Supervised Learning

Decision Trees

Decision Trees

- ID3 (Iterative Dichotomiser) [Quinlan, 1986]
- C4.5 [Quinlan, 1993]

  C4.5 improvements to ID3:
  Handling both continuous and discrete attributes
  Handling training data with missing attribute values
  Handling attributes with differing costs.
  Pruning trees after creation

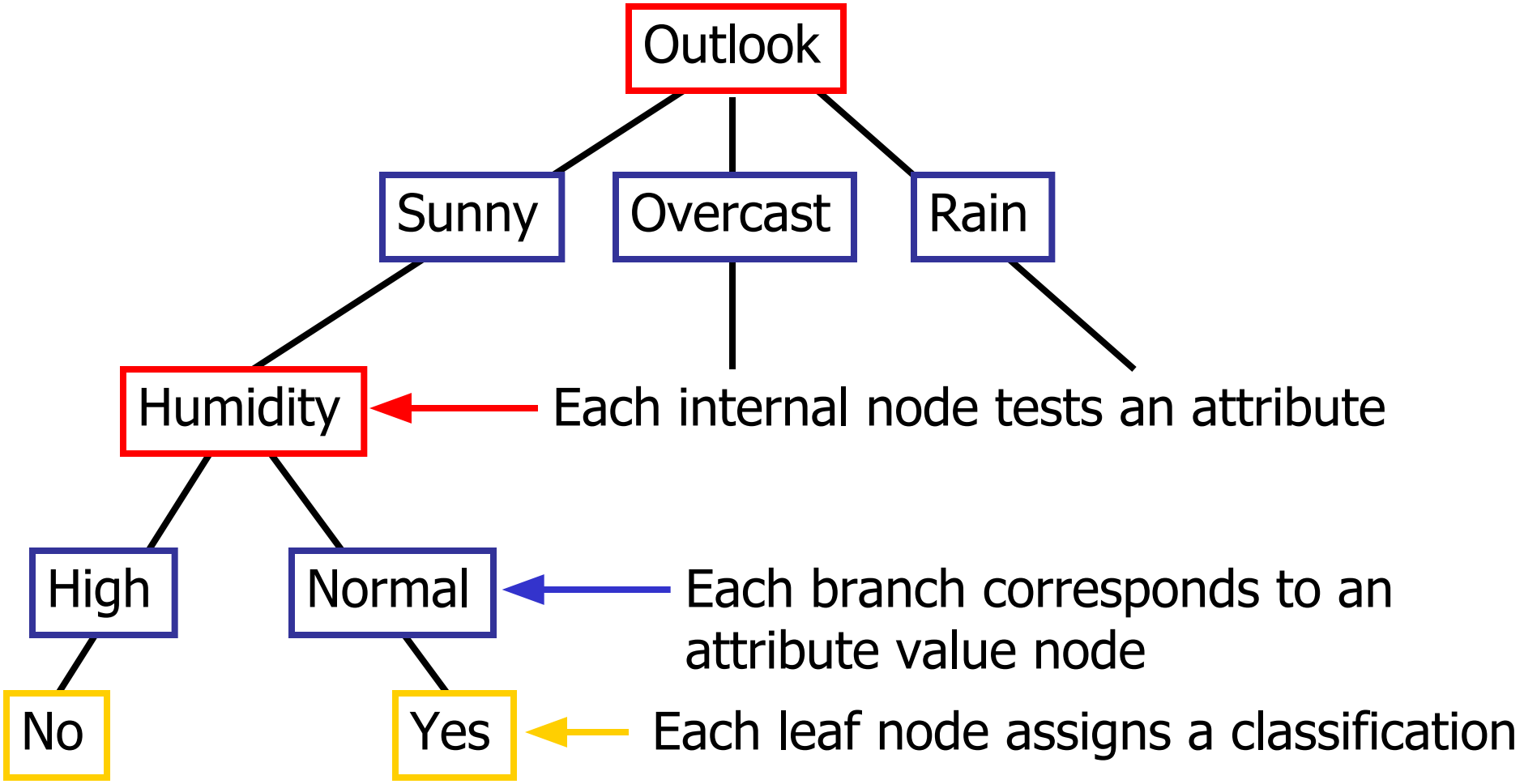
The Swiss-army knife of classifiers

Αν δοκιμάσετε πρώτο ένα ταξινομητή, αυτός θα είναι τα δέντρα αποφάσεων (απόδοση - ταχύτητα - επεξηγηματικότητα)

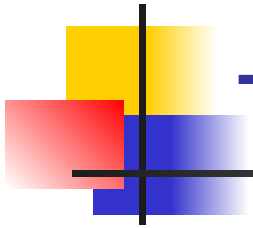
Play Tennis Dataset

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

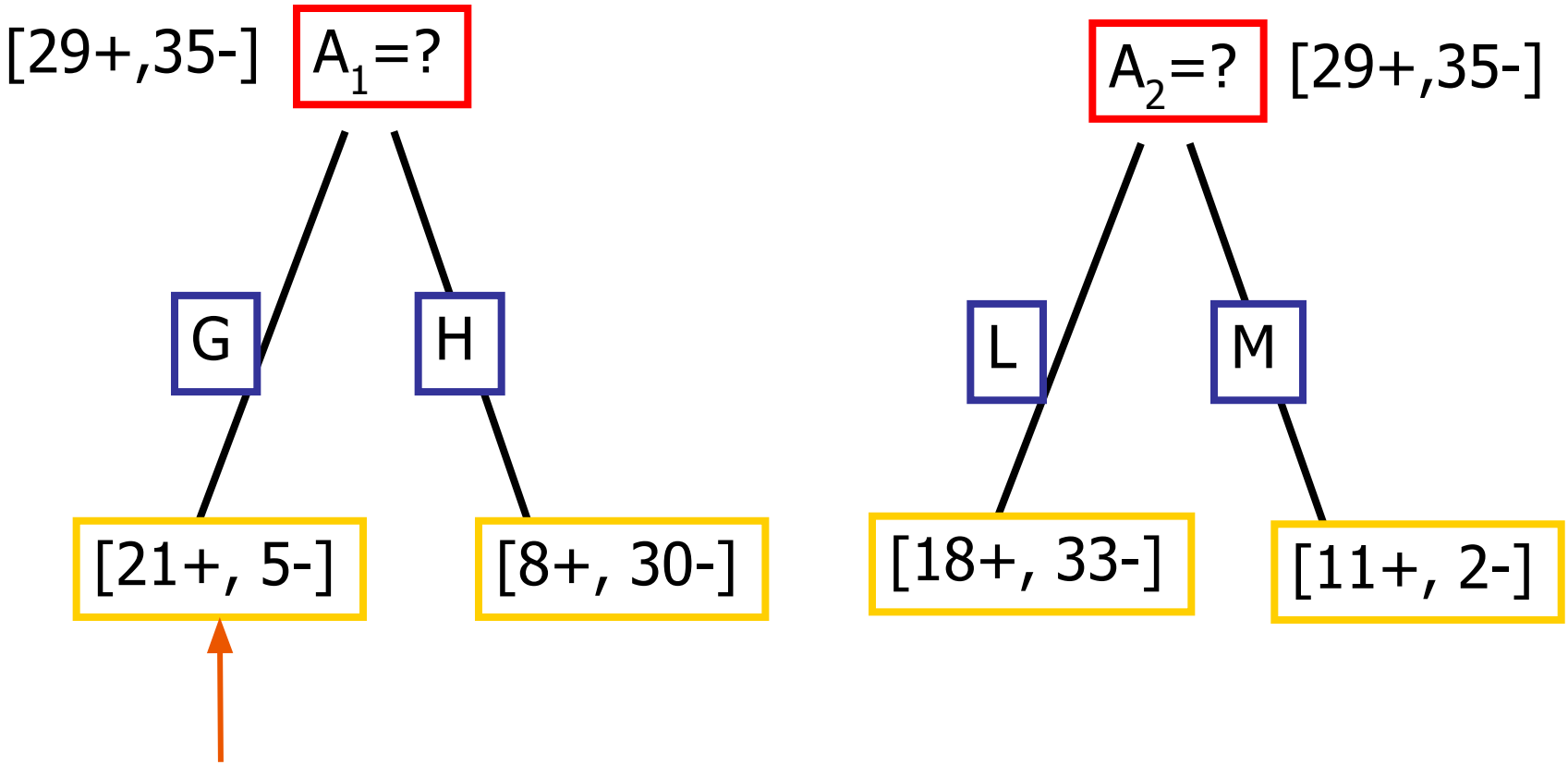


Top-Down Induction of Decision Trees ID3



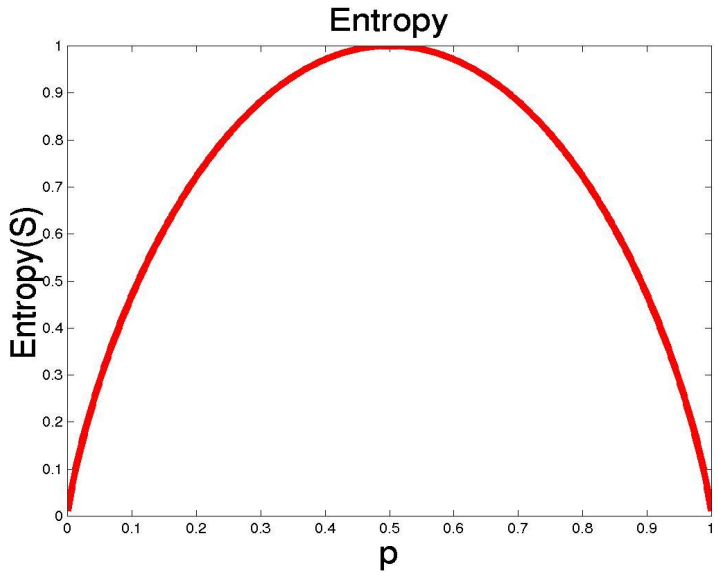
1. A ← the "best" decision attribute for next *node*
2. Assign A as decision attribute for *node*
3. For each value of A create new descendant
4. Sort training examples to leaf node according to the attribute value of the branch
5. If all training examples are perfectly classified (same value of target attribute) stop, else iterate over new leaf nodes.

Which attribute is best?



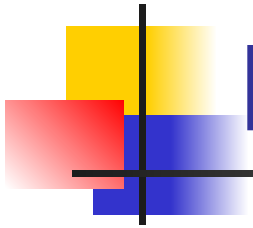
Διαλέγουμε την πλειοψηφική κλάση πχ αν A_1=G τότε "+". Οι περιπτώσεις ισοπαλίας είναι χωρίς σημασία (το σφάλμα είναι το ίδιο).

Entropy



- S is a sample of training examples
- p_+ is the proportion of positive examples
- p_- is the proportion of negative examples
- Entropy measures the impurity of S

$$\text{Entropy}(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$$



Entropy

- Entropy(S)= expected number of bits needed to encode class (+ or -) of randomly drawn members of S (under the optimal, shortest length-code)

Why?

- Information theory optimal length code assign
  $-\log_2 p$ bits to messages having probability p.
- So the expected number of bits to encode (+ or -) of random member of S:

$$-p_+ \log_2 p_+ - p_- \log_2 p_-$$

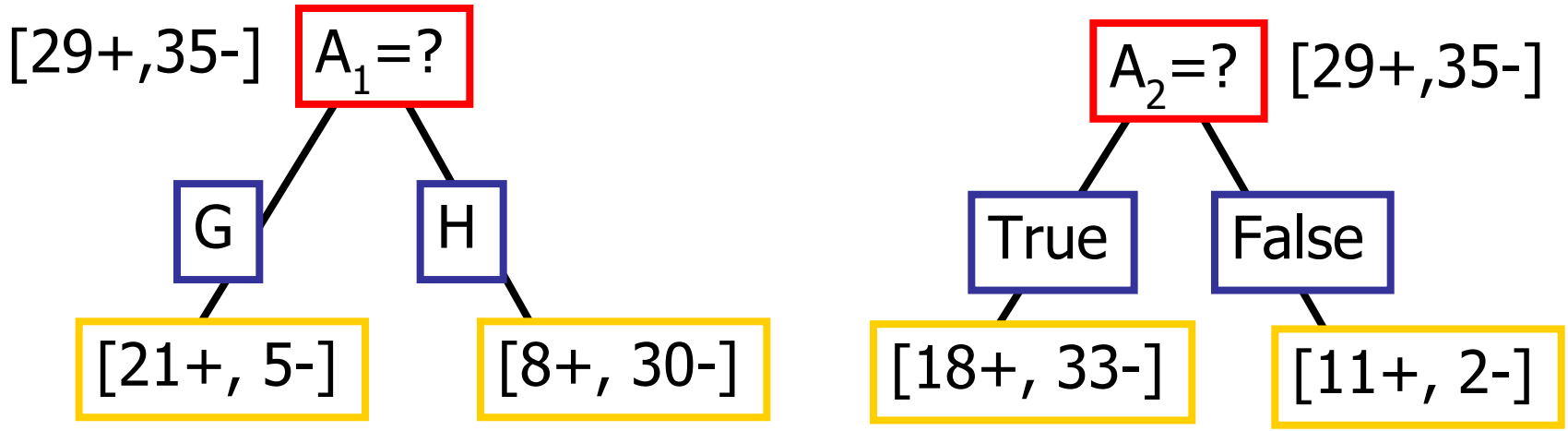
-log2(0.2) = 2.32
-0.2*log2(0.2)-0.8*log2(0.8) = 0.53

Information Gain

- Gain(S,A): expected reduction in entropy due to sorting S on attribute A

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in D_A} \frac{|S_v|}{|S|} Entropy(S_v)$$

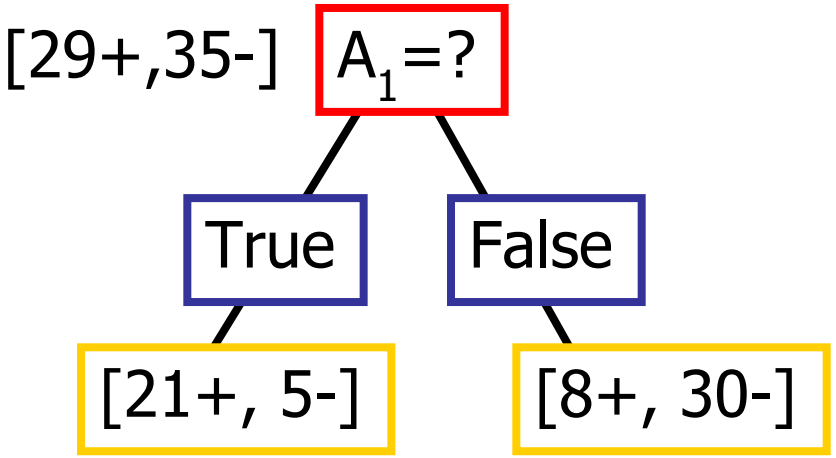
Entropy([29+,35-]) = -29/64 log2 29/64 – 35/64 log2 35/64
= 0.99

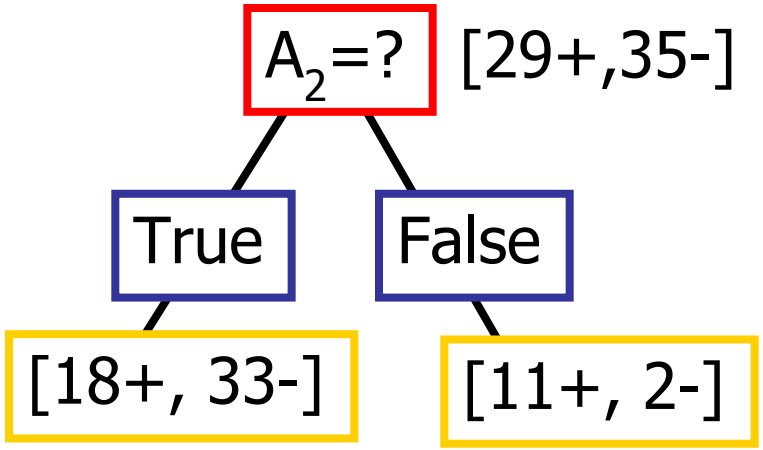


Information Gain

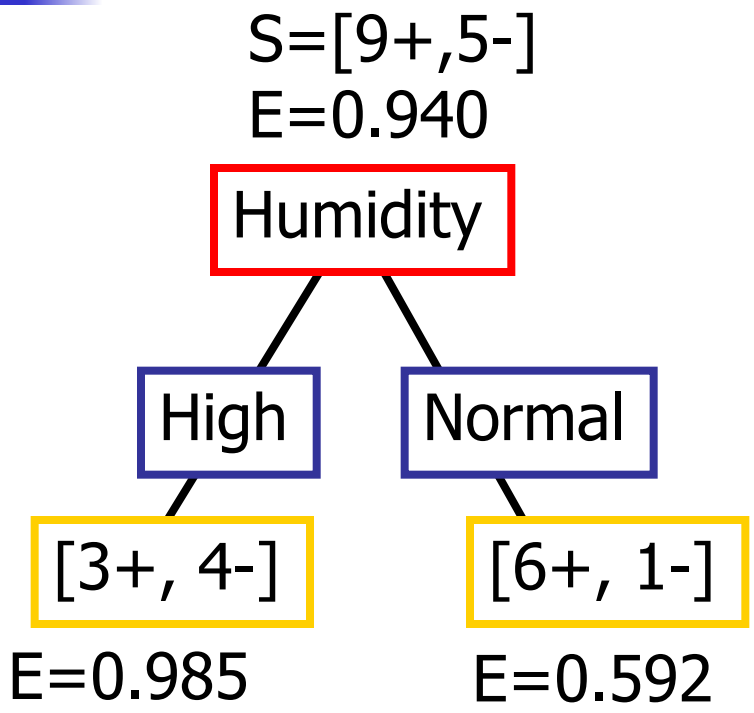
Entropy([21+,5-]) = 0.71
Entropy([8+,30-]) = 0.74
Gain(S,A_1)=Entropy(S)
-26/64*Entropy([21+,5-])
-38/64*Entropy([8+,30-])
=0.27

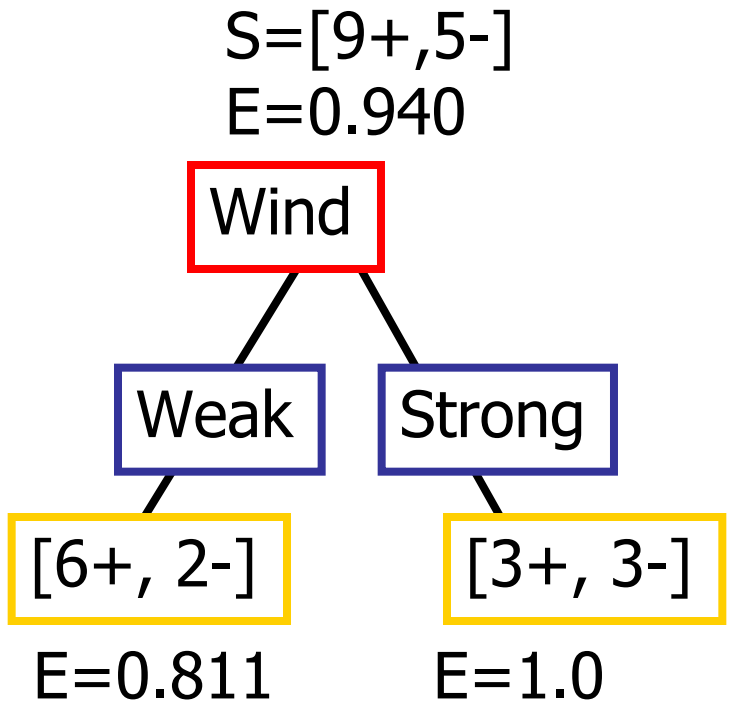
Entropy([18+,33-]) = 0.94
Entropy([11+,2-]) = 0.62
Gain(S,A2)=Entropy(S)
-51/64*Entropy([18+,33-])
-13/64*Entropy([11+,2-])
=0.12





Selecting the Next Attribute



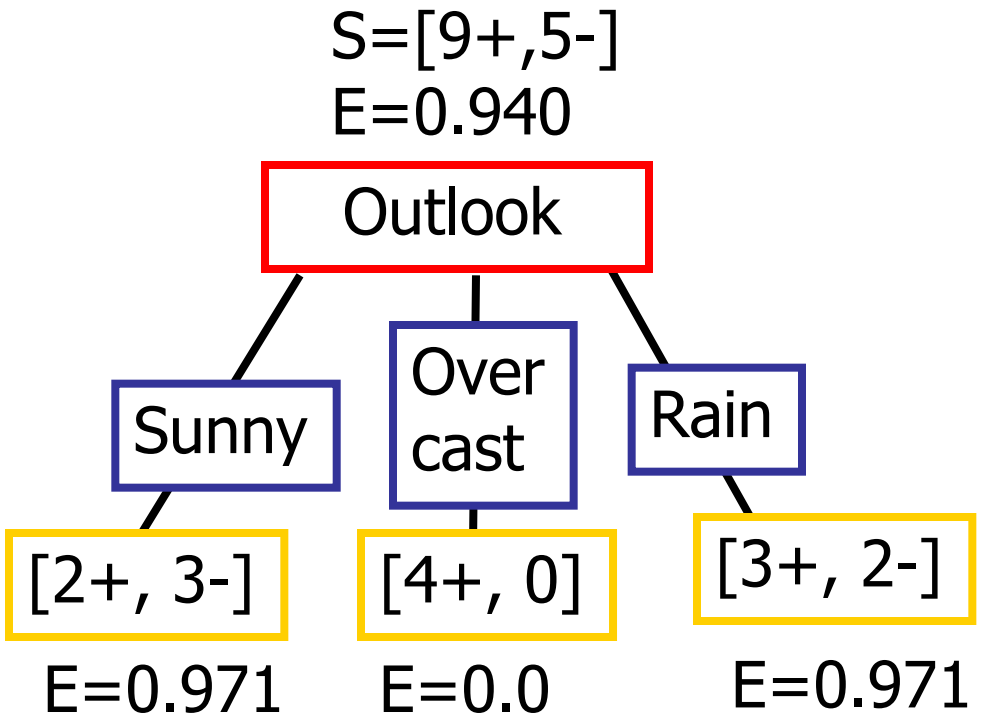


Gain(S,Humidity)
=0.940-(7/14)*0.985
– (7/14)*0.592
=0.151

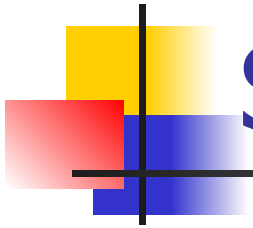
Gain(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0
=0.048

Humidity provides greater info. gain than Wind, w.r.t target classification.

Selecting the Next Attribute



Gain(S,Outlook)
=0.940-(5/14)*0.971
-(4/14)*0.0 – (5/14)*0.0971
=0.247



Selecting the Next Attribute

The information gain values for the 4 attributes are:

- Gain(S,Outlook) =0.247
- Gain(S,Humidity) =0.151
- Gain(S,Wind) =0.048
- Gain(S,Temperature) =0.029

where S denotes the collection of training examples

Other Attribute Selection Measures

Gini index

- CART [Breiman et al., 1984]
- IBM IntelligentMiner [IBM, 1996]

All attributes are assumed continuous-valued

Assume there exist several possible split values for each attribute

May need other tools, such as clustering, to get the possible split values

Can be modified for categorical attribute

Gini Index

- If a data set T contains examples from n classes, gini index, $gini(T)$ is defined as

$$gini\ (T) = 1 - \sum_{j=1}^{n} p_j^2$$

where p_j is the relative frequency of class j in T.

- If a data set T is split into two subsets T_1 and T_2 with sizes N_1 and N_2 respectively, the *gini* index of the split data contains examples from n classes, the *gini* index $gini(T)$ is defined as

$$gini_{split}(T) = \frac{N_1}{N} gini(T_1) + \frac{N_2}{N} gini(T_2)$$

- The attribute provides the smallest $gini_{split}(T)$ is chosen to split the node (*need to enumerate all possible splitting points for each attribute*).