University of Nottingham
UK | CHINA | MALAYSIA

COMP3055 Machine Learning

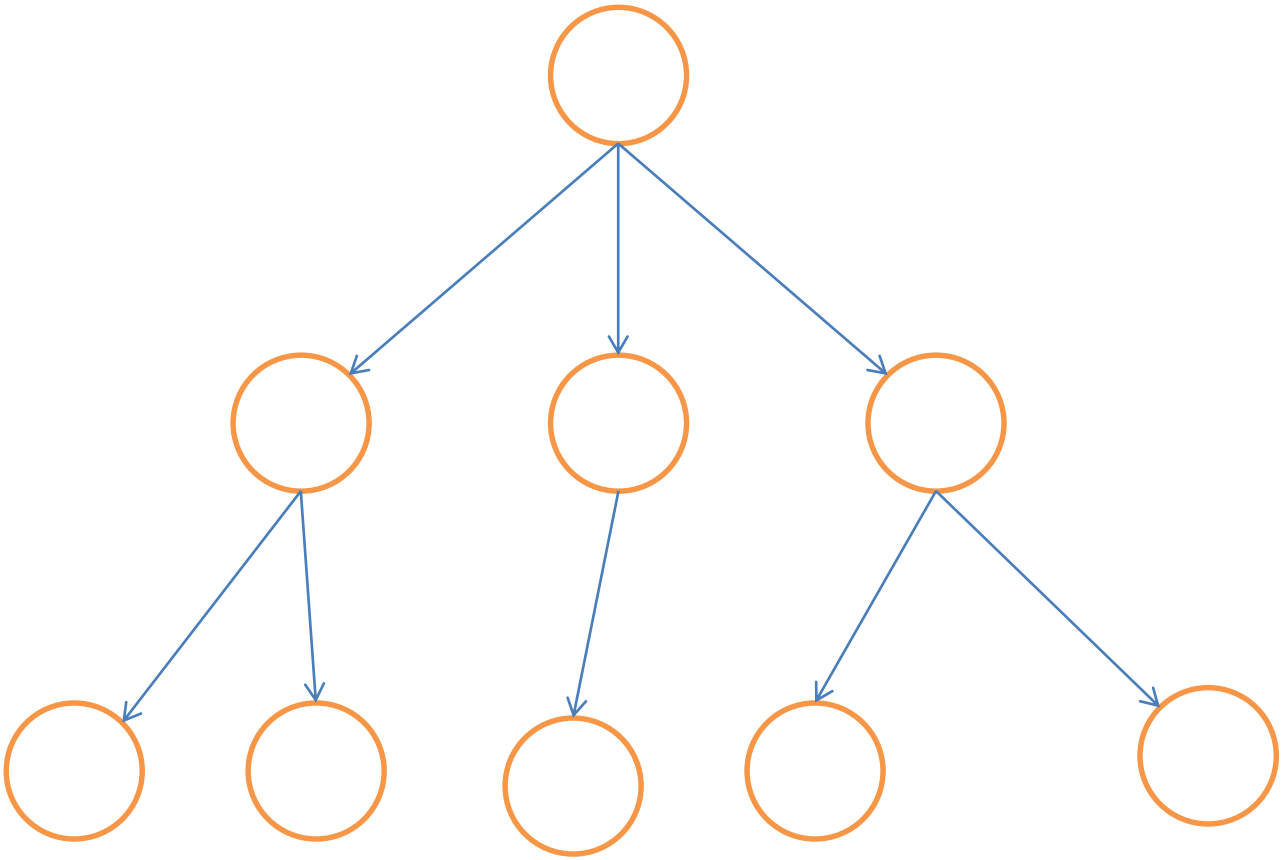
Topic 9 – Decision Tree

Ying Weng

2024 Autumn

Trees

- Node
- Root
- Leaf
- Branch
- Path
- Depth



Decision Trees

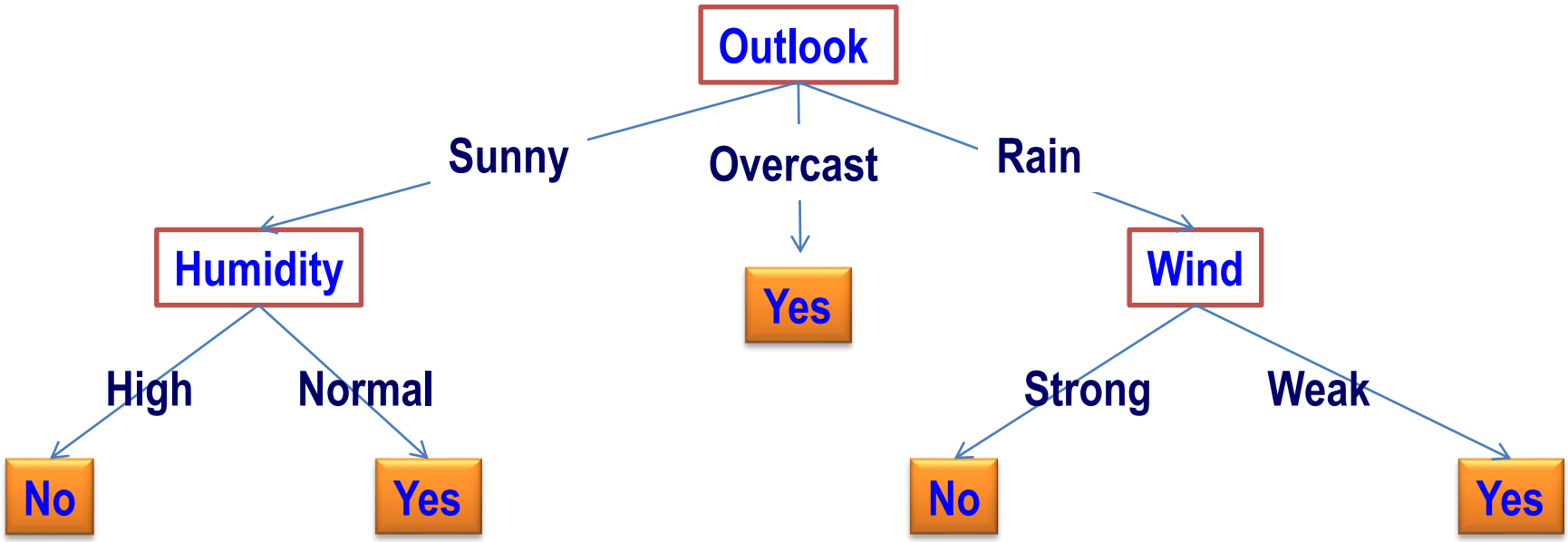
- A hierarchical data structure that represents data by implementing a divide and conquer strategy
- Can be used as a non-parametric classification method
- Given a collection of examples, learn a decision tree that represents it
- Use this representation to classify new examples

Decision Trees

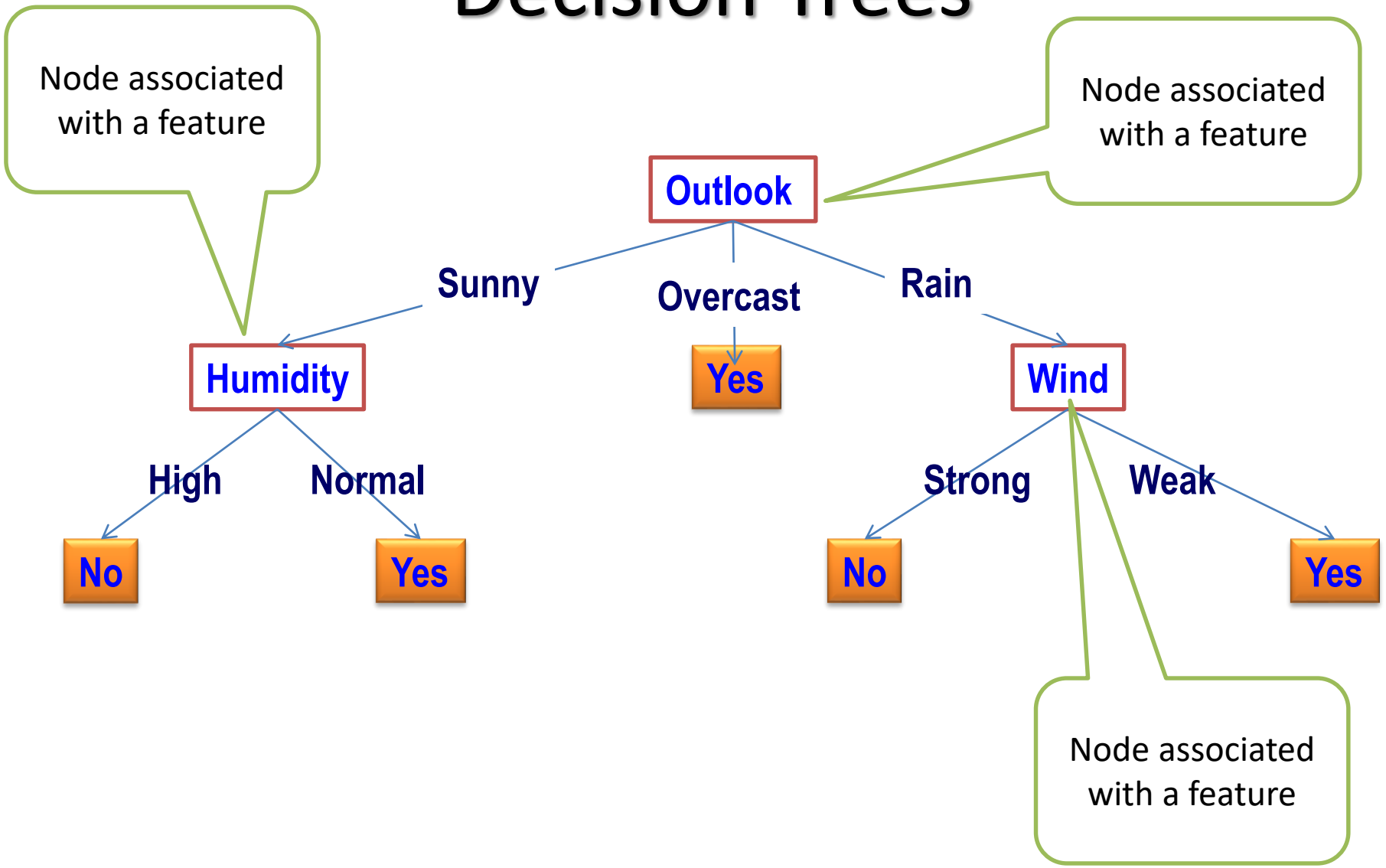
- Each node is associated with a feature (one of the elements of a feature vector that represent an object)
- Each node test the value of its associated feature
- There is one branch for each value of the feature
- Leaves specify the categories (classes)
- Can categorize instances into multiple disjoint categories – multi-class

Decision Trees

- Play Tennis Example
- Feature Vector = (Outlook, Temperature, Humidity, Wind)



Decision Trees

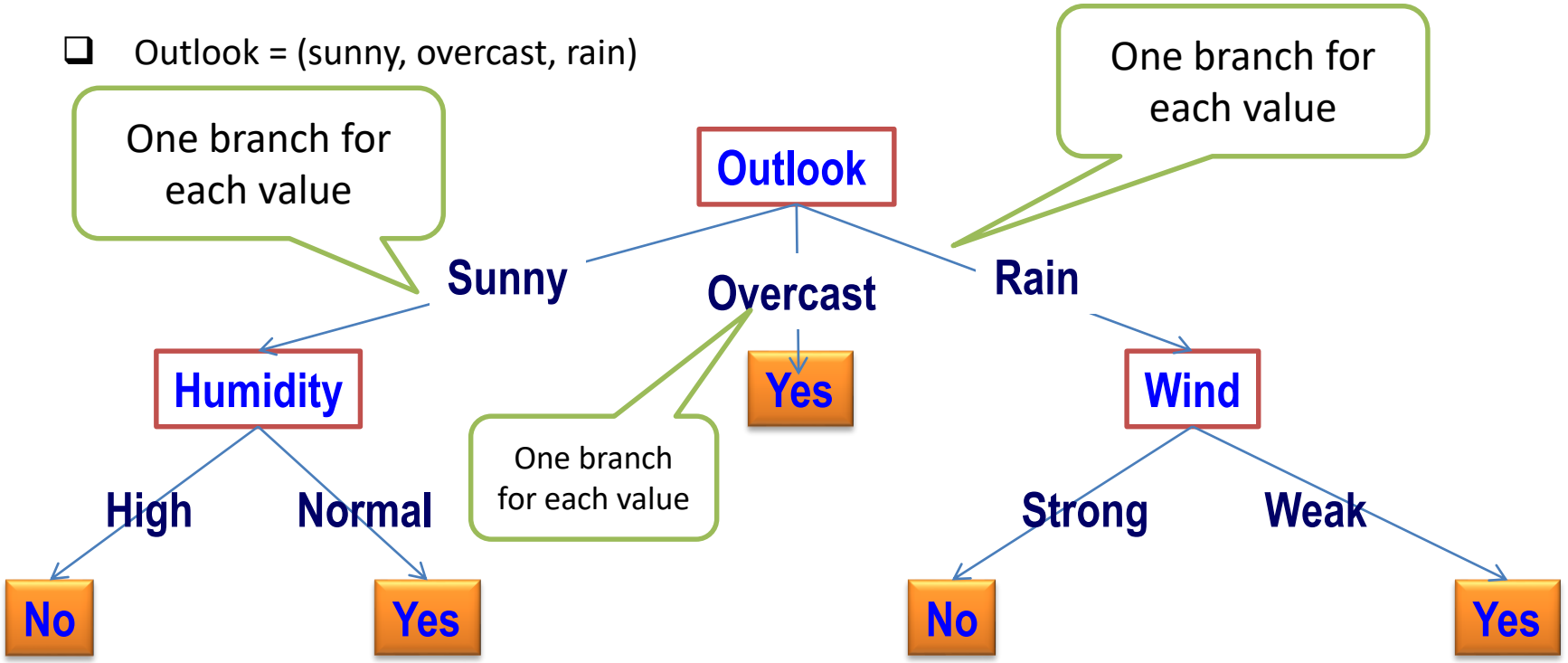


Decision Trees

- Play Tennis Example
- Feature values:
  - Outlook = (sunny, overcast, rain)
  - Temperature =(hot, mild, cool)
  - Humidity = (high, normal)
  - Wind =(strong, weak)

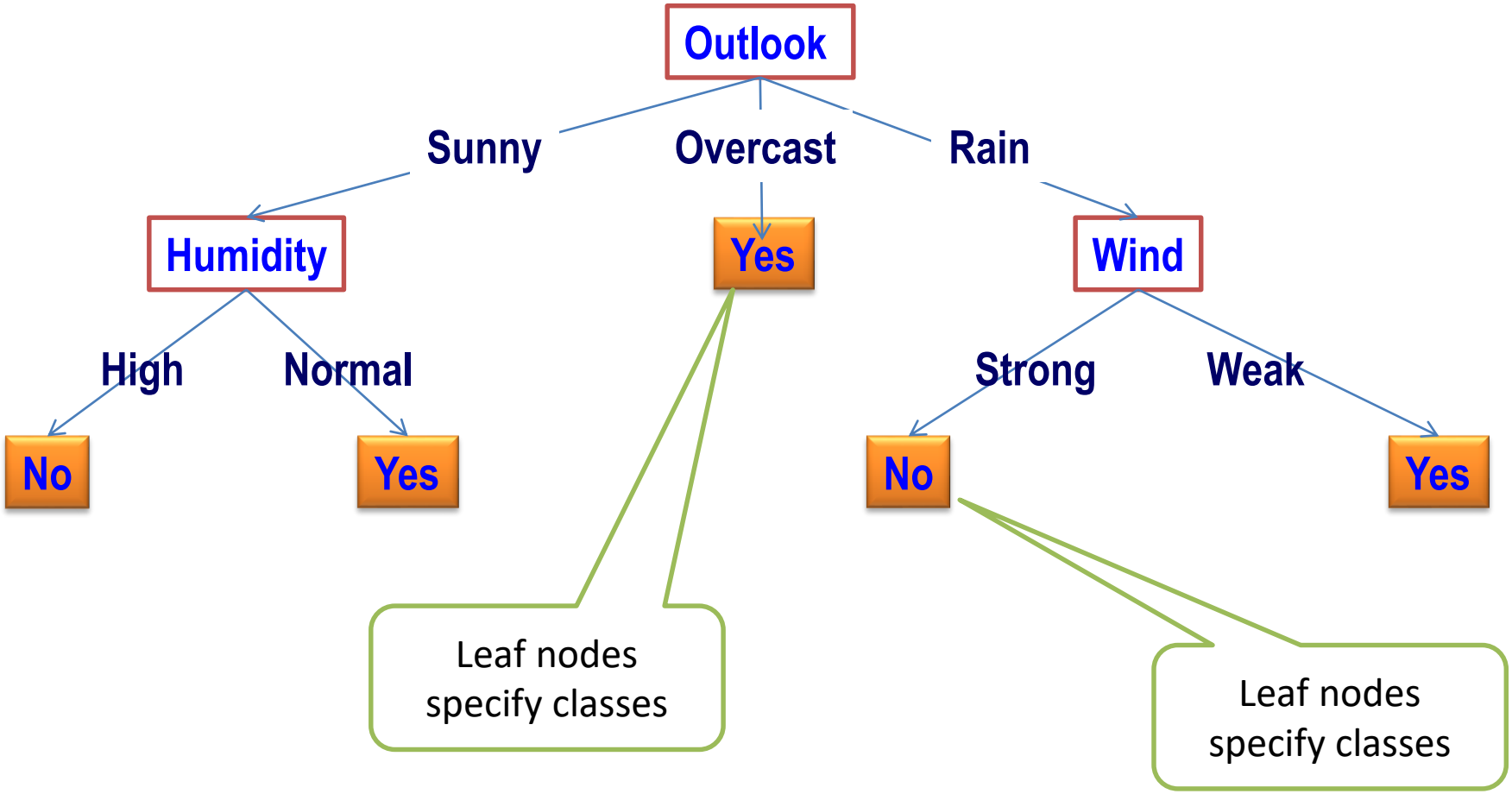
Decision Trees

- Outlook = (sunny, overcast, rain)



Decision Trees

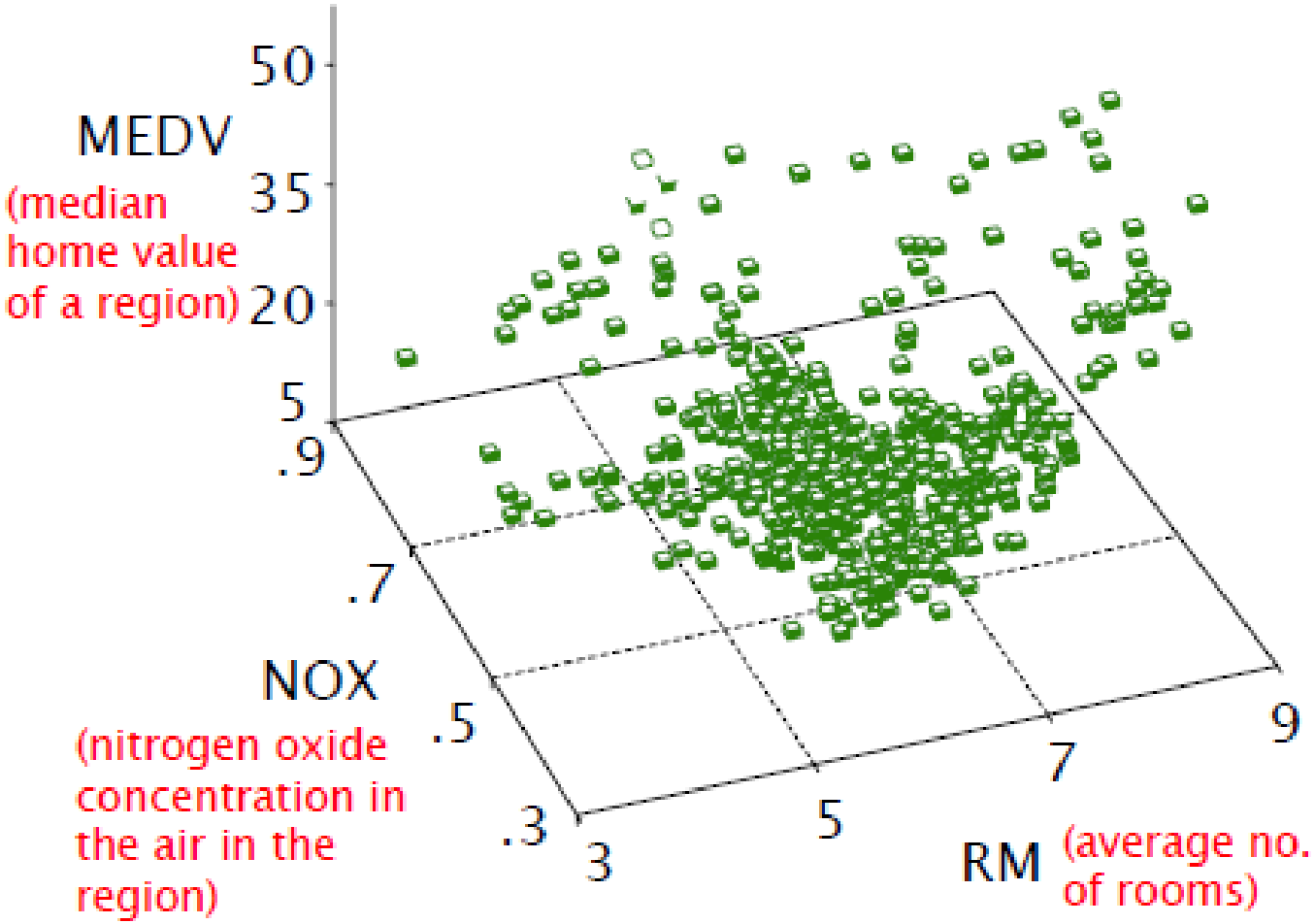
- Class = (Yes, No)



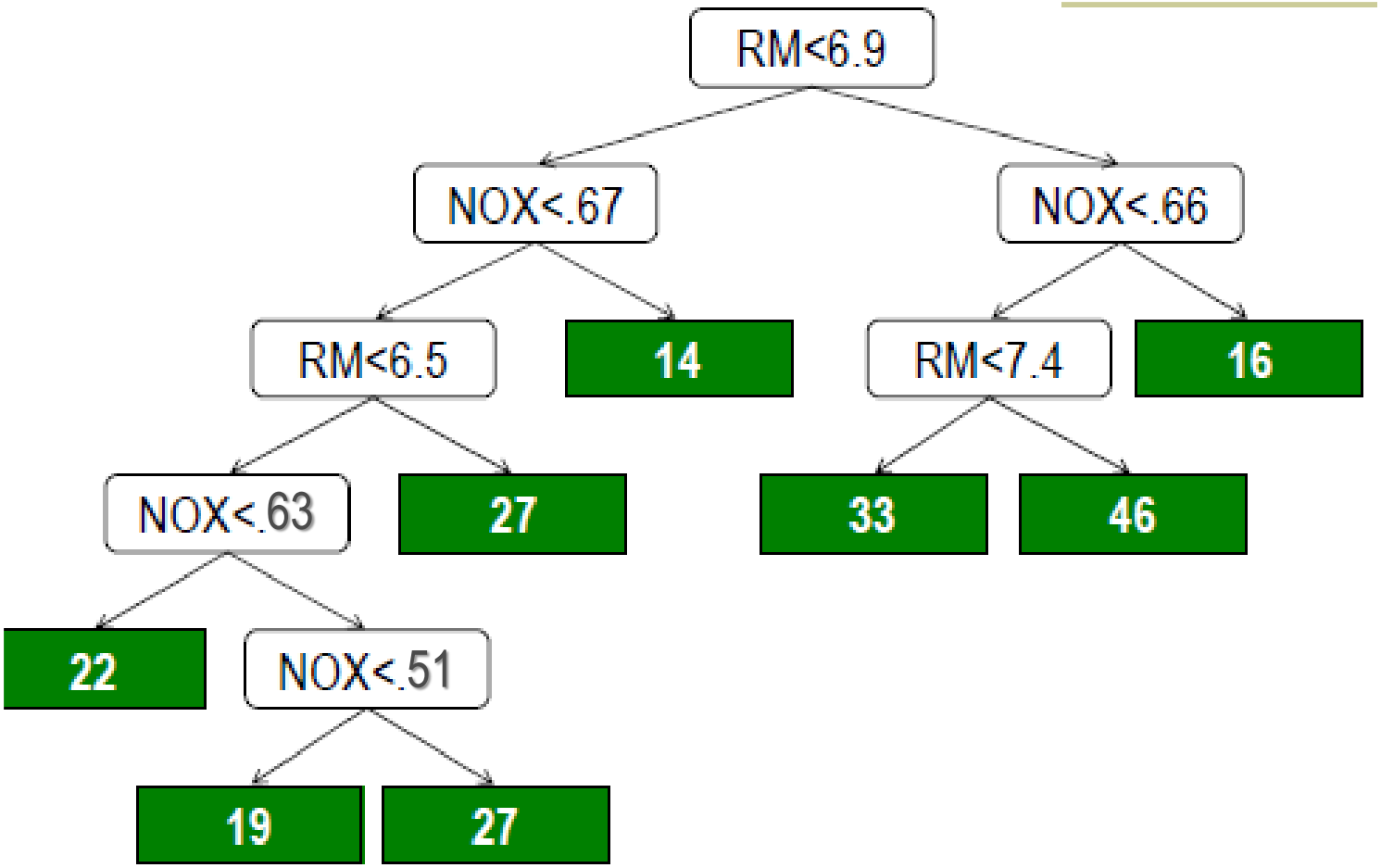
Variation of Decision Trees

- Classification tree
  - The output is discrete (label).
  - The leaves give the predicted class as well as the probability of class membership.
- Regression tree
  - The output is continuous.
  - The leaves give the predicted value of the output.
- Tree with binary splits
- Tree with multiway splits

Boston Housing Data



Regression Tree



Decision Trees

- Design Decision Tree Classifier
  - Picking the root node
  - Recursively branching

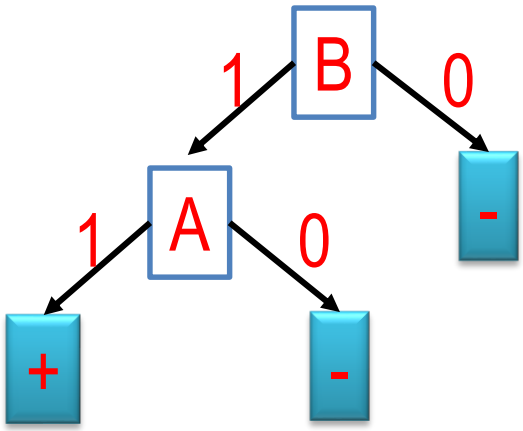
Decision Trees

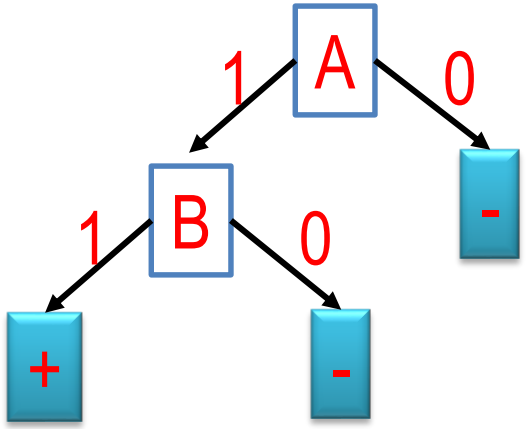
- Picking the root node
  - Consider data with two Boolean attributes (A,B) and two classes + and –

{ (A=0,B=0), - }: 50 examples
{ (A=0,B=1), - }: 50 examples
{ (A=1,B=0), - }: 3 examples
{ (A=1,B=1), + }: 100 examples

Decision Trees

- Picking the root node
  - Trees looks structurally similar; which attribute should we choose?





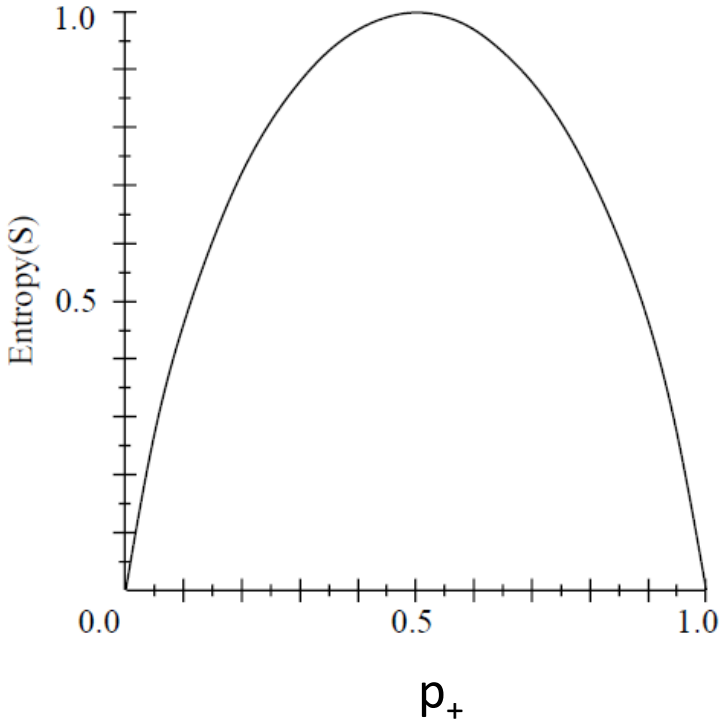
Decision Trees

- Picking the root node

  - The goal is to have the resulting decision tree as small as possible (Occam's Razor)
  - The main decision in the algorithm is the selection of the next attribute to condition on (start from the root node).
  - We want attributes that split the examples to sets that are relatively pure in one label; this way we are closer to a leaf node.
  - The most popular heuristics is based on information gain, originated with the ID3 system of Quinlan.

Entropy

- S is a sample of training examples
- p_+ is the proportion of positive examples in S
- p_- is the proportion of negative examples in S
- Entropy measures the impurity of S



$$Entropy(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

Entropy(IIP)

- The idea: associate information with probability
- A random event *E* with probability *P*(*E*) contains:

$$I(E) = log(\frac{1}{P(E)}) = -log(P(E))$$

units of information

- Suppose that grey level values are generated by a random variable, then r_k contains:

$$I(r_k) = -log(P(r_k))$$

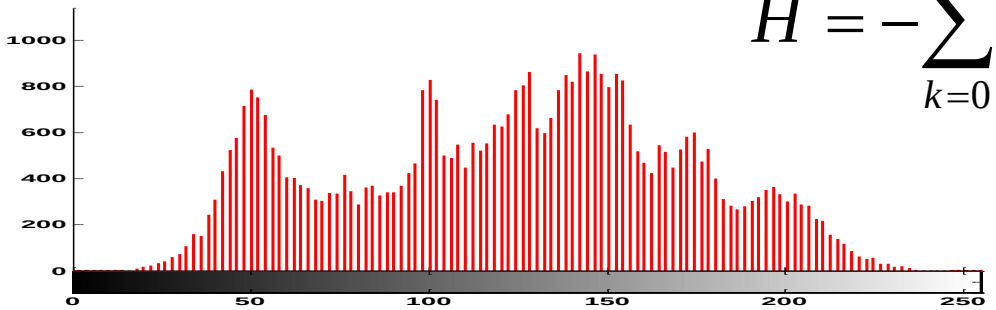
Note: I(E)=0 when P(E)=1

units of information

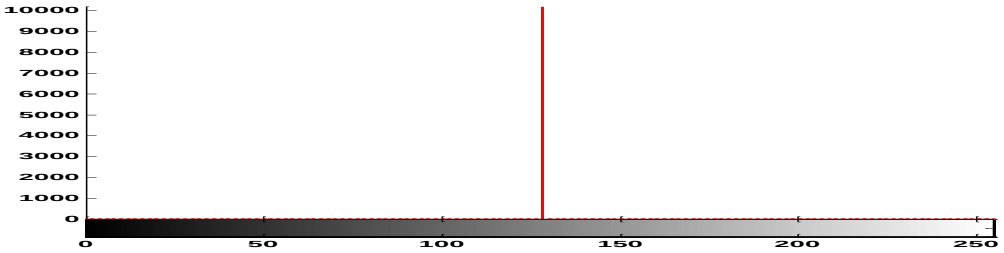
Entropy(IIP)

- Entropy is the average information content of an image, a measure of histogram dispersion

$$H = -\sum_{k=0}^{L-1} p(r_k)\log_2 p(r_k)$$



entropy=7.4635



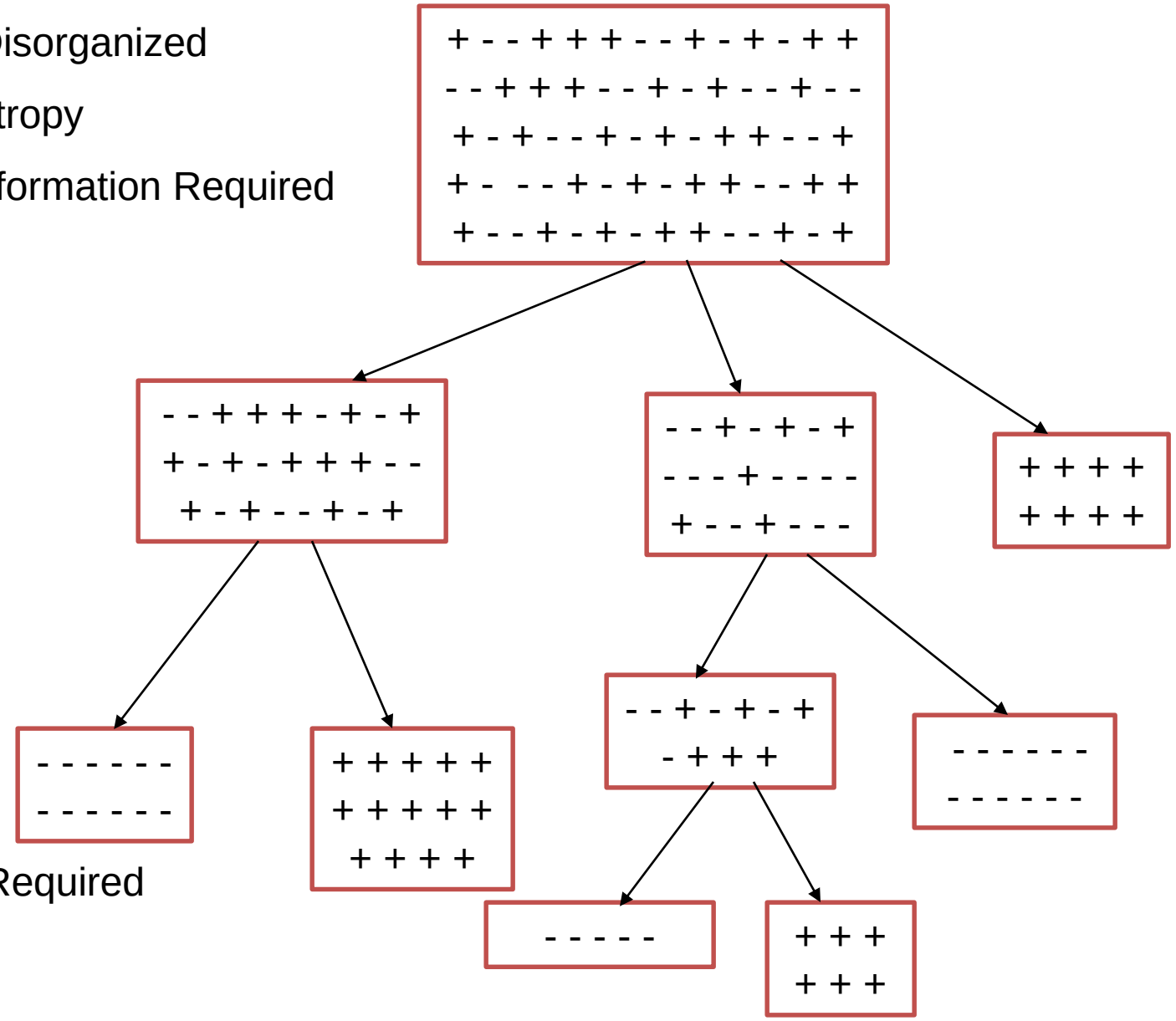
entropy=0

- Can't compress to less than H bits/pixel without losing information

Highly Disorganized

High Entropy

Much Information Required



Highly Organized

Low Entropy

Little Information Required

Information Gain

- Gain (S, A) = expected reduction in entropy due to sorting on A

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

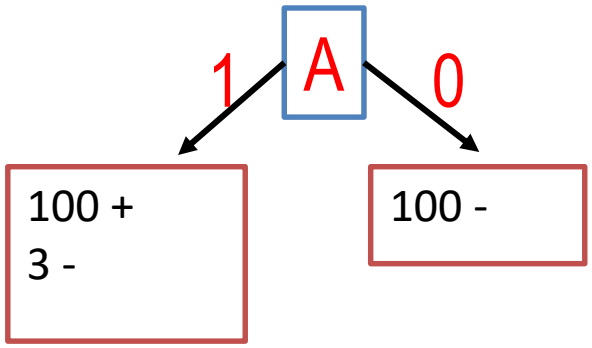
- Values (A) is the set of all possible values for attribute A, S_v is the subset of S which attribute A has value v, |S| and | S_v | represent the number of samples in set S and set S_v respectively

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

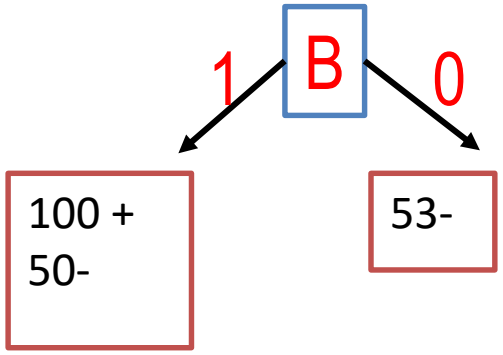
Information Gain

Example: Choose A or B ?

Split on A



Split on B



Example

Play Tennis Example

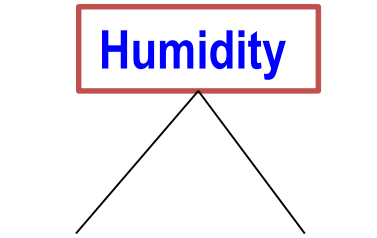
$$\mathbf{Entropy(S)} = -\tfrac{9}{14}\log(\tfrac{9}{14}) - \tfrac{5}{14}\log(\tfrac{5}{14}) = \mathbf{0.94}$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

Example

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

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No



High 3+,4- *E*=0.985

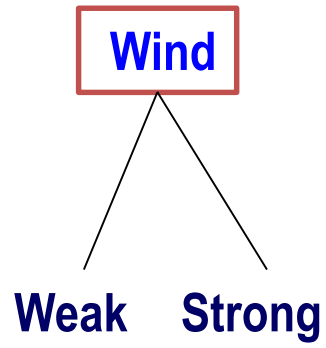
Normal 6+,1- *E*=0.592

***Gain(S,* Humidity*)* = .94 - 7/14 * 0.985 - 7/14 *.592 = 0.151**

Example

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

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No



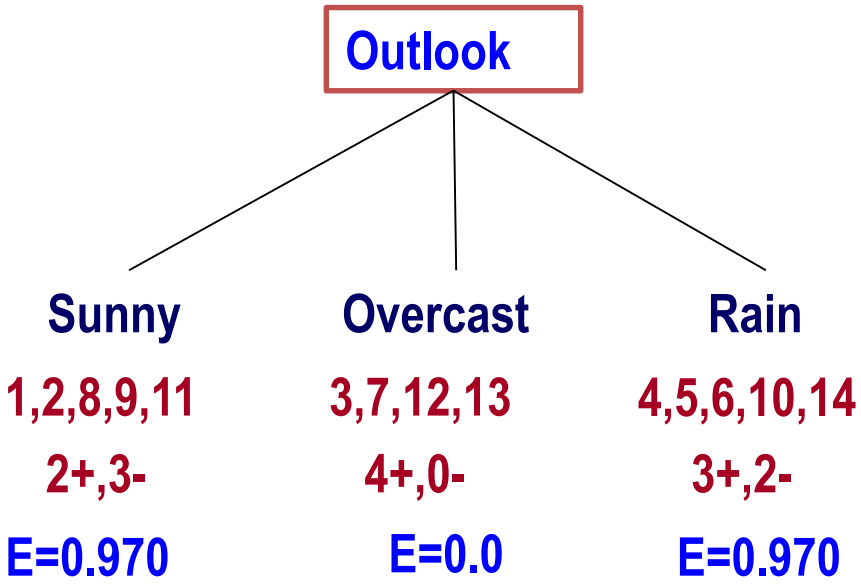
6+,2- 3+,3-

E=0.811 *E*=1.0

Gain(S, Wind*)* = .94 - 8/14 * 0.811 - 6/14 * 1.0 = 0.048

Example

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

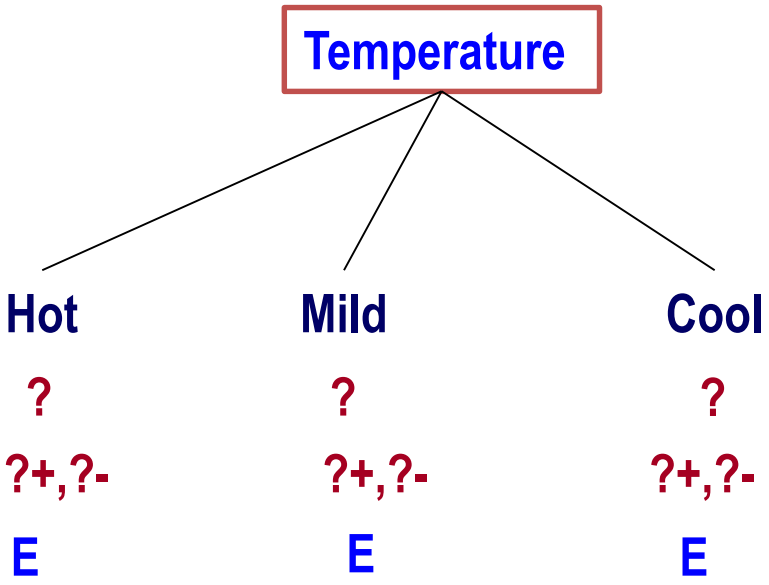


Gain(S, Outlook*)* = 0.246

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

Example

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

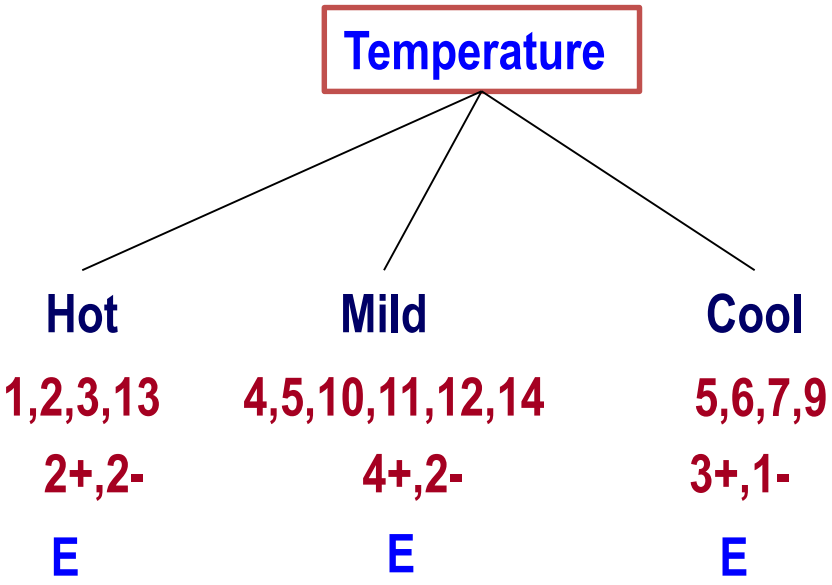


***Gain(S,* Temperature*)* = ?**

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

Example

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

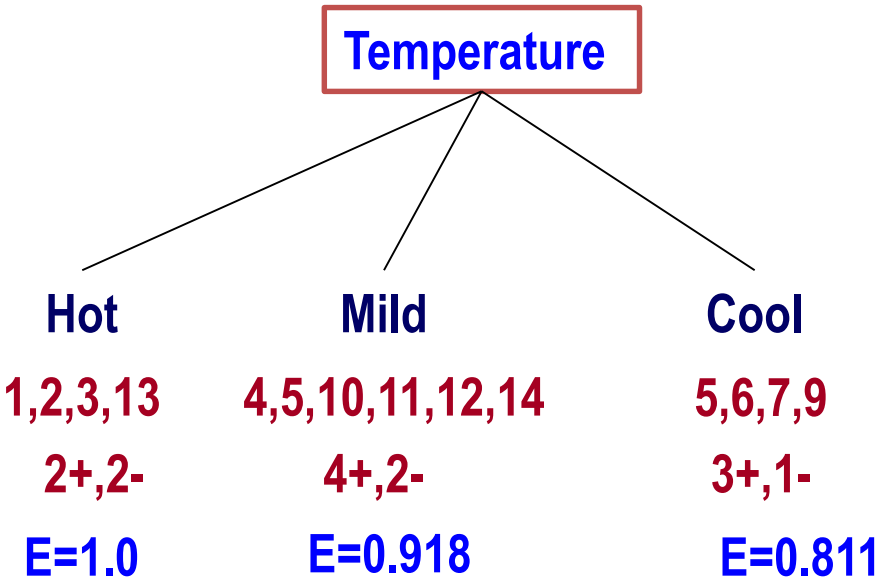


Gain(S, Temperature*)* = ?

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

Example

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



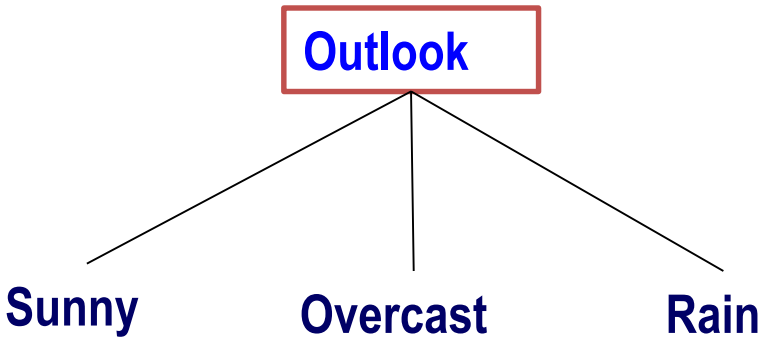
Gain(S, Temperature*)* = 0.029

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

Example

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

Pick Outlook as the root



Gain(S, Outlook) = 0.246

Gain(S, Temperature) = 0.029

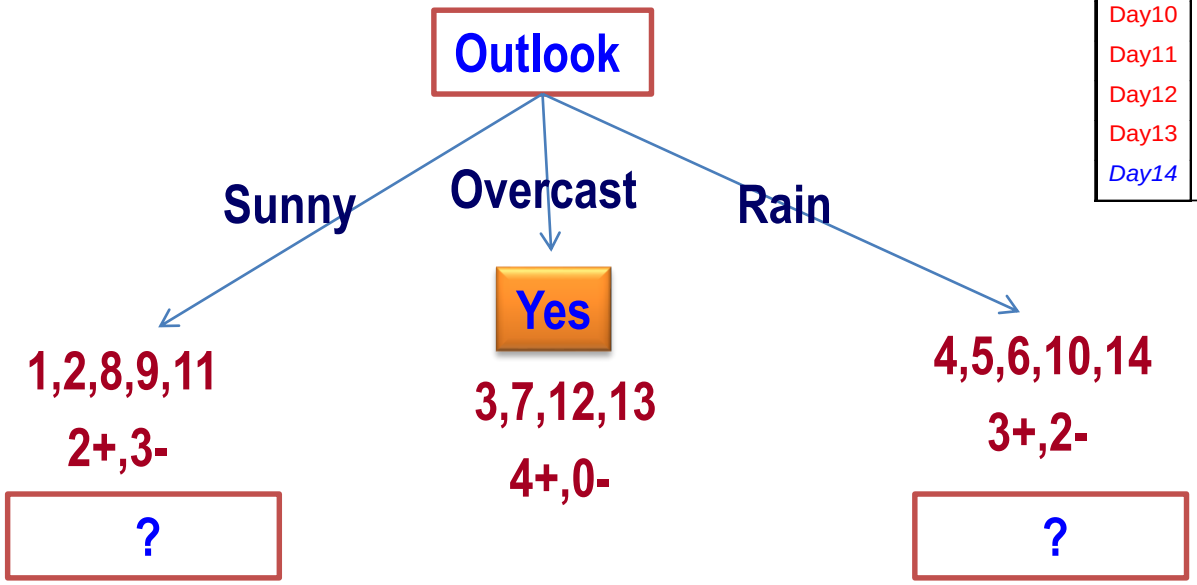
Gain(S, Humidity) = 0.151

Gain(S, Wind) = 0.048

Example

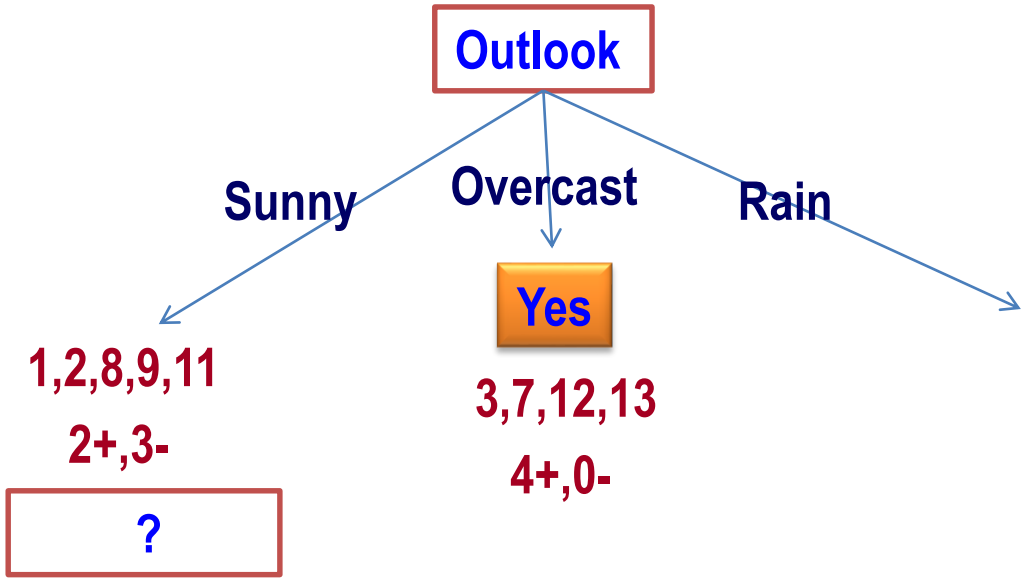
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

Pick Outlook as the root



Continue until: Every attribute is included in path, or, all examples in the leaf have same label

Example



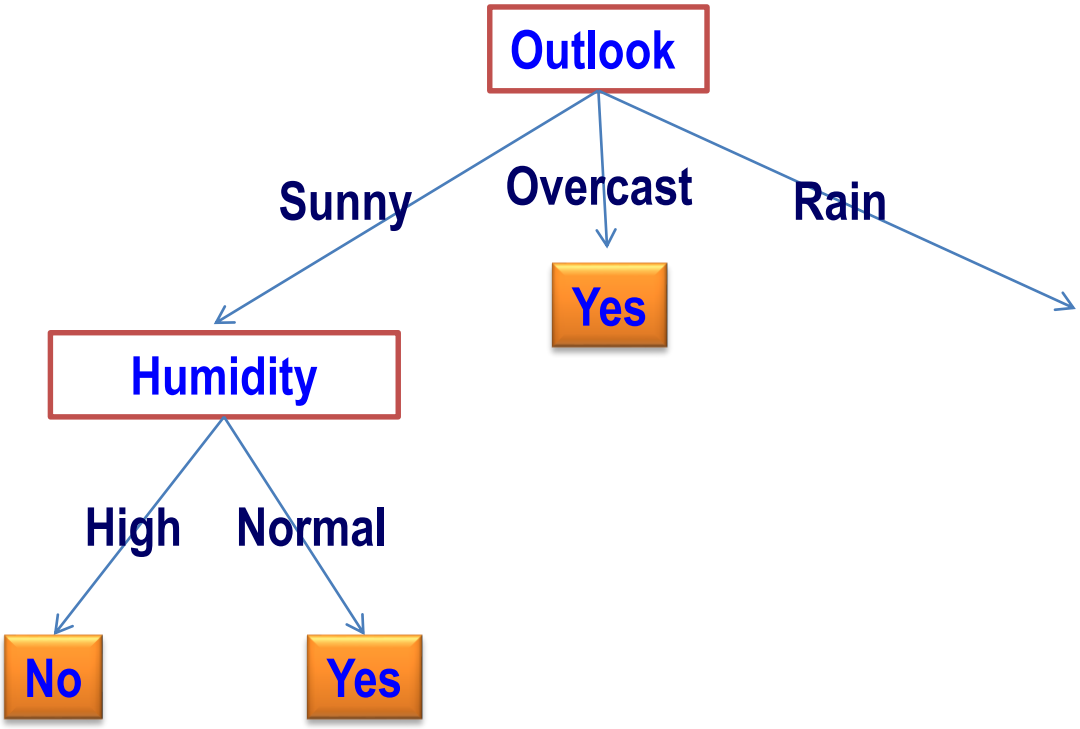
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Gain (S_{sunny}, Humidity) = .97-(3/5) * 0-(2/5) * 0 = .97
Gain (S_{sunny}, Temp) = .97- 0-(2/5) *1 = .57
Gain (S_{sunny}, Wind) = .97-(2/5) *1 - (3/5) *.92 = .02

Example

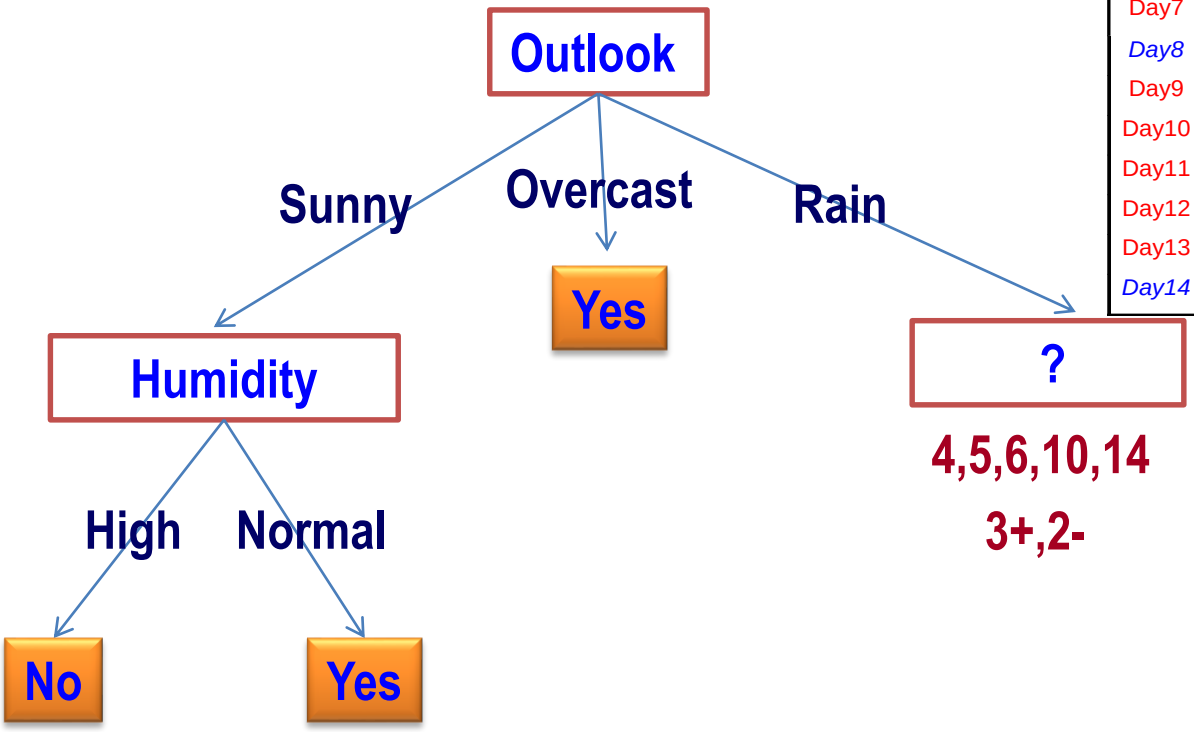
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No



Gain (S_{sunny}, Humidity) = .97-(3/5) * 0-(2/5) * 0 = .97
Gain (S_{sunny}, Temp) = .97- 0-(2/5) *1 = .57
Gain (S_{sunny}, Wind) = .97-(2/5) *1 - (3/5) *.92 = .02

Example

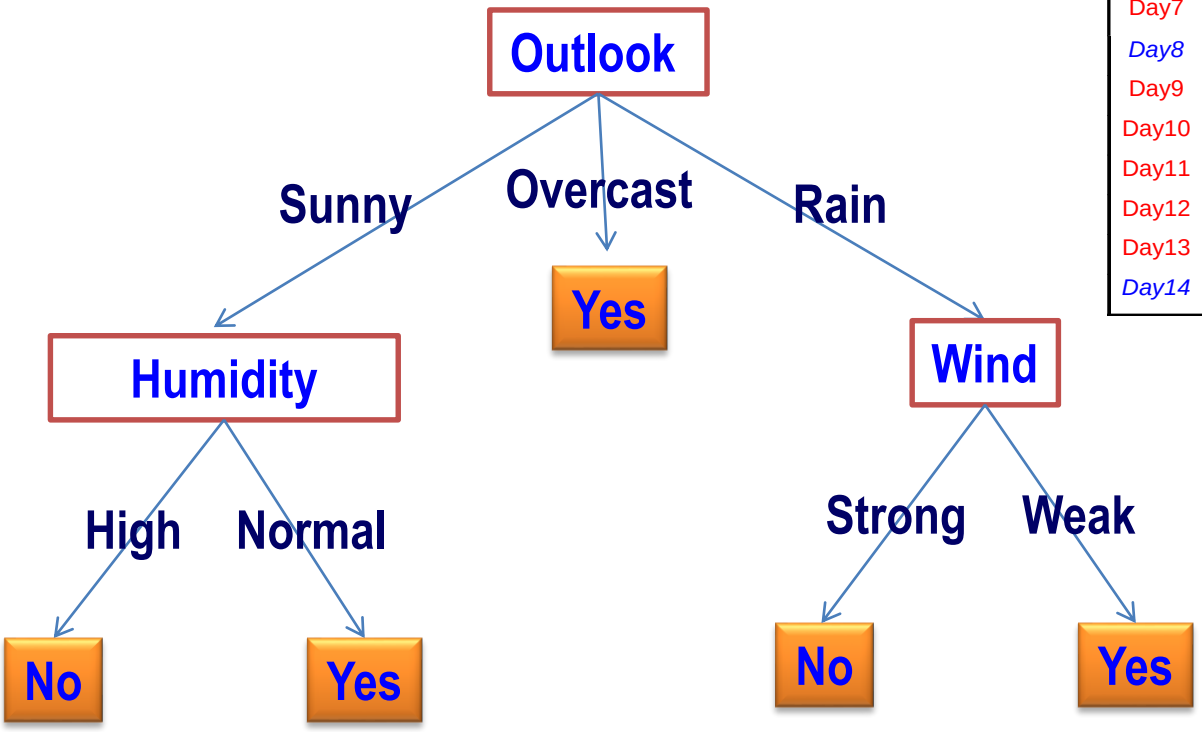
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No



Gain (S_{rain}, Humidity) =
Gain (S_{rain}, Temp) =
Gain (S_{rain}, Wind) =

Example

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No



Summary on Build a Decision Tree

- Recursive partitioning
  - a top-down, greedy algorithm to fit the decision tree for the data.
- Top-down
  - Starting at the root node, split the data into subgroups that are as homogeneous as possible with respect to the output.
- Greedy method
  - Always make a locally optimal choice in the hope that this will lead to a globally optimal solution.

Three Steps in Building Decision Tree

- Selection of the best split
  - Which feature could give the “best” split – information gain?
- Stop-splitting rule
  - When should the splitting stop?
- Assignment of each leaf node to a class
  - Predict the value of the output at each leaf node.

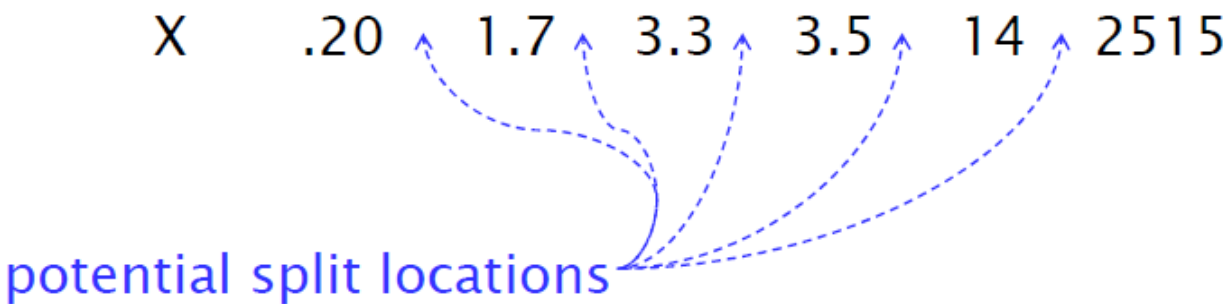
Possible Splits

Split on a discrete feature with ***B*** branches.

Branch (*B*)		
2	**3**	**4**
A, BCD	A, B, CD	A, B, C, D
B, ACD	A, C, BD	
C, ABD	A, D, BC	
D, ABC	B, C, AD	
AB, CD	B, D, AC	
AC, BD	C, D, AB	
AD, BC		

Possible Splits

Split on a continuous feature.



Selection of Best Splits

- Exhaustively examining all possible splits is time consuming.
- By default, algorithms will use exhaustive search if no. of possible splits is less than a given number (e.g. 5000).
- Otherwise, a clustering of levels of a feature is used to limit the possible splits to consider.
- An alternative way is to consider binary splits only

Remarks

- The **process** of selecting the best split on a node:

  1. Select the best split on each feature (i.e. choose number of branches and cut-off points).

  2. Select the best of these.

Stopping Criteria for Tree Induction

- Stop expanding a node when all the data belong to the same class
- Stop expanding a node when all the data have similar feature values
- Early termination

Stop Splitting Rule

- A simple method:
  - Continue splitting until every node is pure or contains only one instance.
  - Fit training data perfectly but may predict poorly on new data.

- Two approaches:
  - Top-down stopping rules (pre-pruning).
  - Bottom-up assessment criteria (post-pruning).

Assignment of Each Leaf Node to A Class

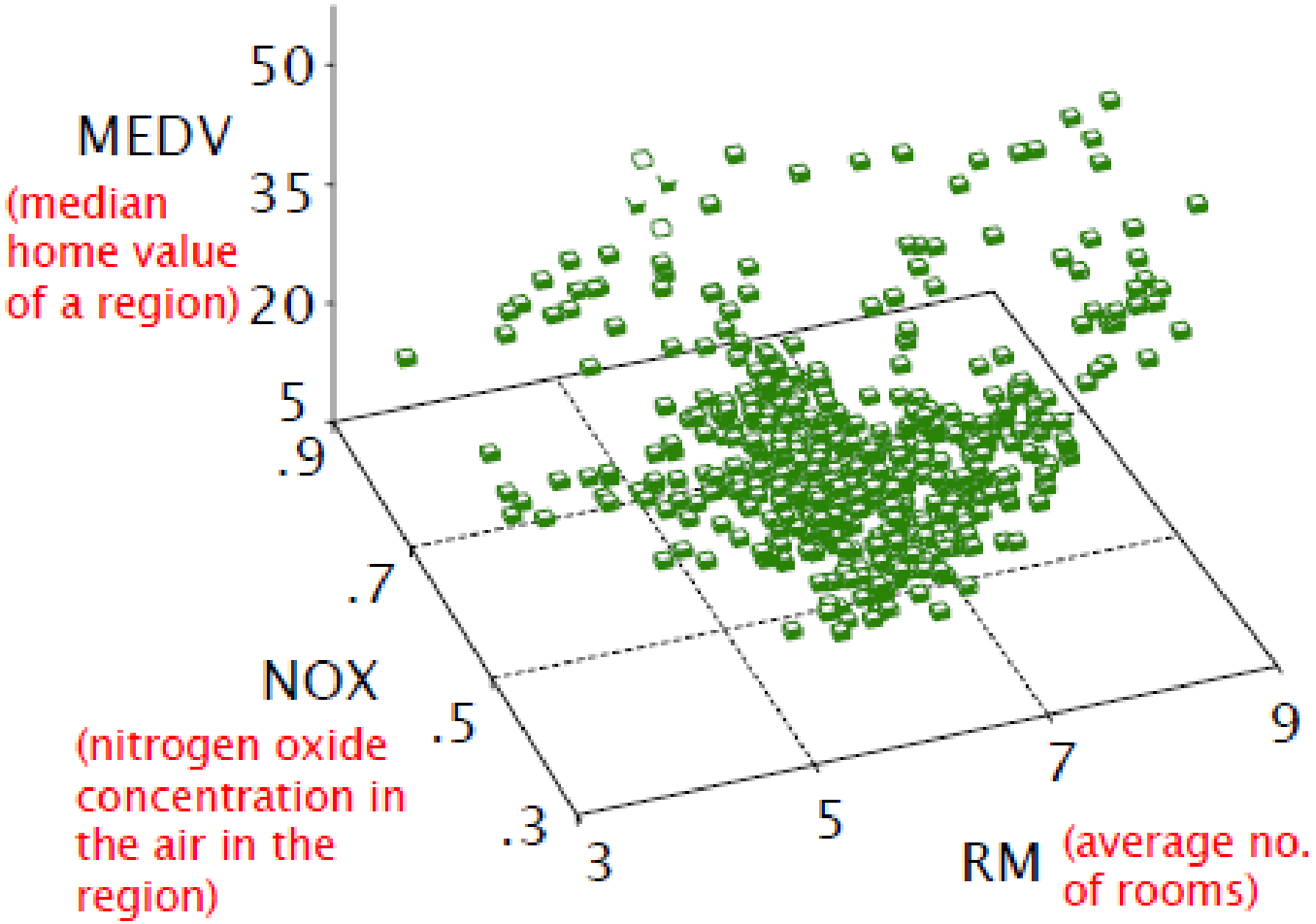
- **Classification tree**
  - Classify an input sample with the most common label in the leaf node (the percentage of the label is the probability).

- **Regression tree**
  - Predict an input in a node by the sample mean of the output values in the node.

Advantages of Trees

- **Easy to interpret**
  - Tree structured presentation.
- **Allow mixed input data types**
  - Discrete or continuous.
- **Allow discrete or continuous output**
- **Robust to outliers in feature vectors**
- **No problem with missing values**
- **Automatically**
  - Accommodates nonlinearity.
  - Selects input variables.

Boston Housing Data



Disadvantages of Trees

- **Most algorithms use univariate splits (split on only one variable)**
  - Solution: Linear combination split (a1x1+a2x2< c?).
- **Unstable fitted tree**
  - Often a small change in the data result in a very different series of splits.
- **Lack of smoothness (step function) in regression tree**
- **Splitting turns continuous input features into discrete features**
- **Spitting using a "greedy" algorithm**
  - While each split is optimal, the overall tree is not.

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary.
- Training error no longer provides a good estimate of how well the tree will perform on previously unseen data.
- Need new ways for estimating errors.

How to Address Overfitting

Pre-Pruning (Early Stopping Rule)

- Stop the algorithm before it becomes a fully-grown tree.
- Typical stopping conditions for a node
  - Stop if all samples belong to the same class.
  - Stop if all the features values are the same.
- More restrictive conditions
  - Stop if number of samples is less than some user-specified threshold.
  - Stop if expanding the current node does not improve impurity measures, e.g., information gain.

How to Address Overfitting

Post-pruning

- Grow decision tree to its entirety.
- Trim the nodes of the decision tree in a bottom-up fashion.
- If generalization error improves after trimming, replace sub-tree by a leaf node.
- Class label of leaf node is determined from majority class of samples in the sub-tree.

Any Questions?