

COMP4616 – Fundamentals of Machine Learning

ASST. PROF. TUĞBA ERKOÇ

SPRING 2025

WEEK 2: DECISION TREES



Decision Tree

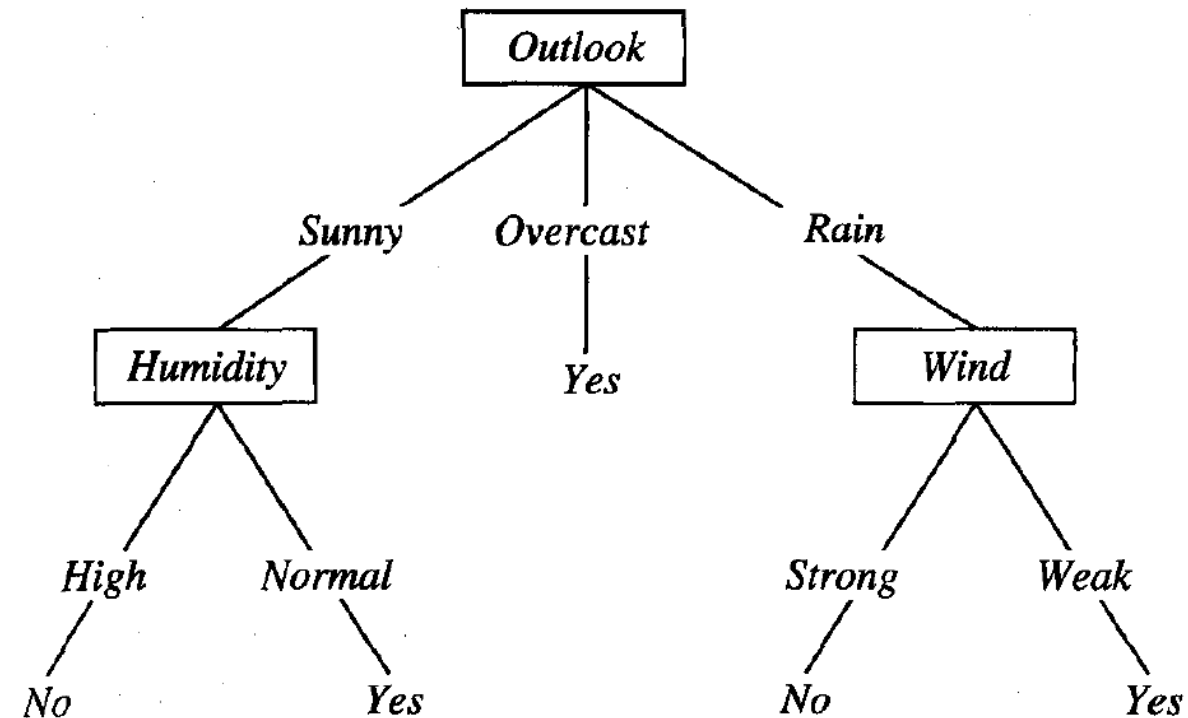
- A **decision tree** is a representation of a function that maps a vector of attribute values to a single output value.
- We call this output value a **decision**.
- A decision tree is a hierarchical model for **supervised learning** by which the local region is identified in a sequence of recursive splits in a smaller number of steps.
- A decision tree reaches its decision by performing a sequence of tests, starting at the root and following the appropriate branch until a leaf is reached.
- Structure of the decision tree is as follows:
 - Each internal node in the tree corresponds to a test of the value of one of the input attributes
 - The branches from the node are labeled with the possible values of the attribute
 - The leaf nodes specify what value is to be returned by the function

Decision Tree

- A decision tree reaches its decision by performing a sequence of tests, starting at the root and following the appropriate branch until a leaf is reached.

A decision tree for the concept **PlayTennis**:

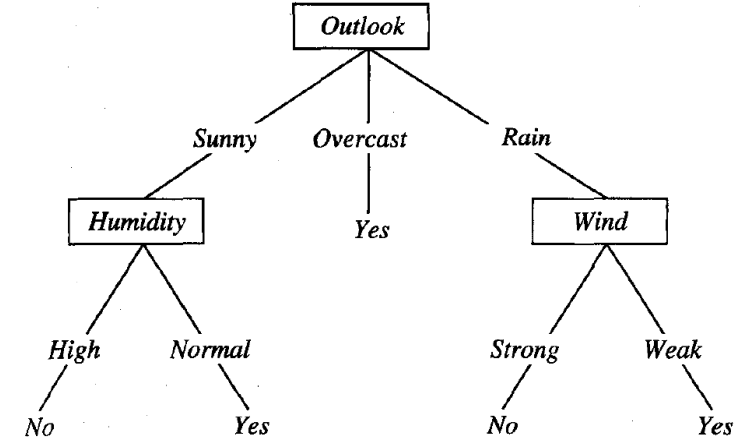
- This tree classifies Saturday mornings according to whether or not they are suitable for playing tennis.
- **For example:**
- $\langle \text{Outlook} = \text{Sunny}, \text{Temperature} = \text{Hot}, \text{Humidity} = \text{High}, \text{Wind} = \text{Strong} \rangle$
- would be sorted down the leftmost branch of this decision tree. The tree predicts that **PlayTennis = No**



Decision Tree

- Each path from the tree root to a leaf corresponds to a conjunction of attribute tests, and the tree itself to a disjunction of these conjunctions
- The decision tree from the previous slide can be expressed as

$$(Outlook = Sunny \wedge Humidity = Normal) \\ \vee (Outlook = Overcast) \\ \vee (Outlook = Rain \wedge Wind = Weak)$$



Decision Tree

- Another example of a decision tree.

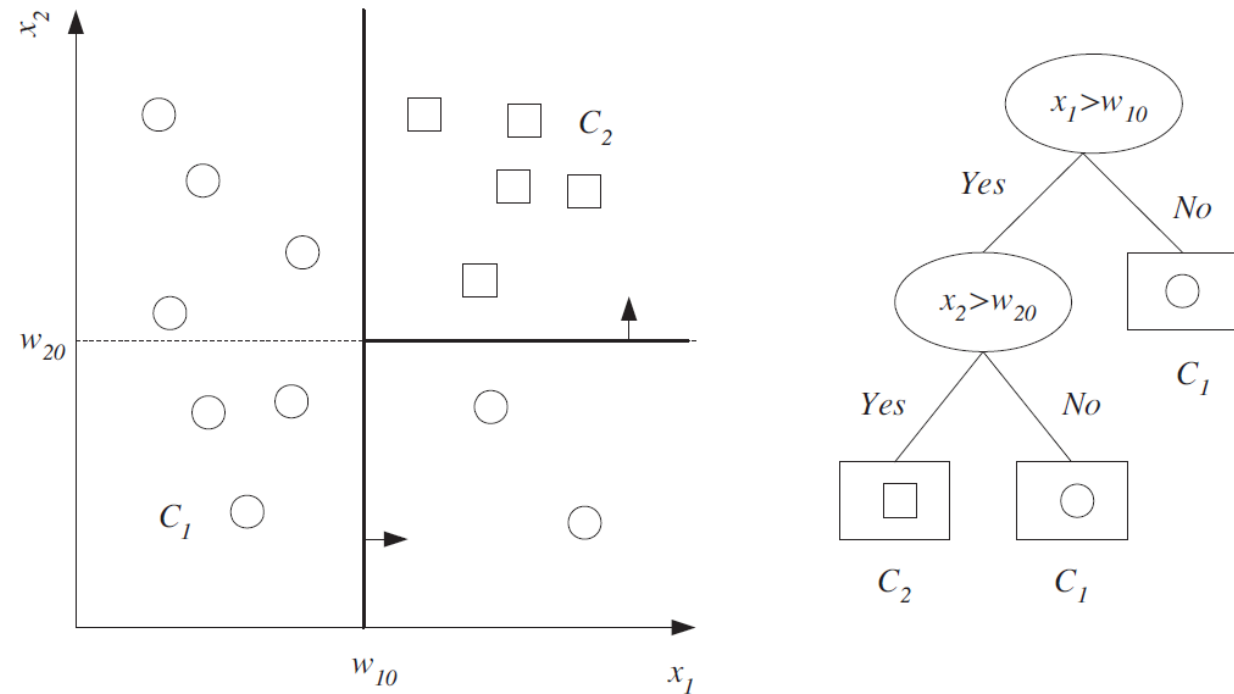


Figure 9.1 Example of a dataset and the corresponding decision tree. Oval nodes are the decision nodes and rectangles are leaf nodes. The univariate decision node splits along one axis, and successive splits are orthogonal to each other. After the first split, $\{\mathbf{x} | x_1 < w_{10}\}$ is pure and is not split further.

APPROPRIATE PROBLEMS FOR DECISION TREE LEARNING

- Decision tree learning is generally best suited to problems with the following characteristics:
 - **Instances are represented by attribute-value pairs**
 - Instances are described by a fixed set of attributes (e.g., Temperature) and their values (e.g., Hot). Attributes generally takes on a small number of disjoint possible values (e.g., Hot, Mild, Cold).
 - Real-valued attributes can also be handled with modified versions of the algorithm.
 - **The target function has discrete output values**
 - Even though the example tree assigns a Boolean classification, decision tree methods easily extend to learning functions with more than two possible output values.
 - **The training data may contain errors.**
 - Decision tree learning methods are robust to errors
 - **The training data may contain missing attribute values.**
 - Decision tree methods can be used even when some training examples have unknown values (e.g., if the Humidity of the day is known for only some of the training examples)

Decision Tree

- For a given training set, there exists many trees that code it with no error, and, for simplicity, we are interested in finding the smallest among them, where tree size is measured as the number of nodes in the tree and the complexity of the decision nodes
- Finding the smallest tree is **NP-complete** (Quinlan 1986), and we are forced to use **local search** procedures based on heuristics that give reasonable trees in reasonable time.
 - **Note:** “NP” stands for “nondeterministic polynomial time”, which represents the set of problems for which a solution can be verified in polynomial time.
 - An NP-complete problem is one for which a solution can be checked quickly, but finding the solution itself may be difficult.
- Tree learning algorithms are **greedy** and, at each step, starting at the root with the complete training data, we look for the best split.

ID3 Algorithm

- ID3 learns decision trees by constructing them top-down, beginning with the question “***which attribute should be tested at the root of the tree?***”
- To answer this question, each instance attribute is evaluated using a statistical test to determine how well it alone classifies the training examples.
- The best attribute is selected and used as the test at the root node of the tree.
- A descendant of the root node is then created for each possible value of this attribute.
- The entire process is then repeated using the training examples associated with each descendant node to select the best attribute to test at that point in the tree

ID3 Algorithm

ID3(*Examples*, *Target_attribute*, *Attributes*)

Examples are the training examples. *Target_attribute* is the attribute whose value is to be predicted by the tree. *Attributes* is a list of other attributes that may be tested by the learned decision tree. Returns a decision tree that correctly classifies the given *Examples*.

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

Which Attribute Is the Best Classifier?

- The central choice in the ID3 algorithm is selecting which attribute to test at each node in the tree.
- We would like to select the attribute that is most useful for classifying examples.
- What is a good quantitative measure of the worth of an attribute?
- We will define a statistical property, called **information gain**, that measures how well a given attribute separates the training examples according to their target classification.
- ID3 uses this information gain measure to select among the candidate attributes at each step while growing the tree.

Entropy Measures Homogeneity of Examples

- **Entropy** characterizes the (im)purity of an arbitrary collection of examples.
- Given a collection **S**, containing positive and negative examples of some target concept, the entropy of S relative to this boolean classification is formulated as

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

where p_{+} , is the proportion of positive examples in S and p_{-} , is the proportion of negative examples in S.

- In all calculations involving entropy we define $0 \log 0$ to be 0.

Entropy Measures Homogeneity of Examples

- Suppose S is a collection of 14 examples of some Boolean concept, including 9 positive and 5 negative examples (use $[9+, 5-]$ to summarize such a sample of data). Then the entropy of S relative to this boolean classification is

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

- Notice that the entropy is 0 if all members of S belong to the same class.
- Note the entropy is 1 when the collection contains an equal number of positive and negative examples
- If the collection contains unequal numbers of positive and negative examples, the entropy is between 0 and 1.

Entropy Measures Homogeneity of Examples

- One interpretation of entropy from information theory is that it specifies the *minimum number of bits of information* needed to encode the classification of an arbitrary member of S
- If p_+ is 1, the receiver knows the drawn example will be positive, so no message need be sent, and the entropy is zero.
- If p_+ is 0.5, one bit is required to indicate whether the drawn example is positive or negative.
- More generally, if the target attribute can take on c different values, then the entropy of S relative to this c -wise classification is defined as

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

where p_i is the proportion of S belonging to class i .

- Note the logarithm is still base 2 because entropy is a measure of the expected encoding length measured in bits.
- Note also that if the target attribute can take on c possible values, the entropy can be as large as $\log_2 c$.

Information Gain Measures the Expected Reduction in Entropy

- **Information gain**, is simply the expected reduction in entropy caused by partitioning the examples according to the selected attribute.
- The information gain, $Gain(S, A)$ of an attribute A , relative to a collection of examples S , is defined as

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


where $Values(A)$ is the set of all possible values for attribute A , and S_v , is the subset of S for which attribute A has value v .

- Note the first term is just the entropy of the original collection S , and the second term is the expected value of the entropy after S is partitioned using attribute A .

Information Gain Measures the Expected Reduction in Entropy

- The expected entropy described by this second term is simply the sum of the entropies of each subset S_v , weighted by the fraction of examples $\frac{|S_v|}{|S|}$ that belong to S_v .

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


$$Remainder(A) = \sum_{k=1}^d \frac{p_k + n_k}{p + n} B\left(\frac{p_k}{p_k + n_k}\right)$$

- $Gain(S, A)$ is therefore the expected reduction in entropy caused by knowing the value of attribute A .

Example

- Suppose S is a collection of training-example days described by attributes including *Wind*, which can have the values *Weak* or *Strong*. Assume S is a collection containing 14 examples, $[9+, 5-]$. Of these 14 examples, suppose 6 of the positive and 2 of the negative examples have *Wind* = *Weak*, and the remainder have *Wind* = *Strong*. The information gain due to sorting the original 14 examples by the attribute *Wind* may then be calculated as

$$\text{Values}(\text{Wind}) = \text{Weak}, \text{Strong}$$

$$S = [9+, 5-]$$

$$S_{\text{Weak}} \leftarrow [6+, 2-]$$

$$S_{\text{Strong}} \leftarrow [3+, 3-]$$

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

$$\begin{aligned}\text{Gain}(S, \text{Wind}) &= \text{Entropy}(S) - \sum_{v \in \{\text{Weak}, \text{Strong}\}} \frac{|S_v|}{|S|} \text{Entropy}(S_v) \\ &= \text{Entropy}(S) - (8/14) \text{Entropy}(S_{\text{Weak}}) \\ &\quad - (6/14) \text{Entropy}(S_{\text{Strong}}) \\ &= 0.940 - (8/14)0.811 - (6/14)1.00 \\ &= 0.048\end{aligned}$$

Information Gain

- The use of information gain is to evaluate the relevance of attributes.
- The attribute which gives **highest information gain** is the **best classifier**
- In the example here, Humidity and Wind, is computed in order to determine which is the better attribute for classifying the 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

Which attribute is the best classifier?

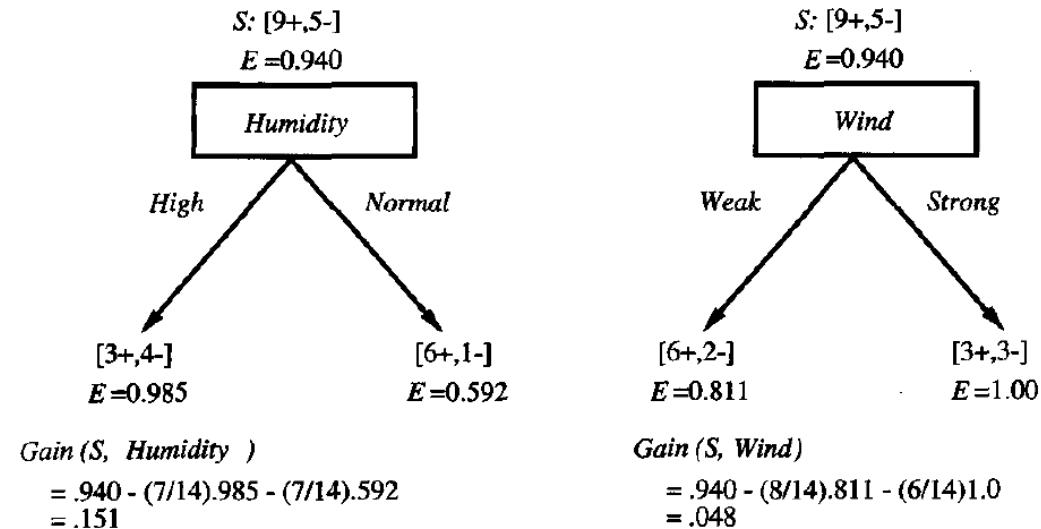


FIGURE 3.3

Humidity provides greater information gain than *Wind*, relative to the target classification. Here, E stands for entropy and S for the original collection of examples. Given an initial collection S of 9 positive and 5 negative examples, $[9+, 5-]$, sorting these by their *Humidity* produces collections of $[3+, 4-]$ (*Humidity* = *High*) and $[6+, 1-]$ (*Humidity* = *Normal*). The information gained by this partitioning is .151, compared to a gain of only .048 for the attribute *Wind*.

Example Decision Tree

- Consider the learning task represented by the training examples given in the table
- The target attribute ***PlayTennis***, which can have values *yes* or *no* for different Saturday mornings, is to be predicted based on other attributes of the morning in question

Day	<i>Outlook</i>	<i>Temperature</i>	<i>Humidity</i>	<i>Wind</i>	<i>PlayTennis</i>
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

Example Decision Tree

- Start with Root node.
- Which attribute should be tested first in the tree?
- ID3 determines the information gain for each candidate attribute (i.e., Outlook, Temperature, Humidity, and Wind), then selects the one with highest information gain.

$$Gain(S, Outlook) = 0.246$$

$$Gain(S, Humidity) = 0.151$$

$$Gain(S, Wind) = 0.048$$

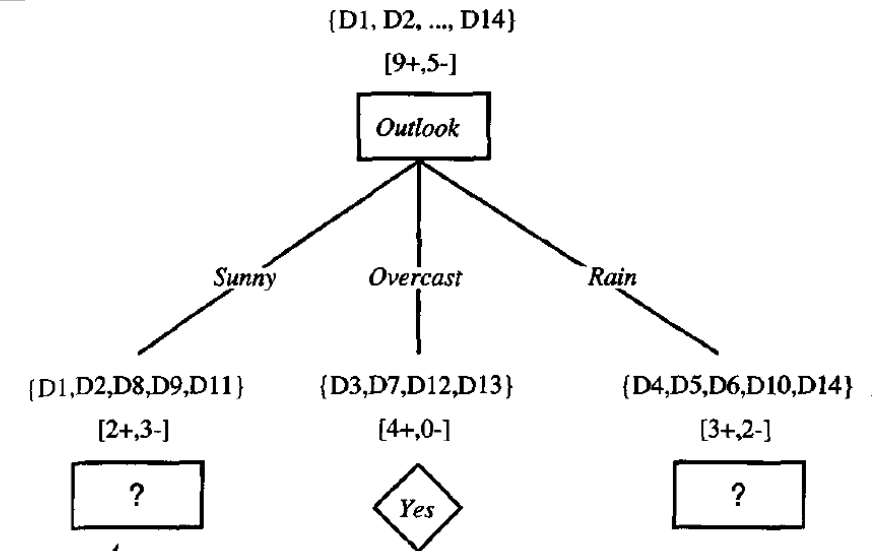
$$Gain(S, Temperature) = 0.029$$

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

Example Decision Tree

- According to the information gain measure, the **Outlook** attribute provides the best prediction of the target attribute, **PlayTennis**, over the training examples.
- Therefore, **Outlook** is selected as the decision attribute for the root node, and branches are created below the root for each of its possible values (i.e., Sunny, Overcast, and Rain).

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



Which attribute should be tested here?

$$S_{\text{sunny}} = \{D1, D2, D8, D9, D11\}$$

$$\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = .970 - (3/5) 0.0 - (2/5) 0.0 = .970$$

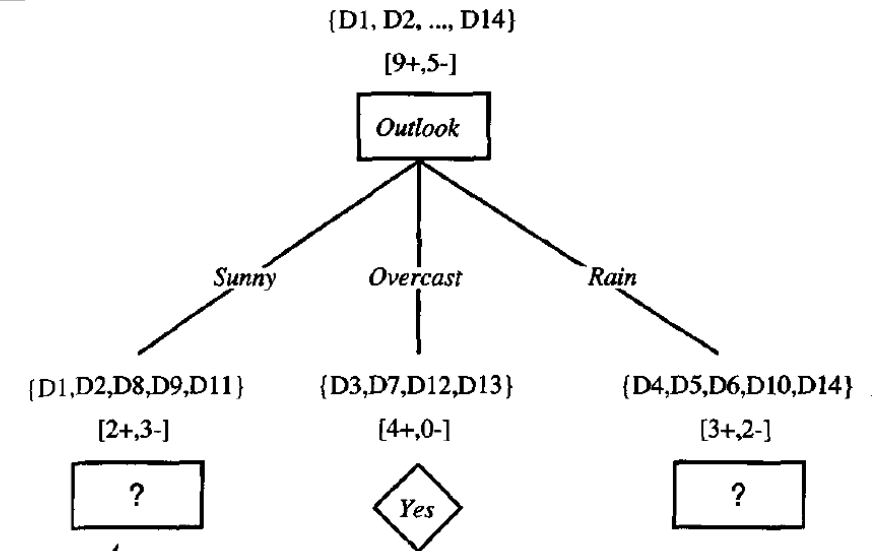
$$\text{Gain}(S_{\text{sunny}}, \text{Temperature}) = .970 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = .570$$

$$\text{Gain}(S_{\text{sunny}}, \text{Wind}) = .970 - (2/5) 1.0 - (3/5) .918 = .019$$

Example Decision Tree

- Note that every example for which **Outlook = Overcast** is also a positive example of **PlayTennis**. Therefore, this node of the tree becomes a leaf node with the classification **PlayTennis = Yes**.

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



Which attribute should be tested here?

$$S_{\text{sunny}} = \{D1, D2, D8, D9, D11\}$$

$$\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = .970 - (3/5) 0.0 - (2/5) 0.0 = .970$$

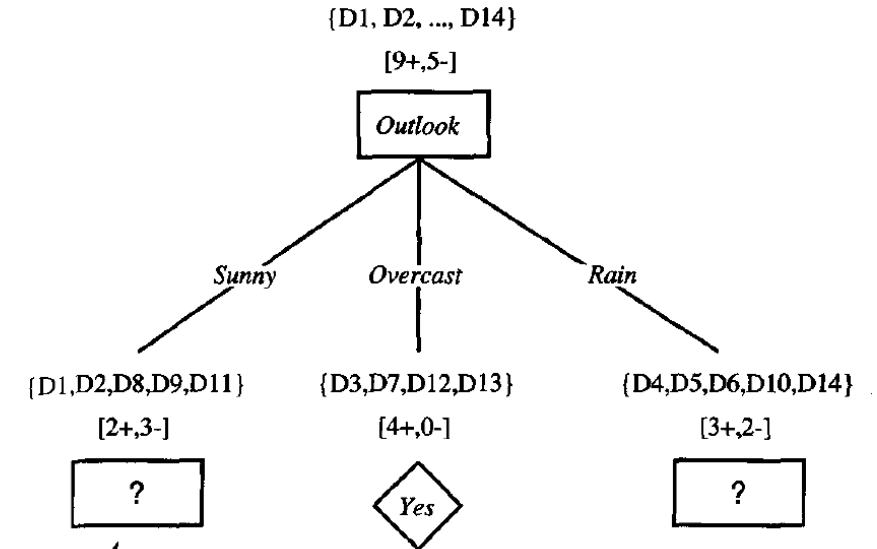
$$\text{Gain}(S_{\text{sunny}}, \text{Temperature}) = .970 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = .570$$

$$\text{Gain}(S_{\text{sunny}}, \text{Wind}) = .970 - (2/5) 1.0 - (3/5) .918 = .019$$

Example Decision Tree

- In contrast, the descendants corresponding to **Outlook = Sunny** and **Outlook = Rain** still have nonzero entropy, and the decision tree will be further elaborated below these nodes.

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



Which attribute should be tested here?

$$S_{\text{sunny}} = \{D1, D2, D8, D9, D11\}$$

$$\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = .970 - (3/5) 0.0 - (2/5) 0.0 = .970$$

$$\text{Gain}(S_{\text{sunny}}, \text{Temperature}) = .970 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = .570$$

$$\text{Gain}(S_{\text{sunny}}, \text{Wind}) = .970 - (2/5) 1.0 - (3/5) .918 = .019$$

Attributes: Outlook, temperature, humidity, windy

Outcomes: Decision on Play golf → Yes/No

Outlook	Temperature	Humidity	Windy	Play Golf
Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Overcast	Cool	Normal	TRUE	Yes
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Sunny	Mild	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Sunny	Mild	High	TRUE	No



Attributes: Outlook, temperature, humidity, windy

Outcomes: Decision on Play golf → Yes/No

1. Frequency table for Play Golf column:

Yes	No
9	5

Outlook	Temperature	Humidity	Windy	Play Golf
Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Overcast	Cool	Normal	TRUE	Yes
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Sunny	Mild	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Sunny	Mild	High	TRUE	No



Attributes: Outlook, temperature, humidity, windy

Outcomes: Decision on Play golf → Yes/No

1. Frequency table for Play Golf column:

Yes	No
9	5

2. Calculate the entropy for the Play Golf column

$$H(\text{PlayGolf}) = -\left(\frac{9}{14} \log_2 \frac{9}{14}\right) - \left(\frac{5}{14} \log_2 \frac{5}{14}\right) = 0.94$$

Outlook	Temperature	Humidity	Windy	Play Golf
Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Overcast	Cool	Normal	TRUE	Yes
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Sunny	Mild	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Sunny	Mild	High	TRUE	No



Now, we need to calculate entropy of the attributes

H(PlayGolf, Outlook) frequency table:

Outlook	Play YES	Play NO	Total
sunny	3	2	5
overcast	4	0	4
rainy	2	3	5

Outlook	Temperature	Humidity	Windy	Play Golf
Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Overcast	Cool	Normal	TRUE	Yes
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Sunny	Mild	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Sunny	Mild	High	TRUE	No



Now, we need to calculate entropy of the attributes

$H(\text{PlayGolf}, \text{Outlook})$ frequency table:

$$\text{Remainder}(A) = \sum_{k=1}^d \frac{p_k + n_k}{p + n} B\left(\frac{p_k}{p_k + n_k}\right)$$

Outlook	Play YES	Play NO	Total
sunny	3	2	5
overcast	4	0	4
rainy	2	3	5

Calculate entropy

$$H(\text{PlayGolf}, \text{Outlook}) = P(\text{sunny})H(\text{sunny}) + P(\text{overcast})H(\text{overcast}) + P(\text{rainy})H(\text{rainy})$$

$$H(\text{PlayGolf}, \text{Outlook}) = \frac{5}{14}H(\text{sunny}) + \frac{4}{14}H(\text{overcast}) + \frac{5}{14}H(\text{rainy})$$



Now, we need to calculate entropy of the attributes

H(PlayGolf, Outlook) frequency table:

Outlook	Play YES	Play NO	Total
sunny	3	2	5
overcast	4	0	4
rainy	2	3	5

$$B(q) = -(q \log_2 q + (1 - q) \log_2 (1 - q))$$

$$H(\text{PlayGolf}, \text{Outlook}) = \frac{5}{14} H(\text{sunny}) + \frac{4}{14} H(\text{overcast}) + \frac{5}{14} H(\text{rainy})$$

$$H(\text{sunny}) = -\left(\frac{3}{5} \log_2 \frac{3}{5}\right) - \left(\frac{2}{5} \log_2 \frac{2}{5}\right) = 0.971$$

Sunny and rainy has the same entropy value. So no need to do another calculation.

Overcast has only one possible outcome so the entropy is 0.

$$H(\text{PlayGolf}, \text{Outlook}) = \frac{5}{14} \times 0.971 + \frac{4}{14} \times 0 + \frac{5}{14} \times 0.971 = 0.693$$



If you repeat the same process for other attributes, you will end up with

- $H(\text{PlayGolf}, \text{Outlook}) = 0.693$
- $H(\text{PlayGolf}, \text{Temperature}) = 0.911$
- $H(\text{PlayGolf}, \text{Humidity}) = 0.788$
- $H(\text{PlayGolf}, \text{Windy}) = 0.892$

The next step is to calculate the information gain for each of the attributes.

The information gain is calculated from the split using each of the attributes. Then the attribute with the largest information gain is used for the split.



The information gain after splitting using the Outlook attribute:

$$\text{Gain}(\text{PlayGolf}, \text{Outlook}) = H(\text{PlayGolf}) - H(\text{PlayGolf}, \text{Outlook})$$

$$\text{Gain}(\text{PlayGolf}, \text{Outlook}) = 0.94 - 0.693 = \mathbf{0.247}$$

$$\text{Gain}(\text{PlayGolf}, \text{Temperature}) = H(\text{PlayGolf}) - H(\text{PlayGolf}, \text{Temperature})$$

$$\text{Gain}(\text{PlayGolf}, \text{Temperature}) = 0.94 - 0.911 = 0.029$$

$$\text{Gain}(\text{PlayGolf}, \text{Humidity}) = H(\text{PlayGolf}) - H(\text{PlayGolf}, \text{Humidity})$$

$$\text{Gain}(\text{PlayGolf}, \text{Humidity}) = 0.94 - 0.788 = 0.152$$

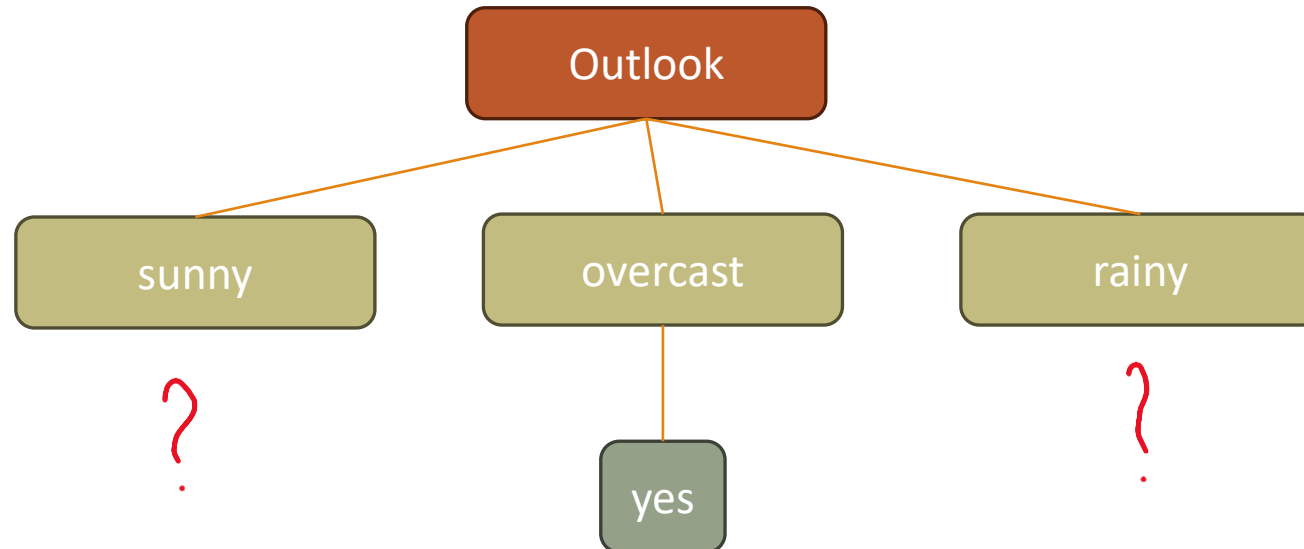
$$\text{Gain}(\text{PlayGolf}, \text{Windy}) = H(\text{PlayGolf}) - H(\text{PlayGolf}, \text{Windy})$$

$$\text{Gain}(\text{PlayGolf}, \text{Windy}) = 0.94 - 0.892 = 0.048$$

Choose the attribute that gives the **highest information gain** after the split



Perform the first split



Decision nodes

Leaf nodes

You still need to split the tree further.



To continue, you need to also split the original table to create sub-tables:

Sunny and the Rainy attributes needs to be split

Apply the previous process for the sub-table for sunny outlook.

Apply the previous process for sub-table rainy outlook.

You will end up with the final decision tree.

Outlook	Temperature	Humidity	Windy	Play Golf
Sunny	Mild	Normal	FALSE	Yes
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Sunny	Mild	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Overcast	Cool	Normal	TRUE	Yes
Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes



