is worth less than it would be for a honest coin (50-50).

Information theory (cont …)

- For a fair (honest) coin, you have no information, and you are willing to pay more (say in terms of $) for advanced information - less you know, the more valuable the information.
- Information theory uses this same intuition, but instead of measuring the value for information in dollars, it measures information contents in **bits**.
- One bit of information is enough to answer a yes/no question about which one has no idea, such as the flip of a fair coin

Entropy in a nut-shell

Low Entropy

..the values (locations of soup) sampled entirely from within the soup bowl

High Entropy

..the values (locations of soup) unpredictable... almost uniformly sampled throughout our dining room

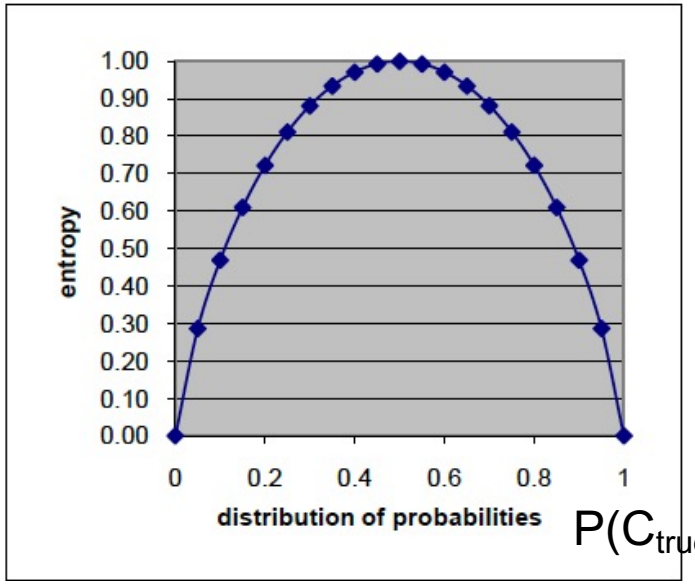
Information theory: Entropy measure

- The entropy formula,

$$entropy(D) = -\sum_{j=1}^{|C|} \Pr(c_j) \log_2 \Pr(c_j)$$

$$\sum_{j=1}^{|C|} \Pr(c_j) = 1,$$

- $\Pr(c_j)$ is the probability of class c_j in data set D
- We use entropy as a measure of impurity or disorder of data set D. (Or, a measure of information in a tree)



Binary

Entropy measure: let us get a feeling

1. The data set D has 50% positive examples (Pr(*positive*) = 0.5) and 50% negative examples (Pr(*negative*) = 0.5).

$$entropy(D) = -0.5 \times \log_2 0.5 - 0.5 \times \log_2 0.5 = 1$$

2. The data set D has 20% positive examples (Pr(*positive*) = 0.2) and 80% negative examples (Pr(*negative*) = 0.8).

$$entropy(D) = -0.2 \times \log_2 0.2 - 0.8 \times \log_2 0.8 = 0.722$$

3. The data set D has 100% positive examples (Pr(*positive*) = 1) and no negative examples, (Pr(*negative*) = 0).

$$entropy(D) = -1 \times \log_2 1 - 0 \times \log_2 0 = 0$$

- As the data become purer and purer, the entropy value becomes smaller and smaller. This is useful to us!

$$entropy(D) = -\sum_{j=1}^{|C|} \Pr(c_j) \log_2 \Pr(c_j)$$

Information gain

- Given a set of examples D, we first compute its entropy:

$$entropy(D) = -\sum_{j=1}^{|C|} \Pr(c_j) \log_2 \Pr(c_j)$$

- If we make attribute A_i, with v values, the root of the current tree, this will partition D into v subsets $D_1, D_2 \ldots, D_v$. The expected entropy if A_i is used as the current root:

$$entropy_{A_i}(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times entropy(D_j)$$

S1	S2	S3	D
y	n	n	y
n	y	y	y
n	y	n	n
n	n	n	n
y	y	n	y

Information gain (cont …)

- Information gained by selecting attribute A_i to branch or to partition the data is

$$gain(D, A_i) = entropy(D) - entropy_{A_i}(D)$$

- We choose the attribute with the highest gain to branch/split the current tree.

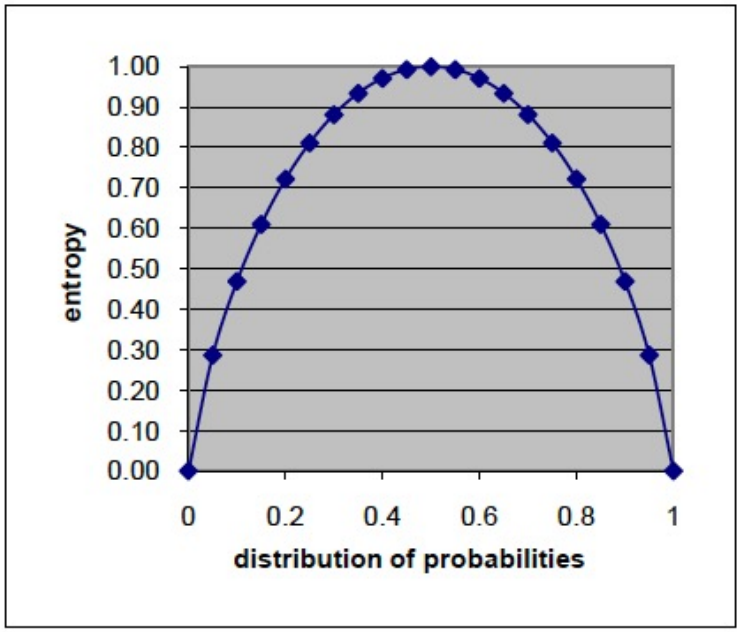
What is Information Gain used for?

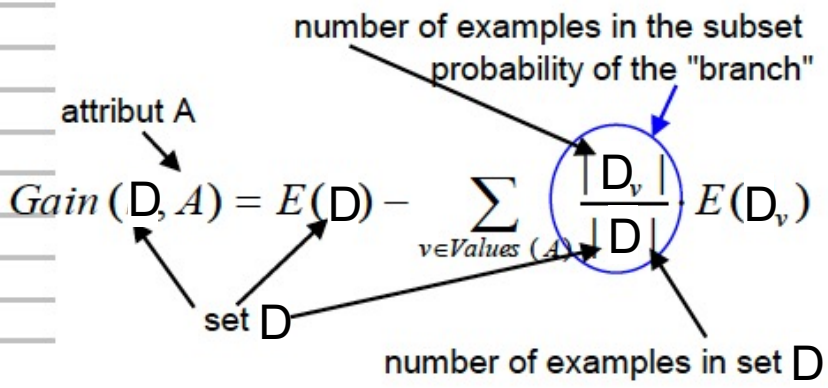
If you are going to collect information from someone (e.g. asking questions sequentially in a decision tree), the “best” question is the one with the highest information gain.

Information gain is useful for model selection

Entropy and Information Gain

probability of class 1	probability of class 2	entropy $E(p_1, p_2)$ =
p_1	$p_2 = 1-p_1$	$-p_1*\log_2(p_1) - p_2*\log_2(p_2)$
0	1	0.00
0.05	0.95	0.29
0.10	0.90	0.47
0.15	0.85	0.61
0.20	0.80	0.72
0.25	0.75	0.81
0.30	0.70	0.88
0.35	0.65	0.93
0.40	0.60	0.97
0.45	0.55	0.99
0.50	0.50	1.00
0.55	0.45	0.99
0.60	0.40	0.97
0.65	0.35	0.93
0.70	0.30	0.88
0.75	0.25	0.81
0.80	0.20	0.72
0.85	0.15	0.61
0.90	0.10	0.47
0.95	0.05	0.29
1	0	0.00





An example

$$entropy(D) = \frac{6}{15} \times \log_2 \frac{6}{15} + \frac{9}{15} \times \log_2 \frac{9}{15} = 0.971$$

ID	Age	Has_Job	Own_House	Credit_Rating	Class
1	young	false	false	fair	No
2	young	false	false	excellent	No
3	young	true	false	good	Yes
4	young	true	true	good	Yes
5	young	false	false	fair	No
6	middle	false	false	fair	No
7	middle	false	false	good	No
8	middle	true	true	good	Yes
9	middle	false	true	excellent	Yes
10	middle	false	true	excellent	Yes
11	old	false	true	excellent	Yes
12	old	false	true	good	Yes
13	old	true	false	good	Yes
14	old	true	false	excellent	Yes
15	old	false	false	fair	No

true

$$\frac{6}{6} \times log_2 \frac{6}{6} + \frac{0}{0} \times log_2 \frac{0}{0}$$

false

$$\frac{6}{9} \times log_2 \frac{6}{9} + \frac{3}{9} \times log_2 \frac{3}{9}$$

$$entropy_{Own\_house}(D) = \frac{6}{15} \times entropy(D_1) + \frac{9}{15} \times entropy(D_2)$$
$$= \frac{6}{15} \times 0 + \frac{9}{15} \times 0.918$$
$$= 0.551$$

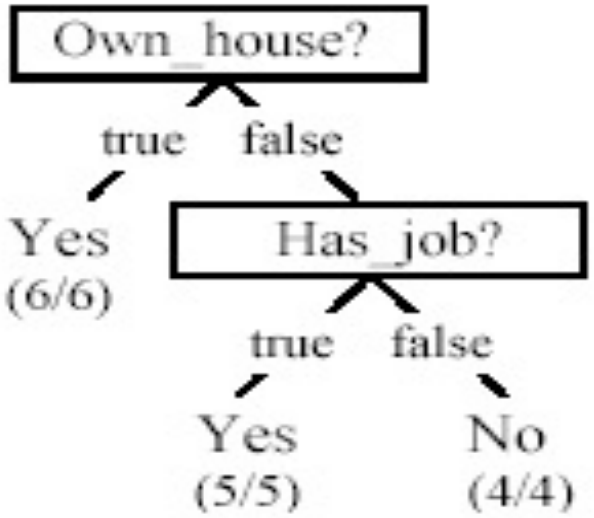
Age	Yes	No	entropy(Di)
young	2	3	0.971
middle	3	2	0.971
old	4	1	0.722

$$entropy_{Age}(D) = \frac{5}{15} \times entropy(D_1) + \frac{5}{15} \times entropy(D_2) + \frac{5}{15} \times entropy(D_3)$$
$$= \frac{5}{15} \times 0.971 + \frac{5}{15} \times 0.971 + \frac{5}{15} \times 0.722$$
$$= 0.888$$

- Own_house is the best choice for the root.

$$gain(D, \text{Age}) = 0.971 - 0.888 = 0.083$$
$$gain(D, \text{Own\_house}) = 0.971 - 0.551 = 0.420$$
$$gain(D, \text{Has\_Job}) = 0.971 - 0.647 = 0.324$$
$$gain(D, \text{Credit\_Rating}) = 0.971 - 0.608 = 0.363$$

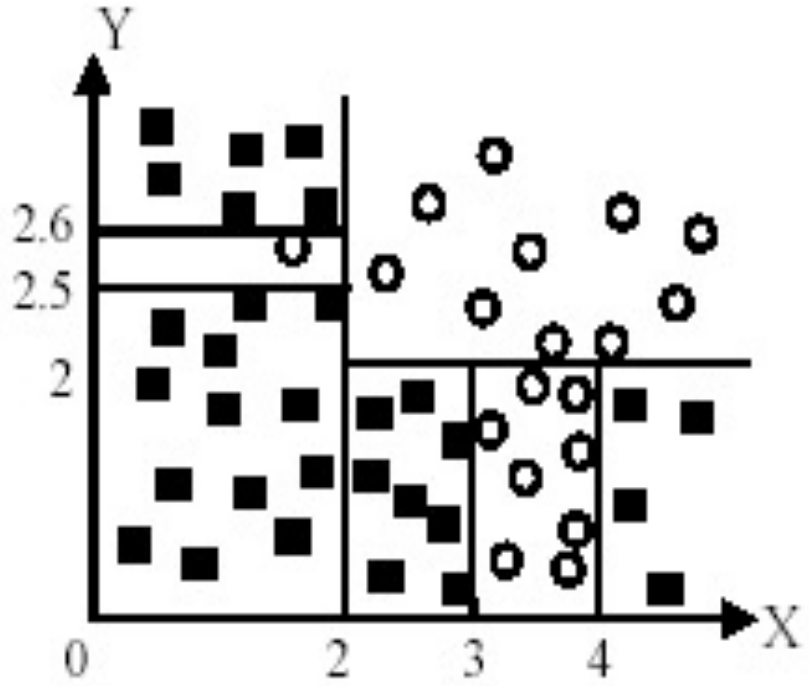
We build the final tree



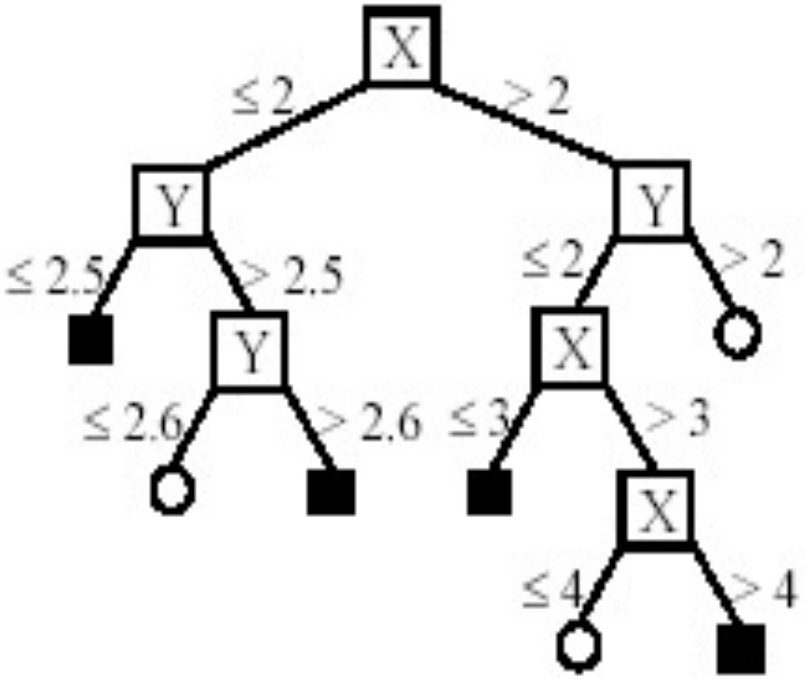
Handling continuous attributes

- Handle continuous attribute by splitting into two intervals (can be more) at each node.
- How to find the best threshold to divide?
  - Use information gain or gain ratio again
  - Sort all the values of an continuous attribute in increasing order $\{v_1, v_2, \ldots, v_r\}$,
  - One possible threshold between two adjacent values v_i and v_{i+1}. Try all possible thresholds and find the one that maximizes the gain (or gain ratio).

An example in a continuous space

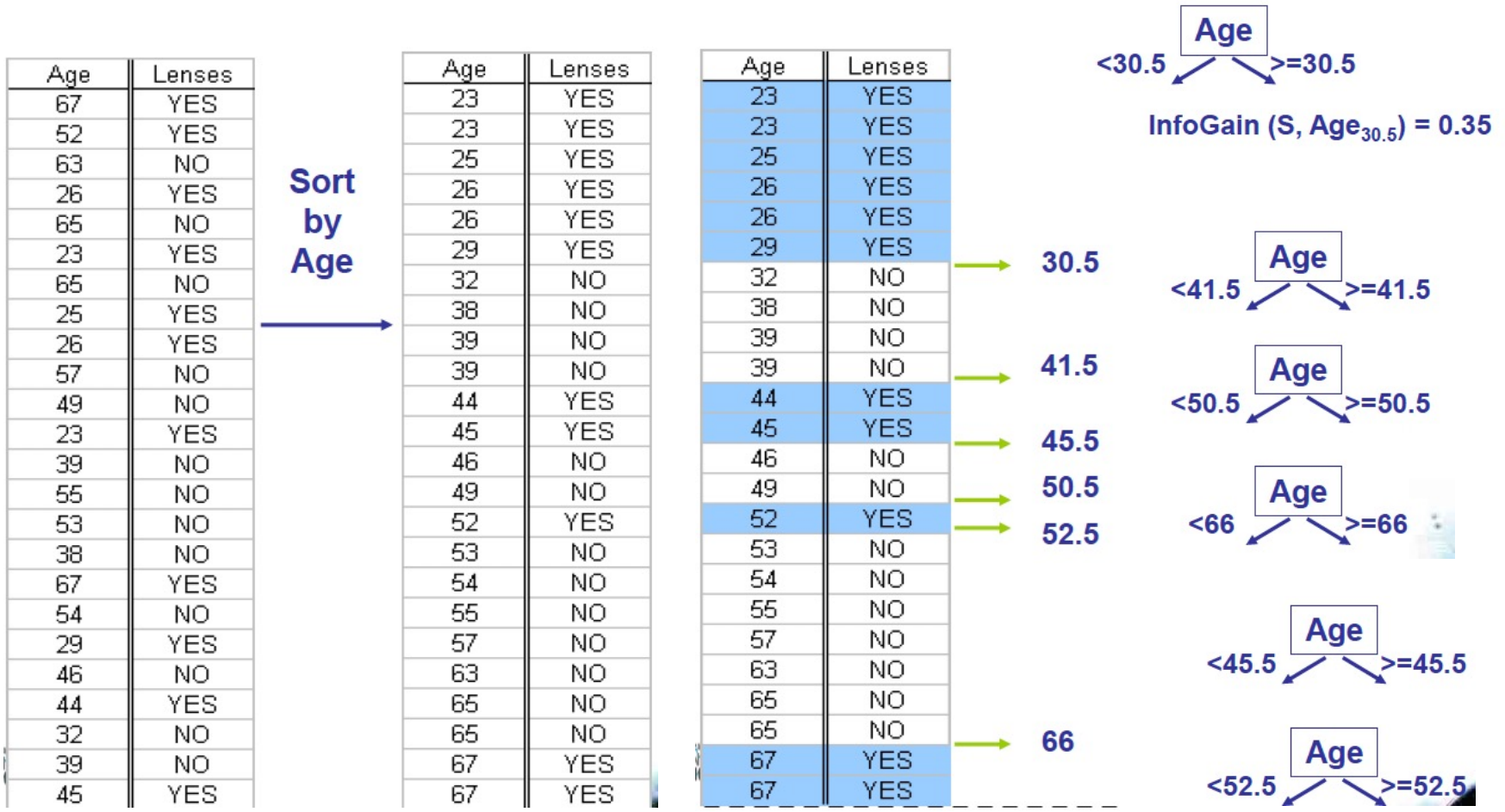


(A) A partition of the data space



(B). The decision tree

Continuous Attributes example



Handling missing values

Algorithm ID3: does not handle missing values

Algorithm C4.5(J48) deals with two problems:

- Missing values in train data:
  - Missing values are not used in gain and entropy calculations
- Missing values in test data:
  - A missing continuous value is replaced with the median* of the training set
  - A missing categorical values is replaced with the most frequent value

* the value separating the higher half from the lower half of a dataset

1, 3, 3, **6**, 7, 8, 9

Median = 6

1, 2, 3, **4**, **5**, 6, 8, 9

Median = (4 + 5) ÷ 2

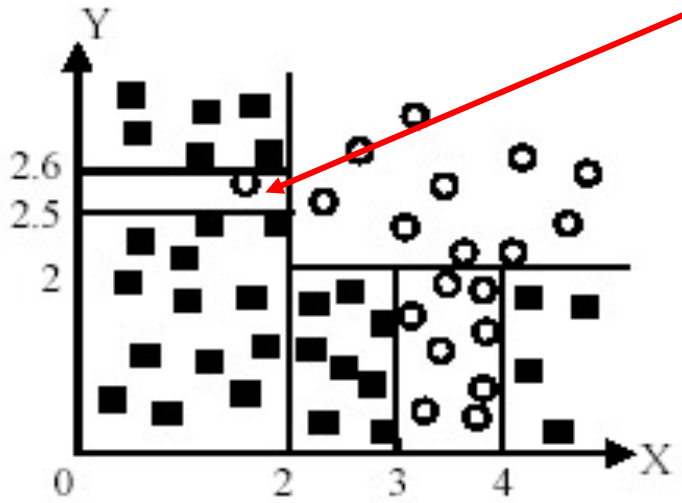
= 4.5

Avoid overfitting in classification

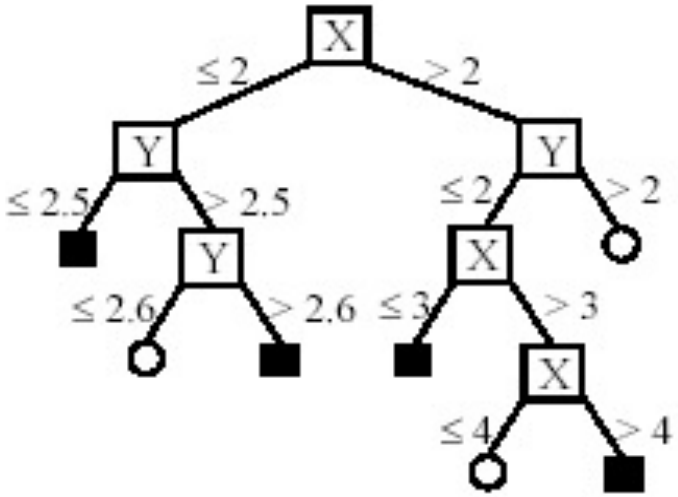
- Overfitting: A tree may overfit the training data
  - Good accuracy on training data but poor on test data
  - Symptoms: tree too deep and too many branches, some may reflect anomalies due to noise or outliers
- Two approaches to avoid overfitting
  - Pre-pruning: Halt tree construction early
    - Difficult to decide because we do not know what may happen subsequently if we keep growing the tree.
  - Post-pruning: Remove branches or sub-trees from a "fully grown" tree.
    - This method is commonly used. C4.5 uses a statistical method to estimates the errors at each node for pruning.
    - A validation set may be used for pruning as well.

An example

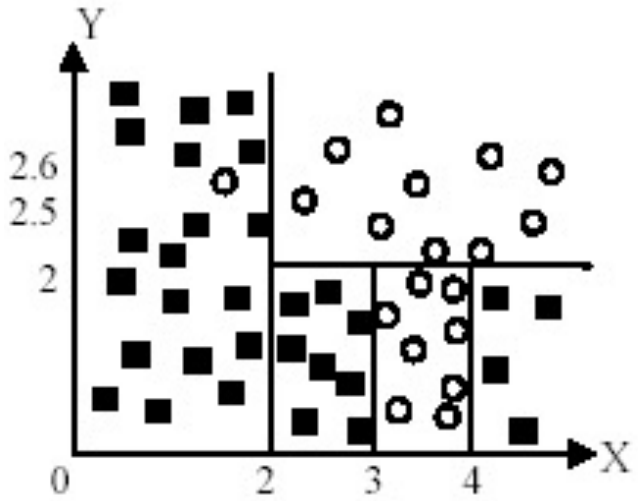
Likely to overfit the data

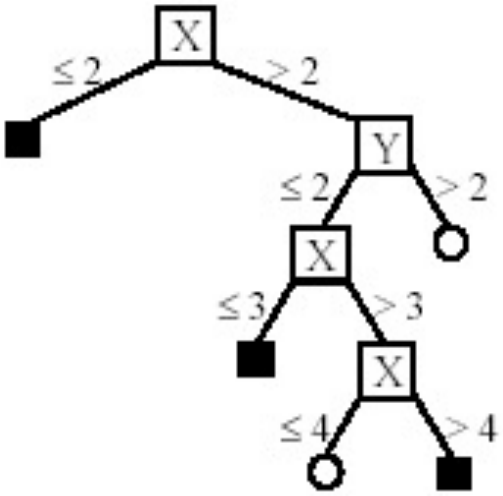


(A) A partition of the data space



(B). The decision tree





Other issues in decision tree learning

- From tree to rules, and rule pruning
- Handling of missing values
- Handling skewed distributions
- Handling attributes and classes with different costs.
- Attribute construction
- Etc.

Exercise - WEKA

- CSV vs ARFF
- J48 classification
  - Weather dataset analysis
  - Bank dataset analysis
  - Results
    - Accuracy
    - Confusion matrix