ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

23AD2001O

BAYESIAN BELIEF NETWORKS, DECISION TREES, KNN

CO3

SESSION 19

AIM OF THE SESSION

To familiarize students with the basic concept of Bayesian Networks,Decision Trees and KNN

INSTRUCTIONAL OBJECTIVES

This Session is designed to:

1. Bayesian belief network
2. Decision Trees
3. What is KNN?

LEARNING OUTCOMES

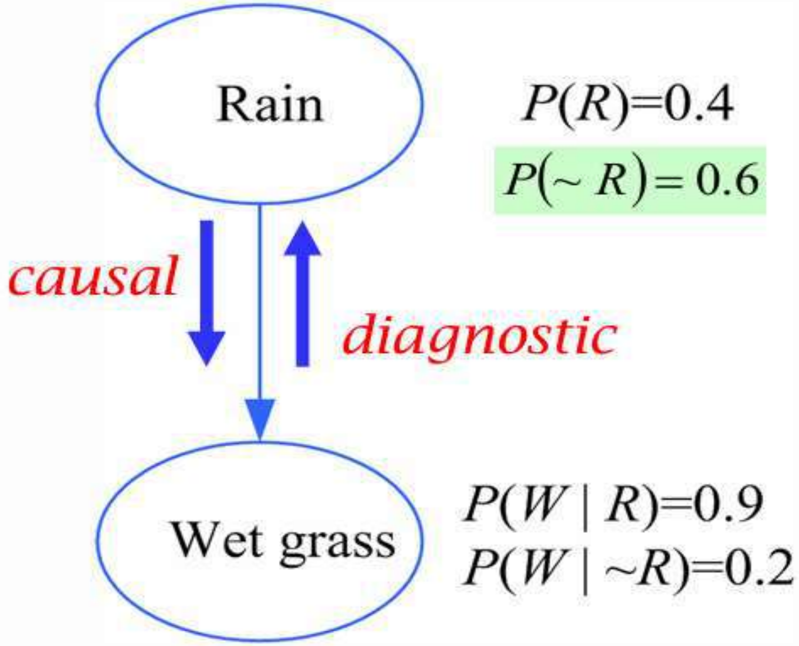
At the end of this session, you should be able to:

1. Define Bayesian Network
2. Describe Decision Tree
3. Apply KNN

Bayesian Networks

- Bayesian Networks: also known as graphical models, probabilistic networks
  - Nodes are hypotheses (random variables) and the probability corresponds to our belief in the truth of the hypothesis.
  - Arcs are direct influences between hypotheses.
  - The structure is represented as a directed acyclic graph (DAG).
  - The parameters are the conditional probabilities in the arcs.

Causes and Bayes' Rule



Diagnostic inference:
Knowing that the grass is wet, what is the probability that rain is the cause?

$$
\begin{aligned}
P(R \mid W) &= \frac{P(W \mid R)P(R)}{P(W)} \\
&= \frac{P(W \mid R)P(R)}{P(W \mid R)P(R) + P(W \mid \sim R)P(\sim R)} \\
&= \frac{0.9 \times 0.4}{0.9 \times 0.4 + 0.2 \times 0.6} \\
&= 0.75
\end{aligned}
$$

Conditional Independence

- *X* and *Y* are independent if

$$P(X,Y)=P(X)P(Y)$$

- *X* and *Y* are conditionally independent given *Z* if

$$P(X,Y|Z)=P(X|Z)P(Y|Z)$$

or $$P(X|Y,Z)=P(X|Z)$$

- Three canonical cases for conditional independence:
  - Head-to-tail connection
  - Tail-to-tail connection
  - Head-to-head connection

Bayesian Networks

$$P(C,S,R,W) = P(C)P(S|C)P(R|C)P(W|S,R)$$

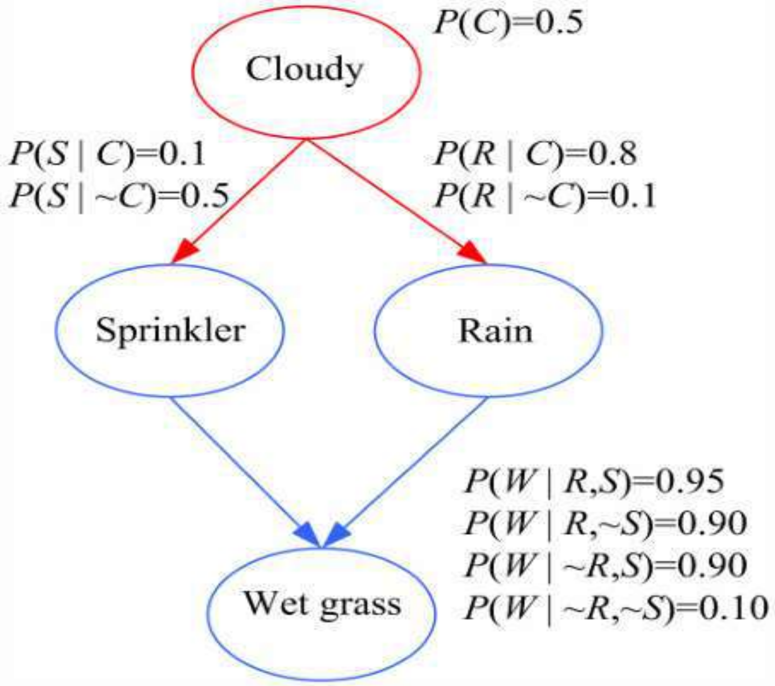
$$P(X_1,\ldots,X_d) = \prod_{i=1}^{d} P(X_i \mid \text{parents}(X_i))$$

Belief propagation (Pearl, 1988)

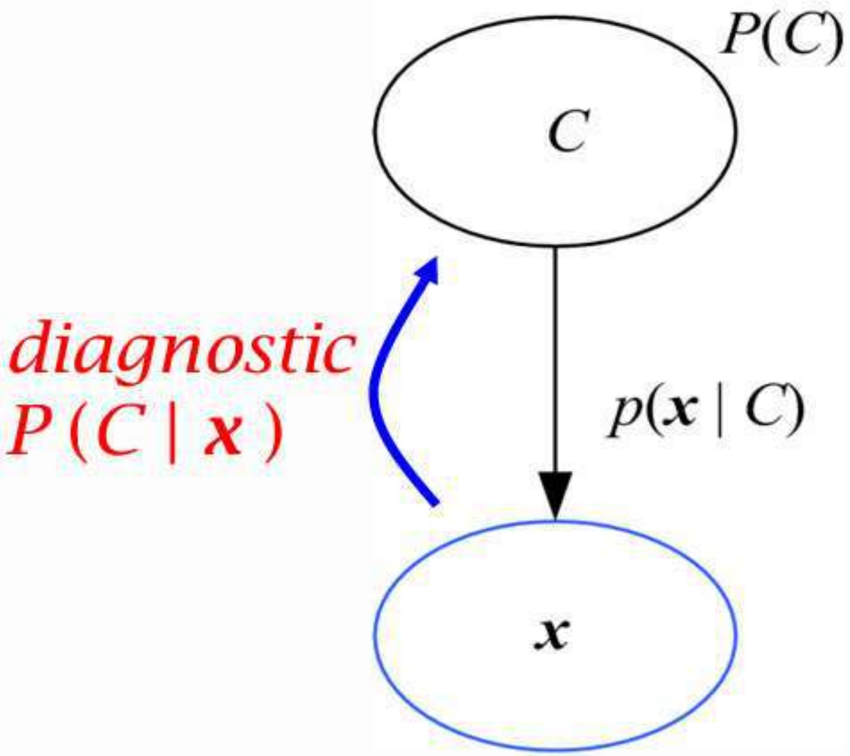
use for inference when the network is a tree

Junction trees (Lauritzen and Spiegelhalter, 1988)

convert a given directed acyclic graph to a tree



Bayesian Networks: Classification



Bayes' rule inverts the arc:

$$P(C \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C)P(C)}{p(\mathbf{x})}$$

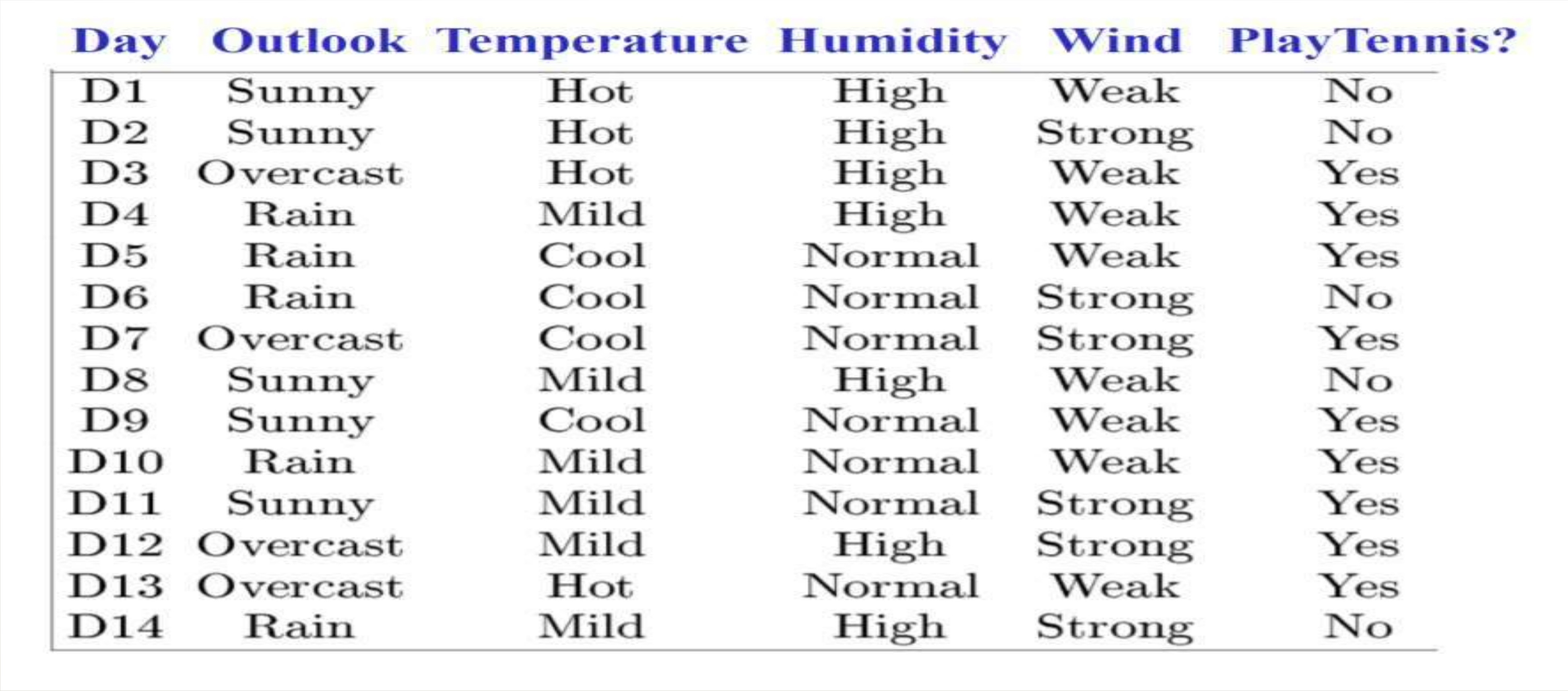
DECISION TREE LEARNING

- **Decision tree learning** is a method for approximating discrete-valued target functions.
- The learned function is represented by **a decision tree**.
  - A learned decision tree can also be re-represented as a set of if-then rules.
- Decision tree learning is one of the most widely used and practical methods for inductive inference.
- It is robust to noisy data and capable of learning disjunctive expressions.

DECISION TREE LEARNING

- Decision tree learning method searches a completely expressive hypothesis.
- Avoids the difficulties of restricted hypothesis spaces.
- Its inductive bias is a preference for small trees over large trees.
- The decision tree algorithms such as ID3, C4.5 – (CART) are very popular inductive inference algorithms, and they are successfully applied to many leaning tasks.

TRAINING DATASET(PLAY TENNIS)



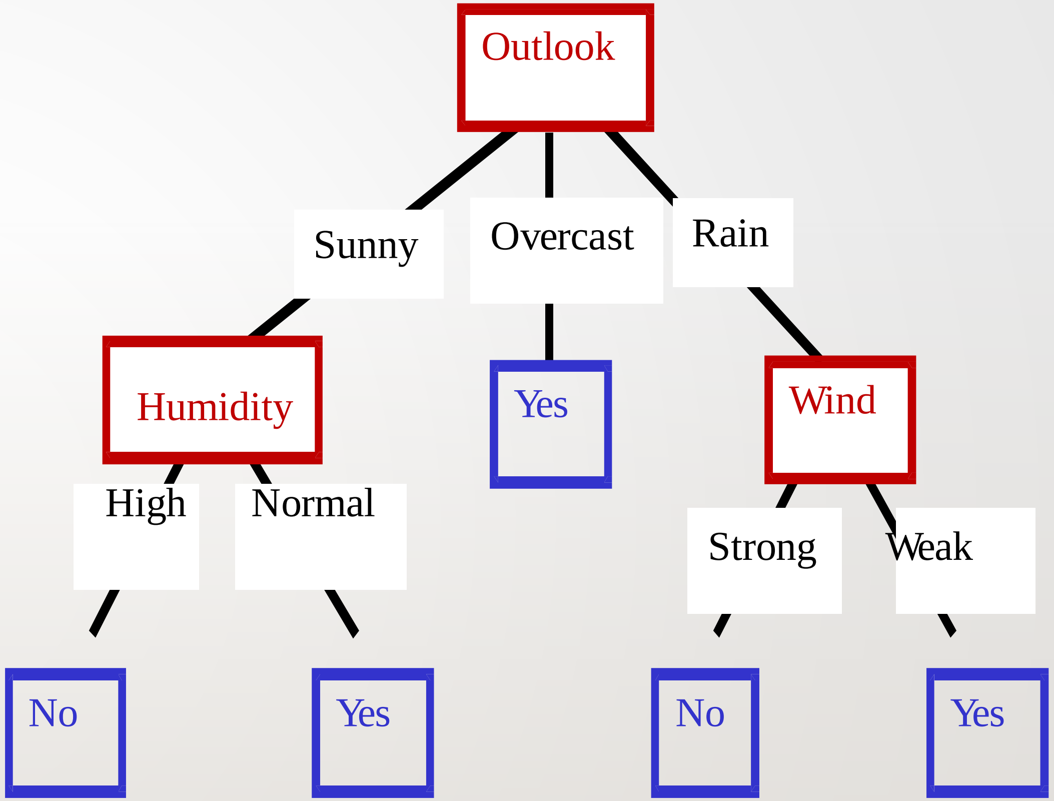
Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	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

DECISION TREE

Decision trees represent a disjunction of conjunctions of constraints on the attribute values of instances.

Each path from the tree root to a leaf corresponds to a conjunction of attribute tests, and

The tree itself is a disjunction of these conjunctions.



$$(\text{Outlook} = \text{Sunny} \wedge \text{Humidity} = \text{Normal})$$

$$(\text{Outlook} = \text{Overcast})$$

$$(\text{Outlook} = \text{Rain} \wedge \text{Wind} = \text{Weak})$$

DECISION TREE

- Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance.
- Each node in the tree specifies a test of some attribute of the instance.
- Each branch descending from a node corresponds to one of the possible values for the attribute.
- Each leaf node assigns a classification.
- The instance
- (Outlook=Sunny, Temperature=Hot, Humidity=High, Wind=Strong) is classified as a negative instance.

WHEN TO CONSIDER DECISION TREES

- Instances are represented by attribute-value pairs.
- Fixed set of attributes, and the attributes take a small number of disjoint possible values.
- The target function has discrete output values.
- Decision tree learning is appropriate for a boolean classification, but it easily extends to learning functions with more than two possible output values.
- Disjunctive descriptions may be required.
- decision trees naturally represent disjunctive expressions.

- The training data may contain errors.
  - Decision tree learning methods are robust to errors, both errors in classifications of the training examples and errors in the attribute values that describe these examples.
- The training data may contain missing attribute values.
  - Decision tree methods can be used even when some training examples have unknown values.
- Decision tree learning has been applied to problems such as learning to classify
  - medical patients by their disease,
  - equipment malfunctions by their cause, and
  - loan applicants by their likelihood of defaulting on payments.

ID3 ALGORITHM

- ID3*(Examples,TargetAttribute,Attributes*)
- Create a *Root* node for the tree
- If all *Examples* are positive, Return the single-node tree *Root*, with label = +
- If all *Examples* are negative, Return the single-node tree *Root*, with label = -
- If *Attributes* is empty, Return the single-node tree Root, with label = most common value of *TargetAttribute* in *Examples*
- Otherwise Begin
- A □ the attribute from *Attributes* that best classifies *Examples*
- The decision attribute for *Root* □A
- For each possible value, vi, of A,
- Add a new tree branch below *Root*, corresponding to the test A = vi
- Let $Examples_{vi}$ be the subset of *Examples* that have value vi for A
- If $Examples_{vi}$ is empty
- Then below this new branch add a leaf node with label = most common value of *TargetAttribute* in *Examples*
- Else below this new branch add the subtree ID3($Examples_{vi}$, *TargetAttribute*, *Attributes* – {A})
- End
- Return *Root*

TOP-DOWN INDUCTION OF DECISION TREES -- ID3

- A ← the **“best”** decision attribute for next *node*
- Assign A as decision attribute for *node*
- For each value of A create new descendant *node*
- Sort training examples to leaf node according to the attribute value of the branch
- If all training examples are perfectly classified (same value of target attribute) STOP,
- else iterate over new leaf nodes.

INFORMATION GAIN

- ***entropy*** is a measure of the impurity in a collection of training examples
- ***information gain*** is a measure of the effectiveness of an attribute in classifying the training data.
- ***information gain*** measures the expected reduction in entropy by partitioning the examples according to an attribute.

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

- S – a collection of examples
- A – an attribute
- Values(A) – possible values of attribute A
- S_v – the subset of S for which attribute A has value v

ENTROPY

- Given a collection S, containing positive and negative examples of some target concept, the ***entropy of S*** relative to this boolean classification is:

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

- **S** is a sample of training examples
- **p_+** is the proportion of positive examples
- **p_-** is the proportion of negative examples

ID3 - TRAINING EXAMPLES

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

ENTROPY

Entropy([9+,5-] = – (9/14) $\log_2$(9/14) – (5/14) $\log_2$(5/14) = 0.940

Entropy([12+,4-] = – (12/16) $\log_2$(12/16) – (4/16) $\log_2$(4/16) = 0.811

Entropy([12+,5-] = – (12/17) $\log_2$(12/17) – (5/17) $\log_2$(5/17) = 0.874

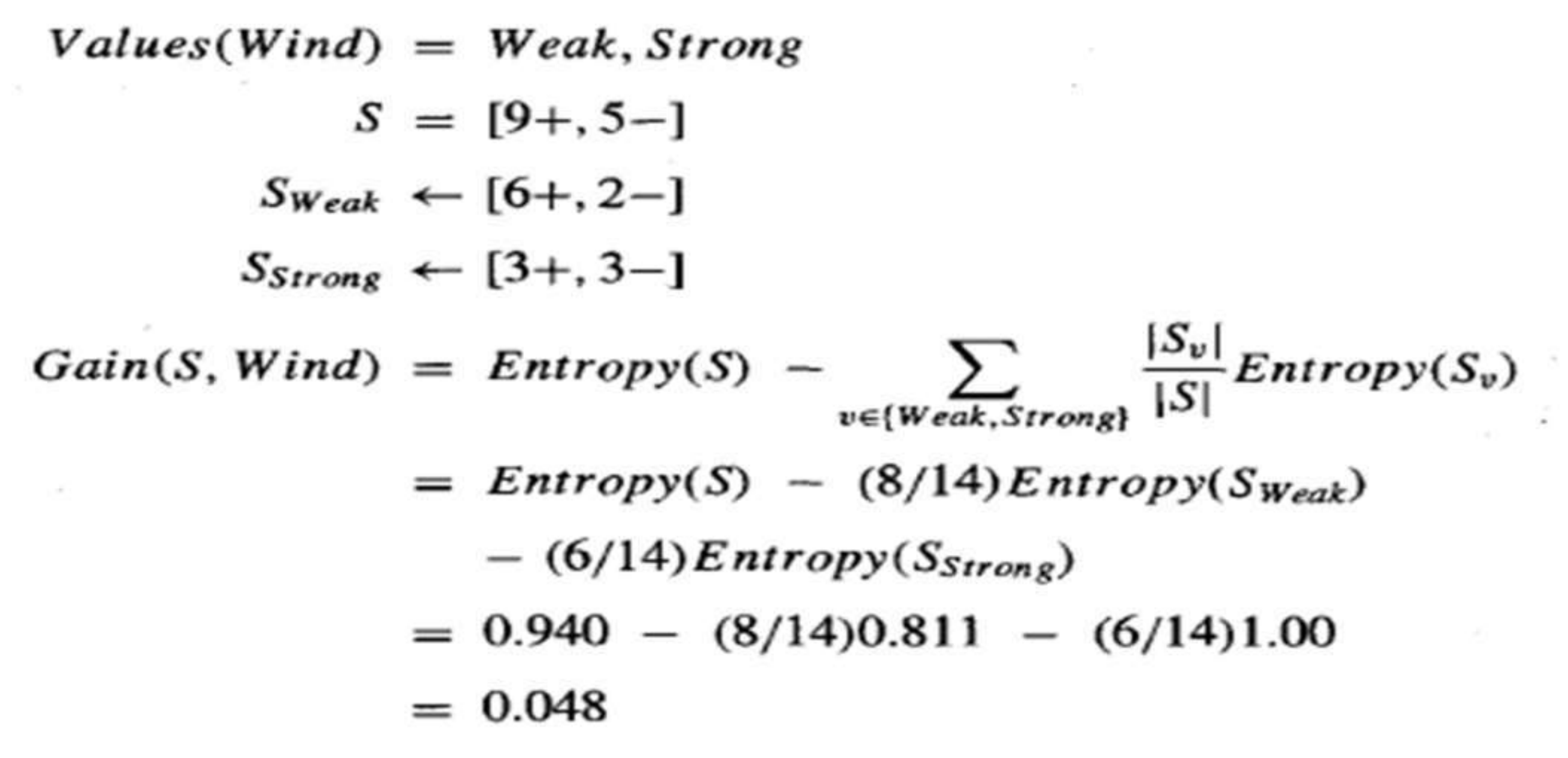
Entropy([8+,8-] = – (8/16) $\log_2$(8/16) – (8/16) $\log_2$(8/16) = 1.0

Entropy([8+,0-] = – (8/8) $\log_2$(8/8) – (0/8) $\log_2$(0/8) = 0.0

Entropy([0+,8-] = – (0/8) $\log_2$(0/8) - (8/8) $\log_2$(8/8) = 0.0

- It is assumed that $\log_2$(0) is 0

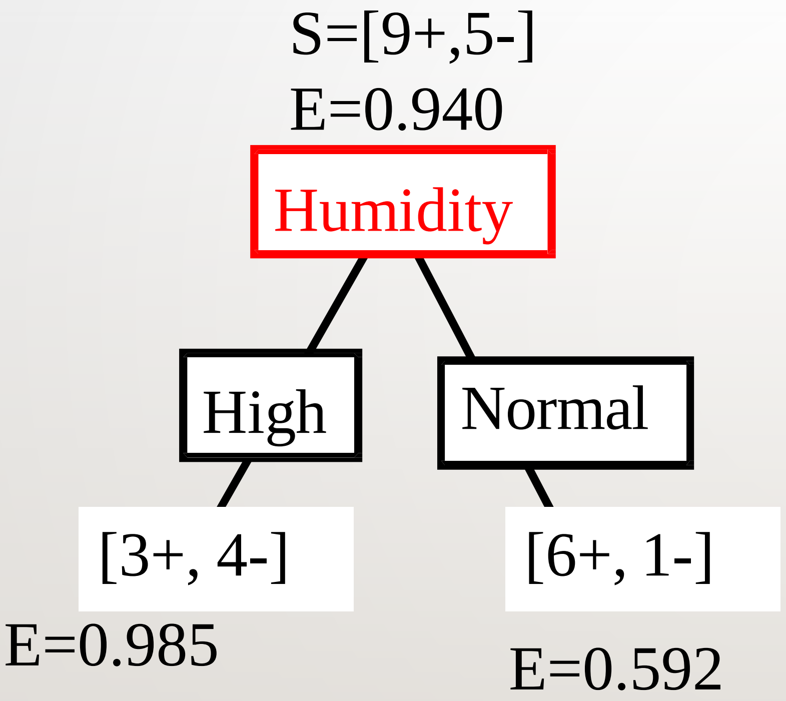
INFORMATION GAIN

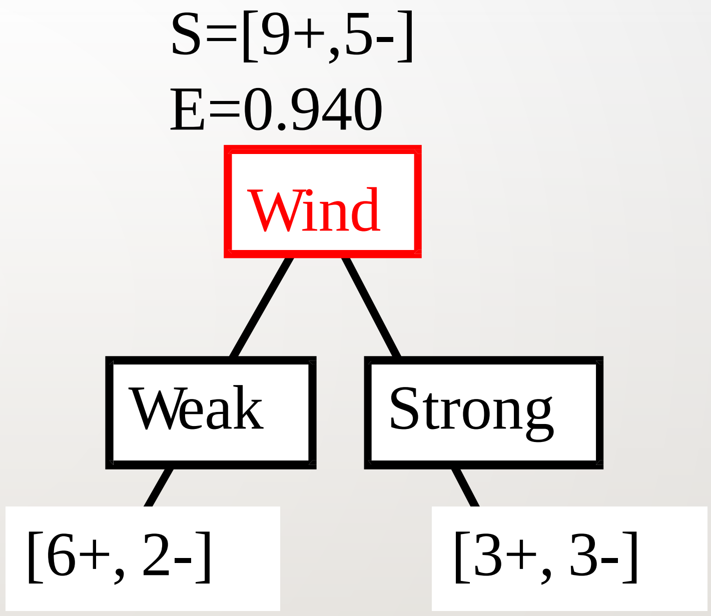


$$\begin{aligned} Values(Wind) &= Weak, Strong \\ S &= [9+, 5-] \\ S_{Weak} &\leftarrow [6+, 2-] \\ S_{Strong} &\leftarrow [3+, 3-] \\ Gain(S, Wind) &= Entropy(S) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S|} Entropy(S_v) \\ &= Entropy(S) - (8/14) Entropy(S_{Weak}) \\ &\quad - (6/14) Entropy(S_{Strong}) \\ &= 0.940 - (8/14)0.811 - (6/14)1.00 \\ &= 0.048 \end{aligned}$$

ID3 – SELECTING NEXT ATTRIBUTE

$$\text{Entropy}([9+,5-] = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.940$$





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**

ID3 - S_{SUNNY}

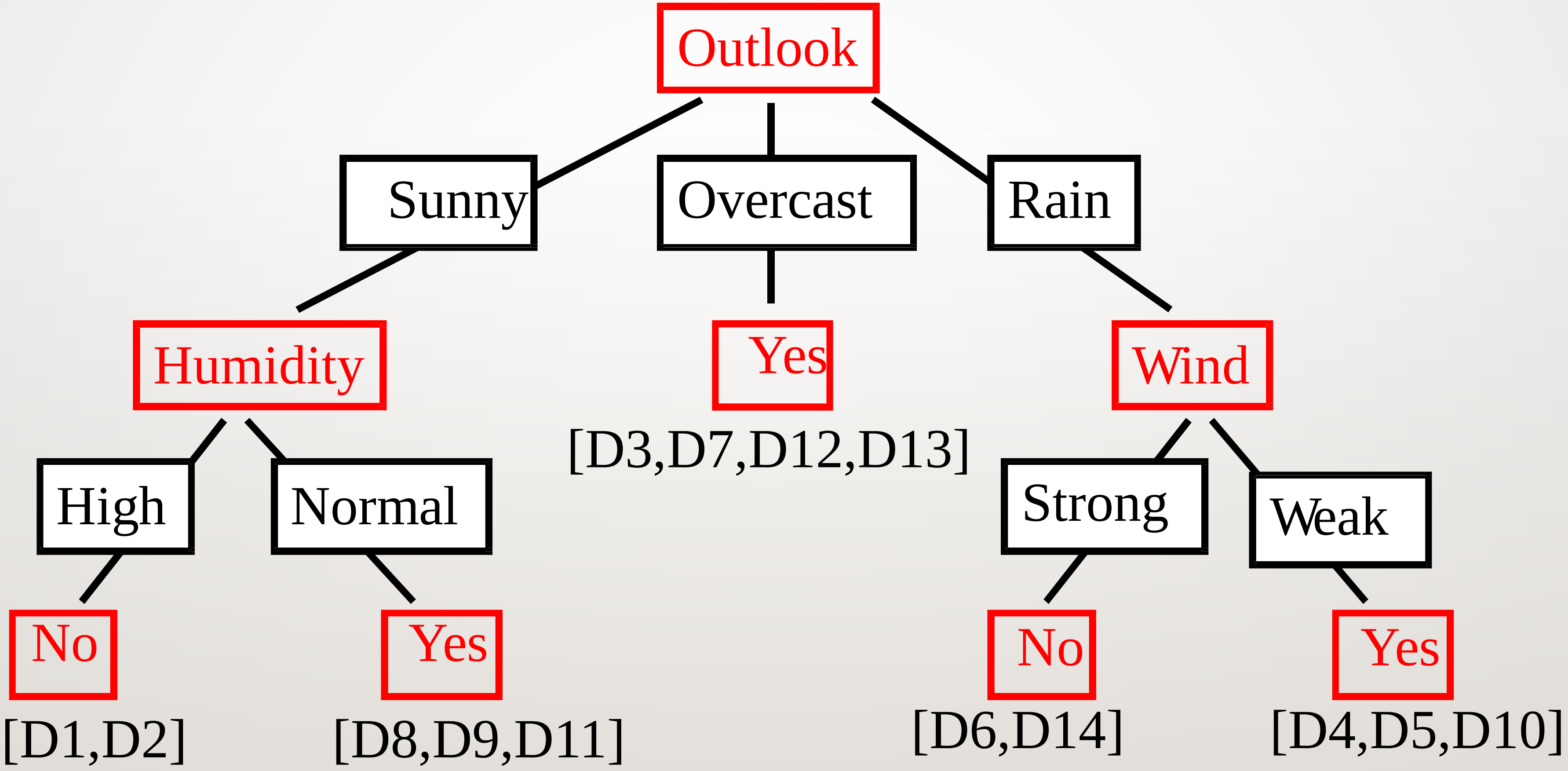
Gain(S_{sunny} , Humidity) = 0.970-(3/5)0.0 – 2/5(0.0) = **0.970**

Gain(S_{sunny} , Temp.) = 0.970-(2/5)0.0 –2/5(1.0)-(1/5)0.0 = 0.570

Gain(S_{sunny} , Wind) = 0.970= -(2/5)1.0 – 3/5(0.918) = 0.019

So, Hummudity will be selected

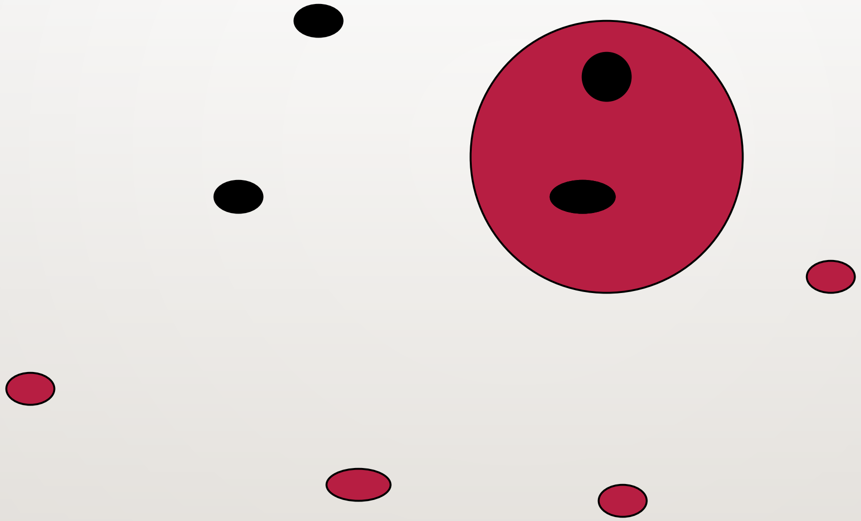
ID3 - RESULT


Outlook
Sunny
Overcast
Rain
Humidity
Yes
[D3,D7,D12,D13]
Wind
High
Normal
Strong
Weak
No
[D1,D2]
Yes
[D8,D9,D11]
No
[D6,D14]
Yes
[D4,D5,D10]

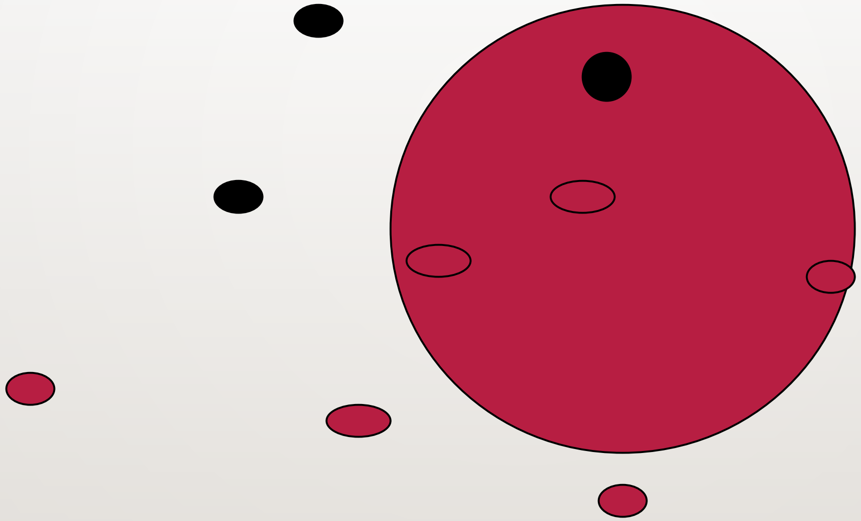
K-NEAREST NEIGHBOR

- Features
  - All instances correspond to points in an n-dimensional Euclidean space
  - Classification is delayed till a new instance arrives
  - Classification done by comparing feature vectors of the different points
  - Target function may be discrete or real-valued

1-NEAREST NEIGHBOR



3-NEAREST NEIGHBOR



K-NEAREST NEIGHBOR

- An arbitrary instance is represented by $(a_1(x), a_2(x), a_3(x),.., a_n(x))$
  - $a_i(x)$ denotes features
- Euclidean distance between two instances

$d(x_i, x_j)$=sqrt (sum for r=1 to n $(a_r(x_i) - a_r(x_j))^2$)

- Continuous valued target function
  - mean value of the k nearest training examples

DISTANCE-WEIGHTED NEAREST NEIGHBOR ALGORITHM

- Assign weights to the neighbors based on their ‘distance’ from the query point
  - Weight ‘may’ be inverse square of the distances

→ All training points may influence a particular instance

  - Shepard’s method

MERITS AND DEMERITS

+Highly effective inductive inference method for noisy training data and complex target functions

+Target function for a whole space may be described as a combination of less complex local approximations

+Learning is very simple

- Classification is time consuming

Self-Assessment Questions

1. In a Bayesian belief network, what do nodes represent?
   a) Data instances
   b) Decision boundaries
   c) Random variables
   d) Class labels

2. In decision trees, what is used to split the data at each node?
   a) Gini impurity
   b) Entropy
   c) Information gain
   d) Chi-square test

TERMINAL QUESTIONS

1. Define Bayesian Belief Networks (BBNs) and explain their significance in probabilistic reasoning.
2. What are the components of a Bayesian Belief Network?
3. Discuss the difference between conditional probability and joint probability in the context of Bayesian Belief Networks.
4. Define Decision Trees and explain their structure.
5. What are the advantages of using Decision Trees for classification and regression?
6. Explain the working principle of the k-Nearest Neighbors algorithm.
7. What is the importance of choosing the right value for 'k' in KNN?

REFERENCE BOOKS AND WEB LINKS

Text Books:

1) Russel and Norvig, ‘Artificial Intelligence’, third edition, Pearson Education, PHI, (2015)
2) Tom M. Mitchell, *Machine Learning, WCB McGraw-Hill, Boston, 1997*.
3) Elaine Rich & Kevin Knight, ‘Artificial Intelligence’, 3nd Edition, Tata McGraw Hill Edition, Reprint (2008)

Web links:

1. https://www.geeksforgeeks.org/agents-artificial-intelligence/
2. https://www.simplilearn.com/what-is-intelligent-agent-in-ai-types-function-article
3. https://www.javatpoint.com/turing-test-in-ai
4. https://www.geeksforgeeks.org/turing-test-artificial-intelligence/
5. https://www.section.io/engineering-education/turing-test-in-ai/

THANK YOU