Decision Trees

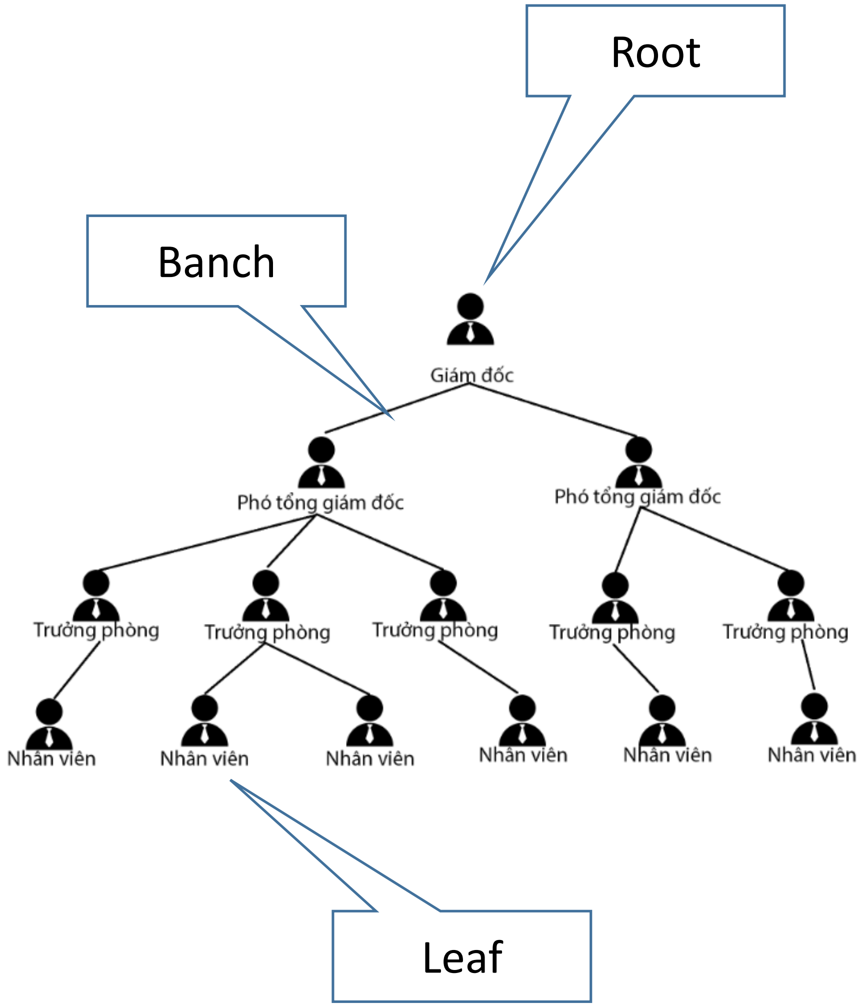
Lương Thái Lê

Outline of the Lecture

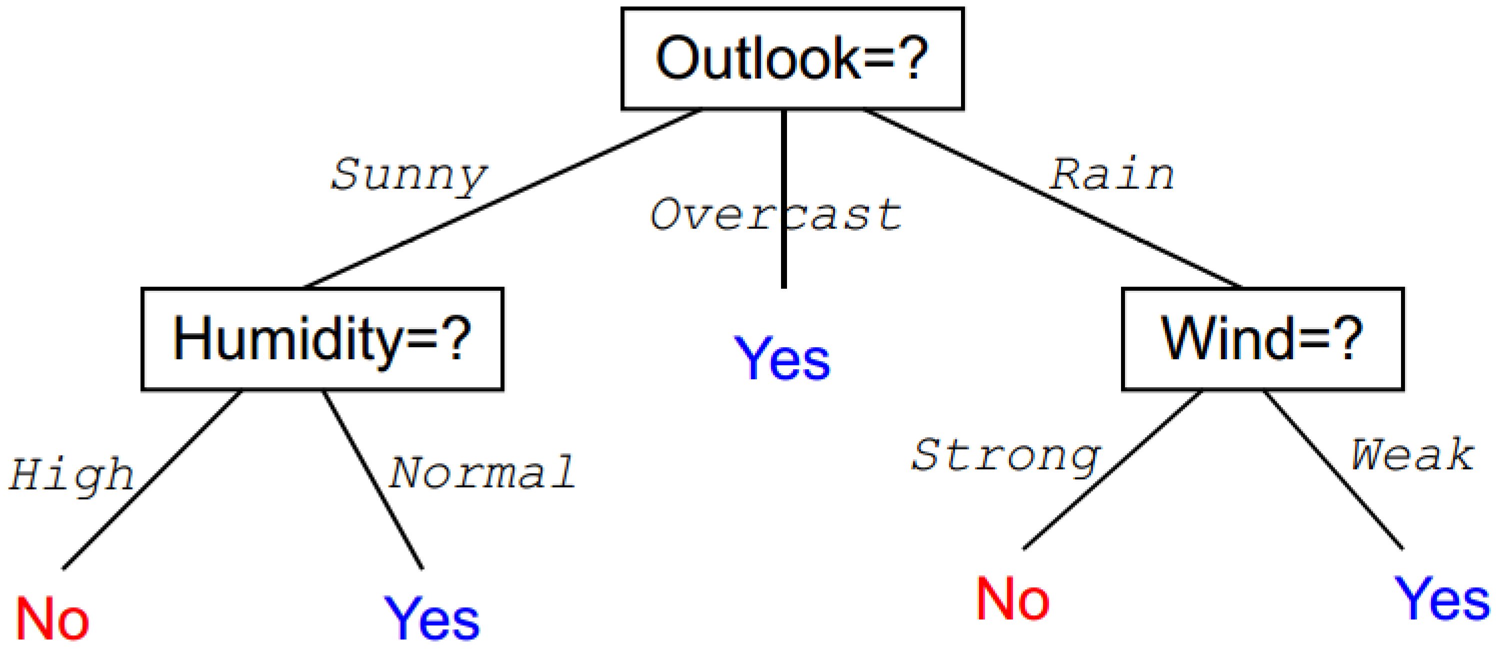
1. **Introduction of Decision Trees (DT)**
2. **DT Algorithms**
3. **Choose the Best Features**
    - **Information Gain**
    - **Example**

Decision Tree (DT) Introduction

- DT is a supervised learning method – classification
- DT learns a classification function represented by a decision tree
- Can be presented by a set of rules IF – THEN
- Can perform even with noise data
- As one of the most common inductive learning methods
- Successfully applied in many application problems
    - Ex: Spam email filtering…



A DT: Example



- (Outlook=Overcast, Temperature=Hot, Humidity=High, Wind=Weak) → Yes
- (Outlook=Rain, Temperature=Mild, Humidity=High, Wind=Strong) → No
- (Outlook=Sunny, Temperature=Hot, Humidity=High, Wind=Strong) → No

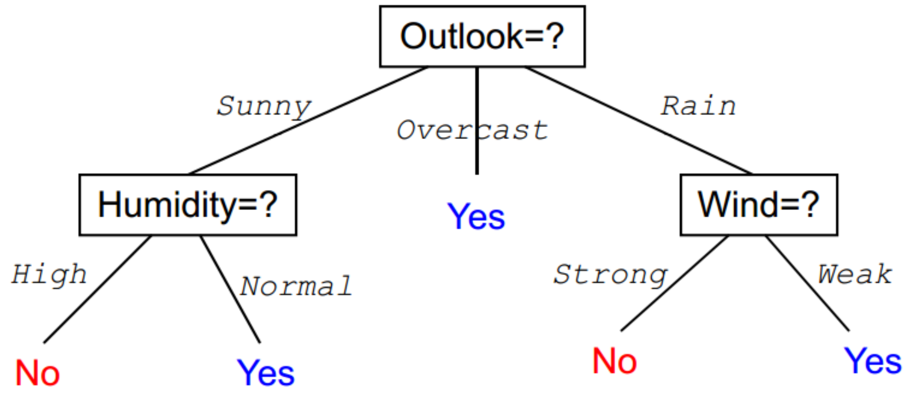
Represent a DT (1)

- Each internal node represents an attribute to be tested for the examples.
- Each branch from a node corresponds to a possible value of the attribute associated with that node
- Each leaf node represents one class c_i in the set of class C
- A learned DT will classify for an example, by traversing the tree from the root node to a leaf node

  => The class label associated with that leaf node will be assigned to the example to be classified

Represent a DT (2)

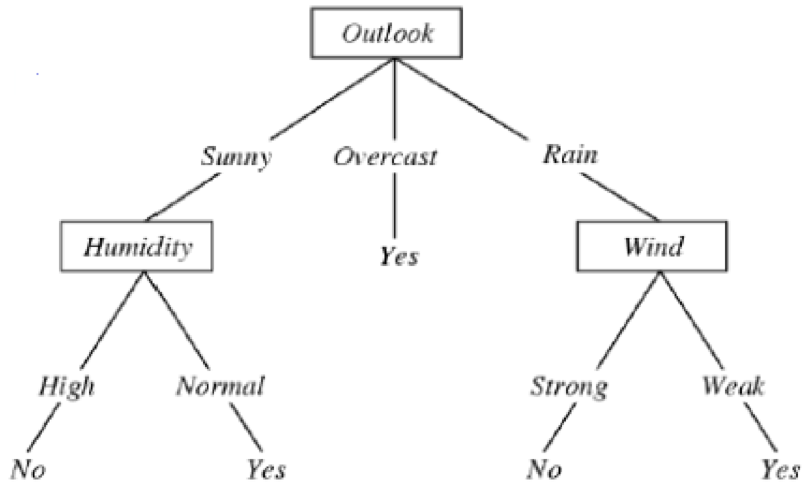
- A DT represents a disjunction of combinations of constraints for the attribute values of the examples
  - Each path from the root node to a leaf node corresponds to a combination of attribute tests



$$[(\text{Outlook}=\texttt{Sunny}) \wedge (\text{Humidity}=\texttt{Normal})] \vee$$
$$(\text{Outlook}=\texttt{Overcast}) \vee$$
$$[(\text{Outlook}=\texttt{Rain}) \wedge (\text{Wind}=\texttt{Weak})]$$

DT – Problem Setting

- Set of possible instances X:
  - each instance x in X is a feature vector
  - x = <x_1, x_2,…, x_n >; Ex: <Humidity=low, Win=weak, Outlook=rain, Temp=hot>
- Unknown target function: $f: X \rightarrow Y$
  - $y \in Y; y = 1\ if\ we\ play\ tennis\ on\ this\ day, else\ y = 0$
- Set of function hypotheses $H = \{h |\ h: X \rightarrow Y\}$
  - each hypothesis h is a decision tree



- **Input:**
  - Training Examples: $\{< x^{(i)}, y^{(i)} >\}$ of unkown target function f
- **Output:**
  - Hypothesis $h \in H$ that best approximates f

Top – down Induction of Decision Trees

[ID3, C4.5, Quinlan]

node = Root

Main loop:

1. $A \leftarrow the\ best\ decision\ attribute\ (feature)\ for\ next\ node$
2. $Assign\ A\ as\ decision\ attribute\ for\ node$
3. $For\ each\ value\ of\ \ A, create\ decendant\ of\ node$
4. $Sort\ training\ examples\ to\ leaf\ nodes$
5. $If\ training\ examples\ perfectly\ classified, then\ STOP\ else\ iterate\ over\ new\ leaf\ node$

Which feature (attribute) is the best?

ID3 Pseudocode (Quinlan - 1979)

ID3 alg *($Training\_Set$, $Class\_Labels$, $Attributes$)*

{

Create the $Root$ node of the decision tree

If all examples of $Training\_Set$ belong to the same class c, Return Decision tree with a $Root$ node is labeled c

If the set $Attributes$ is empty, Return Decision Tree with a $Root$ node attached to a class label ≡ ***Majority _Class_Label****($Training\_Set$)*

$A \leftarrow$ *The attribute in the $Attributes$ set has the* ***"best" classifier*** *for $Training\_Set$*

Test Attribute for $Root$ node $\leftarrow A$

For each possible values v of the attribute A

Add a new branch under the $Root$ node, corresponding to the case: "The value of A is v"

Determine $Training\_Set_v$ ={Instance $x | x \subseteq Training\_Set$, $x_A = v$}

If($Training\_Set_v = \emptyset$) then

Create a leaf node with class label= ***Majority _Class_Label****($Training\_Set$)*

Attach this leaf node to the newly created branch

Else Append to the new created branch a subtree generated by ID3_alg($Training\_Set_v$, $Class\_Labels$, $\{Attributes\} \setminus \{A\}$)

Return $Root$

}

Choose the Best Attribute

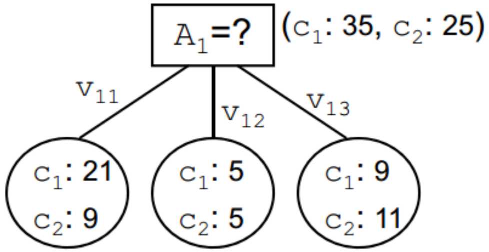
- *How to evaluate an attribute's ability to separate learning examples by their class label?*

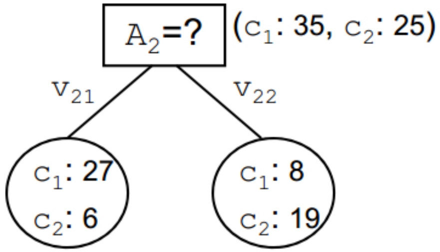
$\Rightarrow$ *Use a statistical evaluation*

Information Gain

- Example:

Which Attribute will be chosen, A_1 or A_2 ?





PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
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	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	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

Entropy

- To evaluate the heterogeneity/impurity of a set
- Entropy of the set S for classification with k classes

$$Entropy(S) = \sum_{i=1}^{k} -p_i log_2\, p_i$$

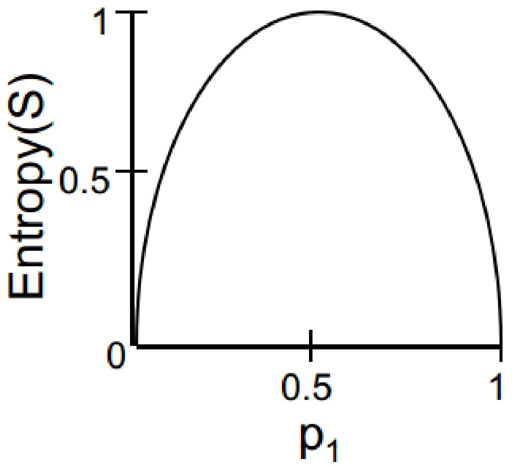
where p_i is the proportion of examples in the set S that belong to class i, and $0.\, log_2 0 = 0$

- Entropy of the set S for classification with 2 classes

$$H(S) \equiv -(\, p_1 log_2 p_1) - (p_2 log_2 p_2)$$

- The meaning of entropy in the field of Information Theory
  - The entropy of the set S indicates the number of bits required to encode the class of an element randomly drawn from the set S.

Entropy – Example with 2 classes



- S includes 14 examples, of which 9 belong to class c_1 (Yes) and 5

examples belong to class c_2 (No)

$$\Rightarrow Entropy(S) = -\left(\frac{9}{14}\right).log_2\left(\frac{9}{14}\right) - \left(\frac{5}{14}\right).log_2\left(\frac{5}{14}\right) \approx 0{,}94$$

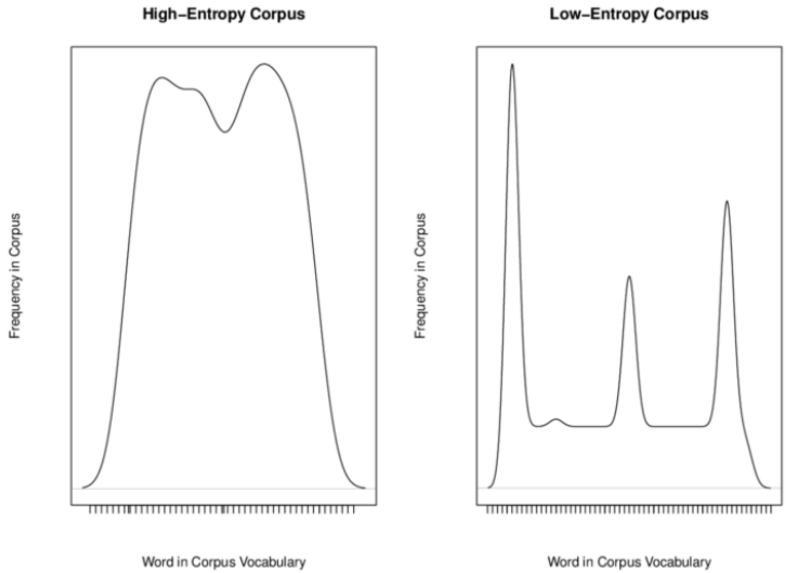
- Entropy = 0, if all examples belong to the same class (c_1 or c_2)
- Entropy = 1, if the number of examples belonging to class c_1 is equal to the number of examples belonging to class c_2
- Entropy = a value in the range (0,1), if the number of examples belonging to class c_1 is different from the number of examples belonging to class c_2

$\Rightarrow$High Entropy:

- x is from a uniform like distribution
- values sampled from it are less predictable

$\Rightarrow$Low Entropy:

- x is from a varied (peaks and valley) distribution
- values sampled from it are easier predictable



Information Gain

- Information Gain of an attribute for a set of examples:
  - The reduction degree in Entropy by partitioning the examples by the values of that property
- Information Gain of attribute A for the set S:

$$IG(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

where `Values(A)` is the set of possible values of the attribute `A` and
$S_v = \{x | x \in S, x_A = v\}$

- Meaning of `IG(S,A)`:
  - The number of bits reduced for the class encoding of an random example from the set `S`, when the value of attribute `A` is known.

=> The best feature is the feature with highest IG

The learning set S (Mitchell-1998)

Day	Outlook	Temperature	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	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	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

Information Example

- Calculate the `Information Gain` value of the `Wind` attribute for the learning set `S : IG(S,Wind)`
- The `Wind` attribute `has 2` possible values : `Weak` and `Strong`
- `S = {`9 for **`Yes`**`,` and 5 for **`No`**`}`
- S_{weak}={6 examples of Yes class and 2 examples of No class with value `Wind=Weak`}
- S_{strong}={3 examples of Yes class and 3 examples of No class with value `Wind=Strong`}

$$IG(S, Wind) = Entropy(S) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$= Entropy(S) - (8/14) Entropy(S_{Weak}) - (6/14) Entropy(S_{Strong})$$

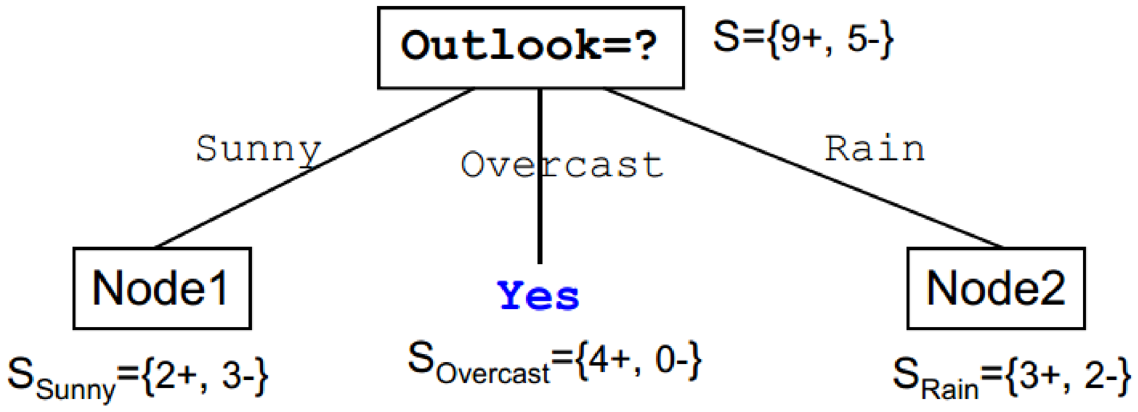
$$= 0{,}94 - (8/14) 0{,}81 - (6/14) 1 = 0{,}048$$

Learning a DT – Example (1)

- For the Root, choose the best feature from the set {`Outlook`, `Temperature`, `Humidity`, `Wind`}
  - IG(S, `Outlook`) = … = 0,246
  - IG(S, `Temperature`) = … =0,029
  - IG(S, `Humidity`) = … = 0,151
  - IG(S, `Wind`) = … = 0,048

The highest IG

=> `Outlook` is chosen to be the test feature for the `Root`

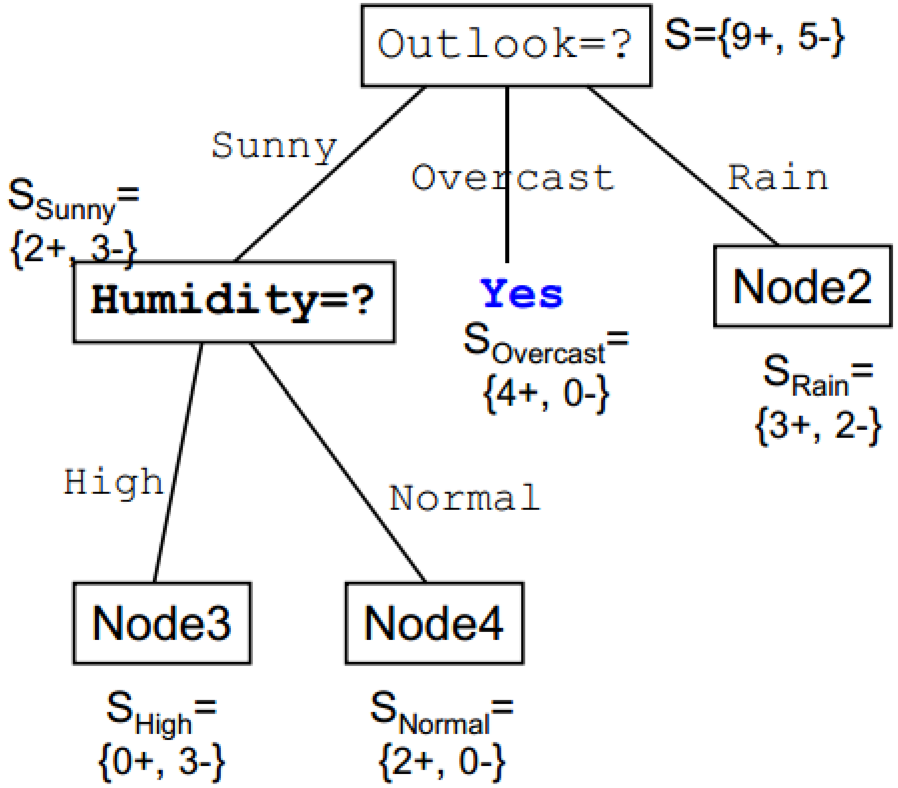


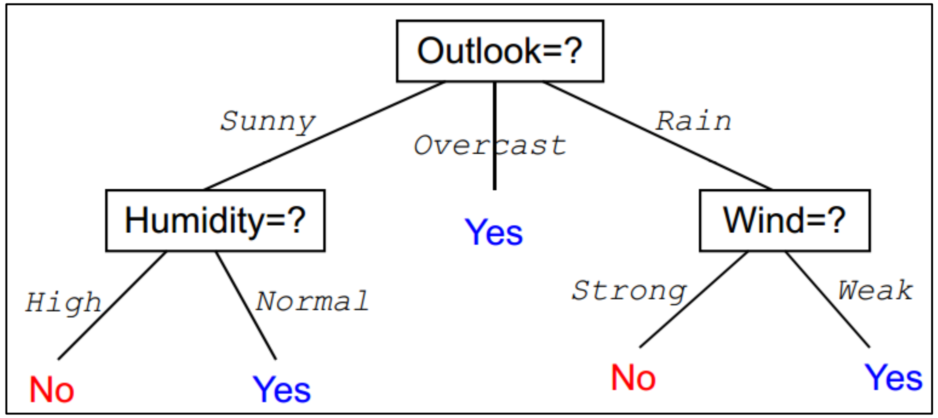
Learning a DT – Example (2)

- For the Node1, choose the best feature from the set {`Temperature, Humidity, Wind`} to be test feature.
  - $IG(S_{Sunny}, Temperature) = \ ... = 0{,}57$
  - $IG(S_{Sunny}, Humidity) = \ ... = 0{,}97$
  - $IG(S_{Sunny}, Wind) = \ ... = 0{,}57$

$\Rightarrow$Choose `Humidity` for Node1

$\Rightarrow$ Similar, we have Node2, Node3, Node4



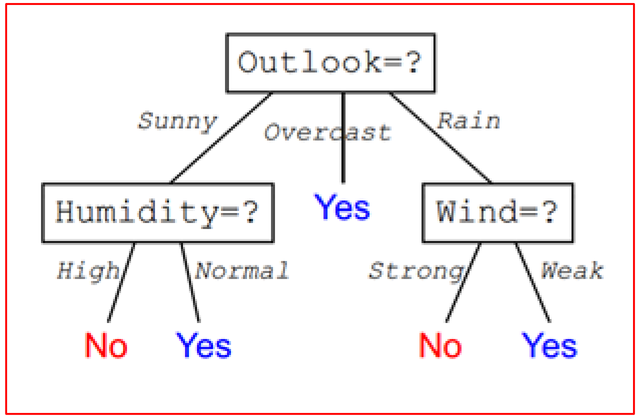


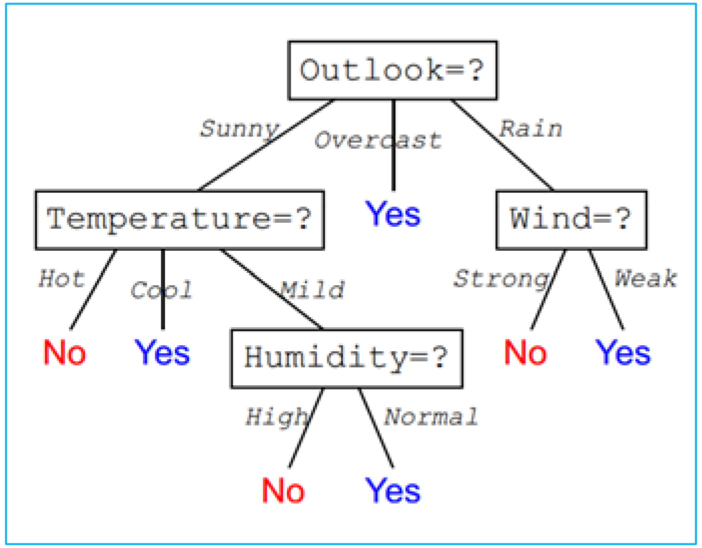
Comment on Stragy of ID3

- ID3 searches only one (but not all) decision trees that fit the training examples
  - chooses the first matching decision tree found during its search
- Use Information Gain to choose the best test feature

  ⇒bias towards multivalued attributes (Ex: Bank account, ID,…) => easily to get overfitting
- During the search, ID3 does not perform backtracking

  => It is only guaranteed to find a locally optimal solution,





Problems in ID3 that Need to be Solve

- Overfitting
- Handling attributes with continuous value (Age, Price…)
- The more suitable evaluations (better than Information Gain) for determining the test attribute for a node
- Handling missing-value attributes training examples
- Handling attributes with different costs

=> C4.5 can handle all above problems

Overfitting Solving

- 2 stragies:
  - Stop learning the decision tree earlier, before it reaches a tree structure that matchs perfect classification of the training set

    => difficult to decide when to stop
  - Learn the full tree (perfectly suitable for the training set), and then perform the tree pruning process.

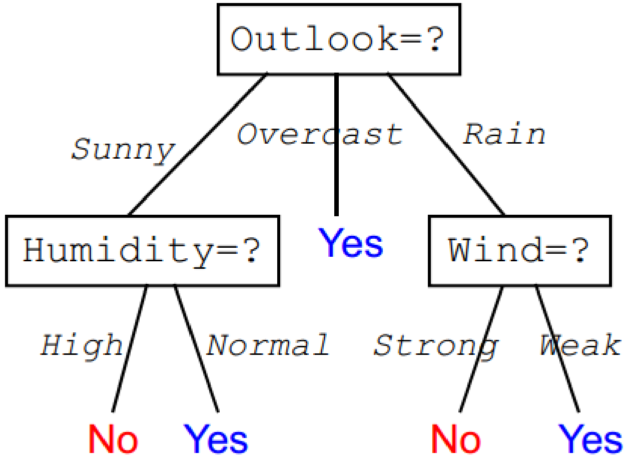
    $\Rightarrow$often give better performance in practice
- How to properly prune trees?
  - Evaluation of classifier performance for an validation set
  - Use *reduced-error pruning* and *rule post-pruning*

Reduced-error Pruning

- Each node of the completely tree is checked for pruning
- A node will be pruned if the tree (after pruning that node) achieves no worse performance than the original tree for the validation set.
- Pruning a node includes:
  - Remove all sub-trees associated with pruned node
  - Convert pruned node to a leaf node (classified label)
    - Attached to this leaf node (pruned node) the class label that dominates the training set associated with that node
- Repeat pruning
  - Always select a node that pruning maximizes the likelihood classification of the decision tree for validation set
  - Stop pruning when it reduces the classifiability of decision tree for the validation set

Rule post-pruning

- Convert the complete decision tree learned into a set of corresponding rules
- Reducing each rule (independently of the others) by removing any conditions that do not help bring about an improvement in the classification efficiency of that rule
- Rearrange the reduced rules according to the classifier ability, and use this order for the classification of future examples



IF (Outlook=Sunny) $\wedge$
(Humidity=Normal)
THEN (PlayTennis=**Yes**)

Features with Continuous values

- Need to convert to discrete-valued attributes, by dividing the continuous interval into a set of non-intersecting intervals.
- For the (continuous) attribute A, create a new attribute of binary type A_v such that: A_v is True if A>v, and False otherwise.
- How to determine the "best" threshold value v?
  - Choose the threshold value v that produces the highest Information Gain value
- Example:
  - Sort the learning examples in ascending value for the `Temperature`
  - Identify learning examples that are contiguous but different from class (`Temperature` 48 & 60; `Temperature` 80 &90)

    $\Rightarrow$average(48,60)=54; average(80,90)=85
  - There are 2 possible threshold values: $\texttt{Temperature}_{54}$ and $\texttt{Temperature}_{85}$
  - The new binary feature $\texttt{Temperature}_{54}$ is selected, because IG(S,$\texttt{Temperature}_{54}$) > IG(S,$\texttt{Temperature}_{85}$)

Temperature	40	48	60	72	80	90
PlayTennis	No	No	Yes	Yes	Yes	No

Gain Ratio – Another Way choosing the best feature

- →Reduce the effect of attributes with many values

$$SplitInformation(S, A) = -\sum_{v \in Values(A)} \frac{|S_v|}{|S|} log_2 \frac{|S_v|}{|S|}$$

$$GainRatio(S, A) = \frac{IG(S, A)}{SplitInformation(S, A)}$$

where Values(A) is the set of possible values of the attribute A

$$S_v = \{x | x \in S, x_A = v\}$$

Handling attributes with missing values (1)

- Suppose attribute $\mathtt{A}$ is a candidate for the test attribute at node n
- How to deal with the example $\mathtt{x}$ has no value for attribute $\mathtt{A}$
- Let $\mathtt{Sn}$ be the set of training examples associated with node n that have a value for the attribute $\mathtt{A}$
    - Solution 1: $\mathtt{x}_{\mathtt{A}}$ is the most common value for attribute $\mathtt{A}$ among the examples belonging to the set $\mathtt{S}_{\mathtt{n}}$
    - Solution 2: $\mathtt{x}_{\mathtt{A}}$ is the most common value for attribute $\mathtt{A}$ among the examples belonging to the set $\mathtt{S}_{\mathtt{n}}$ having the same target class as $\mathtt{x}$

Attributes have different costs

- In some machine learning problems, attributes can be assigned different costs
  - Example: In learning to classify medical diseases, BloodTest has costs $150, while TemperatureTest costs $10
- Tendency to learn cost-based decision trees:
  - Use as many low-cost attributes as possible
  - Only use high-cost attributes when necessary (to help achievereliable classifications)

=> Using assessments other than IG for test attribute identification

$$\frac{Gain^2(S, A)}{Cost(A)}$$

[Tan and Schlimmer, 1990]

$$\frac{2^{Gain(S,A)} - 1}{(Cost(A)+1)^w}$$

[Nunez, 1988; 1991]

When to use DT?

- Learning examples are represented by (attribute, value) pairs.
  - Suitable with discrete-valued attributes
  - For attributes with continuous values, it must be discretized
- The objective function whose output is discrete values
  - Example: Classify the examples into the appropriate class
- The training set may contain noise/error
- The training set may contain missing attributes

Q&A - Thank you!