CDS503: Machine Learning

Topic 4

Classification Non-Parametric – Decision Tree

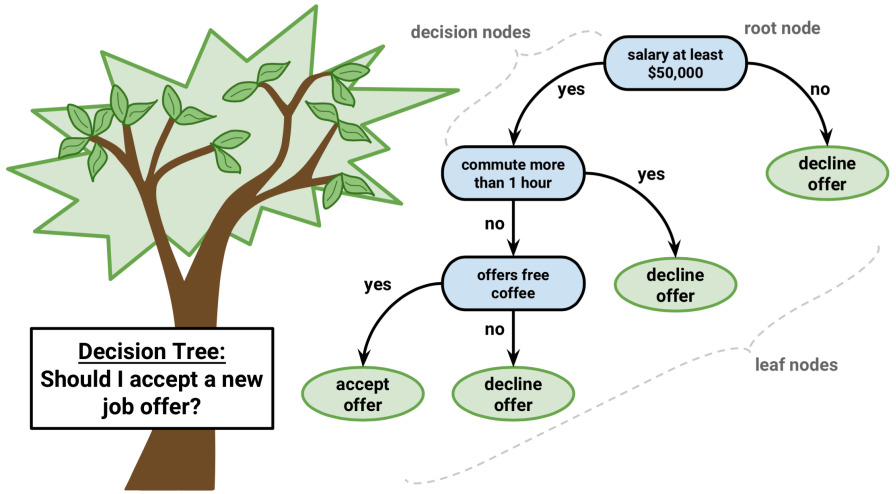
Prof. Madya DR UMI KALSOM YUSOF
SCHOOL OF COMPUTER SCIENCES
UNIVERSITI SAINS MALAYSIA (USM)

Contents

- Introduction
- Constructing Decision Tree
- Rule Extraction from Tree
- Decision Tree Classification Task
- Pruning
- Strengths and Weaknesses

Decision Tree Algorithm

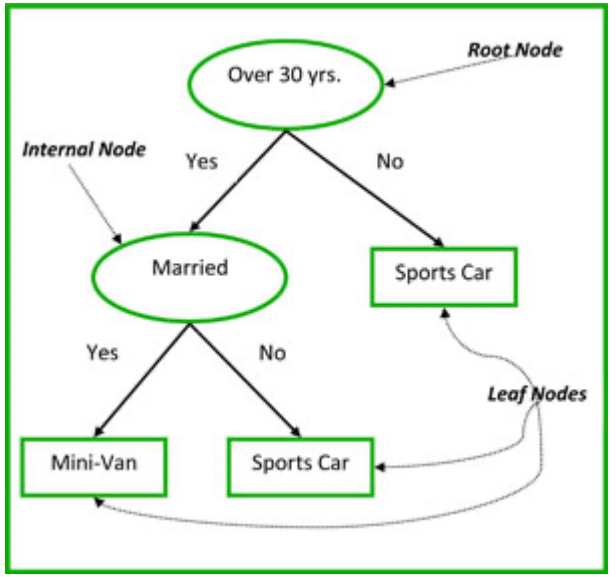
- A classification scheme which generates a tree and a set of rules from given data set.
- The set of records available for developing classification methods is divided into two disjoint subsets – a *training set* and a *test set*.
- The attributes of the records are categorised into two types:
  - Attributes whose domain is numerical are called numerical attributes.
  - Attributes whose domain is not numerical are called the categorical attributes.



Decision Tree Algorithm

- *A decision tree* is a tree with the following properties:
  - An inner node represents an attribute.
  - An edge represents a test on the attribute of the father node.
  - A leaf represents one of the classes.

- Construction of a decision tree
  - Based on the training data
  - Top-Down strategy



Decision Tree *Example*

- The data set has several attributes.
- There is a special attribute: the attribute *class* is the class label.
- The attributes, *temp* (temperature) and *humidity* are numerical attributes
- Other attributes are categorical, that is, they cannot be ordered.
- Based on the training data set, we want to find a set of rules to know what values of *outlook, temperature*, *humidity* and *wind*, determine whether or not to play baseball.

Table 6.1 Training Data Set

OUTLOOK	TEMP(F)	HUMIDITY(%)	WINDY	CLASS
sunny	79	90	true	no play
sunny	56	70	false	play
sunny	79	75	true	play
sunny	60	90	true	no play
overcast	88	88	false	no play
overcast	63	75	true	play
overcast	88	95	false	play
rain	78	60	false	play
rain	66	70	false	no play
rain	68	60	true	no play

Decision Tree *Example*

- We have five leaf nodes.
- In a decision tree, each leaf node represents a rule.

- We have the following rules corresponding to the tree given in figure.

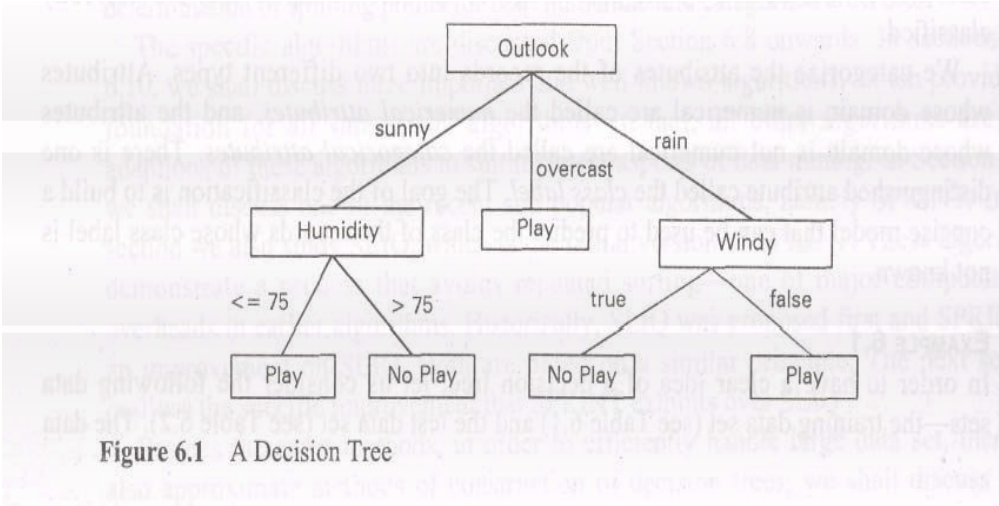
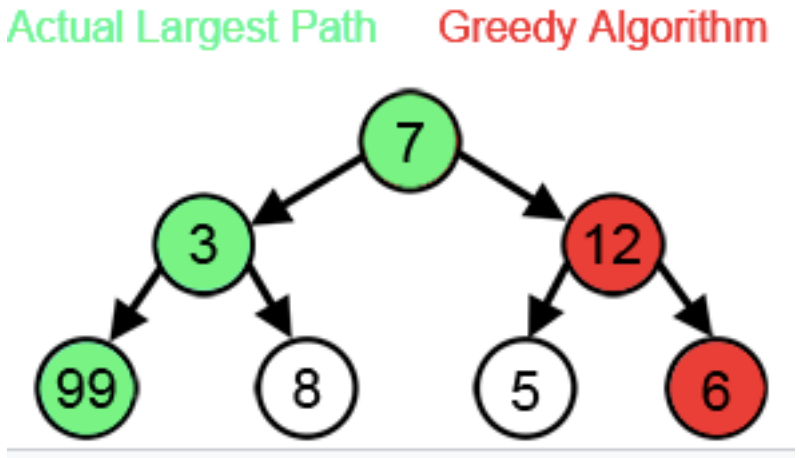


Figure 6.1 A Decision Tree

- *RULE 1 If it is sunny and the humidity is not above 75%, then play.*
- *RULE 2 If it is sunny and the humidity is above 75%, then do not play.*
- *RULE 3 If it is overcast, then play.*
- *RULE 4 If it is rainy and not windy, then play.*
- *RULE 5 If it is rainy and windy, then don't play.*

Constructing Decision Trees

Constructing Decision Tree



- Employ greedy strategy to grow a tree
- Making locally optimum decisions about which attribute to use for partitioning data

- **Hunt's algorithm**
  - Tree is constructed in a **top-down recursive divide-and-conquer** manner
  - At start, all training examples are at the root
  - Attributes are categorical (discretized if continuous-valued)
  - An attribute is selected according to a heuristic or statistical measure to be the decision attribute of that node
  - Outgoing arrows are the values of the selected attribute and the examples are partitioned based on that attribute

Constructing Decision Tree

- How should the splitting procedure stop?
  - All samples for a given node belong to the same class
  - No remaining attributes for further partitioning (majority voting is employed for classifying the leaf if not all the labels are the same)
  - No samples left

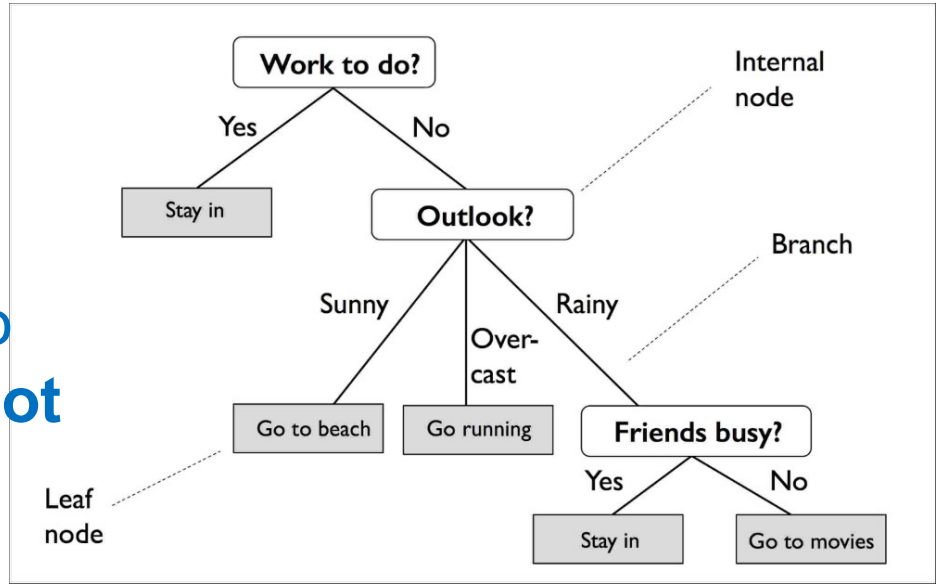
Constructing Decision Tree

- Hunt's algorithm is the basis of many tree induction algorithms
  - ID3
  - CART
  - C4.5

	Splitting criteria	Attribute type	Missing values	Pruning strategy	Outlier detection
ID3	Information Gain	Only categorical value	Do not handle missing values	No pruning is done	Susceptible to outliers
CART	Gini Index	Categorical & numeric value	Handle missing values	Cost-complexity pruning	Can handle outliers
C4.5	Gain Ratio	Categorical & numeric value	Handle missing values	Error-based pruning	Susceptible to outliers

ID3 (Iterative Dichotomizer)

- Quinlan (1986)
- Each node corresponds to a splitting attribute
- Each arc is a possible value of that attribute.
- At each node the splitting attribute is selected to be the most informative among the **attributes not yet considered** in the path from the root.



- *Entropy* is used to measure how informative is a node.
- The algorithm uses the criterion of *information gain* to determine the goodness of a split.
  - The attribute with the greatest information gain is taken as the splitting attribute, and the data set is split for all distinct values of the attribute.

Entropy

- Entropy measures the homogeneity (purity) of a set of examples.
- It gives the information content of the set in terms of the class labels of the examples.
- Consider that you have a set of examples, S with two classes, P and N. Let the set have p instances for the class P and n instances for the class N.
- So the total number of instances we have is t = p + n. The view [p, n] can be seen as a class distribution of S.

The entropy for S is defined as

$$Entropy(S) = - (p/t).\log_2(p/t) - (n/t).\log_2(n/t)$$

- Example: Let a set of examples consists of 9 instances for class positive, and 5 instances for class negative. (p = 9 and n = 5)

So $Entropy(S) = - (9/14).\log_2(9/14) - (5/14).\log_2(5/14)$

$= - (0.64286)(-0.6375) - (0.35714)(-1.48557)$

$= (0.40982) + (0.53056)$

$= 0.940$

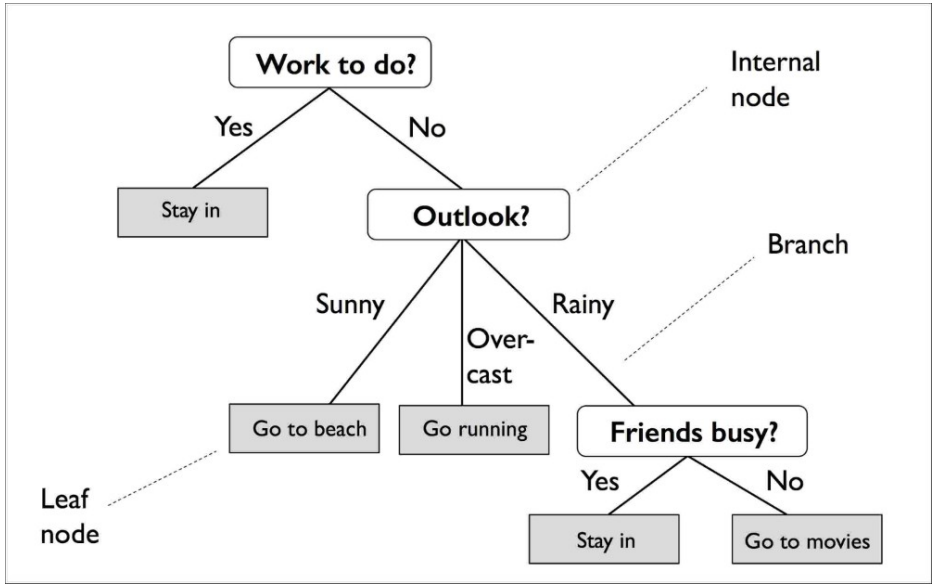
Entropy

The entropy for a completely pure set is 0 and is 1 for a set with equal occurrences for both the classes.

i.e. Entropy[14,0] = - (14/14).log2(14/14) - (0/14).log2(0/14)
= -1.log2(1) - 0.log2(0)
= -1.0 - 0
= 0 completely pure

i.e. Entropy[7,7] = - (7/14).log2(7/14) - (7/14).log2(7/14)
= - (0.5).log2(0.5) - (0.5).log2(0.5)
= - (0.5).(-1) - (0.5).(-1)
= 0.5 + 0.5
= 1 Equal occurrences for **both** the classes

Selecting the Best Split



- Which attribute is the best?
  - Measures for selecting the best split – **degree impurity** in the child node (smaller degree of impurity, more skewed class distribution)
  - Node with class distribution (0, 1) – zero impurity
  - Node with uniform class distribution (0.5, 0.5) – the highest impurity

Impurity Measures

- Entropy (Quinlan, ID3, C4.5): Measure of the amount of uncertainty or randomness in data (lower values imply less uncertainty, higher values imply high uncertainty)

$$Entropy = \sum_j -p_j log_2 p_j$$

- Gini Index (Breiman, CART)

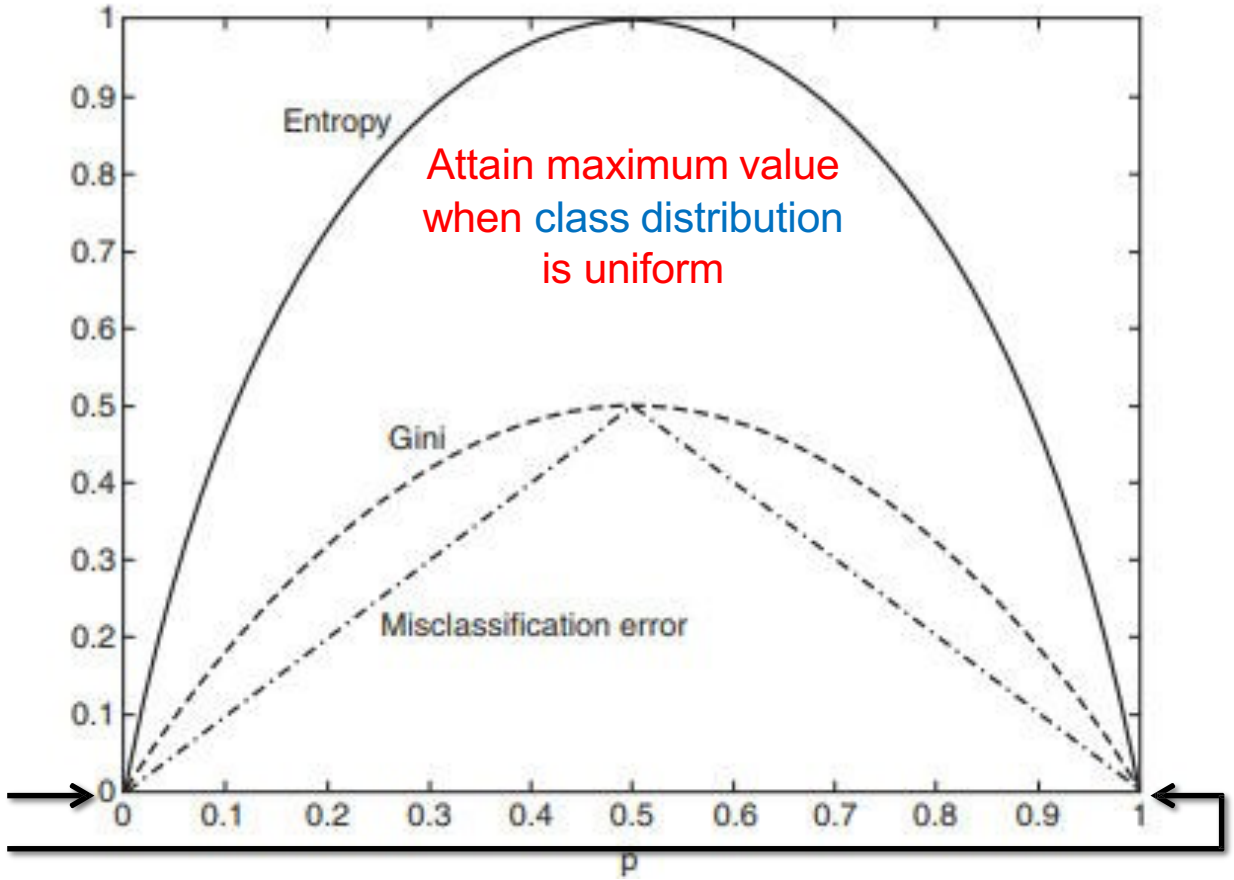
$$Gini\ Index = 1 - \sum_j p_j^2$$

- Classification Error

$$Classification\ Error = 1 - max\{p_j\}$$

Impurity Measures

- Comparison among impurity measures for binary classification problems



Attribute Selection Measures

- Gain: Compare the degree of impurity of parent node (before splitting) to child node (after splitting) (the larger the difference, the better the test condition)
- **Information gain**: Expected reduction in **entropy** caused by partitioning the examples according to a given attribute
  - Select the attribute A with the highest information gain

Working Example

Constructing Decision Tree

Decide whether the **weather is amenable to playing baseball**.

Information Gain

Which attribute is selected as a root? Therefore, information gain for all the four weather attributes has to be computed.

Day	Outlook	Temperature	Humility	Wind	Play ball
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D7	Overcast	Cool	Normal	Strong	Yes
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
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D6	Rain	Cool	Normal	Strong	No
D8	Sunny	Mild	High	Weak	No
D14	Rain	Mild	High	Strong	No

Step 1: Calculate entropy of the target (Play Base Ball, S)

Entropy(S) = Entropy(5/14, 9/14)
= Entropy (0.36, 0.64)
= - (0.36 $\log_2$ 0.36) – (0.64 $\log_2$ 0.64)
= **0.94**

"Yes" members "No" members

$$Entropy(S) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.9403$$

	Yes	No
Play ball	9	5

Information gain for outlook:

Day	Outlook	Temperature	Humility	Wind	Play ball
D7	Overcast	Cool	Normal	Strong	Yes
D3	Overcast	Hot	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D12	Overcast	Mild	High	Strong	Yes
D5	Rain	Cool	Normal	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No

Outlook	Playball			
Overcast	Yes	4	4	14
Overcast	No	0		
Rain	Yes	3	5	
Rain	No	2		
Sunny	Yes	2	5	
Sunny	No	3		

Step 2: Split the data set based on different attributes.
Calculate entropy for each branch - Outlook

$$Entropy(S_{overcast}) = -\frac{4}{4}\log_2\frac{4}{4} = 0$$

$$Entropy(S_{rain}) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.9710$$

$$Entropy(S_{sunny}) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.9710$$

What is the Gain(S,Outlook)?

$$Gain(S, Outlook)$$
$$= Entropy(S) - \frac{4}{14}Entropy(S_{overcast}) - \frac{5}{14}Entropy(S_{rain}) - \frac{5}{14}Entropy(S_{sunny})$$
$$= 0.9403 - \left(\frac{4}{14}\times 0\right) - \left(\frac{5}{14}\times 0.9710\right) - \left(\frac{5}{14}\times 0.9710\right) = 0.2468$$

Information gain for temperature:

Day	Outlook	Temperature	Humility	Wind	Play ball
D5	Rain	Cool	Normal	Weak	Yes
D7	Overcast	Cool	Normal	Strong	Yes
D9	Sunny	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D4	Rain	Mild	High	Weak	Yes
D12	Overcast	Mild	High	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D14	Rain	Mild	High	Strong	No
D10	Rain	Mild	Normal	Weak	Yes

temperature	Play ball			
cool	Yes	3	4	14
cool	No	1		
hot	Yes	2	4	
hot	No	2		
mild	Yes	4	6	
mild	No	2		

Step 2: Split the data set based on different attributes. Calculate entropy for each branch - temperature

$$Entropy(S_{cool}) = -\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4} = 0.8113$$

$$Entropy(S_{hot}) = -\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4} = 1.0000$$

$$Entropy(S_{mild}) = -\frac{4}{6}\log_2\frac{4}{6} - \frac{2}{6}\log_2\frac{2}{6} = 0.9183$$

What is the Gain(S,Temperature)?

$$Gain(S, Temperature) = Entropy(S) - \frac{4}{14}Entropy(S_{cool}) - \frac{4}{14}Entropy(S_{hot}) - \frac{6}{14}Entropy(S_{mild})$$

$$= 0.9403 - \left(\frac{4}{14} \times 0.8113\right) - \left(\frac{4}{14} \times 1.0000\right) - \left(\frac{6}{14} \times 0.9183\right) = 0.0292$$

Constructing Decision Tree

Information gain for humidity:

Day	Outlook	Temperature	Humility	Wind	Play ball
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D12	Overcast	Mild	High	Strong	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No
D14	Rain	Mild	High	Strong	No
D5	Rain	Cool	Normal	Weak	Yes
D7	Overcast	Cool	Normal	Strong	Yes
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No

Humidity	Play ball			
high	Yes	3	7	14
high	No	4		
normal	Yes	6	7	
normal	No	1		

Step 2: Split the data set based on different attributes. Calculate entropy for each branch - humidity

$$Entropy(S_{high}) = -\frac{3}{7}\log_2\frac{3}{7} - \frac{4}{7}\log_2\frac{4}{7} = 0.9852$$

$$Entropy(S_{normal}) = -\frac{6}{7}\log_2\frac{6}{7} - \frac{1}{7}\log_2\frac{1}{7} = 0.5917$$

What is the Gain(S,Humidity)?

$$Gain(S, Humidity) = Entropy(S) - \frac{7}{14}Entropy(S_{high}) - \frac{7}{14}Entropy(S_{normal})$$

$$= 0.9403 - \left(\frac{7}{14} \times 0.9852\right) - \left(\frac{7}{14} \times 0.5917\right) = 0.1518$$

Information gain for wind:

Day	Outlook	Temperature	Humility	Wind	Play ball
D7	Overcast	Cool	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D2	Sunny	Hot	High	Strong	No
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D1	Sunny	Hot	High	Weak	No
D8	Sunny	Mild	High	Weak	No

Step 2: Split the data set based on different attributes. Calculate entropy for each branch - wind

Wind	Play ball			
strong	Yes	3	6	14
strong	No	3		
weak	Yes	6	8	
weak	No	2		

$$Entropy(S_{strong}) = -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} = 1.0000$$

$$Entropy(S_{weak}) = -\frac{6}{8}\log_2\frac{6}{8} - \frac{2}{8}\log_2\frac{2}{8} = 0.8113$$

What is the Gain(S,Wind)?

$$Gain(S, Wind) = Entropy(S) - \frac{6}{14}Entropy(S_{strong}) - \frac{8}{14}Entropy(S_{weak})$$

$$= 0.9403 - \left(\frac{6}{14} \times 1.000\right) - \left(\frac{8}{14} \times 0.8113\right) = 0.0481$$

Working Example

Step 3: Choose attribute with the largest information gain as the decision node, divide the dataset by its branches and repeat the same process on every branch.

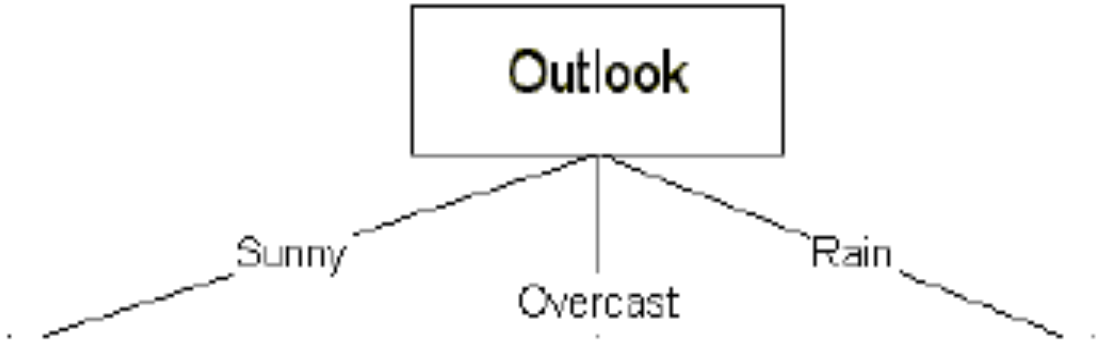
$Gain(S, Wind) = 0.0481$

$Gain(S, Humidity) = 0.1518$

$Gain(S, Temperature) = 0.0292$

→ $Gain(S, Outlook) = 0.2468$

Outlook is selected as it has the highest information gain.



Working Example

– The next question: what attribute should be tested at the "Sunny" branch node?

Day	Outlook	Temperature	Humidity	Wind	Play ball
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No

$$Entropy(S_{sunny}) = 0.9710$$

Show your works that arrive to the decision

$$Gain(S_{sunny}, Wind) = 0.0200$$

$$Gain(S_{sunny}, Humidity) = 0.9710$$

$$Gain(S_{sunny}, Temperature) = 0.5700$$

Humidity is selected as it has the highest information gain.

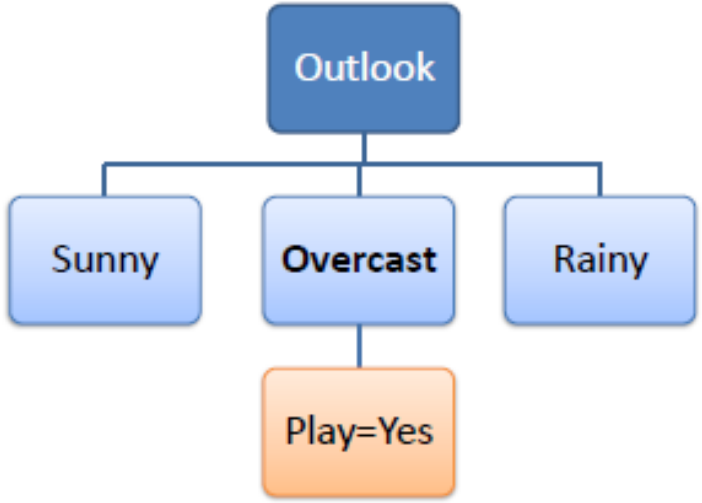
– This process goes on until all data are classified perfectly or we run out of attributes.

Working Example

Day	Outlook	Temperature	Humility	Wind	Play ball
D7	Overcast	Cool	Normal	Strong	Yes
D3	Overcast	Hot	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D12	Overcast	Mild	High	Strong	Yes
D5	Rain	Cool	Normal	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No

Step 4a: A branch with entropy of 0 is a **leaf node**.

Temp	Humidity	Windy	Play Golf
Hot	High	FALSE	Yes
Cool	Normal	TRUE	Yes
Mild	High	TRUE	Yes
Hot	Normal	FALSE	Yes

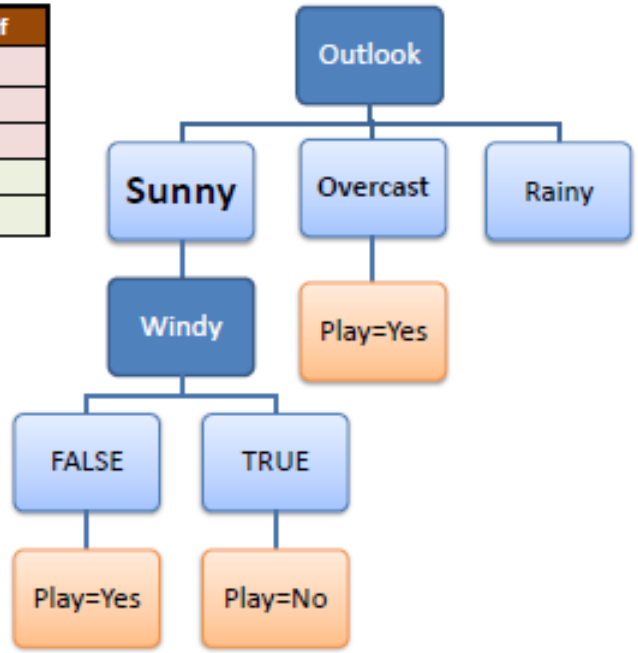


Working Example

Day	Outlook	Temperature	Humility	Wind	Play ball
D7	Overcast	Cool	Normal	Strong	Yes
D3	Overcast	Hot	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D12	Overcast	Mild	High	Strong	Yes
D5	Rain	Cool	Normal	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No

Step 4b: A branch with entropy more than 0 needs further splitting.

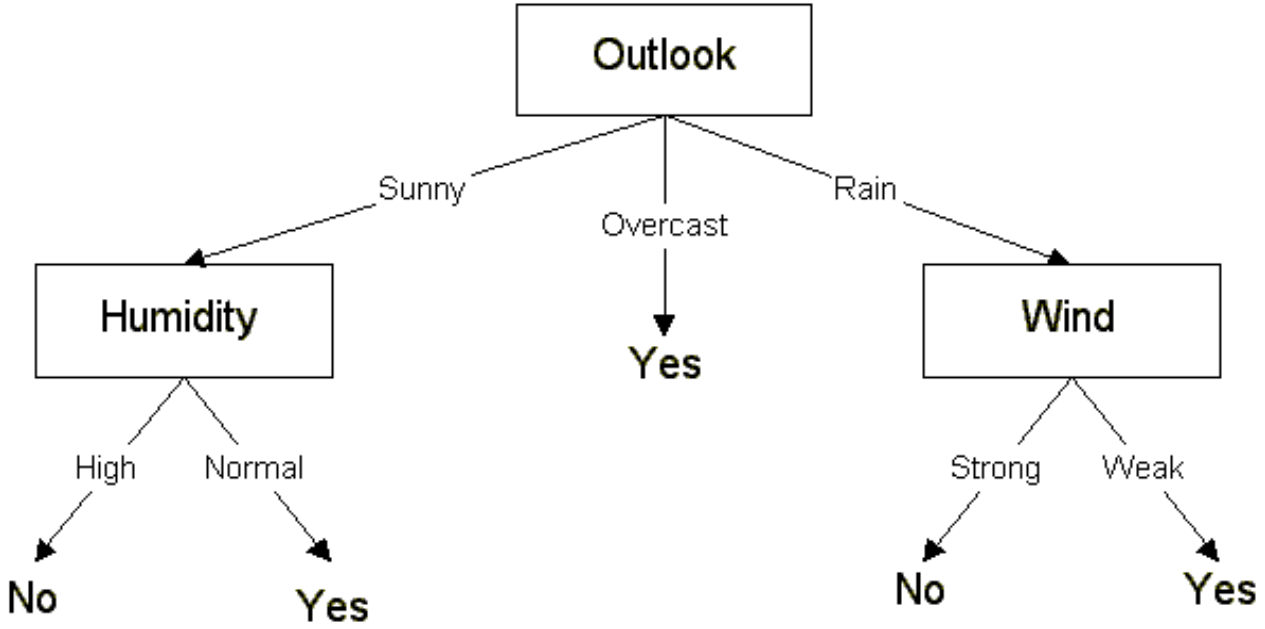
Temp.	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No

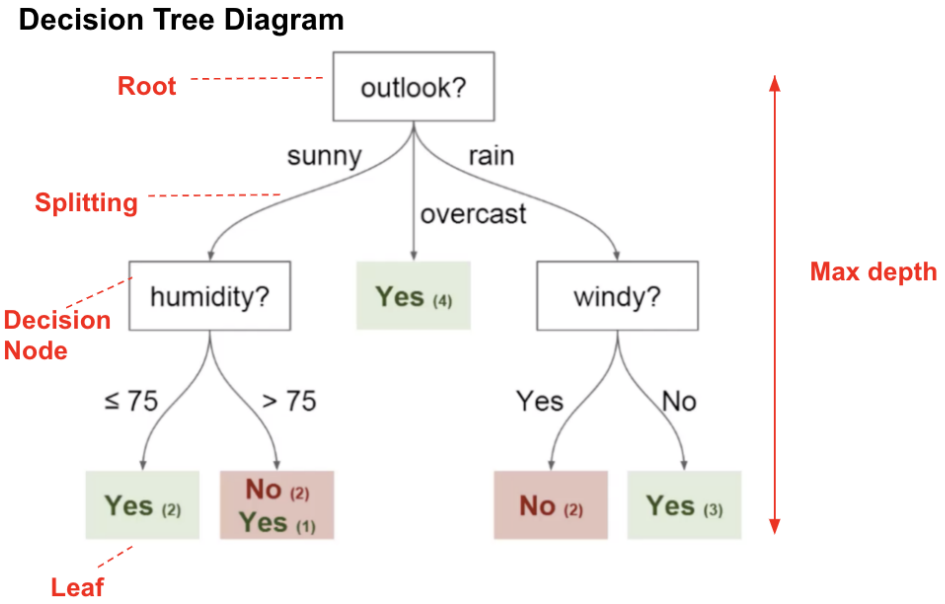


Yes = 9, No = 5, Total = 14

Working Example

– The decision tree is finally produced as follows:





Constructing Decision Tree using Gini Index

$$Gini\ Index = 1 - \sum_j p_j^2$$

Day	Outlook	Temperature	Humility	Wind	Play ball
D7	Overcast	Cool	Normal	Strong	Yes
D3	Overcast	Hot	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D12	Overcast	Mild	High	Strong	Yes
D5	Rain	Cool	Normal	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No

Outlook	Playball			
Overcast	Yes	4	4	14
Overcast	No	0		
Rain	Yes	3	5	
Rain	No	2		
Sunny	Yes	2	5	
Sunny	No	3		

Gini Index for outlook:

Step 1: Calculate probability for outlook

p(outlook=overcast)	4/14
p(outlook=rain)	5/14
p(outlook=sunny)	5/14

Step 2a: Calculate for Gini index (outlook=overcast)

if (outlook=overcast & Play ball = Yes), probability	4/4
if (outlook=overcast & Play ball = No), probability	0/4

Gini index (outlook=overcast)	1- ((4/4)^2 + (0/4)^2) = **0**

Step 2b: Calculate for Gini index (outlook=rain)

if (outlook=rain & Play ball = Yes), probability	3/5
if (outlook=rain & Play ball = No), probability	2/5

Gini index (outlook=rain)	1- ((3/5)^2 + (2/5)^2) **= 0.48**

Step 2c: Calculate for Gini index (outlook=sunny)

if (outlook=sunny & Play ball = Yes), probability	2/5
if (outlook=sunny & Play ball = No), probability	3/5
Gini index (outlook=sunny)	1- ((2/5)^2 + (3/5)^2) = **0.48**

Weighted sum of the Gini indices – Gini Index for Outlook:

= (4/14)0 + (5/14)0.48 + (5/14)0.48
= 0.34

Constructing Decision Tree using Gini Index

$$Gini\ Index = 1 - \sum_j p_j^2$$

Day	Outlook	Temperature	Humility	Wind	Play ball
D7	Overcast	Cool	Normal	Strong	Yes
D3	Overcast	Hot	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D12	Overcast	Mild	High	Strong	Yes
D5	Rain	Cool	Normal	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No

temperature	Play ball			
cool	Yes	3	4	14
cool	No	1		
hot	Yes	2	4	
hot	No	2		
mild	Yes	4	6	
mild	No	2		

Gini Index for Temperature:

Step 1: Calculate probability for temperature

p(temperature = hot)	4/14
p(temperature = mild)	6/14
p(temperature = cool)	4/14

Step 2a: Calculate for Gini index (temperature = hot)

if (temperature = hot & Play ball = Yes), probability	2/4
if (temperature = hot & Play ball = No), probability	2/4
Gini index (temperature = hot)	1- ((2/4)^2 + (2/4)^2) = 0

Step 2b: Calculate for Gini index (temperature =mild)

if (temperature = mild & Play ball = Yes), probability	4/6
if (temperature = mild & Play ball = No), probability	2/6
Gini index (temperature = mild)	1- ((4/6)^2 + (2/6)^2) = 0.44

Step 2c: Calculate for Gini index (temperature = cool)

if (temperature = cool & Play ball = Yes), probability	3/4
if (temperature = cool & Play ball = No), probability	1/4
Gini index (temperature = cool)	1- ((3/4)^2 + (1/4)^2) = 0.38

Weighted sum of the Gini indices –
Gini Index for Temperature:

= (4/14)0 + (6/14)0.44 + (4/14)0.38 = 0.77

Constructing Decision Tree using Gini Index

$$Gini\ Index = 1 - \sum_j p_j^2$$

Day	Outlook	Temperature	Humility	Wind	Play ball
D7	Overcast	Cool	Normal	Strong	Yes
D3	Overcast	Hot	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D12	Overcast	Mild	High	Strong	Yes
D5	Rain	Cool	Normal	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No

Humidity	Play ball			
high	Yes	3	7	14
high	No	4		
normal	Yes	6	7	
normal	No	1		

Gini Index for Humidity:

Step 1: Calculate probability for Humidity

p(humidity = high)	7/14
p(humidity = normal)	7/14

Step 2a: Calculate for Gini index (Humidity = high)

if (humidity = high & Play ball = Yes), probability	3/7
if (humidity =high & Play ball = No), probability	4/7

Gini index (humidity = high)	1- ((3/7)^2 + (4/7)^2) = 0.49

Step 2b: Calculate for Gini index (Humidity = normal)

if (humidity = normal & Play ball = Yes), probability	6/7
if (humidity = normal & Play ball = No), probability	1/7

Gini index (humidity = normal)	1- ((6/7)^2 + (1/7)^2) = 0.24

Weighted sum of the Gini indices –
Gini Index for Humidiy: = (7/14)0.49 + (7/14)0.24 = 0.37

Constructing Decision Tree using Gini Index

$$Gini\ Index = 1 - \sum_j p_j^2$$

Day	Outlook	Temperature	Humility	Wind	Play ball
D7	Overcast	Cool	Normal	Strong	Yes
D3	Overcast	Hot	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D12	Overcast	Mild	High	Strong	Yes
D5	Rain	Cool	Normal	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No

Gini Index for Wind:

Step 1: Calculate probability for Wind

p(wind = strong)	6/14
p(wind = weak)	8/14

Step 2a: Calculate for Gini index (Wind = strong)

if (wind = strong & Play ball = Yes), probability	3/6
if (wind = strong & Play ball = No), probability	3/6

Gini index (wind = strong)	1- ((3/6)^2 + (3/6)^2) = 0.5

Step 2b: Calculate for Gini index (Wind = Weak)

if (wind = weak & Play ball = Yes), probability	6/8
if (wind = weak & Play ball = No), probability	2/8

Gini index (wind = weak)	1- ((6/8)^2 + (2/8)^2) = 0.38

Weighted sum of the Gini indices –

Gini Index for Wind: = (6/14)0.5 + (8/14)0.38 = 0.43

Gini Index

Attributes/Features	Gini index
Outlook	0.34
Temperature	0.77
Humidity	0.37
Wind	0.43

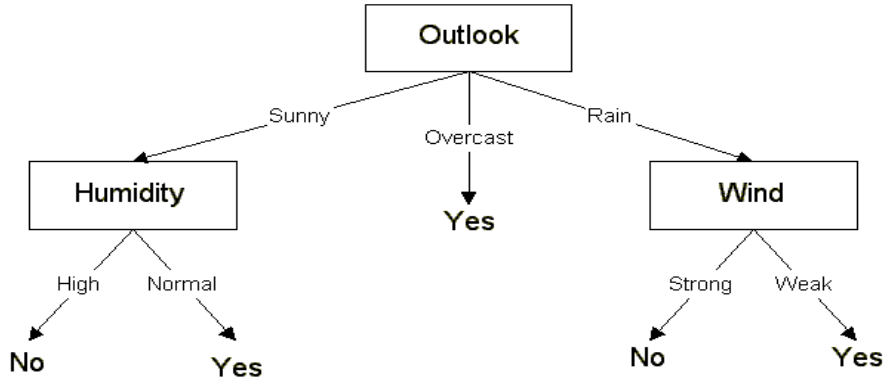
From the result, we observe that 'Outlook' has the lowest Gini Index and hence it will be chosen as the root node for how decision tree works.

Explanation on Gini Index. However, note that the last selection showed in the video is Max. However, the correct answer should be the minimum

https://www.youtube.com/watch?v=2lEcfRuHFV4

Extracting Classification Rules from Trees

- Represent the knowledge in the form of IF-THEN rules
- One rule is created for each path **from the root to a leaf**
- Each attribute-value pair along a path forms a conjunction
- The leaf node holds the class prediction
- Rules are easier for humans to understand



What are the rules?

The decision tree can also be expressed in rule format:

IF outlook = sunny AND humidity = high THEN playball = no
IF outlook = rain AND humidity = high THEN playball = no
IF outlook = rain AND wind = strong THEN playball = yes
IF outlook = overcast THEN playball = yes
IF outlook = rain AND wind = weak THEN playball = yes

Decision Tree Classification Task

Classification of new unknown Instance

- The classification of an unknown input vector is done by traversing the tree from the root node to a leaf node.
- A record enters the tree at the root node.
  - At the root, a test is applied to determine which child node the record will encounter next.
  - This process is repeated until the record arrives at a leaf node.

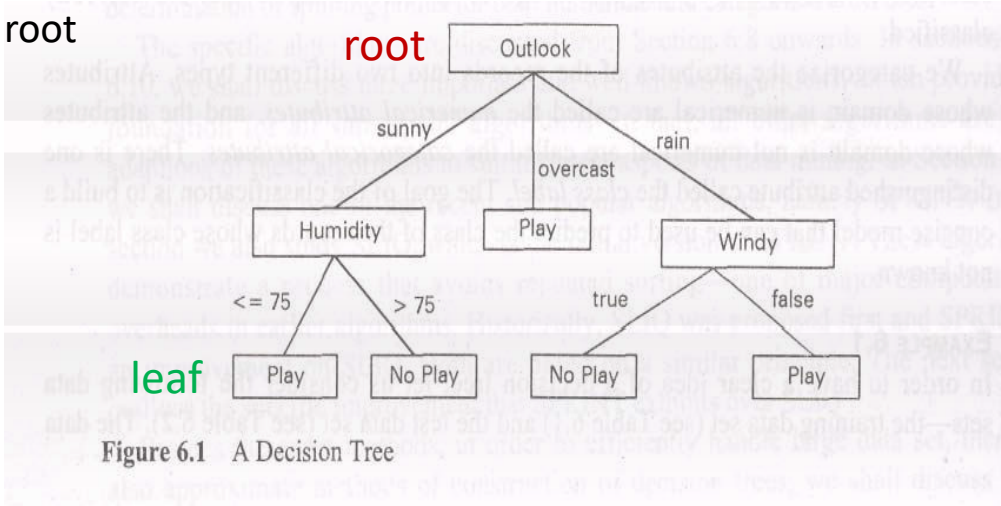


Figure 6.1 A Decision Tree

- All the records that end up at a given leaf of the tree are classified in the same way.
- There is a unique path from the root to each leaf.
- The path is a rule which is used to classify the records.

Classification of new unknown Instance

- In our tree, we can carry out the classification for an unknown record as follows.
  - Let us assume, for the record, that we know the values of the **first four attributes** (but we do not know the value of *class* attribute) as

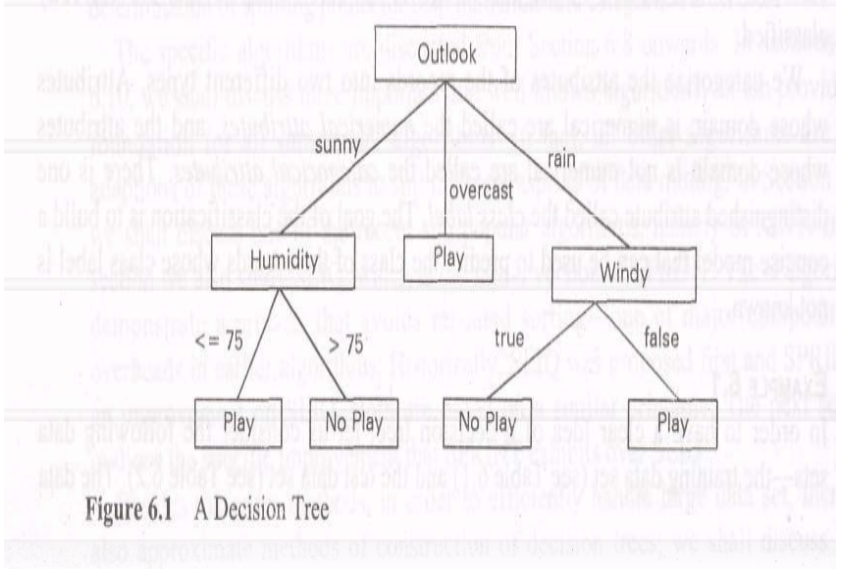


Figure 6.1 A Decision Tree

What is the decision for:
outlook= rain; temp = 70; humidity = 65; and windy= true.

Classification of new unknown Instance

- We start from the root node to check the value of the attribute associated at the root node.
- This attribute is the *splitting attribute* at this node.
- For a decision tree, at every node there is an attribute associated with the node called the *splitting attribute.*
- In our example, *outlook* is the splitting attribute at root.
  - Since for the given record, *outlook* = *rain,* we move to the right- most child node of the root.
  - At this node, the splitting attribute is *windy* and we find that for the record we want classify, *windy* = *true.*
- Hence, we move to the left child node to conclude that the class label Is "*no play*".

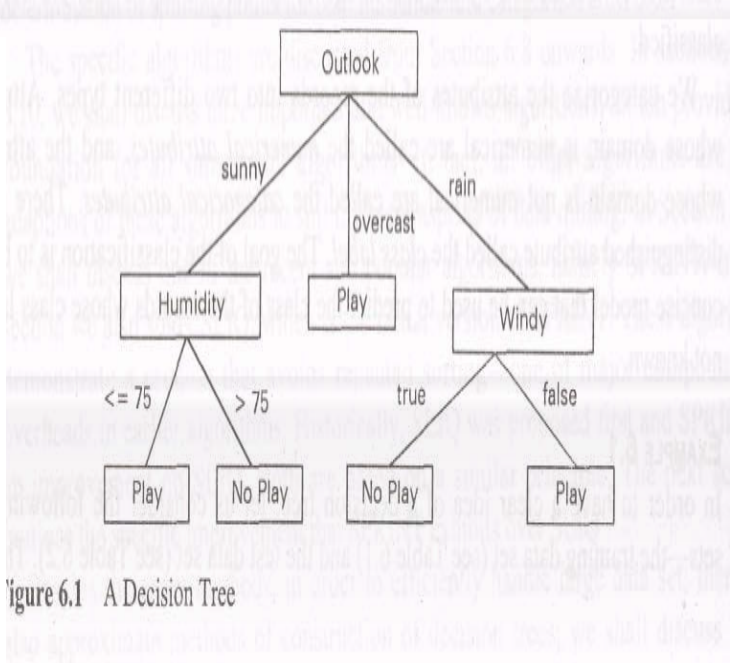


Figure 6.1 A Decision Tree

Classification of new unknown Instance

- The *accuracy of the classifier* is determined by the percentage of the test data set that is correctly classified.
- We can see that for *Rule 1* there are two records of the test data set satisfying *outlook= sunny* and *humidity < 75*, and only one of these is correctly classified *as play.*
  - Thus, the accuracy of this rule is 0.5 (or 50%).
  - Similarly, the accuracy of Rule 2 is also 0.5 (or 50%).
  - The accuracy of Rule 3 is 0.66.

TEST Data Set

OUTLOOK	TEMP(F)	HUMIDITY(%)	WINDY	CLASS
sunny	79	90	true	play
sunny	56	70	false	play
sunny	79	75	true	no play
sunny	60	90	true	no play
overcast	88	88	false	no play
overcast	63	75	true	play
overcast	88	95	false	play
rain	78	60	false	play
rain	66	70	false	no play
rain	68	60	true	play

RULE 1

If it is sunny and the humidity is not above 75%, then play.

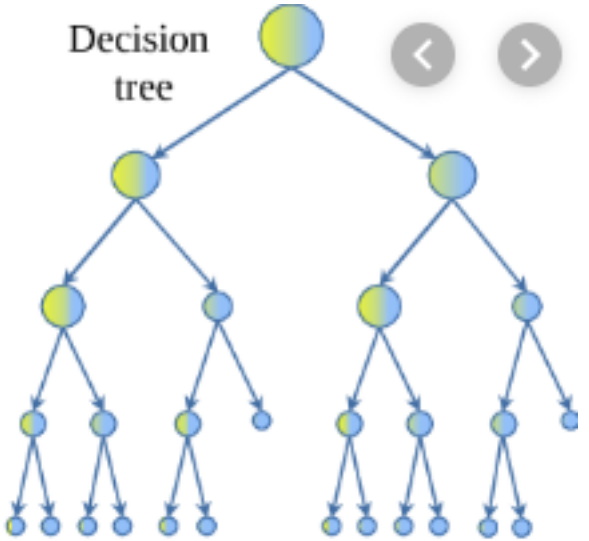
The decision tree can also be expressed in rule format:

IF outlook = sunny AND humidity = high THEN playball = no
IF outlook = rain AND humidity = high THEN playball = no
IF outlook = rain AND wind = strong THEN playball = yes
IF outlook = overcast THEN playball = yes
IF outlook = rain AND wind = weak THEN playball = yes

Pruning

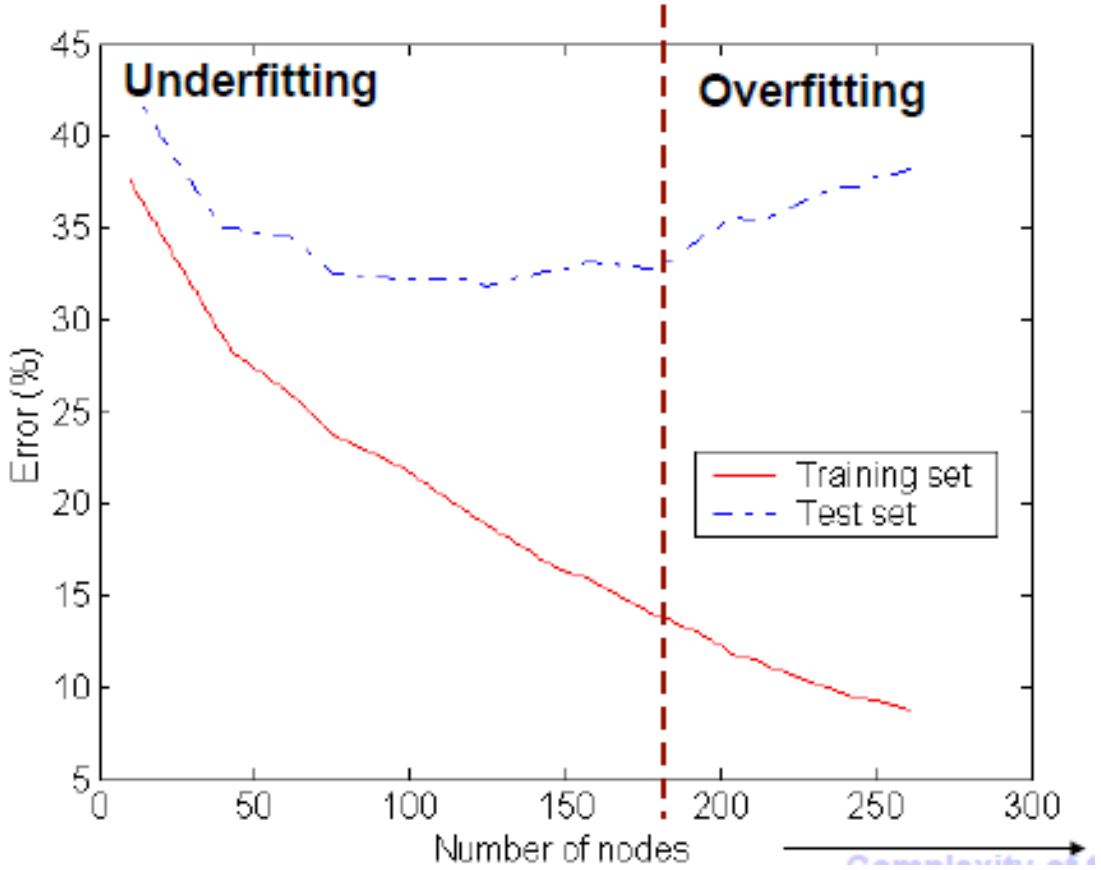
Pruning

- Decision trees are made to classify the item set.
- While classifying we meet with 2 problems
  - Underfitting
    - problem arises when both the training errors and test errors are large
    - This happens when the developed model is made very simple



  - Overfitting
    - training errors are small but test errors are large
    - 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 records.
    - Need new ways for estimating errors.

Underfitting and Overfitting



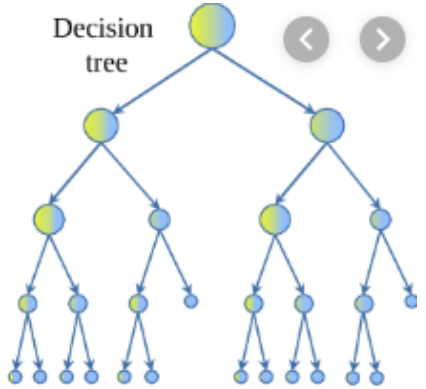
Complexity of the classification (learned) function

Complexity of a decision tree – Number of nodes it uses

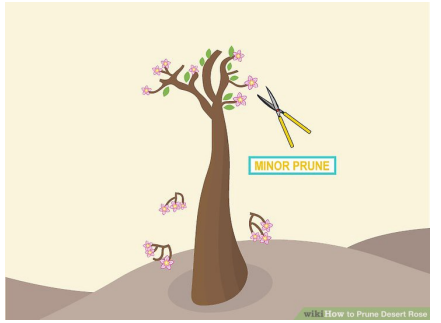
Underfitting: When model is too simple, both training and testing errors are large

Overfitting: When a model overfits, training errors are small but testing errors are large

Avoid Overfitting in Classification



- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Prepruning: Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning: Remove branches from a "fully grown" tree—get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the "best pruned tree"



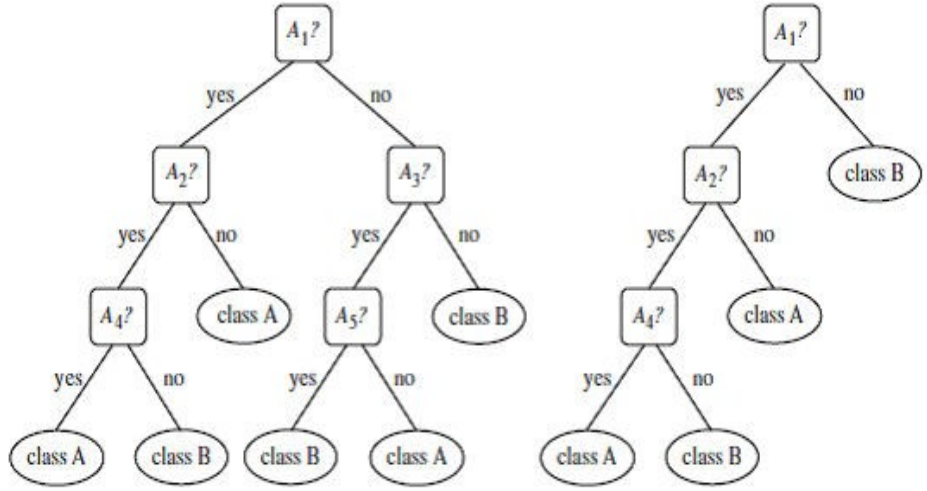
COMPARISION

- Prepruning is faster than post pruning since it don't need to wait for complete construction of decision tree.
- But still Post-pruning is preferable to pre-pruning because of "interaction effect". These are the effects which arise after interaction of several attributes.

- Prepruning suppresses growth by evaluating each attribute individually, and so might overlook effects that are due to the interaction of several attributes and stop too early.
  - Post-pruning, on the other hand, avoids this problem because interaction effects are visible in the fully grown tree.

PREPRUNING

- Prepruning is the halting of subtree construction at some node after checking some measures.
- These measures can be Information gain, Gini index,etc.
- If partitioning the tuple at a node would result in a split that falls below a prespecified threshold, then pruning is done.
- Early stopping - Pre-pruning may stop the growth process prematurely.

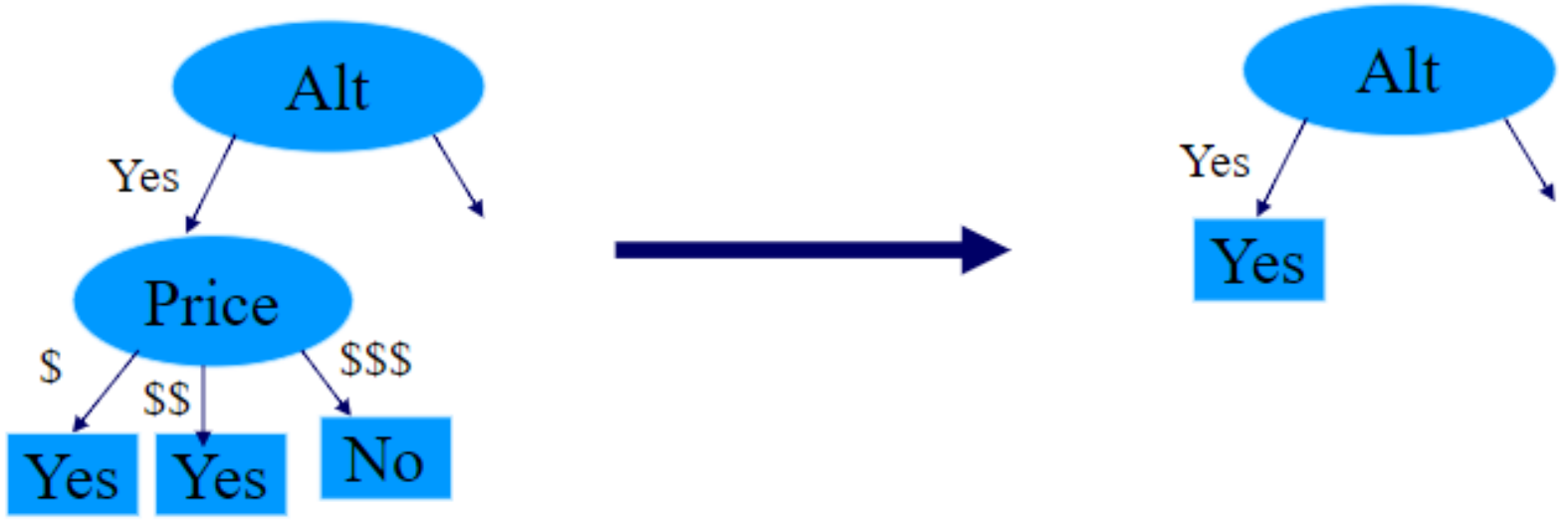


POSTPRUNING

- Grow decision tree to its entirety.
- Trim the nodes of the decision tree in a bottom-up fashion.
- Postpruning is done by replacing the node with leaf.
- If error improves after trimming, replace sub-tree by a leaf node.
- Use a set of data different from training data (validation data) to decide which is the “best- pruned tree”

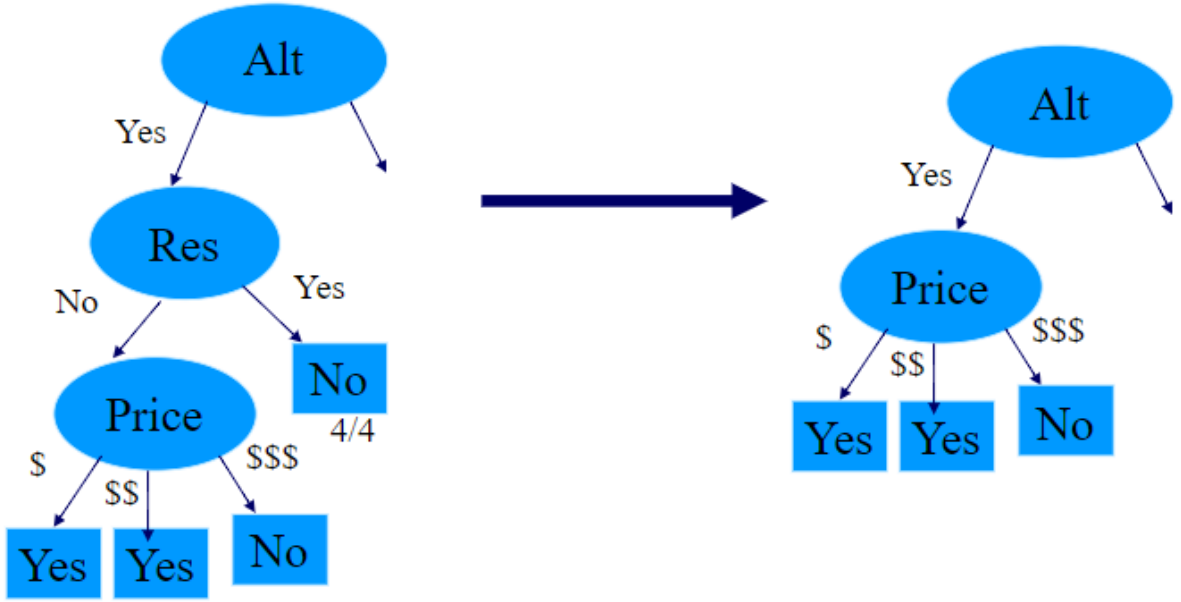
POSTPRUNING

- Subtree replacement: Replace a subtree with a single leaf node



POSTPRUNING

- Subtree raising: Move a subtree to a higher level in the decision tree, subsuming its parent



Advantages vs Disadvantages

- A decision tree construction process is concerned with identifying the splitting attributes and splitting criterion at every level of the tree.

Strengths

- Generate understandable rules.
- Able to handle both numerical and categorical attributes.
- Provide clear indication of which fields are most important for prediction or classification.
- Perform classification without requiring much computation

Weaknesses

- The process of growing a decision tree is computationally expensive. At each node, each candidate splitting field is examined before its best split can be found.
- Some decision tree can only deal with binary-valued target classes.
- Error prone with too many classes

Decision Trees, Random Forests and Boosting

Decision Trees, Random Forests and Boosting are among the top 16 data science and machine learning tools used by data scientists.

The three methods are similar, with a significant amount of overlap.

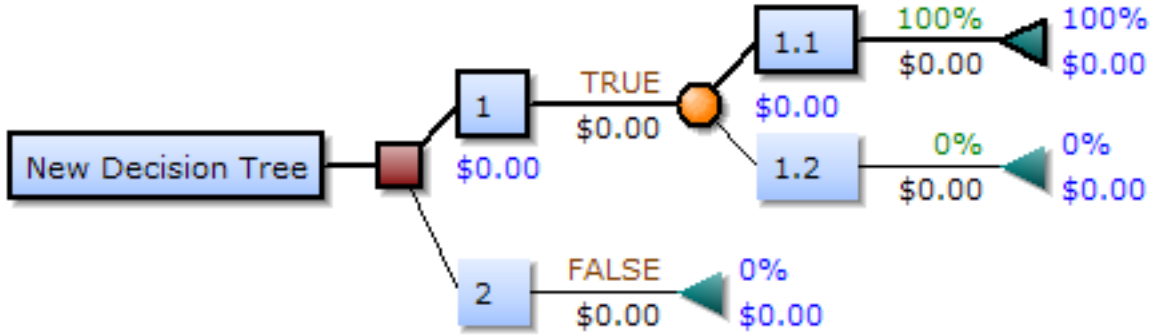
In a nutshell:

- A **decision tree** is a simple, decision making-diagram.
- **Random forests** are a large number of trees, combined (using averages or "majority rules") at the end of the process.
- **Gradient boosting** also combine decision trees, but start the combining process at the beginning, instead of at the end.

Decision Trees

Decision trees are a series of sequential steps designed to answer a question and provide probabilities, costs, or other consequence of making a particular decision.

They are simple to understand, providing a clear visual to guide the decision-making progress. However, it comes with a few *serious disadvantages*, including overfitting, error due to bias and error due to variance.



- **Overfitting** happens for many reasons, including presence of noise and lack of representative instances. It's possible for overfitting with one large (deep) tree.
- **Bias error** happens when you place too many restrictions on target functions. For example, restricting your result with a restricting function (e.g. a linear equation) or by a simple binary algorithm (like the true/false choices in the above tree) will often result in bias.
- **Variance error** refers to how much a result will change based on changes to the training set. Decision trees have high variance, which means that tiny changes in the training data have the potential to cause large changes in the final result.

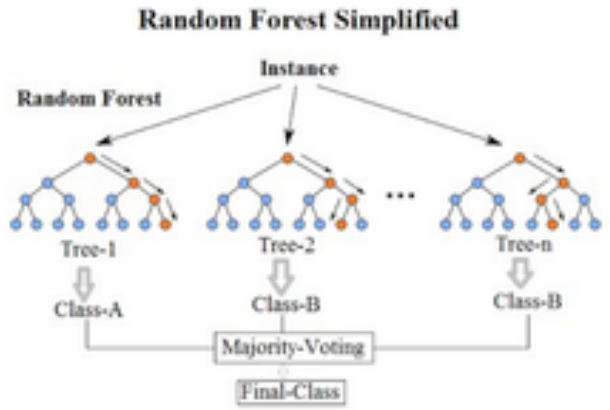
Random Forests

Random forest—a collection of decision trees with a single, aggregated

Random forests are commonly reported as the most accurate learning al

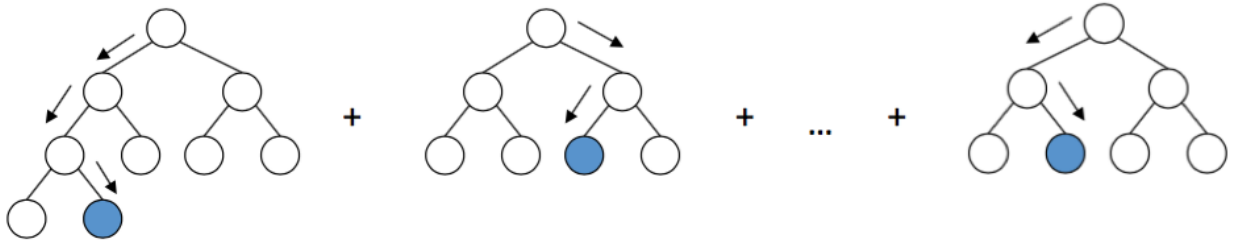
Random forests reduce the variance seen in decision trees by:

1. Using different samples for training,
2. Specifying random feature subsets,
3. Building and combining small (shallow) trees.



- A single decision tree is a weak predictor, but is relatively fast to build. More trees give you a more robust model and prevent overfitting. However, **the more trees you have, the slower the process.**
- The more features you have, the slower the process (which can sometimes take hours or even days); Reducing the set of features can dramatically speed up the process.
- Another distinct difference between a decision tree and random forest is that while a decision tree is easy to read—you just follow the path and find a result—a random forest is a tad more **complicated to interpret.**

Gradient Boosting



Like random forests, gradient boosting is a set of decision trees.

The two main differences are:

- **How trees are built:** random forests builds each tree independently while gradient boosting builds one tree at a time. This additive model (ensemble) works in a forward stage-wise manner, introducing a weak learner to improve the shortcomings of existing weak learners.
- **Combining results**: random forests combine results at the end of the process (by averaging or "majority rules") while gradient boosting combines results along the way.

- If you carefully tune parameters, gradient boosting can result in **better performance** than random forests. However, **gradient boosting may not be a good choice if you have a lot of noise**, as it can result in overfitting.
- Random forests and gradient boosting each excel in different areas.
  - Random forests perform well for multi-class object detection and bioinformatics, which tends to have a lot of statistical noise.
  - Gradient Boosting performs well when you have unbalanced data such as in real time risk assessment.

Thank you