Artificial Intelligence

Fiona Yan Liu

Department of Computing

The Hong Kong Polytechnic University

Learning Outcomes of Intelligent Agents

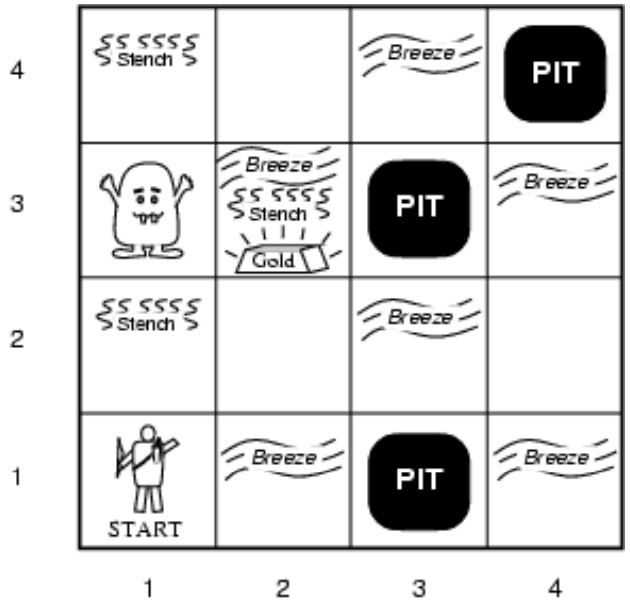
- Search agents
  - Uninformed search
  - Informed search
- Local search
  - Hill climbing search
  - Simulated annealing search
- Constraint satisfaction problem
  - State is defined by variables X_i with values from domain D_i
  - Goal test is a set of constraints specifying allowable combinations of values for subsets of variables
- Games
  - Views any multiagent environment as a game, provided that the impact of each agent on the other is significant

Knowledge based Agents

- Knowledge base (KB)
  - A set of sentences
    - Each sentence is expressed in a language called a knowledge representation language
    - Represent some assertion about the world
  - Inference
    - Derive new sentences from old
    - Tell: add new sentences to the knowledge base
    - Ask: query what is known
- Knowledge-based Agent
  - Tell the knowledge base what it perceives
  - Ask the knowledge base what action is should perform
  - Tell the knowledge base which action was chosen
  - The agent executes the action

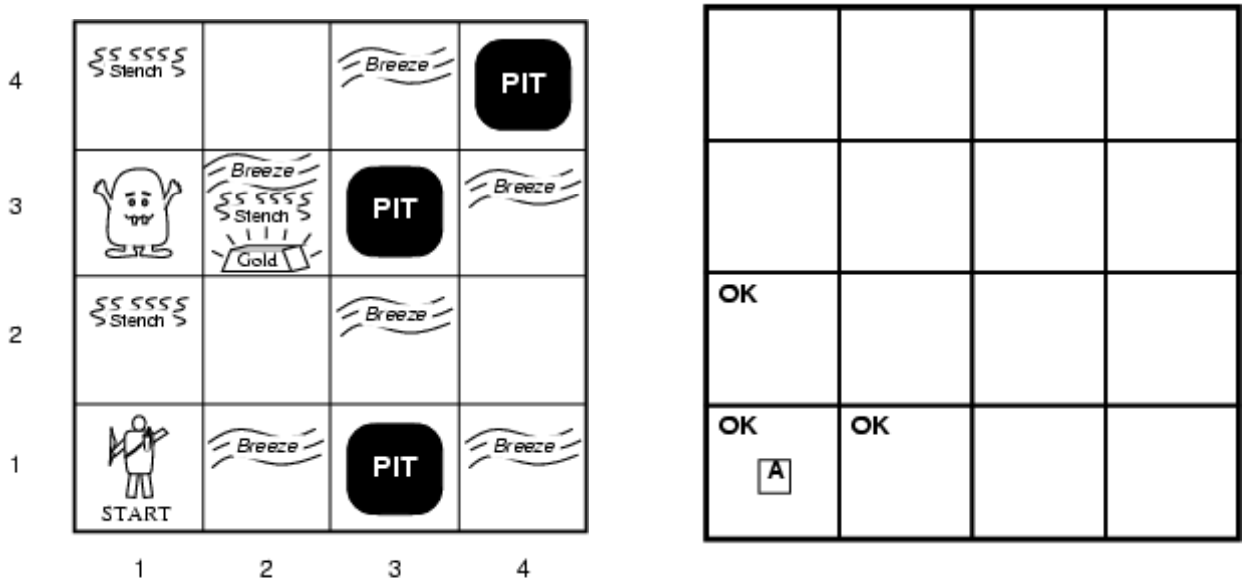
Wumpus World

- Performance measure
  - gold +1000, death -1000
  - -1 per step, -10 for using the arrow
- Environment
  - Squares adjacent to wumpus are smelly
  - Squares adjacent to pit are breezy
  - Glitter iff gold is in the same square
  - Shooting kills wumpus if you are facing it
  - Shooting uses up the only arrow
  - Grabbing picks up gold if in same square
  - Releasing drops the gold in same square
- Sensors
  - Stench, Breeze, Glitter, Bump, Scream
- Actuators
  - Left turn, Right turn, Forward, Grab, Release, Shoot



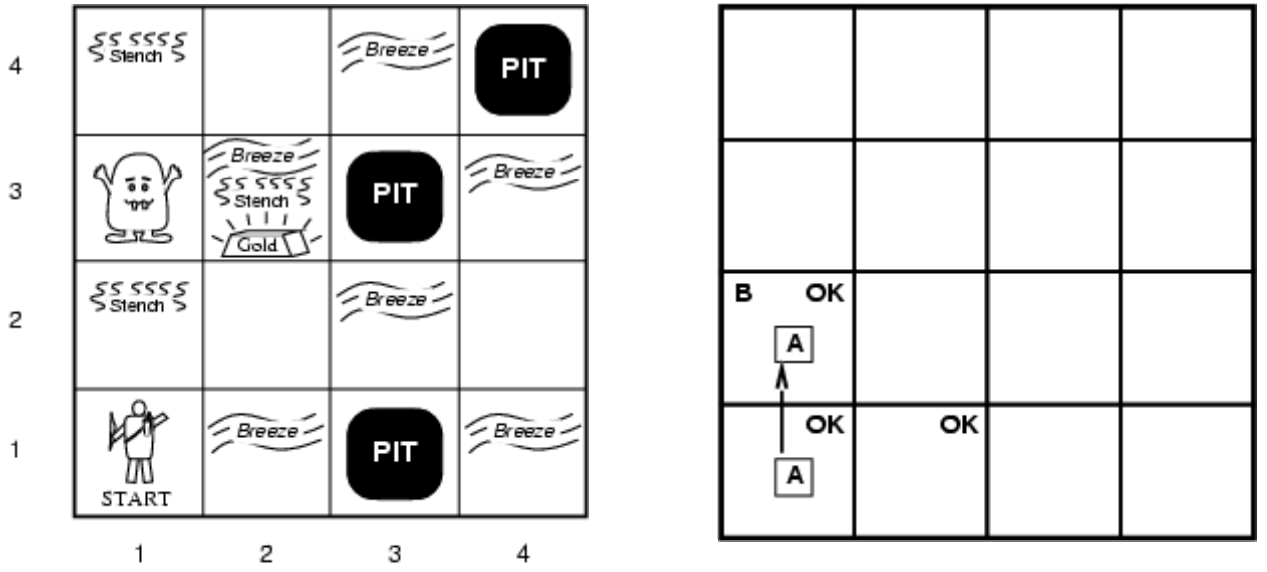
Exploring a Wumpus world

- Perceive: [Stench, Breeze, Glitter, Bump, Scream]
- [1,1]: [None, none, none, none, none]



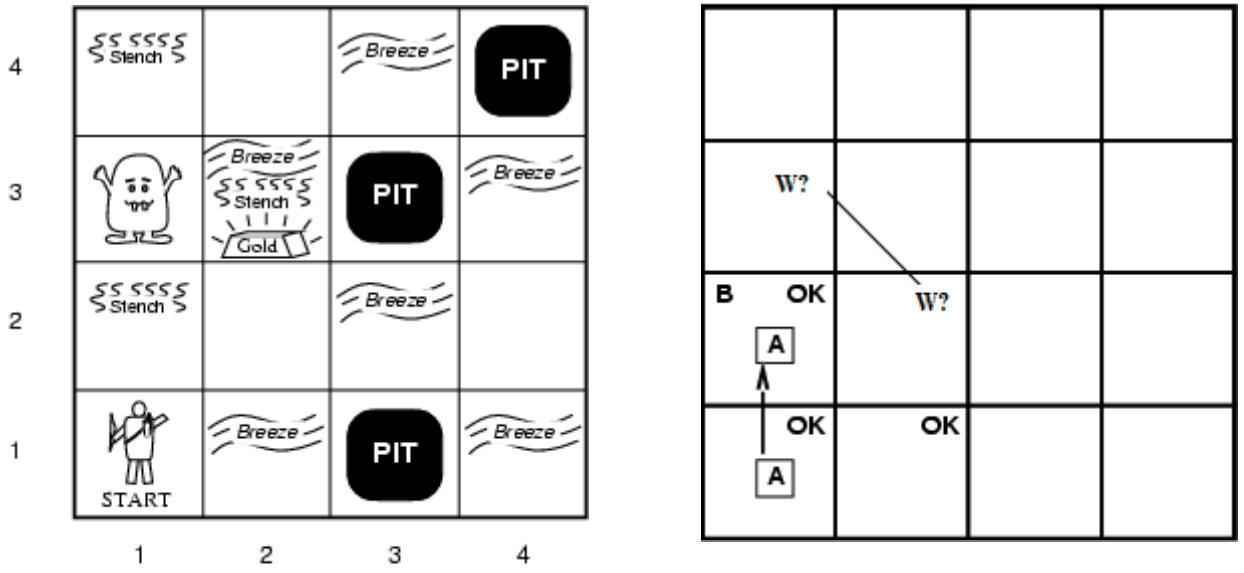
Exploring a Wumpus world

- Perceive: [Stench, Breeze, Glitter, Bump, Scream]
- [1,1]: [None, none, none, none, none]
- [1,2]: [Stench, none, none, none, none]



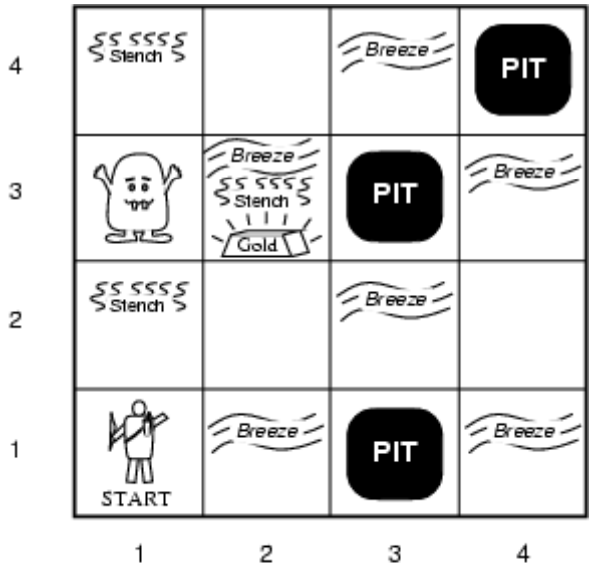
Exploring a Wumpus world

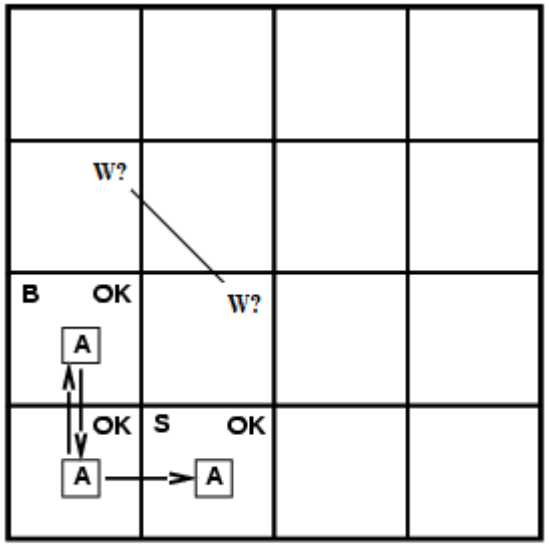
- Perceive: [Stench, Breeze, Glitter, Bump, Scream]
- [1,1]: [None, none, none, none, none]
- [1,2]: [Stench, none, none, none, none]



Exploring a Wumpus world

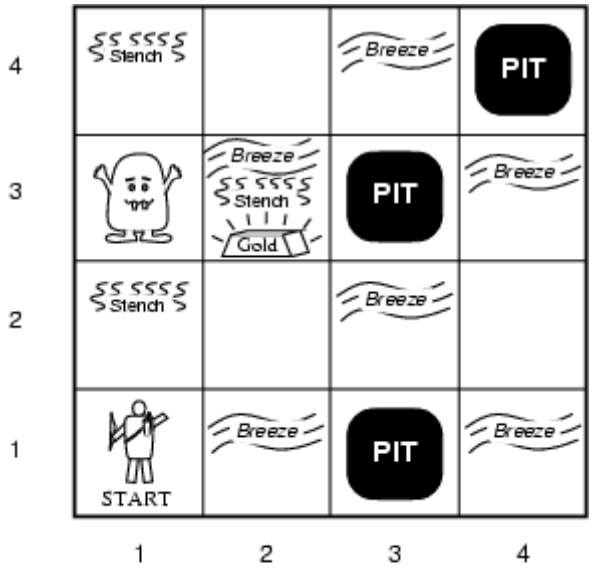
- Perceive: [Stench, Breeze, Glitter, Bump, Scream]
- [1,1]: [None, none, none, none, none]
- [1,2]: [Stench, none, none, none, none]

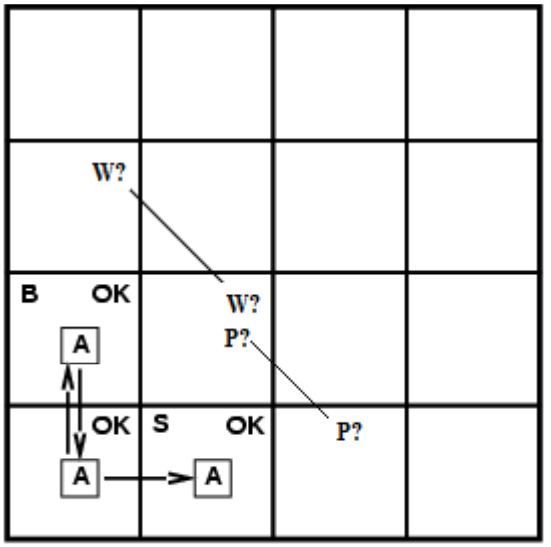




Exploring a Wumpus world

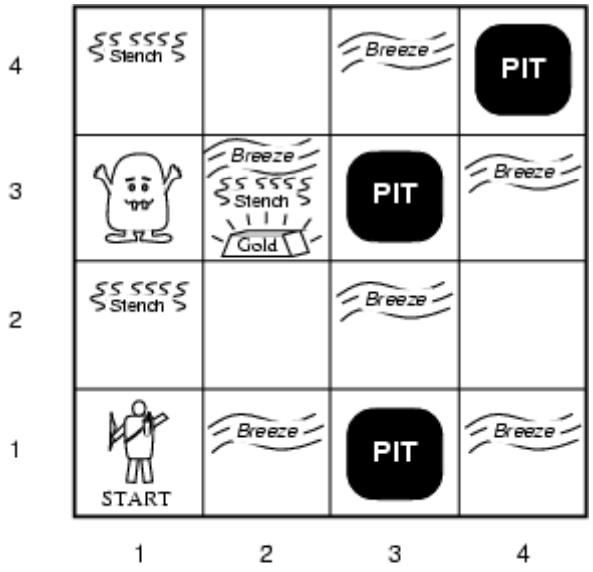
- Perceive: [Stench, Breeze, Glitter, Bump, Scream]
- [1,1]: [None, none, none, none, none]
- [1,2]: [Stench, none, none, none, none]
- [2,1]: [None, breeze, none, none, none]

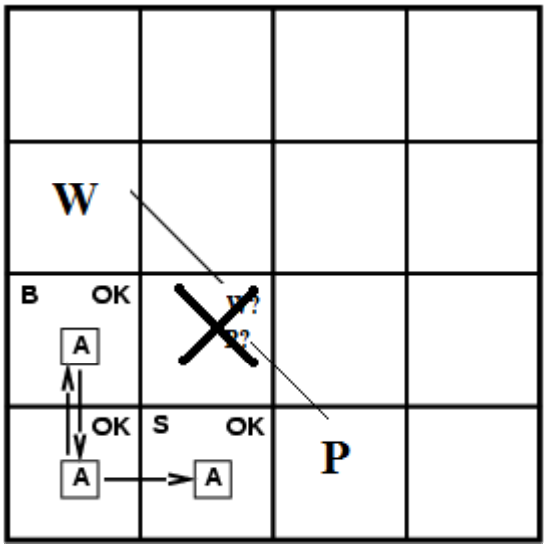




Exploring a Wumpus world

- Perceive: [Stench, Breeze, Glitter, Bump, Scream]
- [1,1]: [None, none, none, none, none]
- [1,2]: [Stench, none, none, none, none]
- [2,1]: [None, breeze, none, none, none]





Representation Language

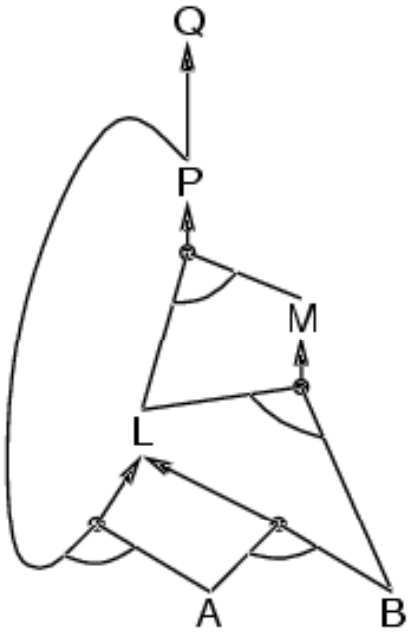
- Knowledge
  - Is contained in agents
  - In the form of sentences
  - In a knowledge representation language
  - That are stored in a knowledge base
- A presentation language is defined by its syntax
  - Specify the structure of sentences
  - And its semantics, which defines the truth of each sentence
- The relationship of entailment between sentences
  - A sentence α entails another sentence β if β is true in all worlds where α is true

    $\alpha \models \beta$ if and only if $M(\alpha) \subseteq M(\beta)$
  - Example:
    - The sentence x = 0 entails the sentence xy=0

Forward Chaining

- Idea: fire any rule whose premises are satisfied in the *KB*,
  - add its conclusion to the *KB*, until query is found

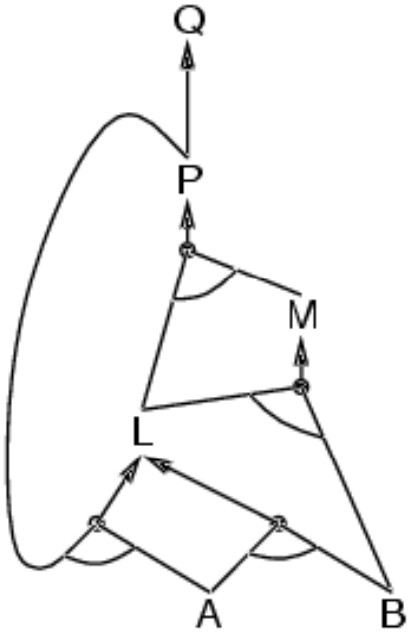
$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward Chaining

- Idea: work backwards from the query q:
- Avoid loops: check if new subgoal is already on the goal stack
- Avoid repeated work: check if new subgoal has already been proved true, or has already failed

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Forward Channing and Backward Channing

- Intend to query Q2
  - A
  - B
  - C
  - A^B => D
  - B^C => E
  - A^C => F
  - A^F => G
  - D^F => K
  - G^K => Q1
  - E => H
  - H^C =>Q2

Decision Tree

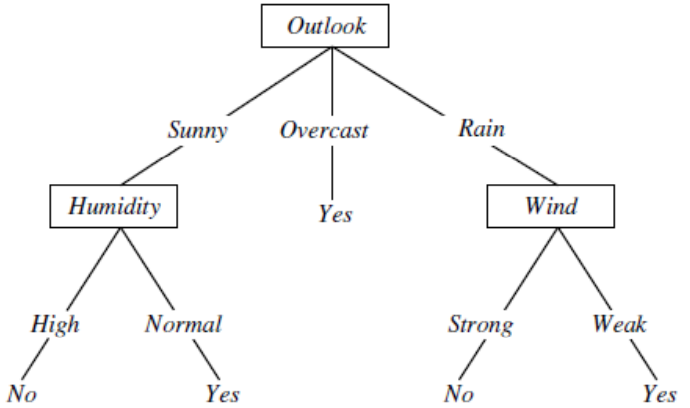
- Given a training set

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

One Example of Decision Tree

- Each internodes test an attribute
- Each branch corresponds a attribute value
- Each leave node assigns a classification

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



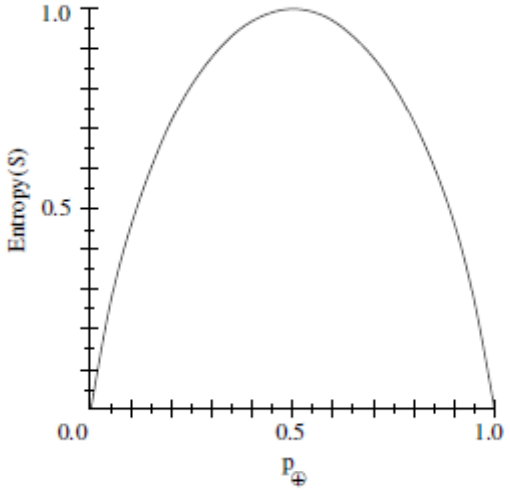
Choosing an Attribute

- *Entropy (s)*
  - Expected number of bits needed to encode class + or – of randomly drawn member of *S*

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

- *Gain (S,A)*
  - Expected reduction in entropy due to sorting on *A*

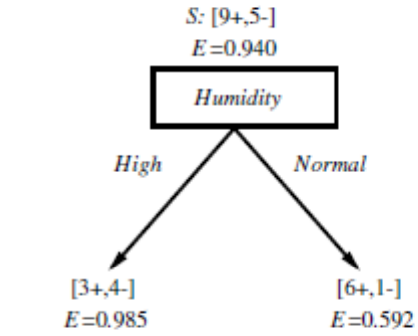
$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v$$



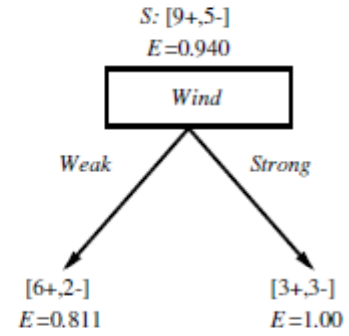
Select Next Attribute

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

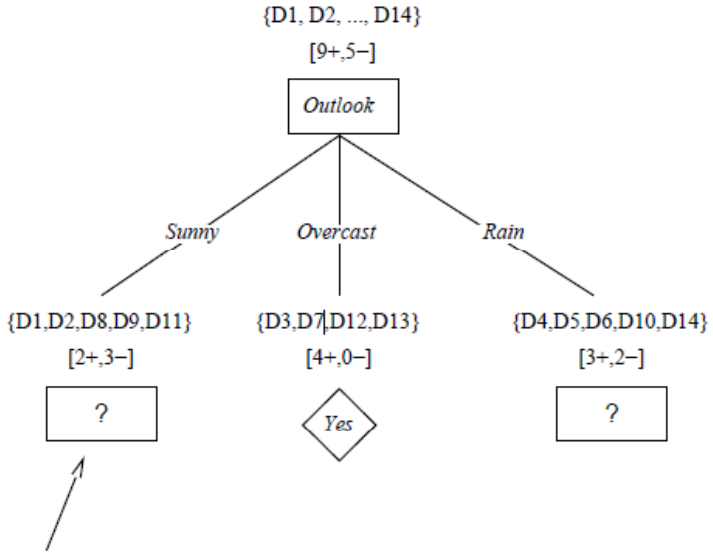
Which attribute is the best classifier?



Gain (S, Humidity)
= .940 - (7/14).985 - (7/14).592
= .151



Gain (S, Wind)
= .940 - (8/14).811 - (6/14)1.0
= .048



Which attribute should be tested here?

S_{sunny} = {D1,D2,D8,D9,D11}

Gain (S_{sunny}, *Humidity*) = .970 − (3/5) 0.0 − (2/5) 0.0 = .970

Gain (S_{sunny}, *Temperature*) = .970 − (2/5) 0.0 − (2/5) 1.0 − (1/5) 0.0 = .570

Gain (S_{sunny}, *Wind*) = .970 − (2/5) 1.0 − (3/5) .918 = .019