Artificial Intelligence

For HEDSPI Project

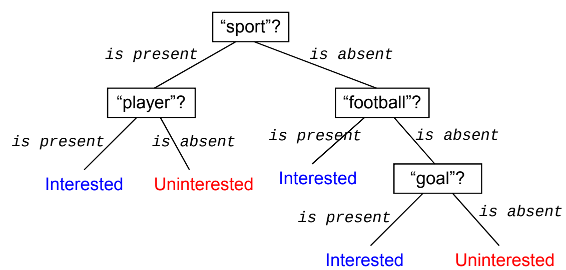
Lecturer 13 – Decision Tree Learning

Lecturers :
Le Thanh Huong
Tran Duc Khanh
Dept of Information Systems
Faculty of Information Technology - HUT

Decision tree – Introduction

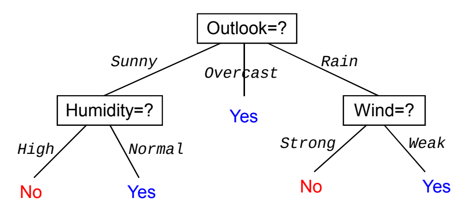
- Decision tree (DT) learning
  - To approximate a ***discrete-valued target function***
  - The target function is represented by a decision tree
- A DT can be represented (interpreted) as a set of IF-THEN rules (i.e., easy to read and understand)
- Capable of learning disjunctive expressions
- DT learning is robust to noisy data
- One of the most widely used methods for inductive inference
- Successfully applied to a range of real-world applications

Example of a DT: Which documents are of my interest?



- (…,"sport",…,"player",…) → Interested
- (…,"goal",…) → Interested
- (…,"sport",…) → Uninterested

Example of a DT: Does a person play tennis?



- (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

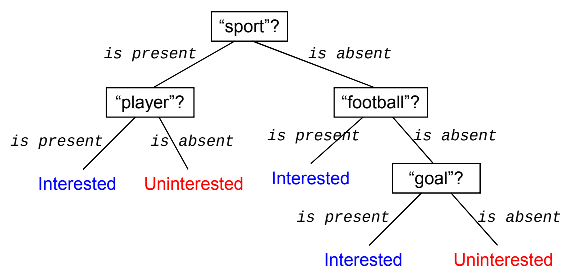
Decision tree – Representation (1)

- Each *internal node* represents an *attribute to be tested* by instances
- Each *branch* from a node corresponds to *a possible value of the attribute* associated with that node
- Each *leaf node* represents a *classification* (e.g., a class label)
- A learned DT classifies an instance by sorting it down the tree, from the root to some leaf node
  - → The classification associated with the leaf node is used for the instance

Decision tree – Representation (2)

- A DT represents a disjunction of conjunctions of constraints on the attribute values of instances
- Each path from the root to a leaf corresponds to a conjunction of attribute tests
- The tree itself is a disjunction of these conjunctions
- Examples
  - → Let's consider the two previous example DTs…

Which documents are of my interest?

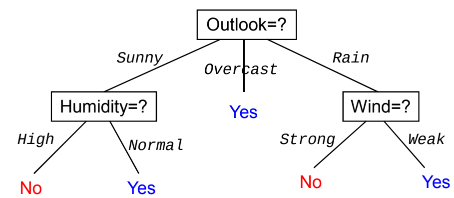


[("sport" is present) ∧ ("player" is present)] ∨

[("sport" is absent) ∧ ("football" is present)] ∨

[("sport" is absent) ∧ ("football" is absent) ∧ ("goal" is present)]

Does a person play tennis?



[(Outlook=Sunny) ∧ (Humidity=Normal)] ∨

(Outlook=Overcast) ∨

[(Outlook=Rain) ∧ (Wind=Weak)]

Decision tree learning – ID3 algorithm

ID3_alg(`Training_Set`, `Class_Labels`, `Attributes`)

Create a node `Root` for the tree

<u>If</u> all instances in `Training_Set` have the same class label c, <u>Return</u> the tree of the single-node `Root` associated with class label c

<u>If</u> the set `Attributes` is empty, <u>Return</u> the tree of the single-node `Root` associated with class label ≡ **Majority_Class_Label**(`Training_Set`)

A ← The attribute in `Attributes` that "best" classifies `Training_Set`

The test attribute for node `Root` ← A

<u>For each</u> possible value v of attribute A

  Add a new tree branch under `Root`, corresponding to the test: "value of attribute A is v"

  Compute $\texttt{Training\_Set}_v$ = {instance x | x ⊆ `Training_Set`, x_A=v}

  <u>If</u> ($\texttt{Training\_Set}_v$ is empty) <u>Then</u>

    Create a leaf node with class label ≡ **Majority_Class_Label**(`Training_Set`)

    Attach the leaf node to the new branch

  <u>Else</u> Attach to the new branch the sub-tree **ID3_alg**($\texttt{Training\_Set}_v$, `Class_Labels`, {`Attributes` \ A})

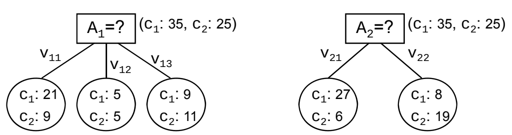
<u>Return</u> `Root`

ID3 algorithm – Intuitive idea

- Perform a greedy search through the space of possible DTs
- Construct (i.e., learn) a DT in a top-down fashion, starting from its root node
- At each node, the test attribute is the one (of the candidate attributes) that best classifies the training instances associated with the node
- A descendant (sub-tree) of the node is created for each possible value of the test attribute, and the training instances are sorted to the appropriate descendant node
- Every attribute can appear at most once along any path of the tree
- The tree growing process continues
  - Until the (learned) DT perfectly classifies the training instances, or
  - Until all the attributes have been used

Selection of the test attribute

- A very important task in DT learning: at each node, how to choose the test attribute?
- To select the attribute that is most useful for classifying the training instances associated with the node
- How to measure an attribute's capability of separating the training instances according to their target classification
  - → Use a statistical measure – *Information Gain*
- Example: A two-class (c_1, c_2) classification problem
  - → Which attribute, A_1 or A_2, should be chosen to be the test attribute?



Entropy

- A commonly used measure in the Information Theory field
- To measure the impurity (inhomogeneity) of a set of instances
- The entropy of a set S relative to a c-class classification

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

where p_i is the proportion of instances in S belonging to class i, and $0.\log_2 0=0$

- The entropy of a set S relative to a two-class classification

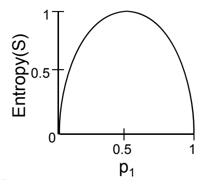
$$\texttt{Entropy(S)} = -p_1.\log_2 p_1 - p_2.\log_2 p_2$$

- Interpretation of entropy (in the Information Theory field)
  - → The entropy of S specifies the expected number of bits needed to encode class of a member randomly drawn out of S
    - Optical length code assigns $-\log_2 p$ bits to message having probability p
    - The expected number of bits needed to encode a class: $p.\log_2 p$

Entropy – Two-class example

- S contains 14 instances, where 9 belongs to class c_1 and 5 to class c_2
- The entropy of S relative to the two-class classification:

  `Entropy(S) = -(9/14).log`$_2$`(9/14)-`
  `(5/14).log`$_2$`(5/14) ≈ 0.94`



- Entropy =0, if all the instances belong to the same class (either c_1 or c_2)
  →Need 0 bit for encoding (no message need be sent)
- Entropy =1, if the set contains equal numbers of c_1 and c_2 instances
  → Need 1 bit per message for encoding (whether c_1 or c_2)
- Entropy = some value in (0,1), if the set contains unequal numbers of c_1 and c_2 instances
  → Need on average <1 bit per message for encoding

Information gain

- Information gain of an attribute relative to a set of instances is
  - the expected reduction in entropy
  - caused by partitioning the instances according to the attribute
- Information gain of attribute A relative to set S

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

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

- In the above formula, the second term is the expected value of the entropy after S is partitioned by the values of attribute A
- Interpretation of `Gain(S,A)`: The number of bits saved (reduced) for encoding class of a randomly drawn member of S, by knowing the value of attribute A

Training set - Example

Let's consider the following dataset (of a person) S:

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

[Mitchell, 1997]

Information gain – Example

- What is the information gain of attribute `Wind` relative to the training set S – `Gain(S,Wind)`?
- Attribute `Wind` have two possible values: `Weak` and `Strong`
- S = {9 positive and 5 negative instances}
- S_{weak} = {6 pos. and 2 neg. instances having `Wind=Weak`}
- S_{strong} = { 3 pos. and 3 neg. instances having `Wind=Strong`}

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

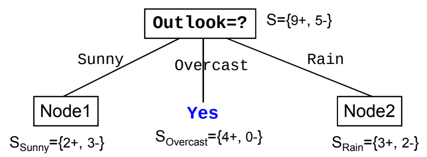
$$= Entropy(S) - (8/14).Entropy(S_{Weak}) - (6/14).Entropy(S_{Strong})$$

$$= 0.94 - (8/14).(0.81) - (6/14).(1) = 0.048$$

Decision tree learning – Example (1)

- At the root node, which attribute of {`Outlook`, `Temperature`, `Humidity`, `Wind`} should be the test attribute?
  - `Gain(S, `**`Outlook`**`) = ... = `**`0.246`** ← The highest IG value
  - `Gain(S, Temperature) = ... = 0.029`
  - `Gain(S, Humidity) = ... = 0.151`
  - `Gain(S, Wind) = ... = 0.048`

→So, `Outlook` is chosen as the test attribute for the root node!



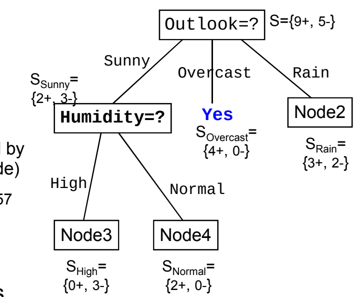
17

Decision tree learning – Example (2)

- At `Node1`, which attribute of {`Temperature`, `Humidity`, `Wind`} should be the test attribute?

  <u>Note</u>! Attribute `Outlook` is excluded, since it has been used by `Node1`'s parent (i.e., the root node)
  - Gain(S_{Sunny}, `Temperature`) =...= 0.57
  - Gain(S_{Sunny}, **`Humidity`**) = ... = **0.97**
  - Gain(S_{Sunny}, `Wind`) = ... = 0.019

→So, `Humidity` is chosen as the test attribute for `Node1`!



18

DT learning – Hypothesis space search (1)

- ID3 searches in a space of hypotheses (i.e., of possible DTs) for one that fits the training instances
- ID3 performs a simple-to-complex, hill-climbing search, beginning with the empty tree
- The hill-climbing search is guided by an evaluation metric – the information gain measure
- ID3 searches only one (rather than all possible) DT consistent with the training instances

19

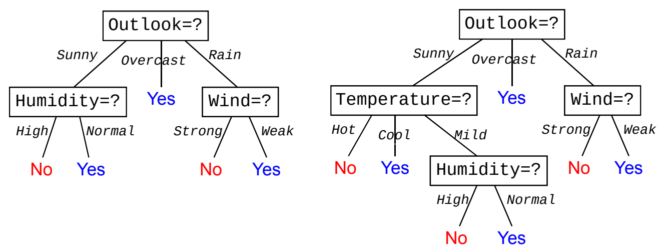
DT learning – Hypothesis space search (2)

- ID3 does not performs backtracking in its search
  - →Guaranteed to converge to a locally (but not the globally) optimal solution
  - →Once an attribute is selected as the test for a node, ID3 never backtracks to reconsider this choice
- At each step in the search, ID3 uses a statistical measure of all the instances (i.e., information gain) to refine its current hypothesis
  - →The resulting search is much less sensitive to errors in individual training instances

20

Inductive bias in DT learning (1)

- Both the two DTs below are consistent with the given training dataset
- So, which one is preferred (i.e., selected) by the ID3 algorithm?



Inductive bias in DT learning (2)

- Given a set of training instances, there may be many DTs consistent with these training instances
- So, which of these candidate DTs should be chosen?
- ID3 chooses the first acceptable DT it encounters in its simple-to-complex, hill-climbing search
  - →Recall that ID3 searches incompletely through the hypothesis space (i.e., without backtracking)
- ID3's search strategy
  - Select in favor of shorter trees over longer ones
  - Select trees that place the attributes with highest information gain closest to the root node

Issues in DT learning

- Over-fitting the training data
- Handling continuous-valued (i.e., real-valued) attributes
- Choosing appropriate measures for attribute selection
- Handling training data with missing attribute values
- Handling attributes with differing costs

→ An extension of the ID3 algorithm with the above mentioned issues resolved results in the C4.5 algorithm