

# Data Mining

## - Classification Algorithms

-II



---

Dr Xue Li

University of Queensland, Brisbane Australia

<http://www.itee.uq.edu.au/~dke>



# Classification Algorithms

---

- Classification by decision tree induction (ID3)
- Bayesian Classification
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)
- Other Classification Methods
- Prediction
- Classification accuracy
- Summary



# Classification by Induction of Decision Trees

---

- Attribute selection by Information Entropy
- Decision tree induction
- Rule generation

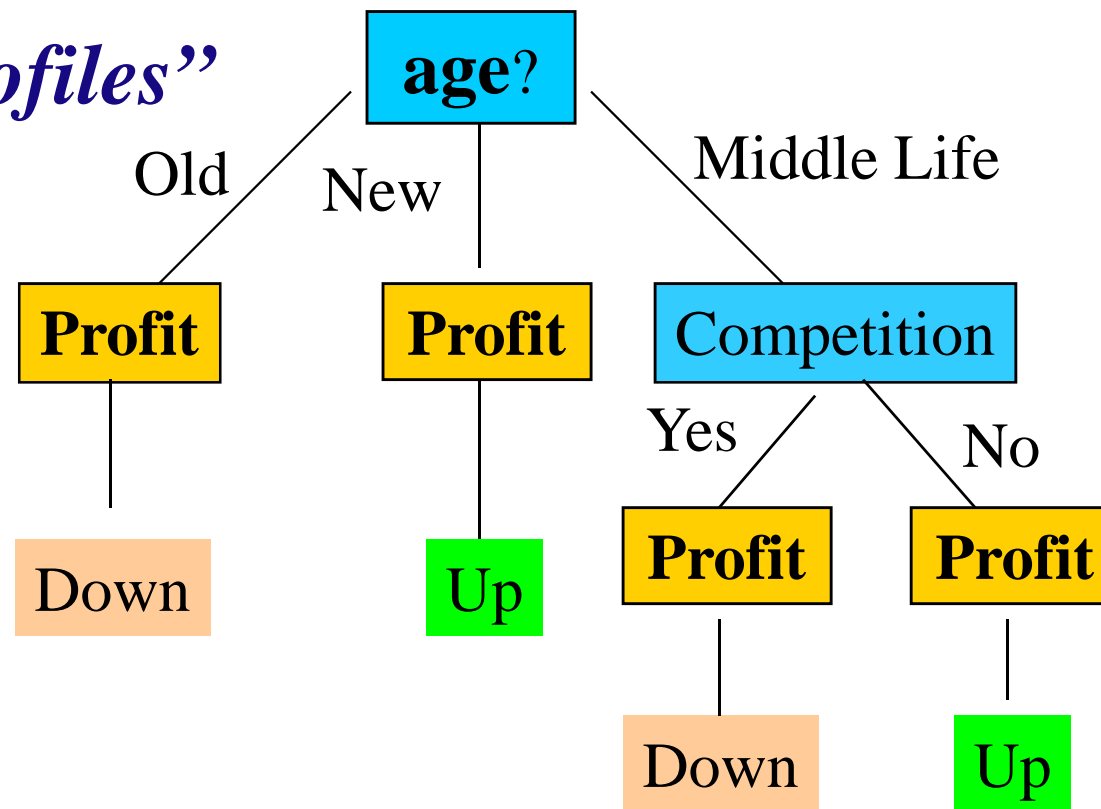
## An example:

find a means of predicting which **company profiles** will lead to a increase or decrease in profits based on the following data:

Profit	Age	Competition	Type
Down	Old	No	Software
Down	Midlife	Yes	Software
Up	Midlife	No	Hardware
Down	Old	No	Hardware
Up	New	No	Hardware
Up	New	No	Software
Up	Midlife	No	Software
Up	New	Yes	Software
Down	Midlife	Yes	Hardware
Down	Old	Yes	Software

# Output: A Decision Tree for

*“Company profiles”*



# A Simple Thought on Decision Tree

- A simple minded algorithm:

**If Age = X and Competition = Y and Type = Z then Profit = ?**

- This structure will yield a branch for each row, not eliminating superfluous attributes, and is unnecessarily complex.
- The complete classification space for  $m$  attributes is of size:

$$\prod_{j=1}^m N_j \quad (N_j \text{ is the number of values of attribute } j)$$

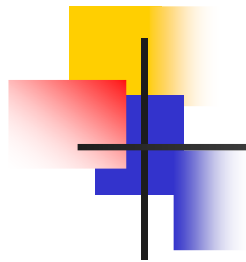


# An Observation on Rule Generation

---

- Rules are based solely upon the input examples.
- Rules should be able to predict results for other cases.
- If prediction is not achieved, then the rules must be alterable either automatically or editorially.

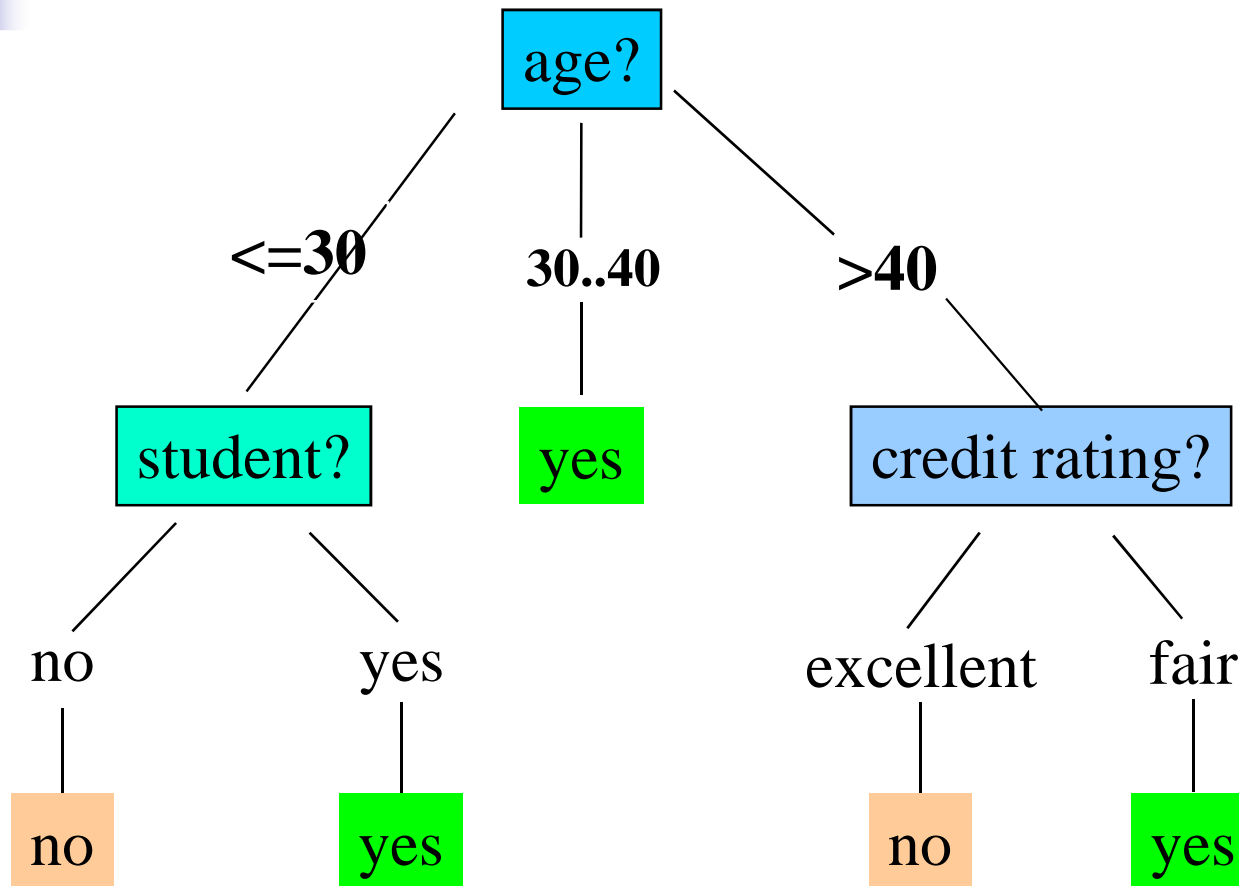
# Training Dataset



An  
example  
from  
Quinlan's  
ID3

age	income	student	credit_rating	buys_computer
$\leq 30$	high	no	fair	no
$\leq 30$	high	no	excellent	no
31..40	high	no	fair	yes
$> 40$	medium	no	fair	yes
$> 40$	low	yes	fair	yes
$> 40$	low	yes	excellent	no
31..40	low	yes	excellent	yes
$\leq 30$	medium	no	fair	no
$\leq 30$	low	yes	fair	yes
$> 40$	medium	yes	fair	yes
$\leq 30$	medium	yes	excellent	yes
31..40	medium	no	excellent	yes
31..40	high	yes	fair	yes
$> 40$	medium	no	excellent	no

# Output: A Decision Tree for *“buys\_computer”*





# The Problem Statement

---

Given a **test set** of data described by a set of **attributes** and an outcome **class**, the problem is to find a minimum **decision tree** that will classify the values of the class based on the values of given attributes.



# Building the Decision Tree

---

- Collect examples to build a **test set** (“training set”).
- Decide the **class**.
- **Select one of the attributes** to be the starting point or root node of the tree.
- **Split** up the test set into a number of smaller tables, each containing examples with the same value of the selected attribute.
- If the values in the **class is partitioned**, then the process is complete. Otherwise, select a new attribute and split the set again.

# Let's have a go ...

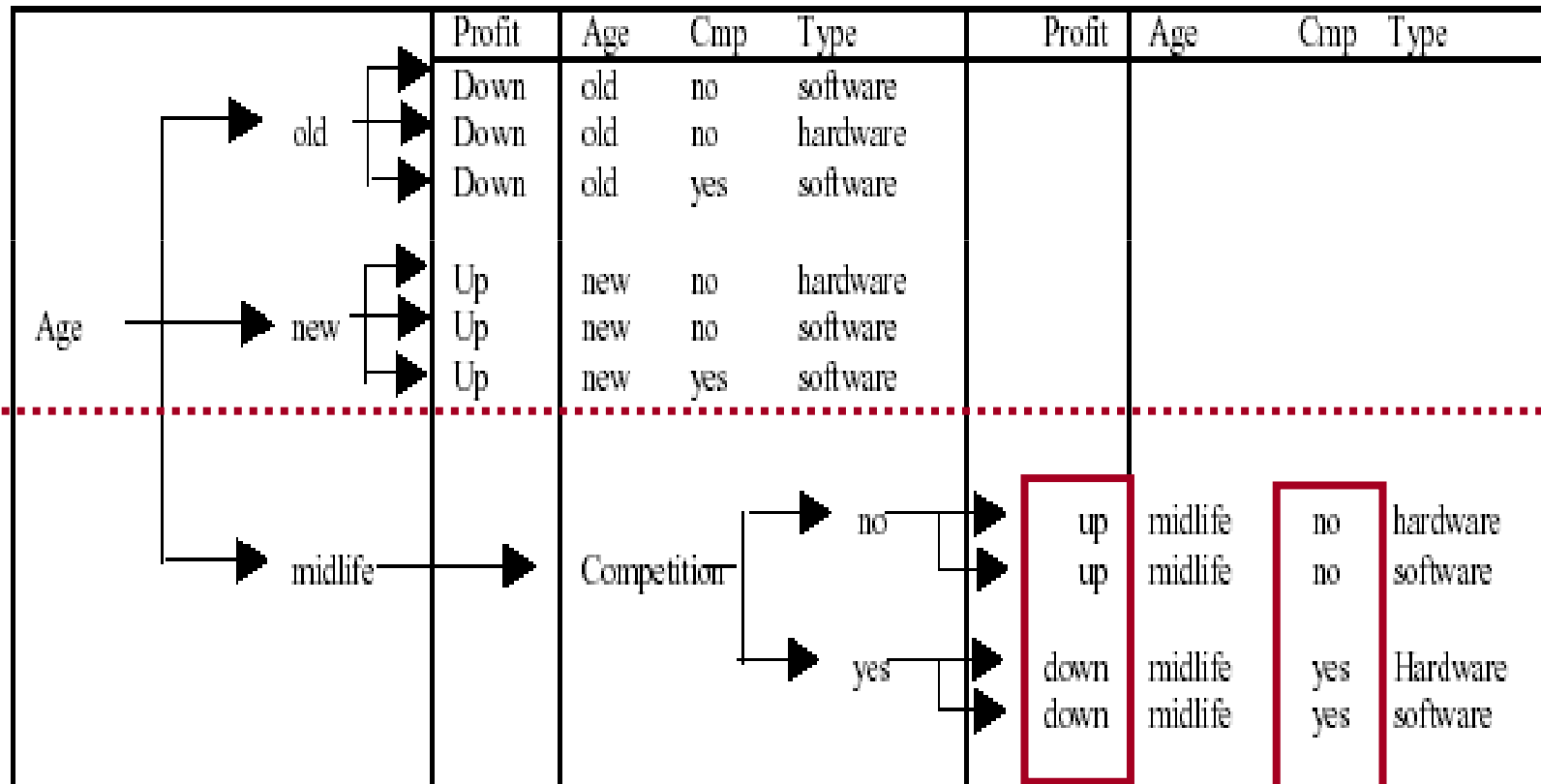
For the Company Profile example we perform a split on attribute **Age**:

	Profit	Age	Competition	Type	
Age	down	old	no	software	
		old	no	hardware	
		old	yes	software	
	up	new	no	hardware	
		new	no	software	
		new	yes	software	
	down	up	midlife	yes	software
			midlife	no	hardware
		up	midlife	no	software
midlife			yes	hardware	

Splitting a table is to sort the attribute and the class values. The “clear-matched” rows of the table will be split out.

# The Test Set Becomes smaller ...

After the Second Split on Attribute Competition





# The Rules Derived from the Tree

---

**R1: IF** age is old **THEN** profit is down.

**R2: IF** age is new **THEN** profit is up.

**R3: IF** age is midlife **AND** competition is no  
**THEN** profit is up.

**R4: IF** age is midlife **AND** competition is  
yes **THEN** profit is down.

Can the original test set  
be re-generated base of  
this set of rules?

Can this set of rules be used to  
predict all cases in the problem  
domain (i.e., the PROFIT prediction)?



We know how to create a tree now, but we still need to know where to start...

---

- Which attribute AGE, TYPE, or COMPETITION is most strongly associated with the class PROFIT?

Note that:

- PROFIT has 2 states: UP or DOWN
- COMPETITION has 2 states: NO or YES
- TYPE has 2 states: SOFTWARE or HARDWARE
- AGE has 3 states: OLD, NEW or MIDLIFE

So, we need to differentiate the “predictability” of the attributes towards the class.

Is there any indication between the values of Competition and Values of Profit? No, not at all.

Competition	Profit
no	down
no	up
no	down
no	up
no	up
no	up
<hr/>	
yes	down
yes	up
yes	down
yes	down

$P(\text{Profit} = \text{up} \mid \text{Comp} = \text{No}) = 4/6 = .67$   
 $P(\text{Profit} = \text{down} \mid \text{Comp} = \text{No}) = 2/6 = .33$   
 $P(\text{Profit} = \text{up} \mid \text{Comp} = \text{Yes}) = 1/4 = .25$   
 $P(\text{Profit} = \text{down} \mid \text{Comp} = \text{Yes}) = 3/4 = .75$

So what?

It has to be compared with the other attributes.



**The Conditional Probability seems to be a right tool to show the relations between the attribute and the class.**

---

**TYPE:**

$$P(\text{Profit} = \text{up} \mid \text{Type} = \text{Software}) = 0.5$$

$$P(\text{Profit} = \text{down} \mid \text{Type} = \text{Software}) = 0.5$$

$$P(\text{Profit} = \text{up} \mid \text{Type} = \text{Hardware}) = 0.5$$

$$P(\text{Profit} = \text{down} \mid \text{Type} = \text{Hardware}) = 0.5$$

**AGE:**

$$P(\text{Profit} = \text{up} \mid \text{Age} = \text{Old}) = 0.0$$

$$P(\text{Profit} = \text{down} \mid \text{Age} = \text{Old}) = 1.0$$

$$P(\text{Profit} = \text{up} \mid \text{Age} = \text{New}) = 1.0$$

$$P(\text{Profit} = \text{down} \mid \text{Age} = \text{New}) = 0.0$$

$$P(\text{Profit} = \text{up} \mid \text{Age} = \text{Midlife}) = 0.5$$

$$P(\text{Profit} = \text{Down} \mid \text{Age} = \text{Midlife}) = 0.5$$

When Age = Old, the probability of Profit =up is zero.

# The problem: How do we select an attribute?



We need to find a formula to determine the most significant attribute in a calculation of the probabilities of the attributes.

How can this be formalized? How can this be extended to deeper levels of the decision tree?

Need some predictive statistic based on conditional probability that can be applied to sort out the 'best' attribute at each level.

**Amongst three attributes: AGE, COMPETITION, and TYPE, does which attribute carry “more” information than others?**



# Can Information be Measured?

---

- Why should information be measurable?
  - Information Theory
  - Information retrieval, coding, and processing techniques
- If it is measurable what can we do about it?
  - Applicability in data mining
  - Quantification of the factors that affect our decisions (to order the importance of different messages that we receive).



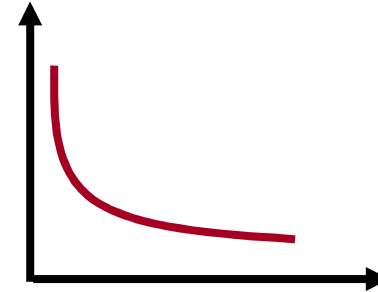
# Information Theory

- How much **information** a message contains depends on the extent to which it resolves **uncertainty**.
- The greater the number of possible messages, the greater the amount of information conveyed (*additivity of accumulated information*)
- The more probable a message is, the less information it conveys (i.e., the amount of information increases as the probability of the message decreases, they are inversely related).

# Information Measuring

- Information additivity

$$I(C) = \sum_{i=1}^m I(c_i)$$



- Inversity between Information and Uncertainty

$$I(c_i) = I(1/p(c_i))$$

- In probability theory, the total probability of two events are the multiplication of two probabilities.

$$I(1/p(c_1)) + I(1/p(c_2)) = I(1/(p(c_1) \times p(c_2)))$$

How do we convert function  $I()$  from the probability  $p()$ ?



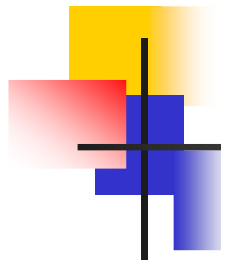
# Information Bit: the Quantity of Information

A similar question: Why is binary system used in a computer system?

- Information is measured in terms of the Order (magnitude) of the inversed probability – the logarithm of  $(1/p)$  for the “information bit”

$$I(1/p) = \log(1/p)$$

- In the simplest case where one of two equally probable messages is selected, each with a probability of  $1/2$ , the quantity of information is  $\log(1/1/2)$ , or  $\log 2$ . The  $\log$  of 2 to the base 2 is 1. Thus by choosing base 2, we are able to deal with the simplest cases and use  $\log_2 2$  as a unit of information measuring of 1.



What is the information quantity of a serials of messages each carrying a different probability?

The **average** information content is calculated by the sum of the probabilities multiplied by the information bits of each message.

For example: in a two letter message **A** or **B** with  $2/3$  and  $1/3$  probabilities, the average information content for it is:

$2/3 \times \log_2(1/2/3) + 1/3 \times \log_2(1/1/3) = 0.918$   
information bits.

Since the probabilities of messages may not be equal, we should weight them according to their probabilities.

# Shannon's Formula

- **Information entropy** is a measure of the uncertainty of classification of that object with regards to all objects being classified.
- Given a set of objects  $C$  and a partitioning  $c_1 \dots c_n$ , the entropy of this classification scheme is given by an addition of the information carried by individual partitions:

$$H(C) = - \sum_{i=1}^n p(c_i) \times \log_2 (p(c_i))$$

Where  $P(C_i)$  is the probability of partition  $c_i$ .




## Example: the information content for the class PROFIT:

---

$$\begin{aligned} H(C) &= - (p(\text{Profit} = \text{up}) * \log_2 (p(\text{Profit} = \text{up})) + \\ &\quad p(\text{Profit} = \text{down}) * \log_2 (p(\text{Profit} = \text{down}))) \\ &= - ( 0.5 * (-1) + 0.5 * (-1)) \\ &= 1 \end{aligned}$$

This shows that one information bit is needed to represent two different states of the PROFIT: **up** and **down**. No information about how the value of up or down is classified based on other attribute values.



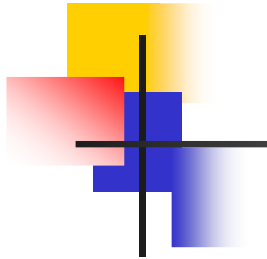
So, if we use Shannon's formula to determine the most significant attribute in the derivation of the decision tree, we need to use the *conditional probability*.

- For a probability of a given attribute value (say, AGE is old), what is the information entropy of the class C?

$$H(C | a_j) = - \sum_{i=1}^n p(c_i | a_j) * \log_2 (p(c_i | a_j)) \quad (\text{Formula 1})$$

- Where  $i = 1 .. n$ . ( $n$  different class values). The function  $p(c_i | a_j)$  is the probability that the class value is  $c_i$  when the attribute has its  $j$ th value.

# Example of Formula 1



$$\begin{aligned} H(C \mid \text{Comp} = \text{no}) &= - [p(\text{up} \mid \text{Comp} = \text{no}) \log_2(p(\text{up} \mid \text{Comp} = \text{no})) + \\ &\quad p(\text{down} \mid \text{Comp} = \text{no}) \log_2(p(\text{down} \mid \text{Comp} = \text{no}))] \\ &= - [4/6 * \log_2(4/6) + 2/6 * \log_2(2/6)] \\ &= 0.918 \end{aligned}$$

and

$$\begin{aligned} H(C \mid \text{Comp} = \text{yes}) &= - [p(\text{up} \mid \text{Comp} = \text{yes}) \log_2(p(\text{up} \mid \text{Comp} = \text{Yes})) + \\ &\quad p(\text{down} \mid \text{Comp} = \text{Yes}) \log_2(p(\text{down} \mid \text{Comp} = \text{Yes}))] \\ &= - [1/4 * \log_2(1/4) + 3/4 * \log_2(3/4)] \\ &= 0.811 \end{aligned}$$

Similar calculations are made for

$$H(C \mid \text{Type} = \text{software}) = 1$$

$$H(C \mid \text{Type} = \text{hardware}) = 1$$

$$H(C \mid \text{Age} = \text{old}) = 0$$

$$H(C \mid \text{Age} = \text{new}) = 0$$

$$H(C \mid \text{Age} = \text{midlife}) = 1$$

# How does each attribute contribute in classifying the class?

If we simplistically add the results as in

$H(C| \text{Type} )$

$= H(C| \text{Type} = \text{software}) + H(C| \text{Type} = \text{hardware})$

$= 2$

whereas

$H(C|\text{Comp})$

$= H(C|\text{Comp}=\text{Yes}) + H(C|\text{Comp}=\text{no})$

$= 0.918 + 0.811$

$= 1.729$

etc.

This does not reflect the weighting using the probabilities of belonging to each class. Thus

$$H(C|A) = \sum_{j=1}^m [ p(a_j) * H(C|a_j) ] \quad (\text{Formula 2})$$

Where  $j = 1 .. m$ .  $m$  is the total number of values for the attribute  $A$ .

# Example of Formula 2



$H(C|Comp)$

$$= p(Comp=yes) * H(C|Comp=yes) +$$

$$p(Comp=no) * H(C|Comp=no)$$

$$= 6/10 * 0.918 + 4/10 * 0.811$$

$$= 0.8752$$

Similarly

$$H(C| Age) = 0.4$$

$$H(C| Type) = 1.0$$

“Type” has an entropy of 1, and is therefore not discriminating, and is not required in a rule!  
Superfluous attributes can now be identified, and eliminated from the rule set..

Since **Age** gives the smallest entropy, and thus the least uncertainty, or greatest predictive power, the root node is taken to be **Age**.

# How is a Significant Attribute Determined?

The final process is then to select the attribute with smallest entropy and the partition the data on the basis of this attribute's values. Thus the final choice is made from

$$\text{MIN}_{t=1}^n \{H(C|A_t)\} \quad (\text{Formula 3})$$

Where  $t = 1 \dots n$ .  $n$  is the total number of attributes for the class  $C$ .

One observation is that all three formulas can be merged as one expression, since they are nested (i.e., Formula 1 is nested in Formula 2, and Formula 2 is nested in formula 3).

# Induction Process Using ID3



---

- Determine main attributes and outcomes of tasks;
- Generate or find a training set;
- Convert numeric data into discrete values by range;
- Apply ID3 (Recursively);
  - Use three formulas to select an attribute to split the test set (sort on the attribute and split the rows that the class values are matched with the attribute values).
  - Repeat the above process to use three formulas to select an attribute again for further splitting of the rest of the test set, until all rows of the test set are considered.
- Prune the tree.

# General Steps in Quinlan's ID3 Algorithm

- Select a training set;
- Repeat
  - induce a rule set to explain the current set
  - find exceptions to this rule set in the remaining examples;
  - form a new rule set from old + existing examples;
- Until no more exceptions to the rule (or 'sufficient processing has taken place).

# Problems in ID3



---

- **Noisy data:** poor measurement, data entry errors, or inappropriate outcome recorded.
- **Unknown attribute values:** can be treated as
  - \* special values denoted \*;
  - \* replaced with some average for the attribute;
  - \* or by pre-processing the data and not including in the training set
- **Excessive Branching:** when an large number of attributes are included. Not all attributes figure as prominently in predicting the outcome. The relatively unimportant attributes should be discarded.



# How big should a Test Set be?

---

A small training set may result in many **empty leaves** in the decision tree. The training set needs to be increased in size, and the process repeated. Some training sets have not identified all the relevant attributes. If this occurs, **then the resultant decision tree will have identical sub-branches for different class attributes. this is called a clash**, and is readily checked by machine and person. The only resolution for this is to call in the expert in order to identify **additional attributes**.



# Reading List: Classification Algorithm

---

- ***Decision Tree Learning on Very Large Data Sets***, By Lawrence O. Hall, Nitesh Chawla and Kevin W. Bowyer.
- ***A Tutorial of Induction of Decision Trees***,  
By Xue Li.
- **Quinlan, J.R.** (1986): ***Induction of Decision Trees***. Machine Learning 1: 81-106. 1986.