

# CS- 510

## DECISION TREE

# Intro.

- Decision tree learning is one of the most widely used and practical methods for inductive inference.
- It is a method for approximating discrete-valued functions that is robust to noisy data and capable of learning disjunctive expressions.
- Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree.
- Learned trees can also be re-represented as sets of if-then rules to improve human readability.

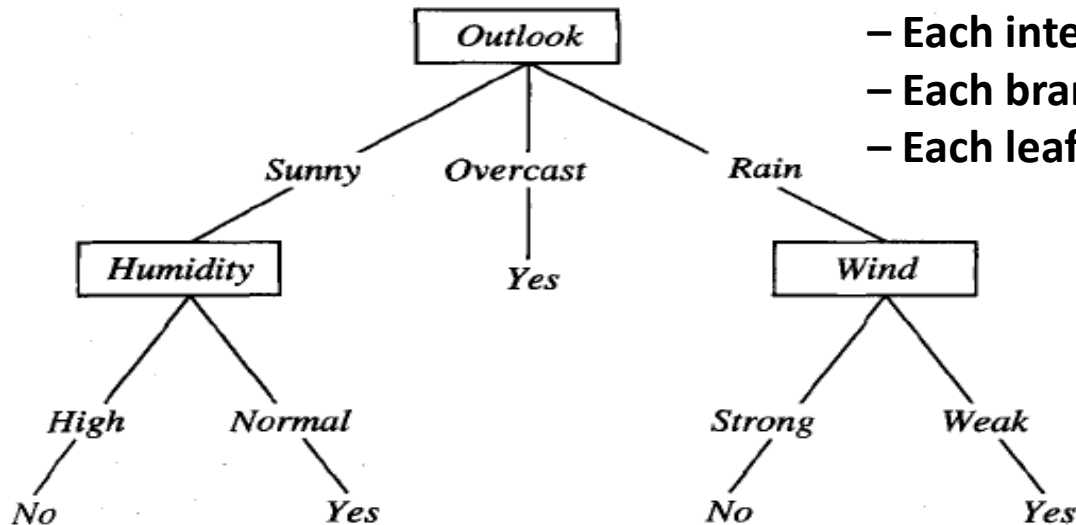
# An Example

Example	<i>Sky</i>	<i>AirTemp</i>	<i>Humidity</i>	<i>Wind</i>	<i>Water</i>	<i>Forecast</i>	<i>EnjoySport</i>
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

# Decision Tree Representation

Decision tree representation:

- Each internal node tests an attribute
- Each branch corresponds to attribute value
- Each leaf node assigns a classification



**$(Outlook = Sunny \wedge Humidity = Normal)$**

**$V(Outlook = Overcast)$**

**$V(Outlook = Rain \wedge Wind = Weak)$**

An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the subtree rooted at the new node.

# APPROPRIATE PROBLEMS FOR DECISION TREE LEARNING

- *Instances are represented by attribute-value pairs.*
- *The target function has discrete output values.*
- *Disjunctive descriptions may be required.*
- *The training data may contain errors.*
- *The training data may contain missing attribute values.*

# DECISION TREE LEARNING 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*

# Target Concept : 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

# 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. *Entropy is a measure of uncertainty associated with a random variable*
- ID3 uses this information gain measure to select among the candidate attributes at each step while growing the tree.



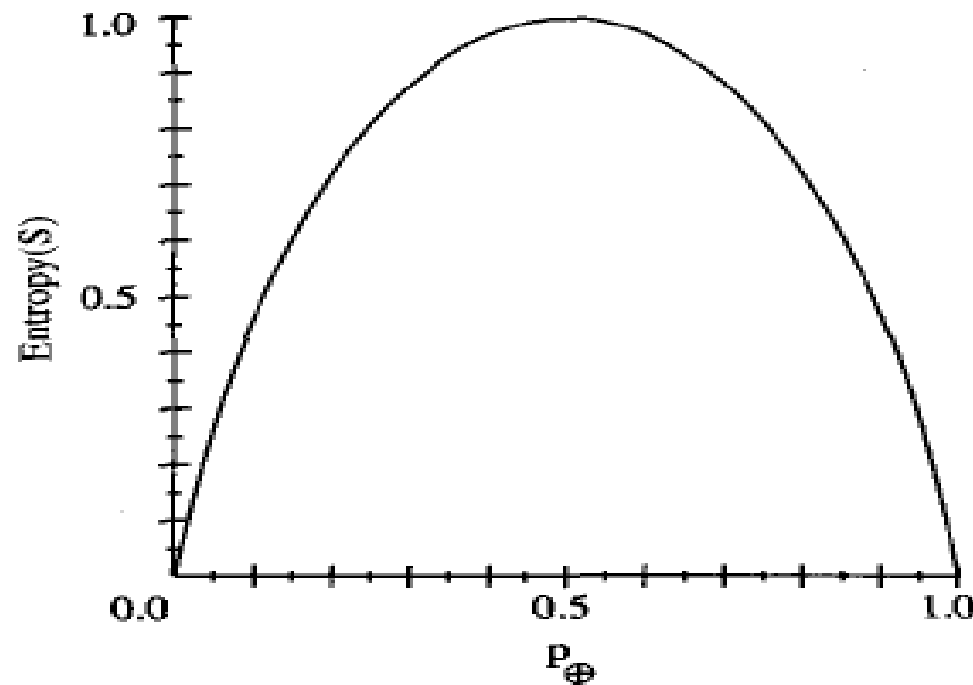
## ***Entropy, that characterizes the (im)purity***

- Given a collection  $S$ , containing positive and negative examples of some target concept, the entropy of  $S$  relative to this Boolean classification is 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.

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

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

# *A typical Case:- $0 \log 0$*



*The entropy function relative to a boolean classification, as the proportion,  $p(+)$ , of positive examples varies between 0 and 1.*

# INFORMATION GAIN

Given entropy as a measure of the impurity in a collection of training examples, we can now define a measure of the effectiveness of an attribute in classifying the training data.

The measure we will use, called ***information gain***, is simply the expected reduction in entropy caused by partitioning the examples according to this attribute.

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

*Values(A)* is the set of all possible values for attribute *A*, and *S<sub>v</sub>* is the subset of *S* for which attribute *A* has value *v* (i.e.,  $S_v = \{s \in S | A(s) = v\}$ ).

$Values(Wind) = Weak, Strong$

$S = [9+, 5-]$

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

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

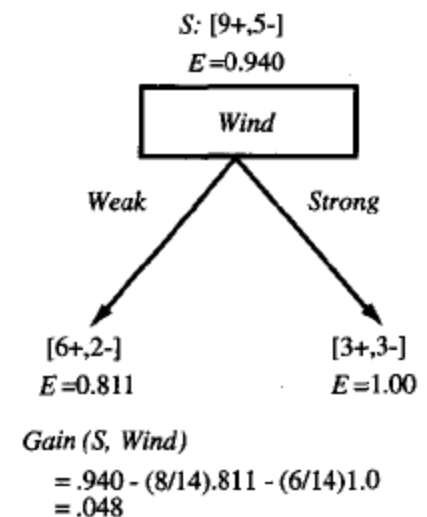
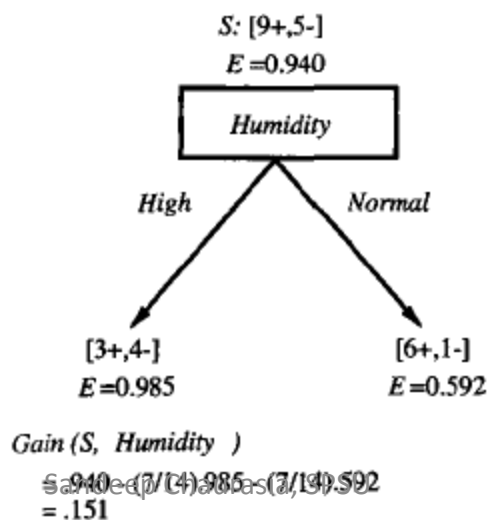
$$\begin{aligned}
 Gain(S, Wind) &= Entropy(S) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S|} Entropy(S_v) \\
 &= Entropy(S) - (8/14)Entropy(S_{Weak}) \\
 &\quad - (6/14)Entropy(S_{Strong}) \\
 &= 0.940 - (8/14)0.811 - (6/14)1.00 \\
 &= 0.048
 \end{aligned}$$

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

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

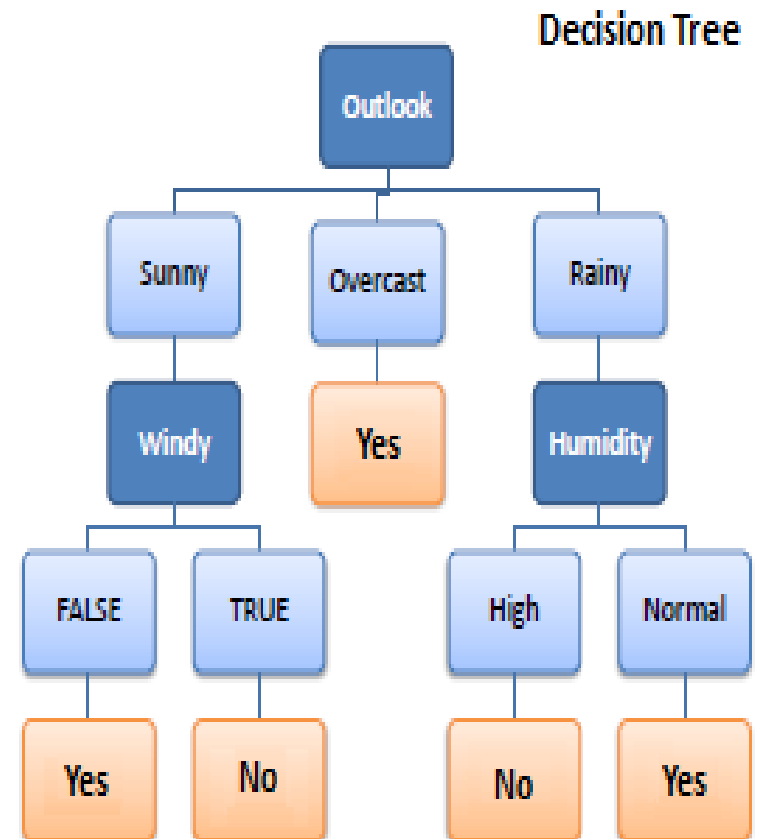
$$Gain(S, Wind) = 0.048$$

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



# Decision Tree - Classification

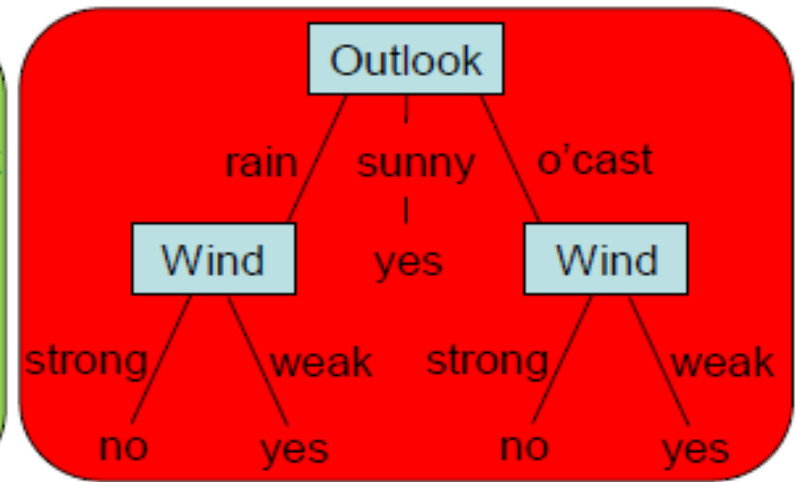
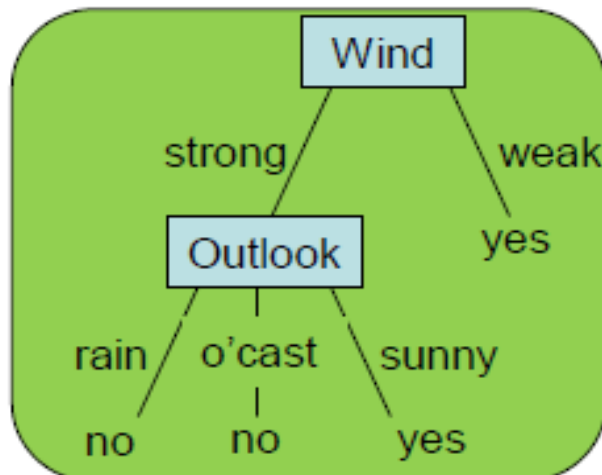
Predictors				Target
Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No



# A simple Example #2 Feature

- Basic Algorithm is recursive...
  - Determine which attribute to use as the top most node of the decision tree (candidates being “Wind” or “Outlook” in the first step using this example)
    - do this by calculating the information gain for each candidate attribute
      - pick the attribute with the highest information gain.

Outlook	Wind	PlayGolf
rain	strong	no
sunny	weak	yes
overcast	weak	yes
rain	weak	yes
sunny	strong	yes
rain	strong	no
overcast	strong	no



# Step 1.

- Determine which attribute to use as the top most node of the decision tree – do this by calculating the information gain for each candidate attribute – pick the attribute with the highest information gain.

$$\text{Gain}(\mathbf{S}, \mathbf{A}) \equiv \text{Entropy}(\mathbf{S}) - \sum_{v \in \text{Values}(\mathbf{A})} (|\mathbf{S}_v| / |\mathbf{S}|) \text{Entropy}(\mathbf{S}_v)$$

The information Gain of attribute **A** in collection **S** where **Values(A)** is the set of possible values for attribute **A** and **S<sub>v</sub>** is the subset of **S** for which attribute **A** has the value **v**

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

Where target classification is boolean

**p<sub>+</sub>** :the proportion of positive examples in collection **S**

**p<sub>-</sub>** :the proportion of negative examples in collection **S**

Outlook	Wind	PlayGolf
rain	strong	no
sunny	weak	yes
overcast	weak	yes
rain	weak	yes
sunny	strong	yes
rain	strong	no
overcast	strong	no

# Calculating the IG for each candidate...

$$\text{Gain}(S, \text{Outlook}) \equiv 0.98 - \sum_{v \in \text{Values}(\text{rain}, \text{sunny}, \text{overcast})} (|S_v| / |S|) \text{Entropy}(S_v)$$

$$\text{Gain}(S, \text{Wind}) \equiv 0.98 - \sum_{v \in \text{Values}(\text{weak}, \text{strong})} (|S_v| / |S|) \text{Entropy}(S_v)$$

Outlook	Wind	PlayGolf
rain	strong	no
sunny	weak	yes
overcast	weak	yes
rain	weak	yes
sunny	strong	yes
rain	strong	no
overcast	strong	no



# Calculating the IG of the Outlook Attribute...

$$\text{Gain}(S, \text{Outlook}) \equiv 0.98 - \sum_{v \in \text{Values}(\text{rain}, \text{sunny}, \text{overcast})} (|S_v| / |S|) \text{Entropy}(S_v)$$

$$\text{Entropy}(S) \equiv -p_+ \log_2 p_+ - p_- \log_2 p_-$$

$$\overset{\text{rain}}{(3/7)} \text{Entropy}([1+, 2-]) + \overset{\text{sunny}}{(2/7)} \text{Entropy}([2+, 0-]) + \overset{\text{overcast}}{(2/7)} \text{Entropy}([1+, 1-])$$

$$\begin{aligned} & (3/7) \text{Entropy}([1+, 2-]) + (2/7) * 0 + (2/7) * 1 \\ & (3/7) * ( -(1/3) * \log_2 (1/3) ) - ( (2/3) * \log_2 (2/3) ) + 0 + (2/7) \\ & 0.43 * ( ( -0.33 * -1.6 ) - ( 0.66 * -0.6 ) ) + 0 + 0.29 \\ & 0.43 * ( ( 0.53 ) - ( -0.4 ) ) + 0 + 0.29 \\ & 0.43 * ( 0.93 ) + 0 + 0.29 \\ & 0.4 + 0 + 0.29 \\ & 0.69 \end{aligned}$$

$$\begin{aligned} \text{Gain}(S, \text{Outlook}) & \equiv 0.98 - 0.69 \\ \text{Gain}(S, \text{Outlook}) & \equiv 0.29 \end{aligned}$$

Outlook	Wind	PlayGolf
rain	strong	no
sunny	weak	yes
overcast	weak	yes
rain	weak	yes
sunny	strong	yes
rain	strong	no
overcast	strong	no

# Calculating the IG of the Wind Attribute...

$$\text{Gain}(S, \text{Wind}) \equiv 0.98 - \sum_{v \in \text{Values}(\text{weak}, \text{strong})} (|S_v| / |S|) \text{Entropy}(S_v)$$

$$\text{Entropy}(S) \equiv -p_+ \log_2 p_+ - p_- \log_2 p_-$$

weak

strong

$$(3/7)\text{Entropy}([3+,0-]) + (4/7)\text{Entropy}([1+,3-])$$

$$(3/7) * 0.0 + (4/7) \text{Entropy}([1+,3-])$$

$$0 + (4/7) * ( -(1/4) * \log_2 (1/4) ) - ( (3/4) * \log_2 (3/4) )$$

$$0 + 0.57 * ( -(0.25 * -2.0) - (0.75 * -0.41) )$$

$$0 + 0.57 * ( -(-0.5) - (-0.31) )$$

$$0 + 0.57 * ( (0.5) - (-0.31) )$$

$$0 + 0.57 * ( 0.81 )$$

$$0 + 0.46$$

$$0.46$$

$$\text{Gain}(S, \text{Wind}) \equiv 0.98 - 0.46$$

$$\text{Gain}(S, \text{Wind}) \equiv 0.52$$

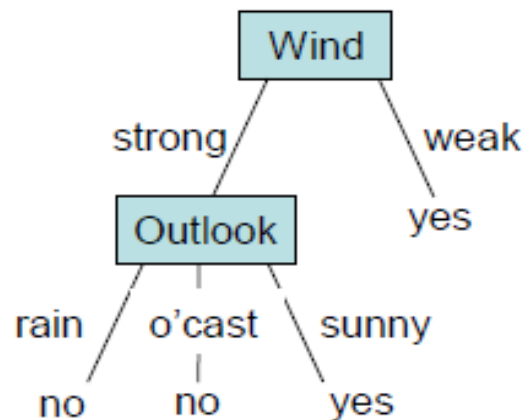
Outlook	Wind	PlayGolf
rain	strong	no
sunny	weak	yes
overcast	weak	yes
rain	weak	yes
sunny	strong	yes
rain	strong	no
overcast	strong	no

# Compare the IGs of each attribute...

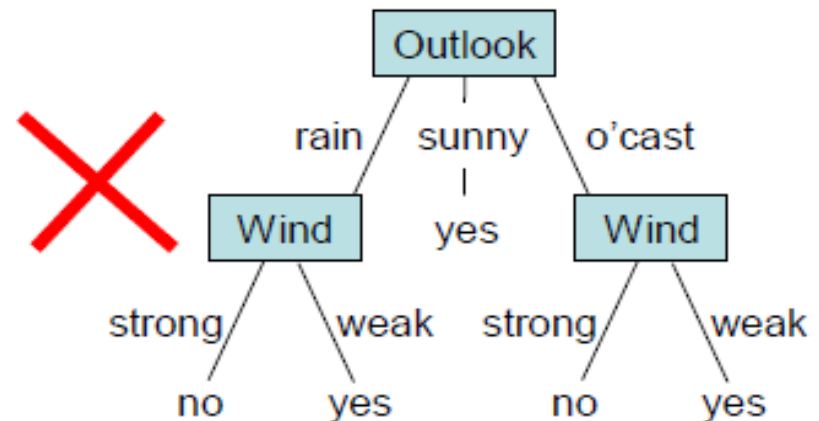
$$\text{Gain}(S, \text{Outlook}) \equiv 0.98 - 0.69$$
$$\text{Gain}(S, \text{Outlook}) \equiv 0.29$$

$$\text{Gain}(S, \text{Wind}) \equiv 0.98 - 0.46$$
$$\text{Gain}(S, \text{Wind}) \equiv 0.52$$

Wind has highest Information Gain so make it top node



Outlook	Wind	PlayGolf
rain	strong	no
sunny	weak	yes
overcast	weak	yes
rain	weak	yes
sunny	strong	yes
rain	strong	no
overcast	strong	no



# Back to the problem

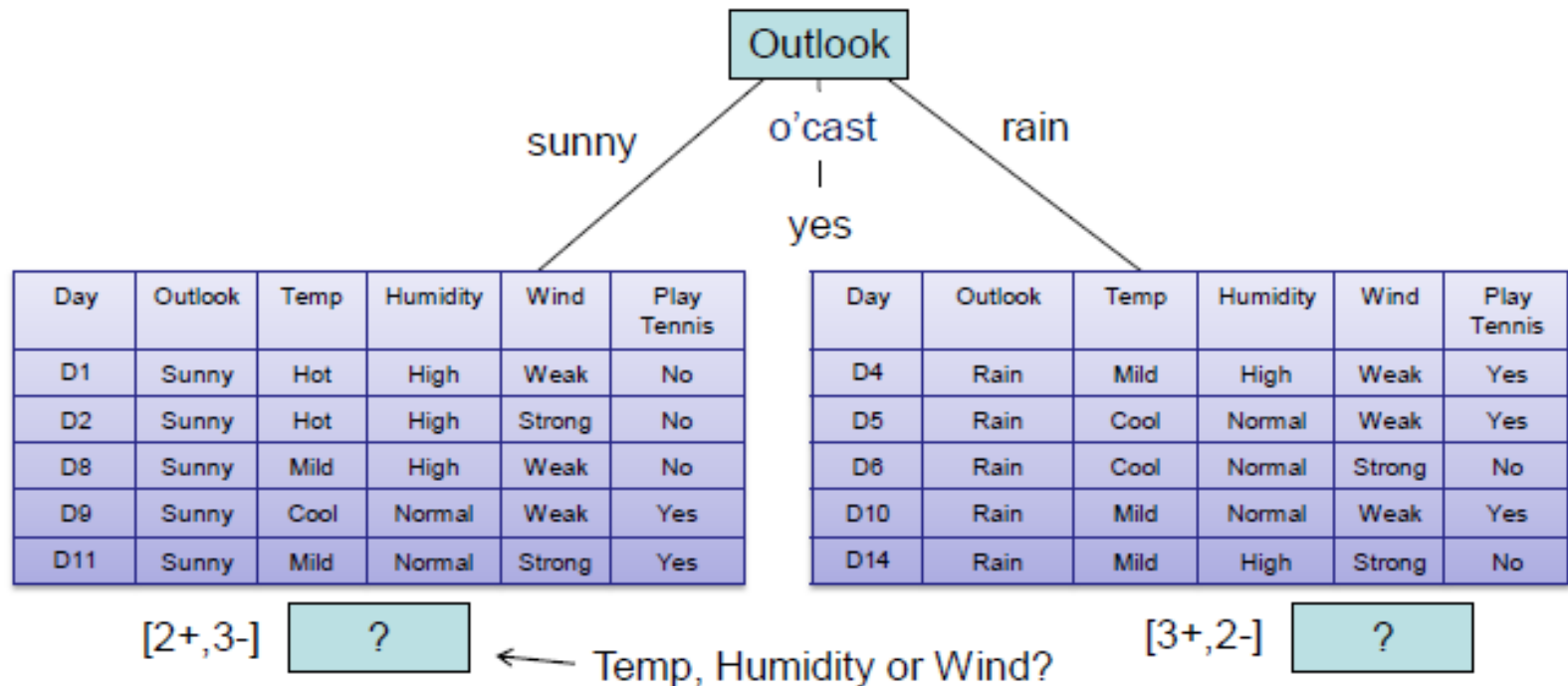
## Sample Training Data...

Day	Outlook	Temp	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

# Step 1...

- Which attribute should we have as the top most node of our decision tree
- Determine the information gain for each candidate attribute...
  - $\text{Gain}(S, \text{Outlook}) = 0.246$
  - $\text{Gain}(S, \text{Humidity}) = 0.151$
  - $\text{Gain}(S, \text{Wind}) = 0.048$
  - $\text{Gain}(S, \text{Temperature}) = 0.029$
- So would have Outlook as top Node

# After Step 1...



$$\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = .97 - (3/5)0.0 - (2/5)0.0 = 0.97$$

$$\text{Gain}(S_{\text{sunny}}, \text{Temp}) = .97 - (2/5)0.0 - (2/5)1.0 - (1/5)0.0 = 0.57$$

$$\text{Gain}(S_{\text{sunny}}, \text{Wind}) = .97 - (3/5)0.93 - (2/5)1.0 = 0.019$$

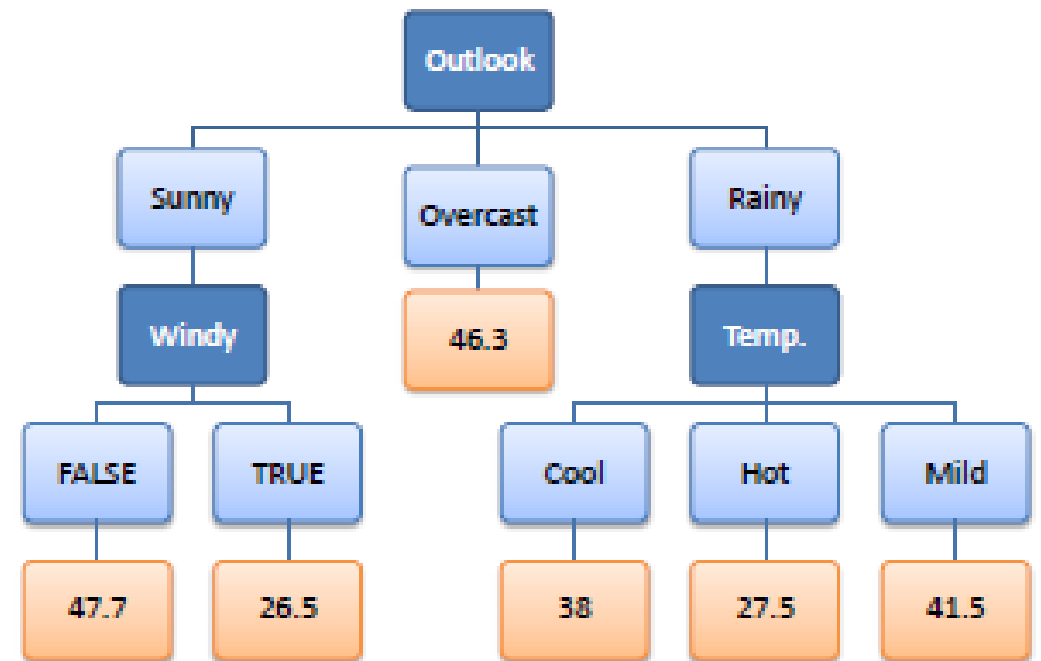
$$\begin{aligned} \text{Entropy}([2+,3-]) &= (- (2/5) * \log_2(2/5)) - ((3/5) * \log_2(3/5)) \\ &= (-0.4 * -1.32) - (0.6 * -0.74) \\ &= 0.53 + 0.44 = 0.97 \end{aligned}$$

Again just 2 dec places  
being used here!

# Decision Tree - Regression

Sandeep Chaurasia

Predictors				Target
Outlook	Temp.	Humidity	Windy	Hours Played
Rainy	Hot	High	False	26
Rainy	Hot	High	True	30
Overcast	Hot	High	False	48
Sunny	Mild	High	False	46
Sunny	Cool	Normal	False	62
Sunny	Cool	Normal	True	28
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	36
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	48
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	62
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30





# Decision Tree Algorithm

- The core algorithm for building decision trees called **ID3** by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. The ID3 algorithm can be used to construct a decision tree for regression by replacing Information Gain with *Standard Deviation Reduction*.

## a) Standard Deviation

Standard Deviation (S) is for tree building (branching).


Coefficient of Deviation (CV) is used to decide when to stop branching. We can use Count (n) as well.

Average (Avg) is the value in the leaf nodes.

Hours Played
25
30
46
45
52
23
43
35
38
46
48
52
44
30

$$\text{Count} = n = 14$$

$$\text{Average} = \bar{x} = \frac{\sum x}{n} = 39.8$$


$$\text{Standard Deviation} = S = \sqrt{\frac{\sum (x - \bar{x})^2}{n}} = 9.32$$

$$\text{Coefficient of Variation} = CV = \frac{S}{\bar{x}} * 100\% = 23\%$$

## b) Standard deviation for **two** attributes (target and predictor):

$$S(T, X) = \sum_{c \in X} P(c)S(c)$$

		Hours Played (StDev)	Count
Outlook	Overcast	3.49	4
	Rainy	7.78	5
	Sunny	10.87	5
			14



$$\begin{aligned} S(\text{Hours, Outlook}) &= P(\text{Sunny}) * S(\text{Sunny}) + P(\text{Overcast}) * S(\text{Overcast}) + P(\text{Rainy}) * S(\text{Rainy}) \\ &= (4/14) * 3.49 + (5/14) * 7.78 + (5/14) * 10.87 \\ &= 7.66 \end{aligned}$$

### Standard Deviation Reduction (SDR)

The standard deviation reduction is based on the decrease in standard deviation after a dataset is split on an attribute.

Step 1: The standard deviation of the target is calculated.

Step 2: The dataset is then split on the different attributes. The standard deviation for each branch is calculated. The resulting standard deviation is subtracted from the standard deviation before the split. The result is the standard deviation reduction.

$$SDR(T, X) = S(T) - S(T, X)$$

$$\begin{aligned} SDR(\text{Hours, Outlook}) &= S(\text{Hours}) - S(\text{Hours, Outlook}) \\ &= 9.32 - 7.66 = 1.66 \end{aligned}$$

		Hours Played (StDev)
Outlook	Overcast	3.49
	Rainy	7.78
	Sunny	10.87
SDR=1.66		

		Hours Played (StDev)
Temp.	Cool	10.51
	Hot	8.95
	Mild	7.65
SDR= 0.48		

		Hours Played (StDev)
Humidity	High	9.36
	Normal	8.37
SDR=0.28		

		Hours Played (StDev)
Windy	False	7.87
	True	10.59
SDR=0.29		

Step 3: The attribute with the largest standard deviation reduction is chosen for the decision node.

★		Hours Played (StDev)
Outlook	Overcast	3.49
	Rainy	7.78
	Sunny	10.87
SDR=1.66		

**Step 4a:** The dataset is divided based on the values of the selected attribute. This process is run recursively on the non-leaf branches, until all data is processed

Outlook

Sunny

Overcast

Rainy

Outlook	Temp	Humidity	Windy	Hours Played
Sunny	Mild	High	FALSE	45
Sunny	Cool	Normal	FALSE	52
Sunny	Cool	Normal	TRUE	23
Sunny	Mild	Normal	FALSE	46
Sunny	Mild	High	TRUE	30

Overcast	Hot	High	FALSE	46
Overcast	Cool	Normal	TRUE	43
Overcast	Mild	High	TRUE	52
Overcast	Hot	Normal	FALSE	44

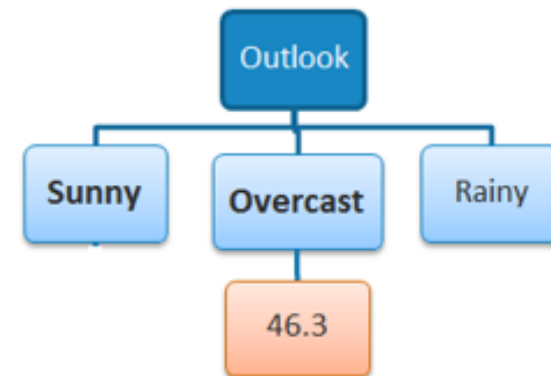
Rainy	Hot	High	FALSE	25
Rainy	Hot	High	TRUE	30
Rainy	Mild	High	FALSE	35
Rainy	Cool	Normal	FALSE	38
Rainy	Mild	Normal	TRUE	48

- In practice, we need some termination criteria. For example, when coefficient of deviation (**CV**) for a branch becomes smaller than a certain threshold (e.g., 10%) and/or when too few instances (**n**) remain in the branch (e.g., 3).

*Step 4b:* "Overcast" subset does not need any further splitting because its CV (8%) is less than the threshold (10%). The related leaf node gets the average of the "Overcast" subset.

Outlook - Overcast

		Hours Played (StDev)	Hours Played (AVG)	Hours Played (CV)	Count
Outlook	Overcast	3.49	46.3	8%	4
	Rainy	7.78	35.2	22%	5
	Sunny	10.87	39.2	28%	5



**Step 4c:** However, the "Sunny" branch has an CV (28%) more than the threshold (10%) which needs further splitting. We select "Temp" as the best best node after "Outlook" because it has the largest SDR.

## Outlook - Sunny

Temp	Humidity	Windy	Hours Played
Mild	High	FALSE	45
Cool	Normal	FALSE	52
Cool	Normal	TRUE	23
Mild	Normal	FALSE	46
Mild	High	TRUE	30
			$S = 10.87$
			$AVG = 39.2$
			$CV = 28\%$

		Hours Played (StDev)	Count
Temp	Cool	14.50	2
	Mild	7.32	3

$$SDR = 10.87 - ((2/5) * 14.5 + (3/5) * 7.32) = 0.678$$

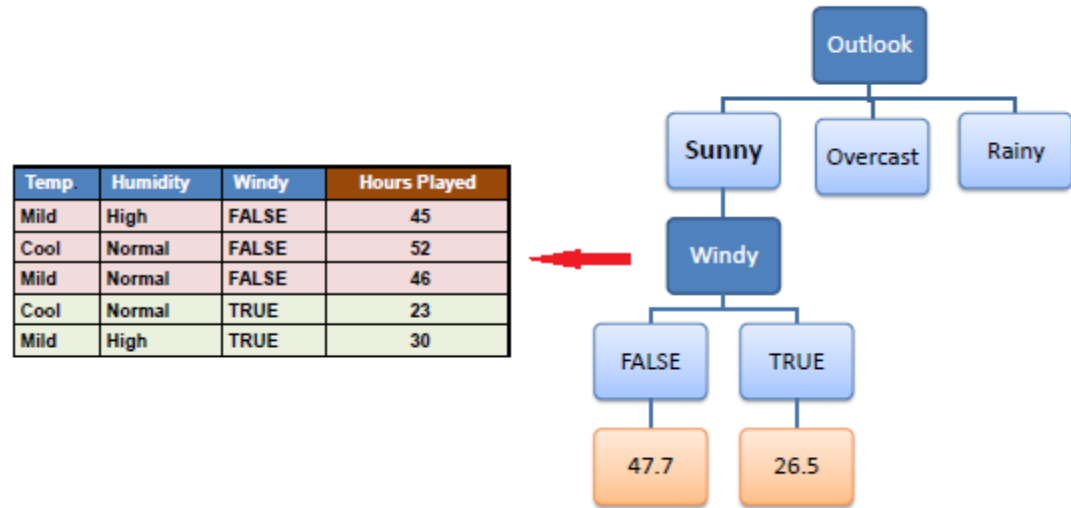
		Hours Played (StDev)	Count
Humidity	High	7.50	2
	Normal	12.50	3

$$SDR = 10.87 - ((2/5) * 7.5 + (3/5) * 12.5) = 0.370$$

		Hours Played (StDev)	Count
Windy	False	3.09	3
	True	3.50	2

$$SDR = 10.87 - ((3/5) * 3.09 + (2/5) * 3.5) = 7.62$$

Because the number of data points for both branches (FALSE and TRUE) is equal or less than 3 we stop further branching and assign the average of each branch to the related leaf node.



Temp	Humidity	Windy	Hours Played
Mild	High	FALSE	45
Cool	Normal	FALSE	52
Mild	Normal	FALSE	46
Cool	Normal	TRUE	23
Mild	High	TRUE	30

Step 4d: Moreover, the "rainy" branch has an CV (22%) which is more than the threshold (10%). This branch needs further splitting. We select "Temp" as the best node because it has the largest SDR.

## Outlook - Rainy

Temp	Humidity	Windy	Hours Played
Hot	High	FALSE	25
Hot	High	TRUE	30
Mild	High	FALSE	35
Cool	Normal	FALSE	38
Mild	Normal	TRUE	48
			S = 7.78
			AVG = 35.2
			CV = 22%

		Hours Played (StDev)	Count
Temp	Cool	0	1
	Hot	2.5	2
	Mild	6.5	2

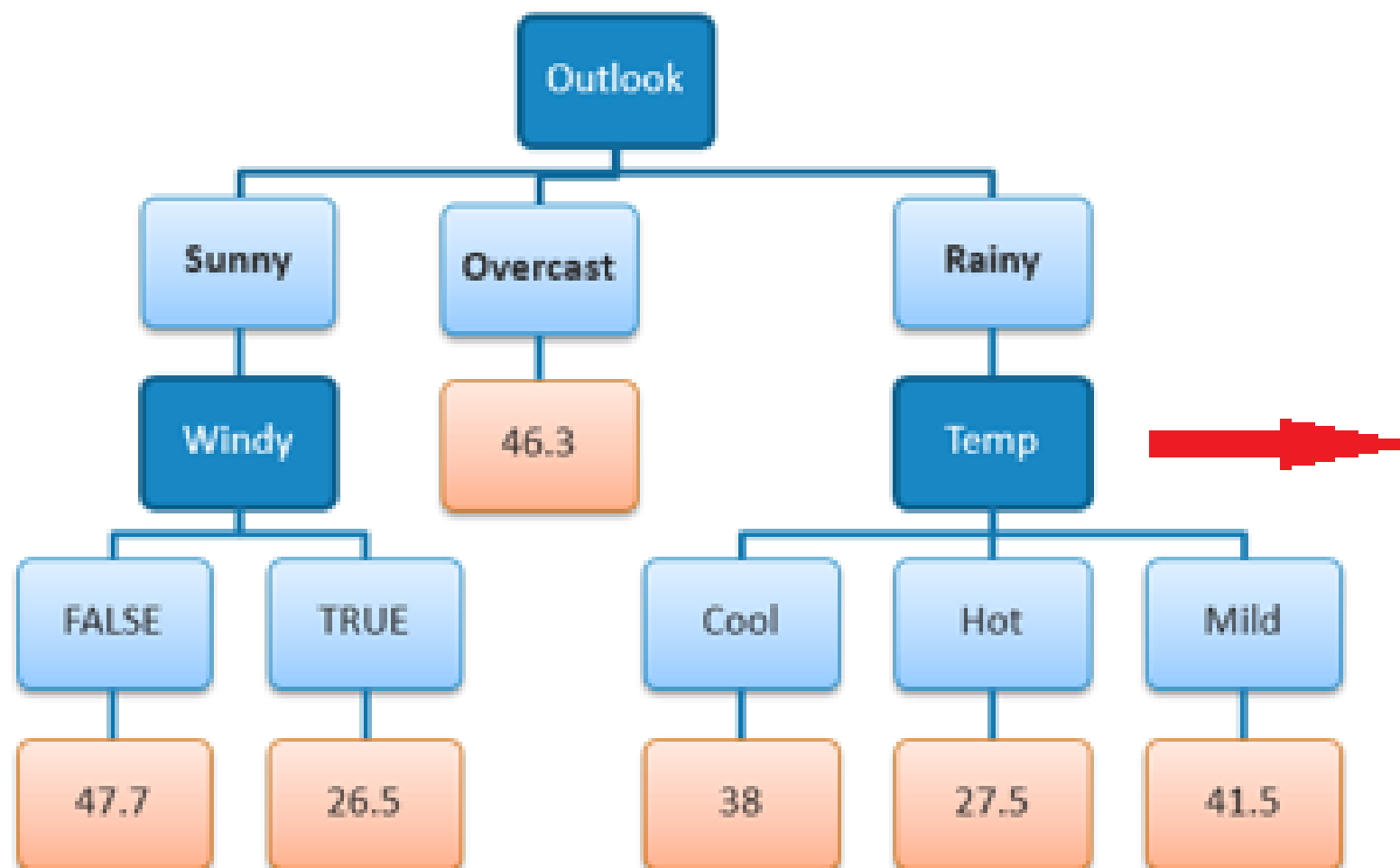
$$SDR = 7.78 - ((1/5)*0 + (2/5)*2.5 + (2/5)*6.5) = 4.18$$

		Hours Played (StDev)	Count
Humidity	High	4.1	3
	Normal	5.0	2

$$SDR = 7.78 - ((3/5)*4.1 + (2/5)*5.0) = 3.32$$

		Hours Played (StDev)	Count
Windy	False	5.6	3
	True	9.0	2

$$SDR = 7.78 - ((3/5)*5.6 + (2/5)*9.0) = 0.82$$



Temp	Hours Played
Cool	38
Hot	25
Hot	30
Mild	35
Mild	48