BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Applied Machine Learning

SEZG568/SSZG568

Dr Y V K RAVI KUMAR
yvk.ravikumar@pilani.bits-pilani.ac.in

innovate achieve lead

BITS Pilani

Pilani Campus

Session 9(30th September,2023)

'Decision Tree'

Session 9 – 30th September,2023

9	Classification Models II: Decision Tree. Entropy and information gain. Construction algorithm. Challenges with Decision Tree	T1: Chapter 6 T2: Chapter 4

Session Content

Module-6

Classification Models II

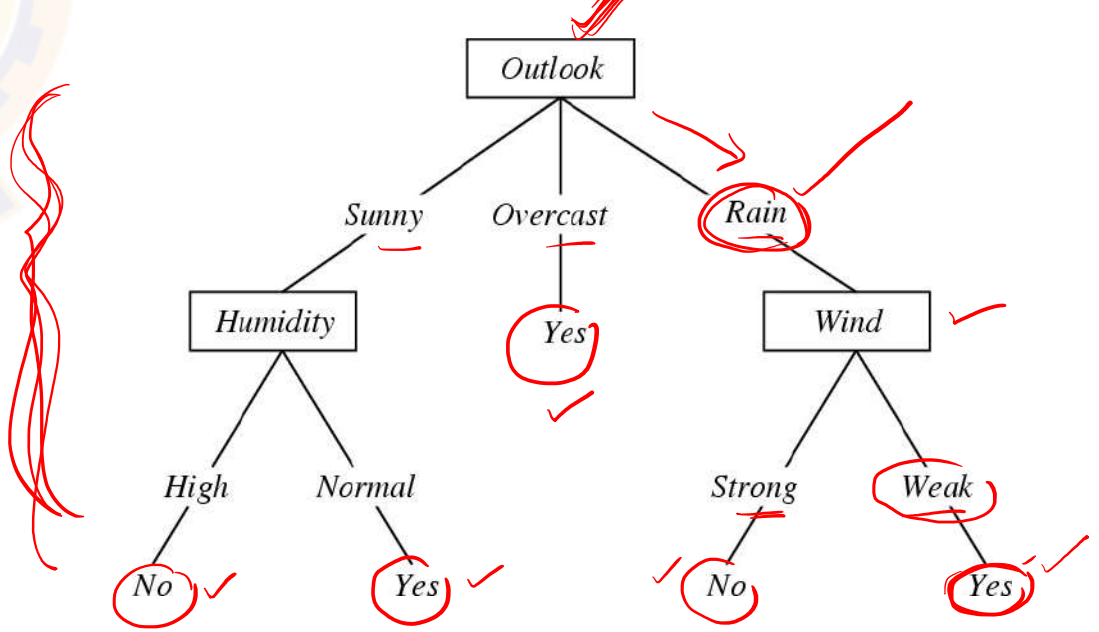
1. Decision Tree
2. Entropy
3. Information gain
4. Challenges with Decision tree

SVM - Kernel

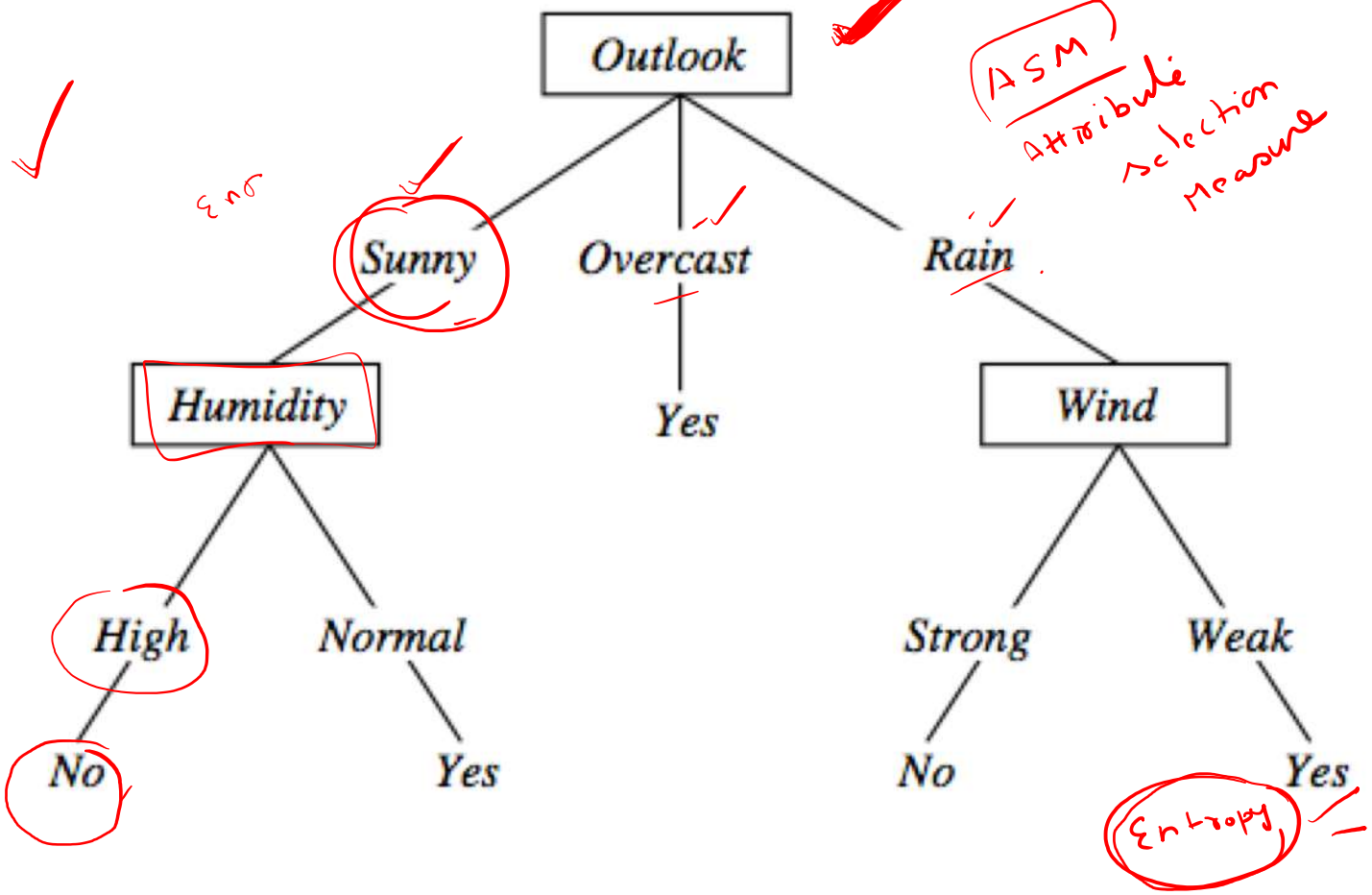
- **Polynomial kernel** – it is mostly used in image processing.
- **Linear Splines kernel in one-dimension –** it is used in text categorization and is helpful in dealing with large spare data vectors.
- **Gaussian Kernel** – it is used when there is no preceding information about the data.
- **Gaussian Radial Basis Function (RBF) –** It is commonly used where there is no previous knowledge about the data.
- **Hyperbolic Tangent Kernel –** it is used in neural networks.
- **Bessel Function of the First kind Kernel –** it is used to eliminate the cross term in mathematical functions.
- **Sigmoid Kernel –** it can be utilized as the alternative for neural networks.
- **ANOVA Radial Basis Kernel –** it is mostly used in regression problems.

Decision Tree Learning

- **Problem Setting**:
  - Set of possible instances *X*
    - each instance *x* in *X* is a feature vector
    - e.g., <*Humidity=low, Wind=weak, Outlook=rain, Temp=hot*>
  - Unknown target function $f : X \rightarrow Y$
    - *Y* is discrete valued
  - Set of function hypotheses $H=\{ h \mid h : X \rightarrow Y \}$
    - each hypothesis *h* is a decision tree
    - trees sorts *x* to leaf, which assigns *y*



Decision tree representation (PlayTennis)



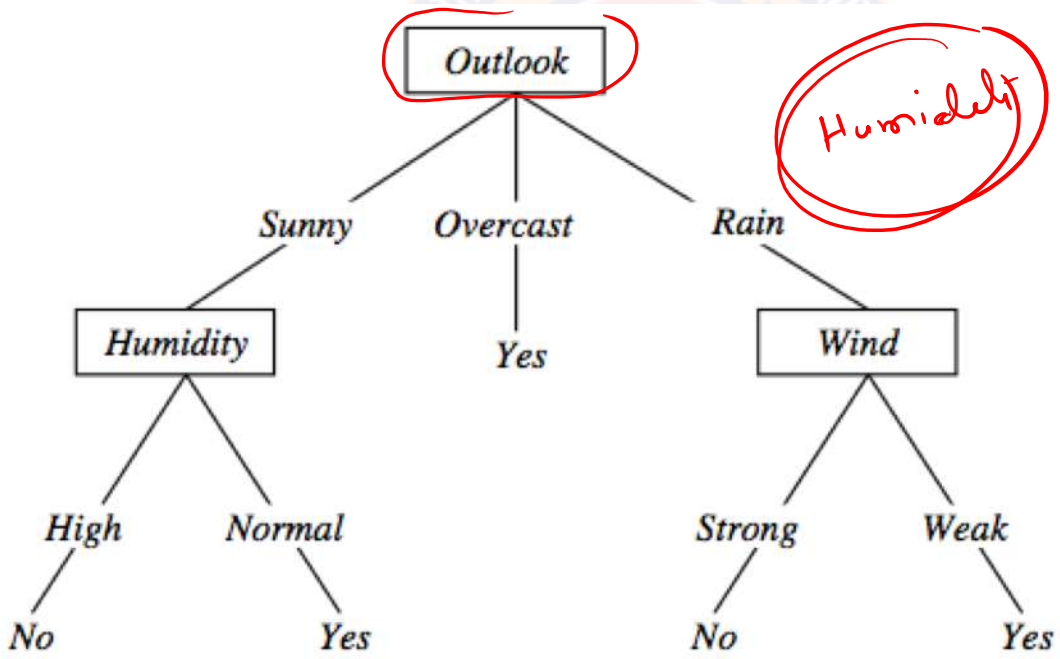
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal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

⟨*Outlook=Sunny, Temp=Hot, Humidity=High, Wind=Strong*⟩ ? No

Decision trees expressivity

Decision trees represent a disjunction of conjunctions on constraints on the value of attributes

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



How to choose the best attribute?

Choosing the Best Attribute

Key problem: choosing which attribute to split a given set of examples

- Some possibilities are:
  - **Random:** Select any attribute at random
  - **Least-Values:** Choose the attribute with the smallest number of possible values
  - **Most-Values:** Choose the attribute with the largest number of possible values
  - **Max-Gain:** Choose the attribute that has the largest expected *information gain*
    - i.e., attribute that results in smallest expected size of subtrees rooted at its children
- The ID3 algorithm uses the Max-Gain method of selecting the best attribute

Measure of Information and Entropy

- The amount of information (surprise element) conveyed by a message is inversely proportional to its probability of occurrence. That is

$$I_k \alpha \frac{1}{p_k}$$

- The mathematical operator satisfies above properties is the logarithmic operator.

$$I_k = \log_r \frac{1}{p_k} \; units$$

- Entropy of discrete random variable X={x_1, x_2…x_n}

$$\mathrm{H}(X) = \mathrm{E}[\mathrm{I}(X)] = \mathrm{E}[-\log(\mathrm{P}(X))].$$

since: $\log_2(1/P(event)) = -\log_2 P(event)$

- As uncertainty increases, entropy increases
- Entropy across all values

$$\mathrm{H}(X) = -\sum_{i=1}^{n} \mathrm{P}(x_i) \log_b \mathrm{P}(x_i)$$

Entropy in general

- Entropy measures the amount of information in a random variable
- For binary classification [two-valued random variable]

$$H(X) = -p_+ \log_2 p_+ - p_- \log_2 p_- \qquad X = \{+, -\}$$

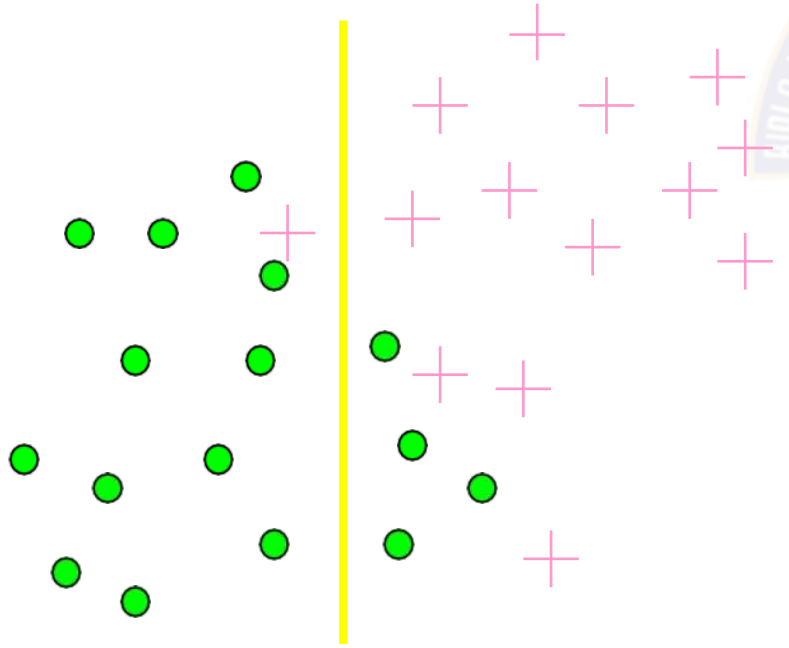
- For classification in c classes

$$H(X) = -\sum_{i=1}^{c} p_i \log_2 p_i = \sum_{i=1}^{c} p_i \log_2 1/p_i \qquad X = \{i, \ldots, c\}$$

Information gain

Which test is more informative?

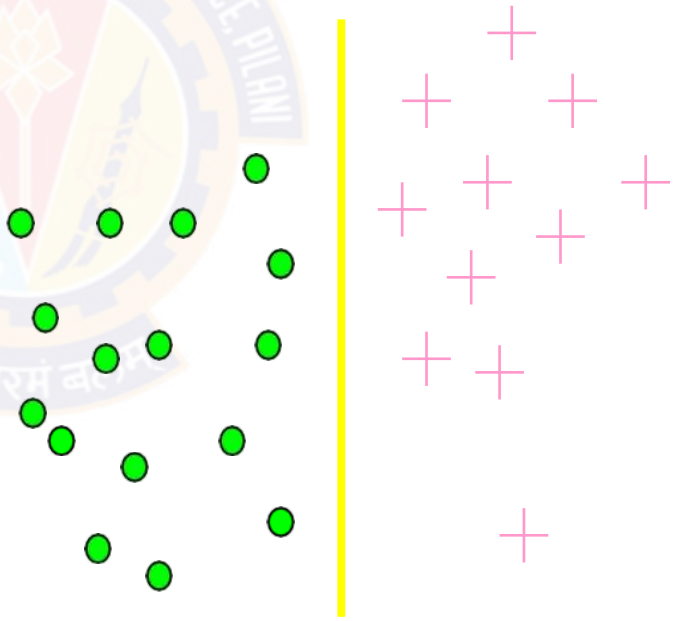
Split over whether Balance exceeds 50K



Less or equal 50K

Over 50K

Split over whether applicant is employed



Unemployed

Employed

Information gain

Impurity/Entropy (informal)

- Measures the level of **impurity** in a group of examples

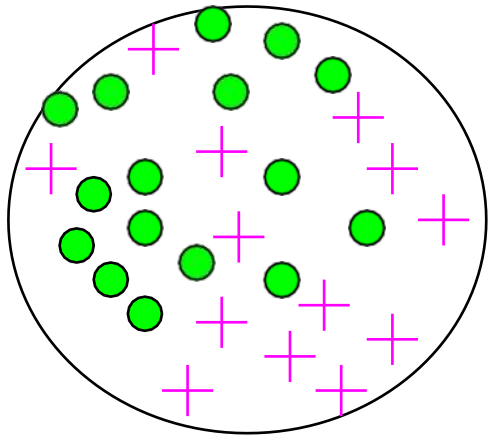
Information gain

Impurity

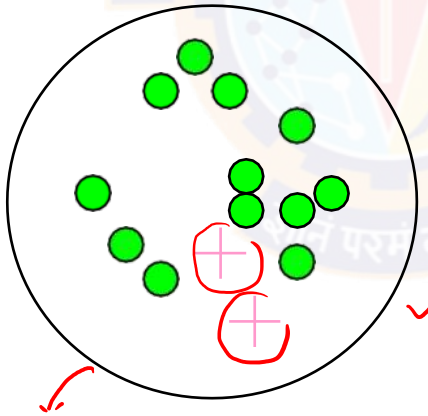
Entropy $= -p_1 \log(p_1) - p_2 \log(p_2)$

$= -\frac{12}{12}\log\left(\frac{12}{12}\right) - 0 = 0$

Very impure group

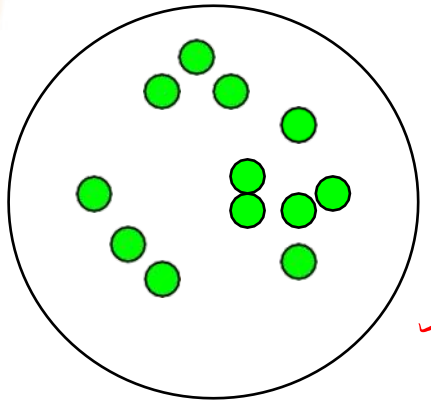


Less impure



$= -\frac{2}{14}\log\left(\frac{2}{14}\right) - \frac{12}{14}\log\left(\frac{12}{14}\right)$

Minimum impurity



12 → 6 - class-1, 6 class-2

Entropy $= -\frac{6}{12}\log\left(\frac{6}{12}\right) - \frac{6}{12}\log\left(\frac{6}{12}\right)$

$= -\frac{1}{2}\log\left(\frac{1}{2}\right) - \frac{1}{2}\log\left(\frac{1}{2}\right)$

$= 1$

Information gain

Entropy: a common way to measure impurity

- Entropy *H(X)* of a random variable X

# of possible values for X

$$H(X) = -\sum_{i=1}^{n} P(X = i) \log_2 P(X = i)$$

- *H(X)* is the expected number of bits needed to encode a randomly drawn value of *X*(under most efficient code)

Information gain

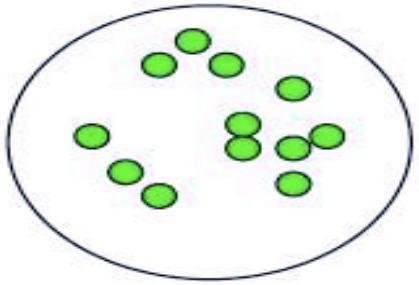
2-Class Cases:

Entropy $$H(x) = -\sum_{i=1}^{n} P(x = i) \log_2 P(x = i)$$

- What is the entropy of a group in which all examples belong to the same class?
  - entropy = - 1 $\log_2 1 = 0$

not a good training set for learning

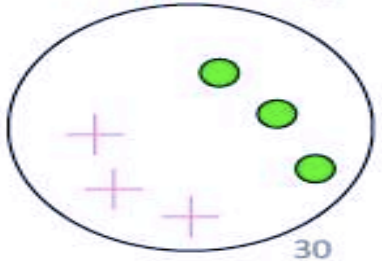
Minimum impurity



- What is the entropy of a group with 50% in either class?
  - entropy = -0.5 $\log_2 0.5 - 0.5 \log_2 0.5 = 1$

good training set for learning

Maximum impurity



Based on slide by Pedro Domingos

Information gain

- We want to determine which attribute in a given set of training feature vectors is most useful for discriminating between the classes to be learned.
- Information gain tells us how important a given attribute of the feature vectors is.
- We will use it to decide the ordering of attributes in the nodes of a decision tree.

Entropy in binary classification

- Entropy measures the *impurity* of a collection of examples. It depends from the distribution of the random variable *p.*
  - *S* is a collection of training examples
  - p_+ the proportion of positive examples in *S*
  - p_- the proportion of negative examples in *S*

$$Entropy\,(S) \equiv -p_+ \log_2 p_+ - p_- \log_2 p_- \quad [0 \log_2 0 = 0]$$

$$Entropy\,([14+, 0-]) = -14/14 \log_2 (14/14) - 0 \log_2 (0) = 0$$

$$Entropy\,([9+, 5-]) = -9/14 \log_2 (9/14) - 5/14 \log_2 (5/14) = 0.94$$

$$Entropy\,([7+, 7-]) = -7/14 \log_2 (7/14) - 7/14 \log_2 (7/14) = 1/2 + 1/2 = 1$$

- Note: the log of a number < 1 is negative, $0 \leq p \leq 1$, $0 \leq entropy \leq 1$

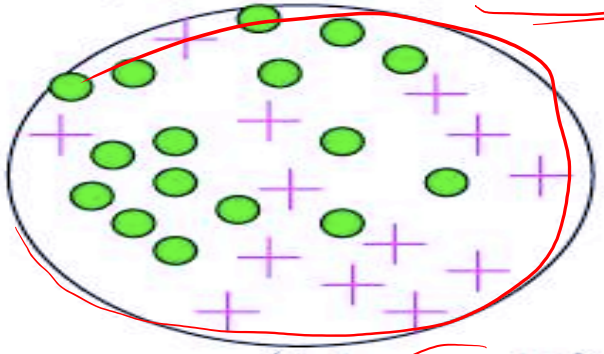
Information gain

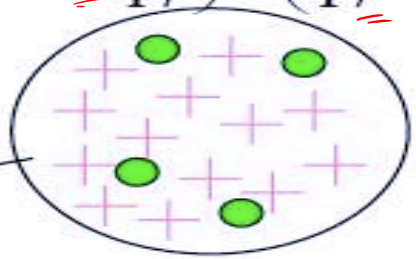
Calculating Information Gain

Information Gain = entropy(parent) – [average entropy(children)]

child entropy $-\left(\frac{13}{17}\cdot\log_2\frac{13}{17}\right)-\left(\frac{4}{17}\cdot\log_2\frac{4}{17}\right)=0.787$

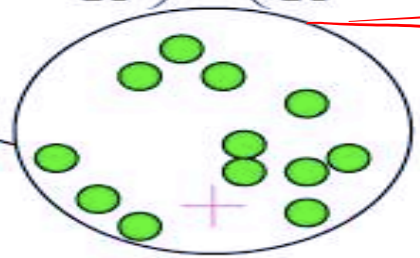
Entire population (30 instances)





17 instances

child entropy $-\left(\frac{1}{13}\cdot\log_2\frac{1}{13}\right)-\left(\frac{12}{13}\cdot\log_2\frac{12}{13}\right)=0.391$



13 instances

parent entropy $-\left(\frac{14}{30}\cdot\log_2\frac{14}{30}\right)-\left(\frac{16}{30}\cdot\log_2\frac{16}{30}\right)=0.996$

(Weighted) Average Entropy of Children $=\left(\frac{17}{30}\cdot 0.787\right)+\left(\frac{13}{30}\cdot 0.391\right)=0.615$

Information Gain= 0.996 - 0.615 = 0.38

Based on slide by Pedro Domingos

Information gain as entropy reduction

- *Information gain* is the *expected* reduction in entropy caused by partitioning the examples on an attribute.
- The higher the information gain the more effective the attribute in classifying training data.

- Expected reduction in entropy knowing A

$$Gain(S, A) = Entropy(S) - \sum_{v \in values(A)} \frac{|Sv|}{|S|} Entropy(S_v)$$

- $values(A)$ possible values for A
- S_v subset of S for which A has value v

Gini

salary	Age	Purchase
20000	21	Yes
10000	45	No
60000	27	Yes
15000	31	No
12000	18	No

Gini : $1 - (P_{Yes}^2 + P_{No}^2)$

$= 1 - \left(\left(\frac{2}{5}\right)^2 + \left(\frac{3}{5}\right)^2\right)$

$= 1 - \left(\frac{4}{25} + \frac{9}{25}\right)$

$= 0.48$ — More randomness

salary	Age	Purchase
34000	31	No
15000	25	No
69000	57	Yes
25000	21	No
32000	28	No

Gini : $1 - \left(\frac{1}{25} + \frac{16}{25}\right)$

$= 0.32$ — Less randomness

comment / observations

IG

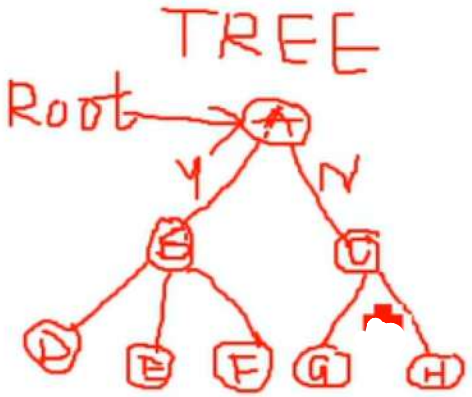
$= (\quad) - (\quad)$

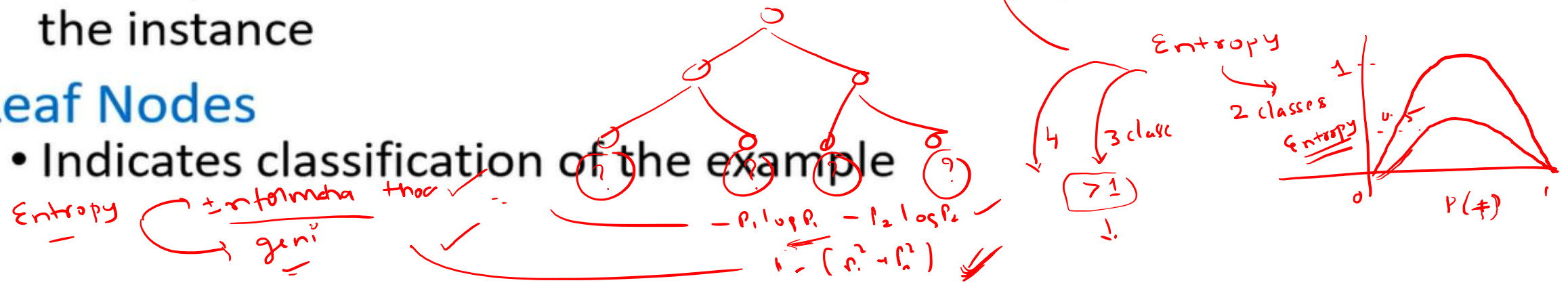
Gini

$= 1 - (P_{Yes}^2 + P_{No}^2)$

Decision Tree

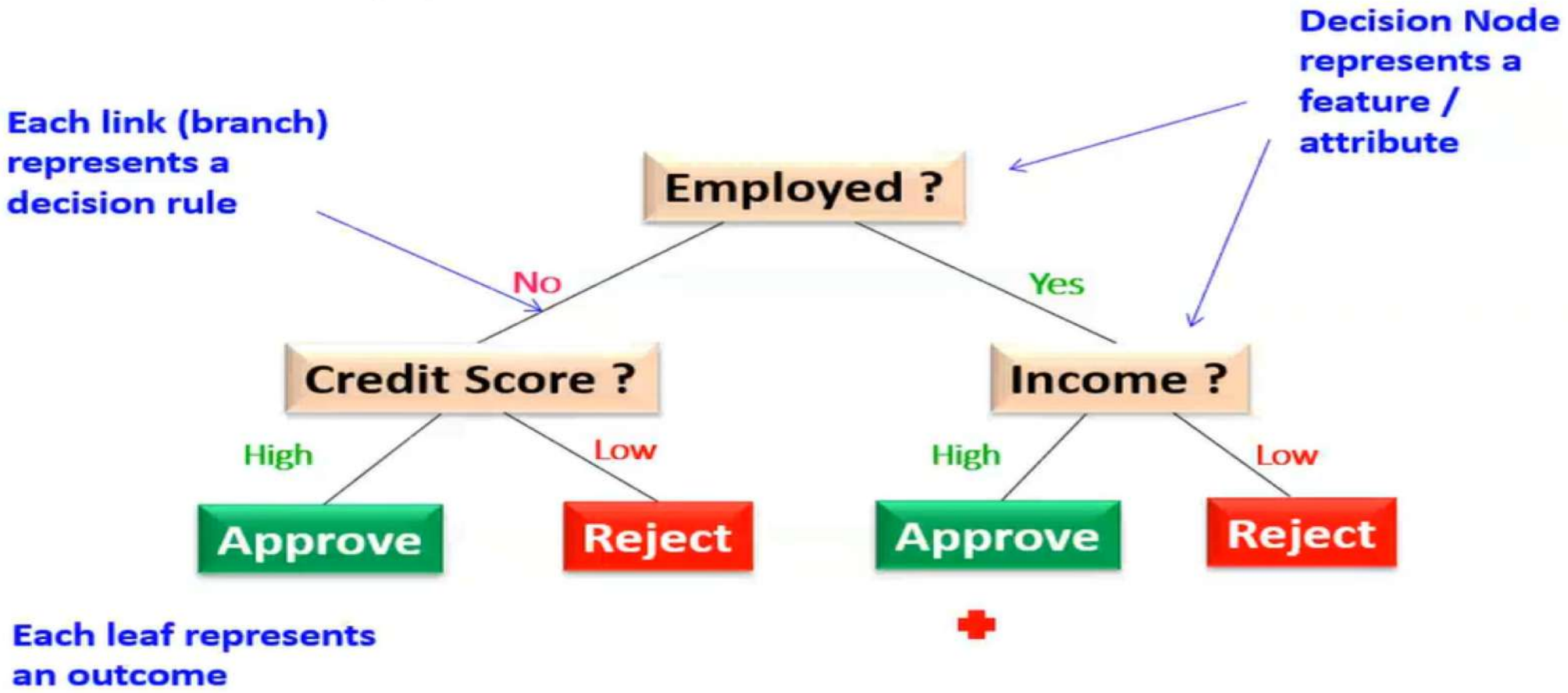
- Can be used for regression or classification
- Popular usage is as classifier
- Decision Nodes
  - Specify a choice / test
  - Have more than one results
  - Test is performed on the value of the feature / attribute of the instance
- Leaf Nodes
  - Indicates classification of the example





Decision Tree - Example

- Whether to approve the loan ?



Example

- Create a decision tree to predict whether tennis will be played on the day ?

S. No.	Outlook	Temperature	Humidity	Windy	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rainy	Mild	High	Weak	Yes
5	Rainy	Cool	Normal	Weak	Yes
6	Rainy	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rainy	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rainy	Mild	High	Strong	No

How to Proceed?

Create a root node

- How to choose a root node ?
  - **The attribute that best classifies the training data, use this attribute at the root of the tree.**
- How to choose the best attribute ?
  - **ID3 algorithm**

It is a greedy algorithm that grows the tree top-down, at each node selecting the attribute (using statistical information) that best classifies the local training examples. This process continues until the tree perfectly classifies the training examples, or until all attributes have been used.

ID3 Algorithm

- Repeat steps 1 to 3 until desired tree is created
  1. compute the **entropy** for data-set **Entropy(*S*)**
  2. for every attribute/feature:
     - 2.1 calculate entropy for all attribute values **Entropy(*A*=*v*)**
     - 2.2 take **average information entropy** for the current attribute (considering all attribute values)
     - 2.3 calculate **gain** for the current attribute
  3. pick the **highest gain attribute**.

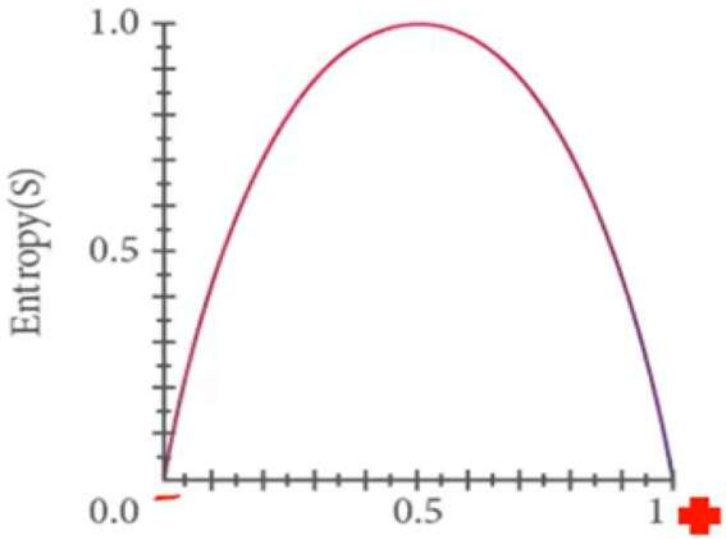
Stopping Criteria

- If all the examples belong to same class
- If there is no more attribute to test

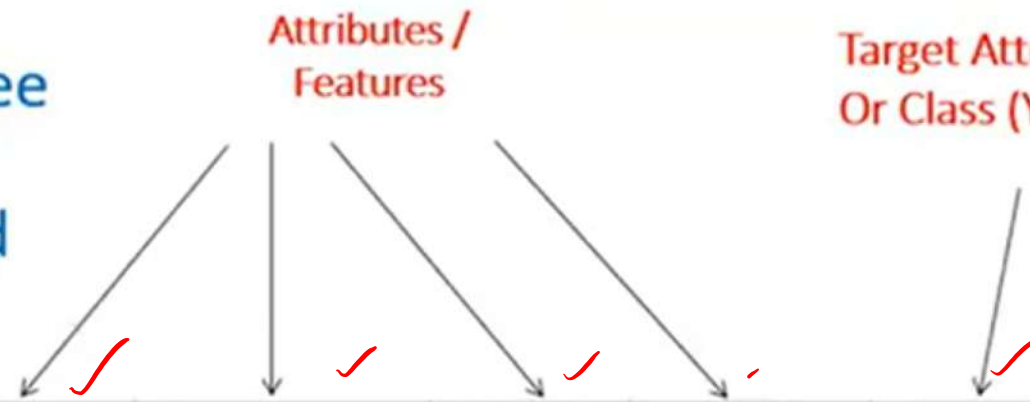
Entropy

- Represents the (im)purity of an arbitrary collection of examples.
- Entropy is **0** if all members of *S* belong to the same class.
- 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.

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



- Create a decision tree to predict whether tennis will be played on the day ?



S. No.	Outlook	Temperature	Humidity	Windy	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rainy	Mild	High	Weak	Yes
5	Rainy	Cool	Normal	Weak	Yes
6	Rainy	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rainy	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rainy	Mild	High	Strong	No

Yes = 9

No = 5

Step 1

- **Calculate Entropy (S)**

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

- where p_i is the proportion of S belonging to class i
- c – number of different values for target attribute (number of classes)

Step 2

- **Calculate Average Information**

$$Information(A) = \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

- where *Values* (*A*) is the set of all possible values for attribute *A*, S_v is the subset of *S* for which attribute *A* has value *v* (i.e. $S_v = \{s \in S \mid A(s) = v\}$)

- **Calculate Information Gain**

$$Gain(S, A) = Entropy(S) - Information(A)$$

Expected reduction in entropy caused by knowing the value of attribute *A*

Entropy of the original collection *S*

Expected value of the entropy after *S* is partitioned using attribute *A*

SOLUTION

Step 1

Calculate Entropy (S)

In the given set of data S,

Yes = 9 **No = 5**

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

S. No.	Outlook	Temperature	Humidity	Windy	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rainy	Mild	High	Weak	Yes
5	Rainy	Cool	Normal	Weak	Yes
6	Rainy	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rainy	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rainy	Mild	High	Strong	No

$$\begin{aligned} Entropy(S) &= -\frac{9}{9+5}\log_2\left(\frac{9}{9+5}\right) - \frac{5}{9+5}\log_2\left(\frac{5}{9+5}\right) \\ &= -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) \\ &= -(-0.64286) * (-0.63743) - (-0.35714) * (-1.48543) \\ &= 0.940286 \end{aligned}$$

Step 2

Compute Information (*A*)

- E.g. Attribute '**outlook**' has three values

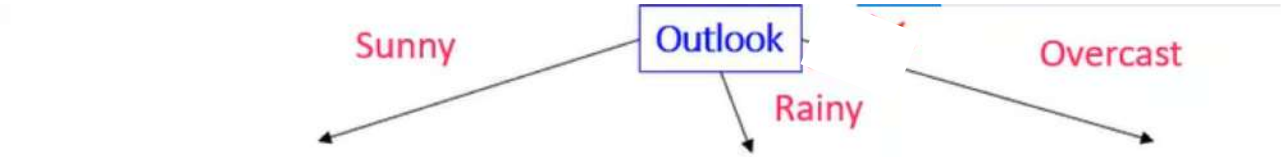
  { 'Sunny', 'Rainy', 'Overcast' }

$$Information(A) = \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

S. No.	Outlook	Temperature	Humidity	Windy	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rainy	Mild	High	Weak	Yes
5	Rainy	Cool	Normal	Weak	Yes
6	Rainy	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rainy	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rainy	Mild	High	Strong	No

- First compute entropy for each value of attribute 'outlook'
  - Entropy(outlook='Sunny')
  - Entropy(outlook='Rainy')
  - Entropy(outlook='Overcast')

Step 2



Outlook	PlayTennis
Sunny	No
Sunny	No
Sunny	No
Sunny	Yes
Sunny	Yes

Outlook	PlayTennis
Rainy	Yes
Rainy	Yes
Rainy	No
Rainy	Yes
Rainy	No

Outlook	PlayTennis
Overcast	Yes
Overcast	Yes
Overcast	Yes
Overcast	Yes

Outlook	Yes	No	Entropy
Sunny	2	3	0.971
Rainy	3	2	0.971
Overcast	4	0	0

S. No.	Outlook	Temperature	Humidity	Windy	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rainy	Mild	High	Weak	Yes
5	Rainy	Cool	Normal	Weak	Yes
6	Rainy	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rainy	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rainy	Mild	High	Strong	No

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

$$E(\text{Outlook=overcast}) = -1\log(1) - 0\log(0) = 0$$

$$E(\text{Outlook=rainy}) = -\frac{3}{5}\log\left(\frac{3}{5}\right) - \frac{2}{5}\log\left(\frac{2}{5}\right) = 0.971$$

Step 2

Compute Information (A)

$$Information('outlook') = \sum_{v \in \{'Sunny','Rainy','Overcast'\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Information(A) = \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$S_{'Sunny'} = [2(Yes) + 3(No)] = 5$$

$$S_{'Rainy'} = [3(Yes) + 2(No)] = 5$$

$$S_{'Overcast'} = [4(Yes) + 0(No)] = 4$$

$$S = [9(Yes) + 5(No)] = 14$$

Entropy(outlook='Sunny') = 0.971
Entropy(outlook='Rainy') = 0.971
Entropy(outlook='Overcast') = 0

$$Information('outlook') = \frac{5}{14} * 0.971 + \frac{5}{14} * 0.971 + \frac{4}{14} * 0$$

$$= 0.693$$

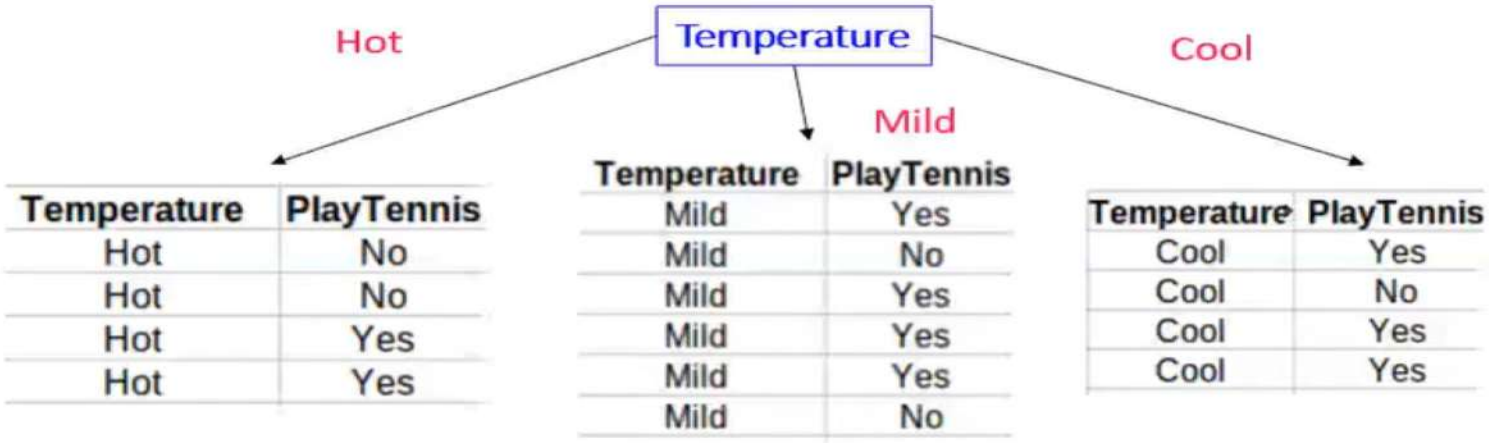
Compute Gain (S, A)

$$Gain(S,'outlook') = Entropy(S) - Information('outlook')$$

$$= 0.940 - 0.693$$

$$= 0.247$$

Step 2



Temperature	PlayTennis
Hot	No
Hot	No
Hot	Yes
Hot	Yes

Temperature	PlayTennis
Mild	Yes
Mild	No
Mild	Yes
Mild	Yes
Mild	Yes
Mild	No

Temperature	PlayTennis
Cool	Yes
Cool	No
Cool	Yes
Cool	Yes

Temperature	Yes	No	Entropy
Hot	2	2	
Mild	4	2	
Cool	3	1	

S. No.	Outlook	Temperature	Humidity	Windy	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rainy	Mild	High	Weak	Yes
5	Rainy	Cool	Normal	Weak	Yes
6	Rainy	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rainy	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rainy	Mild	High	Strong	No

Entropy(temperature='Hot') = $-(2/4)*\log_2(2/4) - (2/4)*\log_2(2/4) = 1.0$

Entropy(temperature='Mild') = $-(4/6)*\log_2(4/6) - (2/6)*\log_2(2/6) = 0.918$

Entropy(temperature='Cool') = $-(3/4)*\log_2(3/4) - (1/4)*\log_2(1/4) = 0.811$

Step 2

Compute Information (*A*)

$$Information('temperature') = \sum_{v \in \{'Hot','Mild','Cool'\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$S_{'Hot'} = [2(Yes) + 2(No)] = 4$$
$$S_{'Mild'} = [4(Yes) + 2(No)] = 6$$
$$S_{'Cool'} = [3(Yes) + 1(No)] = 4$$
$$S = [9(Yes) + 5(No)] = 14$$

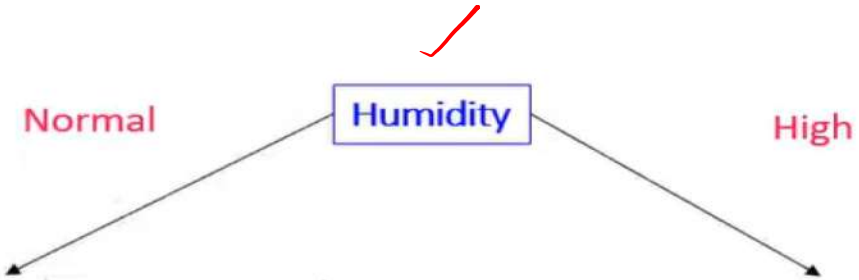
Entropy(temperature='Hot') = 1.0
Entropy(temperature='Mild') = 0.918
Entropy(temperature='Cool') = 0.811

$$Information('temperature') = \frac{4}{14} * 1 + \frac{6}{14} * 0.918 + \frac{4}{14} * 0.811$$
$$= 0.286 + 0.393 + 0.232$$
$$= 0.911$$

Compute Gain (*S, A*)

$$Gain(S,'temperature') = Entropy(S) - Information('temperature')$$
$$= 0.940 - 0.911$$
$$= 0.029$$

Step 2



Humidity	PlayTennis
Normal	Yes
Normal	No
Normal	Yes
Normal	Yes
Normal	Yes
Normal	Yes
Normal	Yes

Humidity	PlayTennis
High	No
High	No
High	Yes
High	Yes
High	No
High	Yes
High	No

Humidity	Yes	No	Entropy
High	3	4	0.985
Normal	6	1	0.591

S. No.	Outlook	Temperature	Humidity	Windy	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rainy	Mild	High	Weak	Yes
5	Rainy	Cool	Normal	Weak	Yes
6	Rainy	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rainy	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rainy	Mild	High	Strong	No

Entropy(humidity='High') = $-(3/7)*\log_2(3/7) - (4/7)*\log_2(4/7) = 0.985$

Entropy(humidity='Normal') = $-(6/7)*\log_2(6/7) - (1/7)*\log_2(1/7) = 0.592$

Step 2

Compute Information (A)

$$Information('humidity') = \sum_{v \in \{'High','Normal'\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$S_{'High'} = [3(Yes) + 4(No)] = 7$$

$$S_{'Normal'} = [6(Yes) + 1(No)] = 7$$

$$S = [9(Yes) + 5(No)] = 14$$

Entropy(humidity='High') = 0.985
Entropy(humidity='Normal') = 0.592

$$Information('humidity') = \frac{7}{14} * 0.985 + \frac{7}{14} * 0.592$$

$$= 0.493 + 0.296$$

$$= 0.789$$

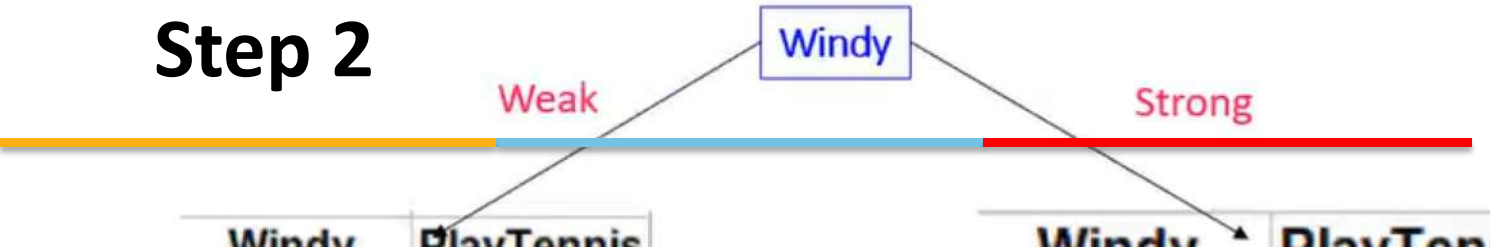
Compute Gain (S, A)

$$Gain(S,'humidity') = Entropy(S) - Information('humidity')$$

$$= 0.940 - 0.789$$

$$= 0.151$$

Step 2



Windy	PlayTennis
Weak	No
Weak	Yes
Weak	Yes
Weak	Yes
Weak	No
Weak	Yes
Weak	Yes
Weak	Yes

Windy	PlayTennis
Strong	No
Strong	No
Strong	Yes
Strong	Yes
Strong	Yes
Strong	No

S. No.	Outlook	Temperature	Humidity	Windy	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rainy	Mild	High	Weak	Yes
5	Rainy	Cool	Normal	Weak	Yes
6	Rainy	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rainy	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rainy	Mild	High	Strong	No

Windy	Yes	No	Entropy
Strong	3	3	1
Weak	6	2	0.811

Entropy(windy='Strong') = $-(3/6)*\log_2(3/6) - (3/6)*\log_2(3/6) = 1.0$

Entropy(windy='Weak') = $-(6/8)*\log_2(6/8) - (2/8)*\log_2(2/8) = 0.811$

Step 2

Compute Information (*A*)

$$Information('windy') = \sum_{v \in \{'Strong','Weak'\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$S_{'Strong'} = [3(Yes) + 3(No)] = 6$$

$$S_{'Weak'} = [6(Yes) + 2(No)] = 8$$

$$S = [9(Yes) + 5(No)] = 14$$

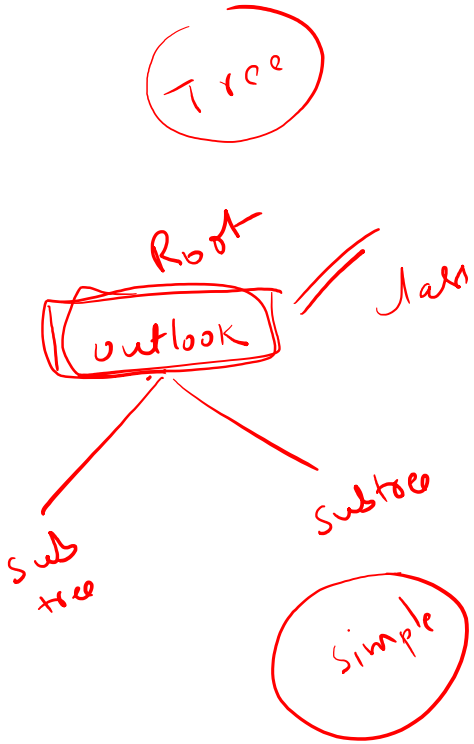
Entropy(windy='Strong') = 1.0
Entropy(windy='Weak') = 0.811

$$Information('windy') = \frac{6}{14} * 1 + \frac{8}{14} * 0.811$$
$$= 0.429 + 0.463$$
$$= 0.892$$

Compute Gain (*S, A*)

$$Gain(S,'windy') = Entropy(S) - Information('windy')$$
$$= 0.940 - 0.892$$
$$= 0.048$$

Step 3

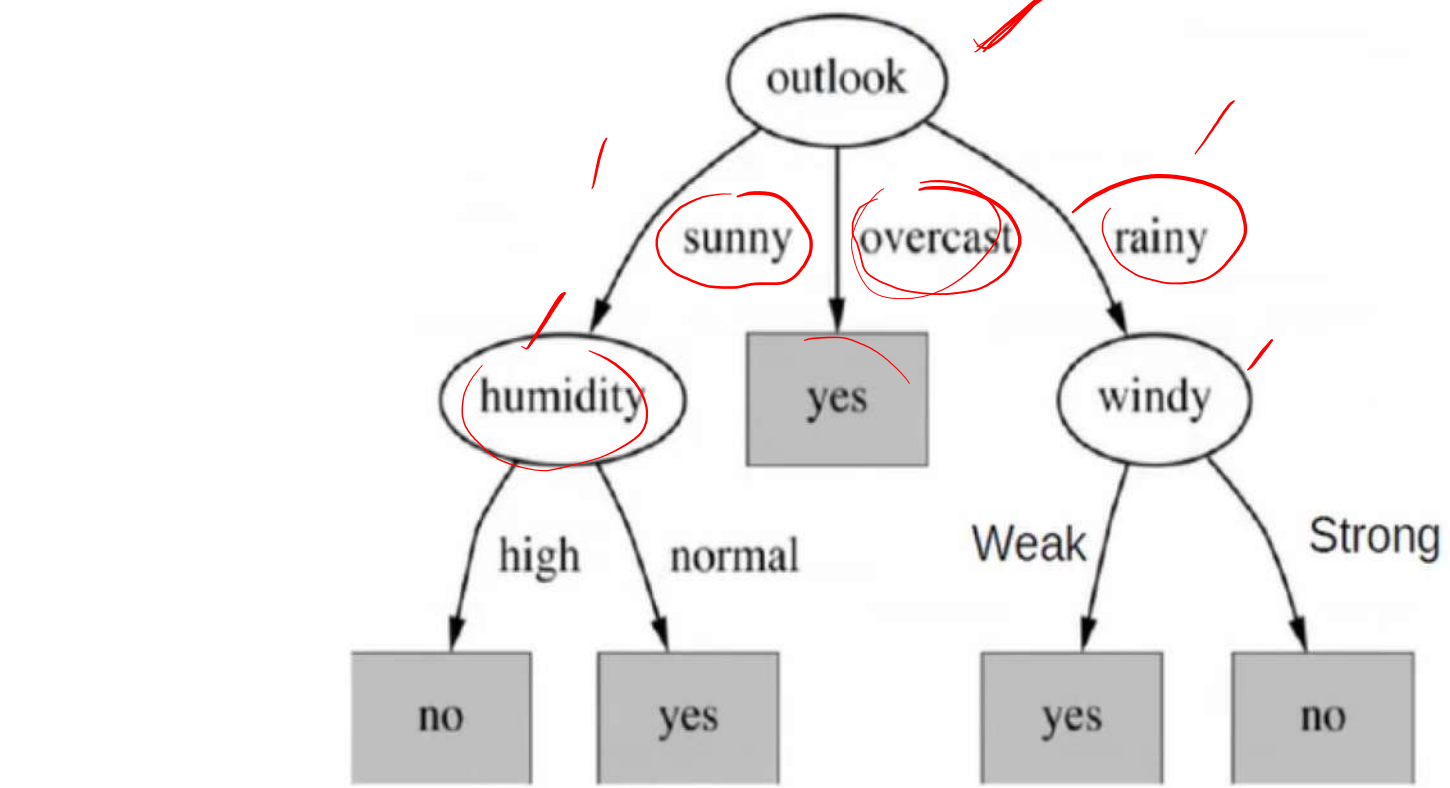


- Pick the highest gain attribute

Attribute	Gain
Outlook	0.247
Temperature	0.029
Humidity	0.151
Windy	0.048

ROOT NODE: Outlook

Continue until we get a complete tree



Prediction Success Table

TRUE	Survive	Dead	Total
Survive	158	20	178
Dead	7	30	37
Total	165	50	215

- % correct = 188/215=0.8744
- In CART terminology the performance of the tree **T** is described by the error rate **R(T)**
  - in our example **R(T)** = 1 – 0.8744 = 0.1256

CART-Example 1

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

- For the given Play Tennis Data set apply the Decision Tree algorithm and find the optimal decision tree.
- Also predict class label for the following example…?

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	Normal	True	?

Example 1

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

Outlook

Overcast	4	Yes	4
		No	0
Sunny	5	Yes	2
		No	3
Rainy	5	Yes	3
		No	2

Attribute	Rules	Error	Total Error
Outlook	Sunny → No	2/5	4/14
	Overcast → Yes	0/4	
	Rainy → Yes	2/5	

Example 1

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

Temp

Hot	4	Yes	2
		No	2
Mild	6	Yes	4
		No	2
Cold	4	Yes	3
		No	1

Attribute	Rules	Error	Total Error
Temp	Hot → No/Yes	2/4	5/14
	Mild → Yes	2/6	
	Cool → Yes	1/4	

Example 1

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

Humidity

High	7	Yes	3
		No	4
Normal	7	Yes	6
		No	1

Attribute	Rules	Error	Total Error
Humidity	High→ No	3/7	4/14
	Normal → Yes	1/7	

Example 1

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

Windy

False	8	Yes	6
		No	2
True	6	Yes	3
		No	3

Attribute	Rules	Error	Total Error
Windy	True → No/Yes	3/6	5/14
	False → Yes	2/8	

Example 1

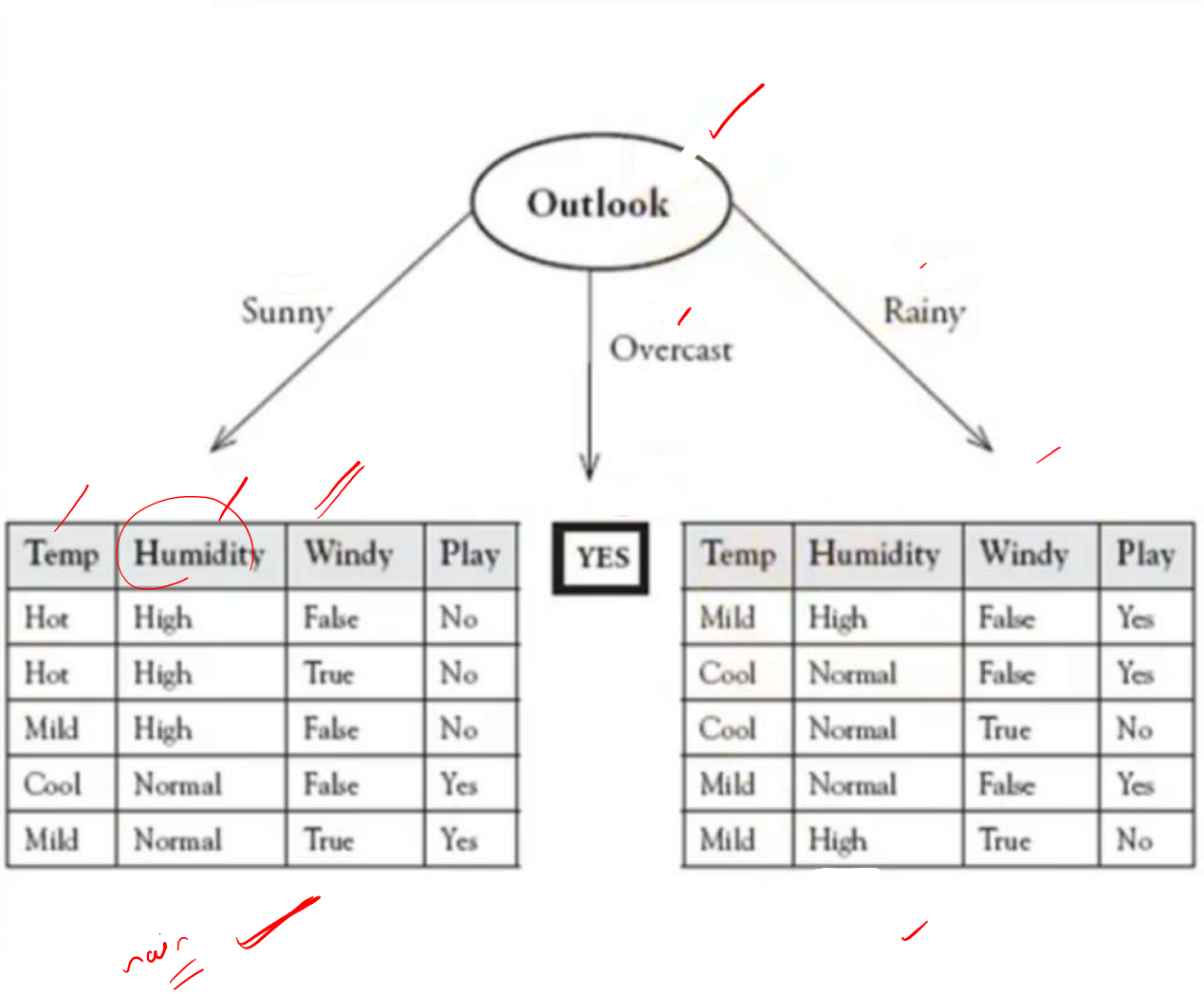
Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny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

Root

Attribute	Rules	Error	Total Error
Outlook	Sunny → No	2/5	**4/14**
	Overcast → Yes	0/4	
	Rainy → Yes	2/5	
Temp	hot → No	2/4	**5/14**
	Mild → Yes	2/6	
	Cool → Yes	1/4	
Humidity	high → No	3/7	**4/14**
	Normal → Yes	1/7	
Windy	False → Yes	2/8	**5/14**
	True → No	3/6	

Example 1

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



Temp	Humidity	Windy	Play
Hot	High	False	No
Hot	High	True	No
Mild	High	False	No
Cool	Normal	False	Yes
Mild	Normal	True	Yes

Temp	Humidity	Windy	Play
Mild	High	False	Yes
Cool	Normal	False	Yes
Cool	Normal	True	No
Mild	Normal	False	Yes
Mild	High	True	No

Example 1

Temp	Humidity	Windy	Play
Hot	High	False	No
Hot	High	True	No
Mild	High	False	No
Cool	Normal	False	Yes
Mild	Normal	True	Yes

Attribute	Rules	Error	Total Error
Temp	Hot → No	0/2	1/5
	Mild → No/Yes	1/2	
	Cool → Yes	0/1	
Humidity	High → No	0/3	0/5
	Normal → Yes	0/2	
Windy	False → No	1/3	2/5
	True → Yes/No	1/2	

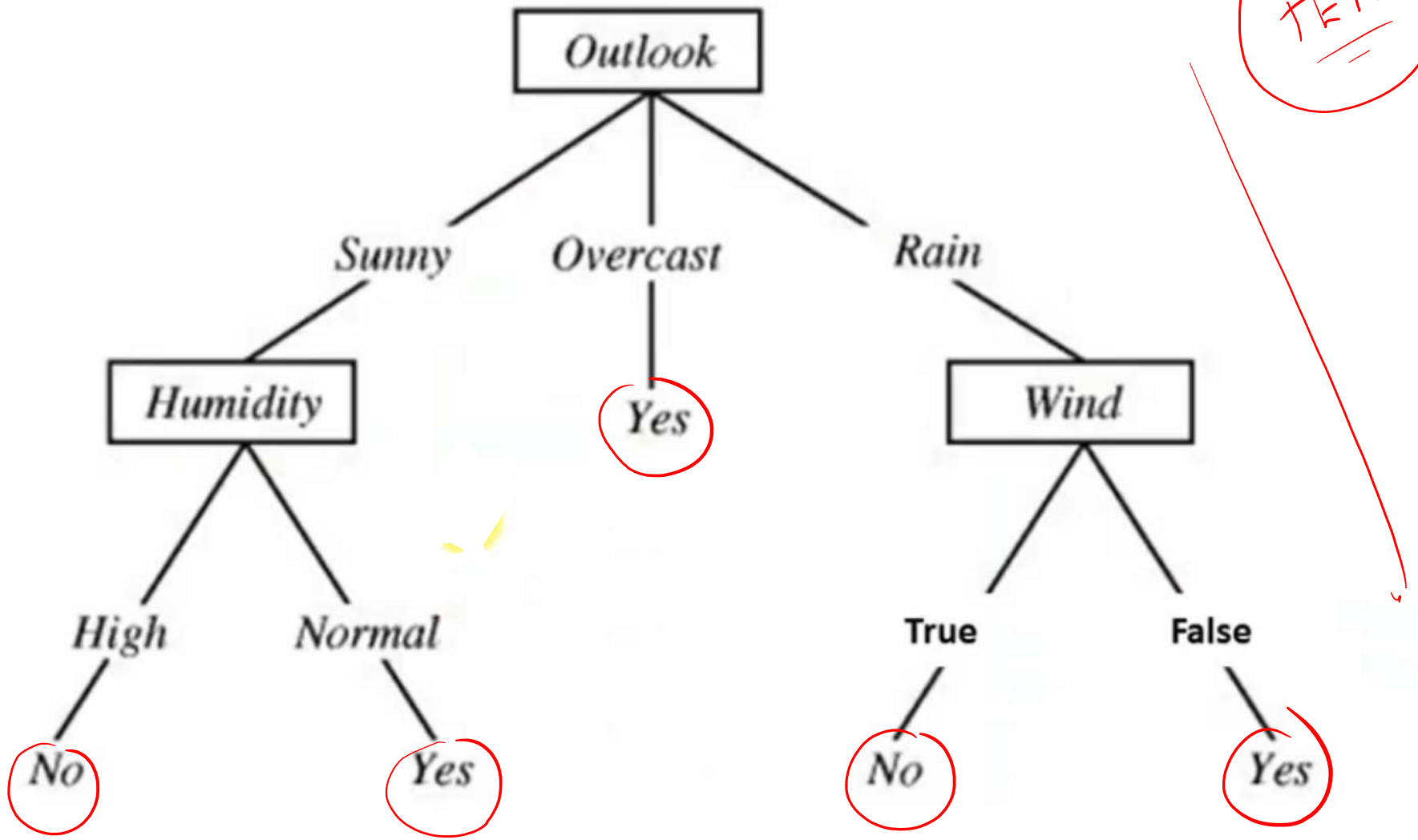
Minim

Example 1

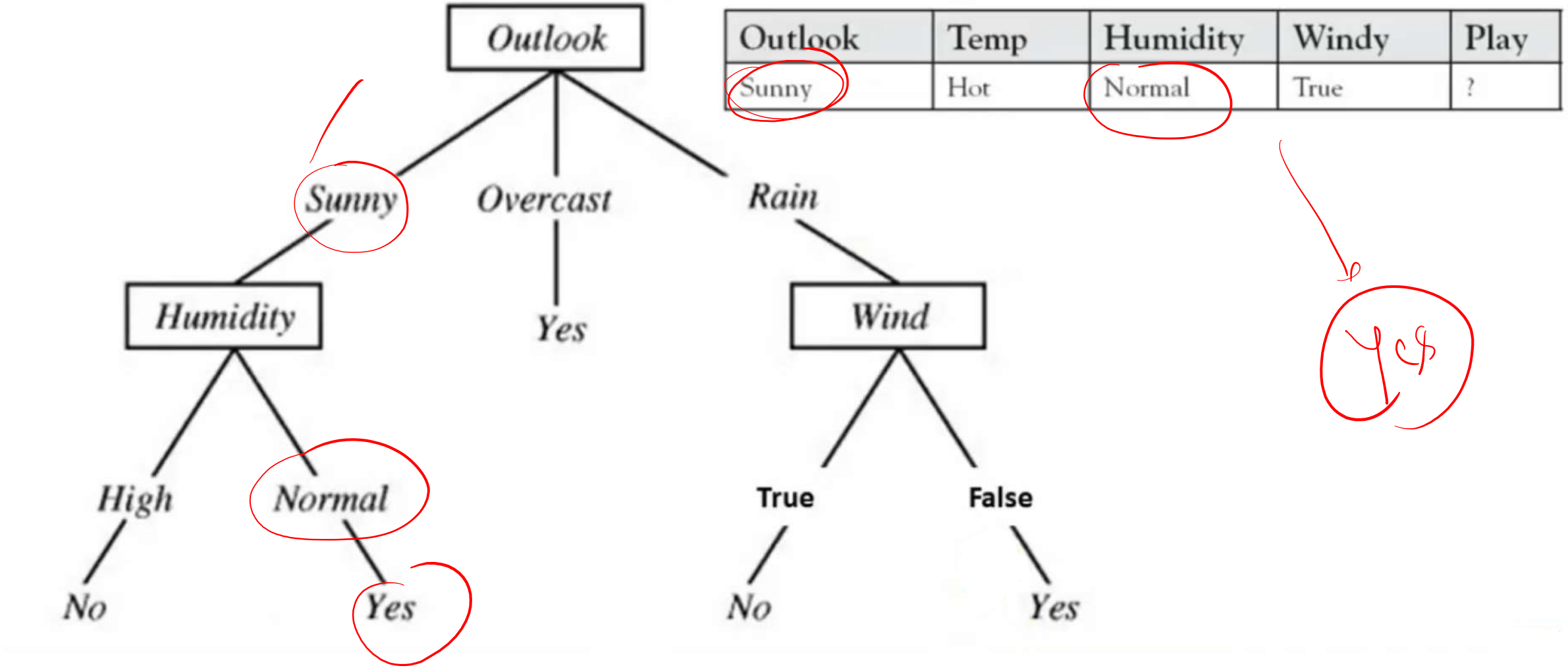
Temp	Humidity	Windy	Play
Mild	High	False	Yes
Cool	Normal	False	Yes
Cool	Normal	True	No
Mild	Normal	False	Yes
Mild	High	True	No

Attribute	Rules	Error	Total Error
Temp	Mild → Yes	1/3	2/5
	Cool → yes/No	1/2	
Humidity	High → No/Yes	1/2	2/5
	Normal → Yes	1/3	
Windy	False → Yes	0/3	0/5
	True → No	0/2	

Example 1


Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
True
False
No
Yes
No
Yes

Example 1



Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	Normal	True	?

Example 2

Age	Job	House	Credit	Loan Approved
Young	False	No	Fair	No
Young	False	No	Good	No
Young	True	No	Good	Yes
Young	True	Yes	Fair	Yes
Young	False	No	Fair	No
Middle	False	No	Fair	No
Middle	False	No	Good	No
Middle	True	Yes	Good	Yes
Middle	False	Yes	Excellent	Yes
Middle	False	Yes	Excellent	Yes
Old	False	Yes	Excellent	Yes
Old	False	Yes	Good	Yes
Old	True	No	Good	Yes
Old	True	No	Excellent	Yes
Old	False	No	Fair	No

- For the given Loan approval Data set apply the Decision Tree algorithm and find the optimal decision tree.
- Also find whether the loan is approved or not for the following example…?

Age	Job	House	Credit	Loan Approved
Young	False	No	Good	?

Example 2

Age	Job	House	Credit	Loan Approved
Young	False	No	Fair	No
Young	False	No	Good	No
Young	True	No	Good	Yes
Young	True	Yes	Fair	Yes
Young	False	No	Fair	No
Middle	False	No	Fair	No
Middle	False	No	Good	No
Middle	True	Yes	Good	Yes
Middle	False	Yes	Excellent	Yes
Middle	False	Yes	Excellent	Yes
Old	False	Yes	Excellent	Yes
Old	False	Yes	Good	Yes
Old	True	No	Good	Yes
Old	True	No	Excellent	Yes
Old	False	No	Fair	No

Age

Young	5	Yes	2
		No	3
Middle	5	Yes	3
		No	2
Old	5	Yes	4
		No	1

Attribute	Rules	Error	Total Error
Age	Young->No	2/5	5/15
	Middle->Yes	2/5	
	Old->Yes	1/5	

Example 2

Age	Job	House	Credit	Loan Approved
Young	False	No	Fair	No
Young	False	No	Good	No
Young	True	No	Good	Yes
Young	True	Yes	Fair	Yes
Young	False	No	Fair	No
Middle	False	No	Fair	No
Middle	False	No	Good	No
Middle	True	Yes	Good	Yes
Middle	False	Yes	Excellent	Yes
Middle	False	Yes	Excellent	Yes
Old	False	Yes	Excellent	Yes
Old	False	Yes	Good	Yes
Old	True	No	Good	Yes
Old	True	No	Excellent	Yes
Old	False	No	Fair	No

Job			
False	10	Yes	4
		No	6
True	5	Yes	5
		No	0

Attribute	Rules	Error	Total Error
Job	False->No	4/10	**4/15**
	True->Yes	0/5	

Example 2

Age	Job	House	Credit	Loan Approved
Young	False	No	Fair	No
Young	False	No	Good	No
Young	True	No	Good	Yes
Young	True	Yes	Fair	Yes
Young	False	No	Fair	No
Middle	False	No	Fair	No
Middle	False	No	Good	No
Middle	True	Yes	Good	Yes
Middle	False	Yes	Excellent	Yes
Middle	False	Yes	Excellent	Yes
Old	False	Yes	Excellent	Yes
Old	False	Yes	Good	Yes
Old	True	No	Good	Yes
Old	True	No	Excellent	Yes
Old	False	No	Fair	No

House

No	9	Yes	3
		No	6
Yes	6	Yes	6
		No	0

Attribute	Rules	Error	Total Error
House	No->No	3/9	**3/15**
	Yes->yes	0/6	

Example 2

Age	Job	House	Credit	Loan Approved
Young	False	No	Fair	No
Young	False	No	Good	No
Young	True	No	Good	Yes
Young	True	Yes	Fair	Yes
Young	False	No	Fair	No
Middle	False	No	Fair	No
Middle	False	No	Good	No
Middle	True	Yes	Good	Yes
Middle	False	Yes	Excellent	Yes
Middle	False	Yes	Excellent	Yes
Old	False	Yes	Excellent	Yes
Old	False	Yes	Good	Yes
Old	True	No	Good	Yes
Old	True	No	Excellent	Yes
Old	False	No	Fair	No

Credit

Fair	5	Yes	1
		No	4
Good	6	Yes	4
		No	2
Excellent	4	Yes	4
		No	0

Attribute	Rules	Error	Total Error
Credit	Fair->No	1/5	**3/15**
	Good->Yes	2/6	
	Excellent->Yes	0/4	

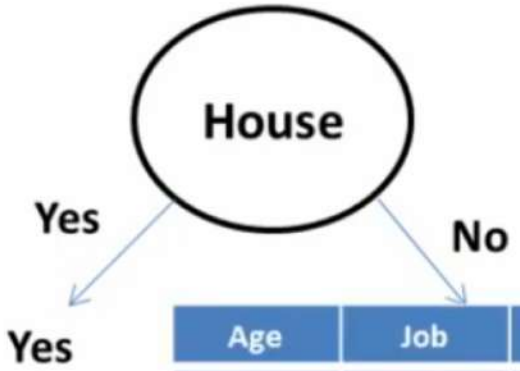
Example 2

Age	Job	House	Credit	Loan Approved
Young	False	No	Fair	No
Young	False	No	Good	No
Young	True	No	Good	Yes
Young	True	Yes	Fair	Yes
Young	False	No	Fair	No
Middle	False	No	Fair	No
Middle	False	No	Good	No
Middle	True	Yes	Good	Yes
Middle	False	Yes	Excellent	Yes
Middle	False	Yes	Excellent	Yes
Old	False	Yes	Excellent	Yes
Old	False	Yes	Good	Yes
Old	True	No	Good	Yes
Old	True	No	Excellent	Yes
Old	False	No	Fair	No

Attribute	Rules	Error	Total Error
Age	Young->No	2/5	**5/15**
	Middle->Yes	2/5	
	Old->Yes	1/5	
Job	False->No	4/10	**4/15**
	True->Yes	0/5	
House	No->No	3/9	**3/15**
	Yes->yes	0/6	
Credit	Fair->No	1/5	**3/15**
	Good->Yes	2/6	
	Excellent->Yes	0/4	

Example 2

Age	Job	House	Credit	Loan Approved
Young	False	No	Fair	No
Young	False	No	Good	No
Young	True	No	Good	Yes
Young	True	Yes	Fair	Yes
Young	False	No	Fair	No
Middle	False	No	Fair	No
Middle	False	No	Good	No
Middle	True	Yes	Good	Yes
Middle	False	Yes	Excellent	Yes
Middle	False	Yes	Excellent	Yes
Old	False	Yes	Excellent	Yes
Old	False	Yes	Good	Yes
Old	True	No	Good	Yes
Old	True	No	Excellent	Yes
Old	False	No	Fair	No



Age	Job	Credit	LA
Young	False	Fair	No
Young	False	Good	No
Young	True	Good	Yes
Young	False	Fair	No
Middle	False	Fair	No
Middle	False	Good	No
Old	True	Good	Yes
Old	True	Excellent	Yes
Old	False	Fair	No

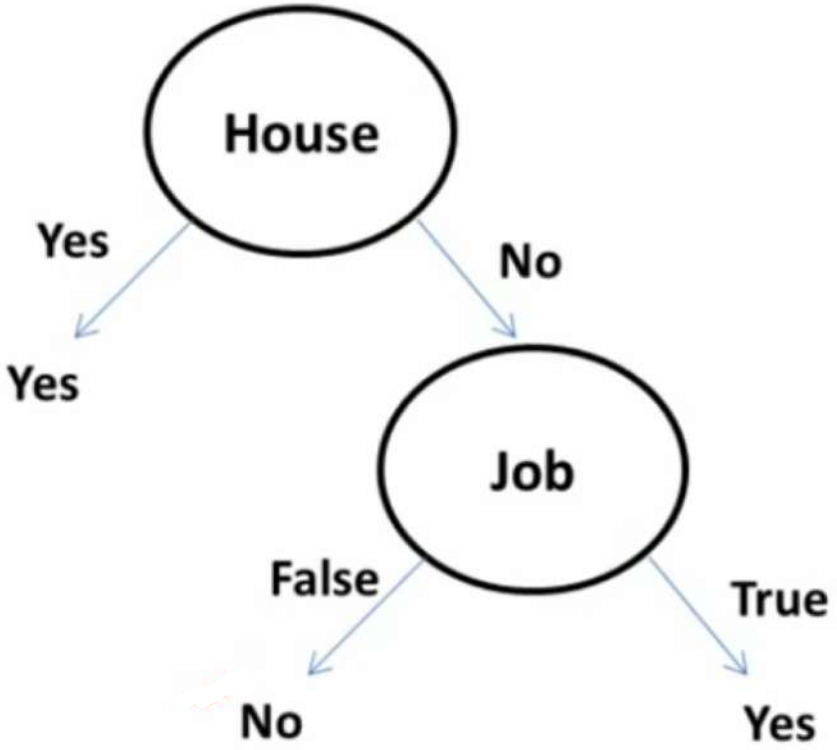
Example 2

Age	Job	Credit	LA
Young	False	Fair	No
Young	False	Good	No
Young	True	Good	Yes
Young	False	Fair	No
Middle	False	Fair	No
Middle	False	Good	No
Old	True	Good	Yes
Old	True	Excellent	Yes
Old	False	Fair	No

Attribute	Rules	Error	Total Error
Age	Young->No	1/4	**2/9**
	Middle->No	0/2	
	Old->Yes	1/3	
Job	False->No	0/6	**0/9**
	True->Yes	0/3	
Credit	Fair->No	0/4	**2/9**
	Good->Yes/No	2/4	
	Excellent->Yes	0/1	

Example 2

Age	Job	Credit	LA
Young	False	Fair	No
Young	False	Good	No
Young	True	Good	Yes
Young	False	Fair	No
Middle	False	Fair	No
Middle	False	Good	No
Old	True	Good	Yes
Old	True	Excellent	Yes
Old	False	Fair	No



Age	Job	House	Credit	Loan Approved
Young	False	No	Good	?

Example 3

City Size	Avg Income	Local Investors	LOHAS Awareness	Decision
Big	High	Yes	High	Yes
Med	Med	No	Med	No
Small	Low	Yes	Low	No
Big	High	No	High	Yes
Small	Med	Yes	High	No
Med	High	Yes	Med	Yes
Med	Med	Yes	Med	No
Big	Med	No	Med	No
Med	High	Yes	Low	No
Small	High	No	High	Yes
Small	Med	No	High	No
Med	High	No	Med	No

- For the given City Size, Avg. Income, Local Investors, LOHAS Awareness Data set apply the Decision Tree algorithm and find the optimal decision tree.
- Also predict the class label for new example.

City Size	Avg Income	Local Investors	LOHAS Awareness	Decision
Med	Med	No	Med	?

Example 3

City Size	Avg Income	Local Investors	LOHAS Awareness	Decision
Big	High	Yes	High	Yes
Med	Med	No	Med	No
Small	Low	Yes	Low	No
Big	High	No	High	Yes
Small	Med	Yes	High	No
Med	High	Yes	Med	Yes
Med	Med	Yes	Med	No
Big	Med	No	Med	No
Med	High	Yes	Low	No
Small	High	No	High	Yes
Small	Med	No	High	No
Med	High	No	Med	No

City Size

Big	3	Yes	2
		No	1
Medium	5	Yes	1
		No	4
Small	4	Yes	1
		No	3

Attribute	Rules	Error	Total Error
City Size	Big->Yes	1/3	3/12
	Medium->No	1/5	
	Small->No	1/4	

Example 3

City Size	Avg Income	Local Investors	LOHAS Awareness	Decision
Big	High	Yes	High	Yes
Med	Med	No	Med	No
Small	Low	Yes	Low	No
Big	High	No	High	Yes
Small	Med	Yes	High	No
Med	High	Yes	Med	Yes
Med	Med	Yes	Med	No
Big	Med	No	Med	No
Med	High	Yes	Low	No
Small	High	No	High	Yes
Small	Med	No	High	No
Med	High	No	Med	No

Avg. Inc.

High	6	Yes	4
		No	2
Medium	5	Yes	0
		No	5
Low	1	Yes	0
		No	1

Attribute	Rules	Error	Total Error
Avg. Inc.	High->Yes	2/6	**2/12**
	Medium->No	0/5	
	Low->No	0/1	

Example 3

City Size	Avg Income	Local Investors	LOHAS Awareness	Decision
Big	High	Yes	High	Yes
Med	Med	No	Med	No
Small	Low	Yes	Low	No
Big	High	No	High	Yes
Small	Med	Yes	High	No
Med	High	Yes	Med	Yes
Med	Med	Yes	Med	No
Big	Med	No	Med	No
Med	High	Yes	Low	No
Small	High	No	High	Yes
Small	Med	No	High	No
Med	High	No	Med	No

Local Investors

Yes	6	Yes	2
		No	4
No	6	Yes	2
		No	4

Attribute	Rules	Error	Total Error
Local Investors	Yes->No	2/6	**4/12**
	No->No	2/6	

Example 3

City Size	Avg Income	Local Investors	LOHAS Awareness	Decision
Big	High	Yes	High	Yes
Med	Med	No	Med	No
Small	Low	Yes	Low	No
Big	High	No	High	Yes
Small	Med	Yes	High	No
Med	High	Yes	Med	Yes
Med	Med	Yes	Med	No
Big	Med	No	Med	No
Med	High	Yes	Low	No
Small	High	No	High	Yes
Small	Med	No	High	No
Med	High	No	Med	No

Lohas Awareness

High	5	Yes	3
		No	2
Med	5	Yes	1
		No	4
Low	2	Yes	0
		No	2

Attribute	Rules	Error	Total Error
Lohas Awareness	High->Yes	2/5	**3/12**
	Med->No	1/5	
	Low->No	0/2	

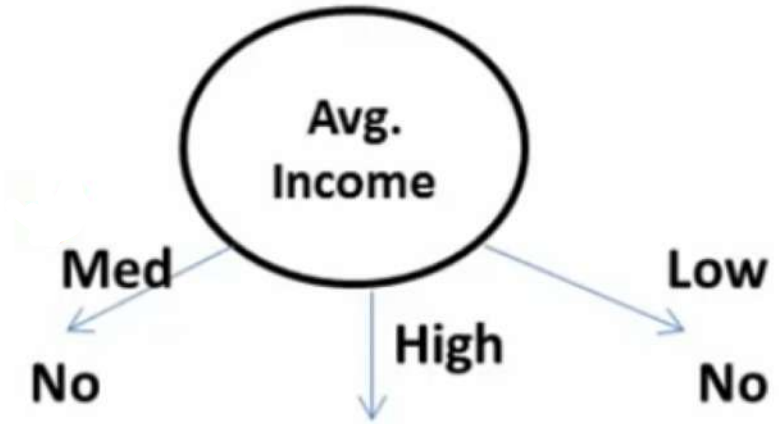
Example 3

City Size	Avg Income	Local Investors	LOHAS Awareness	Decision
Big	High	Yes	High	Yes
Med	Med	No	Med	No
Small	Low	Yes	Low	No
Big	High	No	High	Yes
Small	Med	Yes	High	No
Med	High	Yes	Med	Yes
Med	Med	Yes	Med	No
Big	Med	No	Med	No
Med	High	Yes	Low	No
Small	High	No	High	Yes
Small	Med	No	High	No
Med	High	No	Med	No

Attribute	Rules	Error	Total Error
City Size	Big->Yes	1/3	**3/12**
	Medium->No	1/5	
	Small->No	1/4	
Avg. Inc.	High->Yes	2/6	**2/12**
	Medium->No	0/5	
	Low->No	0/1	
Local Investors	Yes->No	2/6	**4/12**
	Yes->No	2/6	
Lohas Awareness	High->Yes	2/5	**3/12**
	Med->No	1/5	
	Low->No	0/2	

Example 3

City Size	Avg Income	Local Investors	LOHAS Awareness	Decision
Big	High	Yes	High	Yes
Med	Med	No	Med	No
Small	Low	Yes	Low	No
Big	High	No	High	Yes
Small	Med	Yes	High	No
Med	High	Yes	Med	Yes
Med	Med	Yes	Med	No
Big	Med	No	Med	No
Med	High	Yes	Low	No
Small	High	No	High	Yes
Small	Med	No	High	No
Med	High	No	Med	No



City Size	Local Investor	LOHAS Awareness	Decision
Big	Yes	High	Yes
Big	No	High	Yes
Med	Yes	Med	Yes
Med	Yes	Low	No
Small	No	High	Yes
Med	No	Med	No

Example 3

City Size	Local Investor	LOHAS Awareness	Decision
Big	Yes	High	Yes
Big	No	High	Yes
Med	Yes	Med	Yes
Med	Yes	Low	No
Small	No	High	Yes
Med	No	Med	No

City Size

Big	3	Yes	2
		No	1
Medium	5	Yes	1
		No	4
Small	4	Yes	1
		No	3

Attribute	Rules	Error	Total Err
City Size	Big->Yes	1/3	**3/12**
	Medium->No	1/5	
	Small->No	1/4	

Example 3

City Size	Local Investor	LOHAS Awareness	Decision
Big	Yes	High	Yes
Big	No	High	Yes
Med	Yes	Med	Yes
Med	Yes	Low	No
Small	No	High	Yes
Med	No	Med	No

Local Investors

Yes	6	Yes	2
		No	4
No	6	Yes	2
		No	4

Attribute	Rules	Error	Total Error
Local Investors	Yes->No	2/6	**4/12**
	No->No	2/6	

Example 3

City Size	Local Investor	LOHAS Awareness	Decision
Big	Yes	High	Yes
Big	No	High	Yes
Med	Yes	Med	Yes
Med	Yes	Low	No
Small	No	High	Yes
Med	No	Med	No

Lohas Awareness

High	5	Yes	3
		No	2
Med	5	Yes	1
		No	4
Low	2	Yes	0
		No	2

Attribute	Rules	Error	Total Error
Lohas Awareness	High->Yes	2/5	3/12
	Med->No	1/5	
	Low->No	0/2	

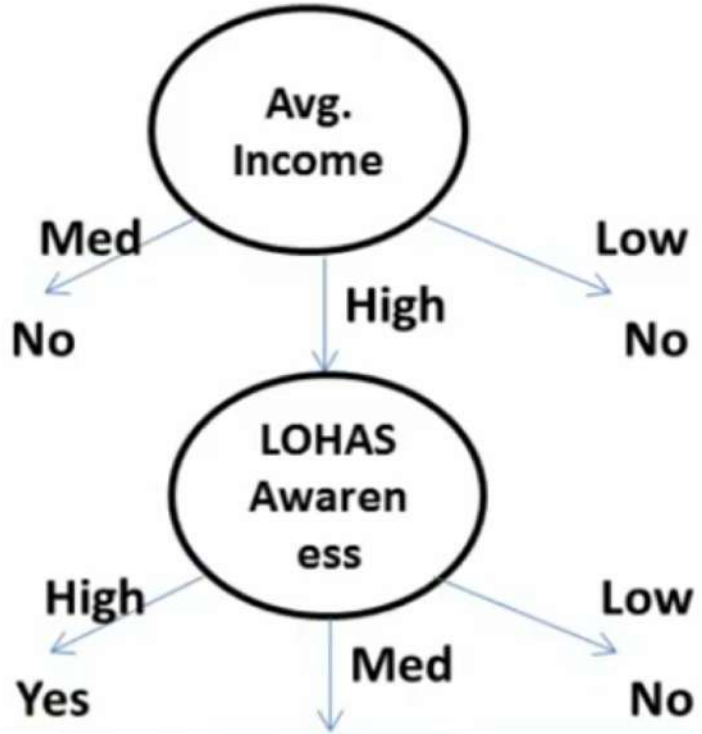
Example 3

City Size	Local Investor	LOHAS Awareness	Decision
Big	Yes	High	Yes
Big	No	High	Yes
Med	Yes	Med	Yes
Med	Yes	Low	No
Small	No	High	Yes
Med	No	Med	No

Attribute	Rules	Error	Total Error
City Size	Big->Yes	0/2	**1/6**
	Medium->No	1/3	
	Small->Yes	0/1	
Local Investors	Yes->Yes	1/3	**2/6**
	No->Yes	1/3	
Lohas Awareness	High->Yes	0/3	**1/6**
	Med->No/Yes	1/2	
	Low->No	0/1	

Example 3

City Size	Local Investor	LOHAS Awareness	Decision
Big	Yes	High	Yes
Big	No	High	Yes
Med	Yes	Med	Yes
Med	Yes	Low	No
Small	No	High	Yes
Med	No	Med	No



City Size	Local Investor	Decision
Med	Yes	Yes
Med	No	No

Example 3

City Size	Local Investor	Decision
Med	Yes	Yes
Med	No	No

City Size

Med	2	Yes	1
		No	1

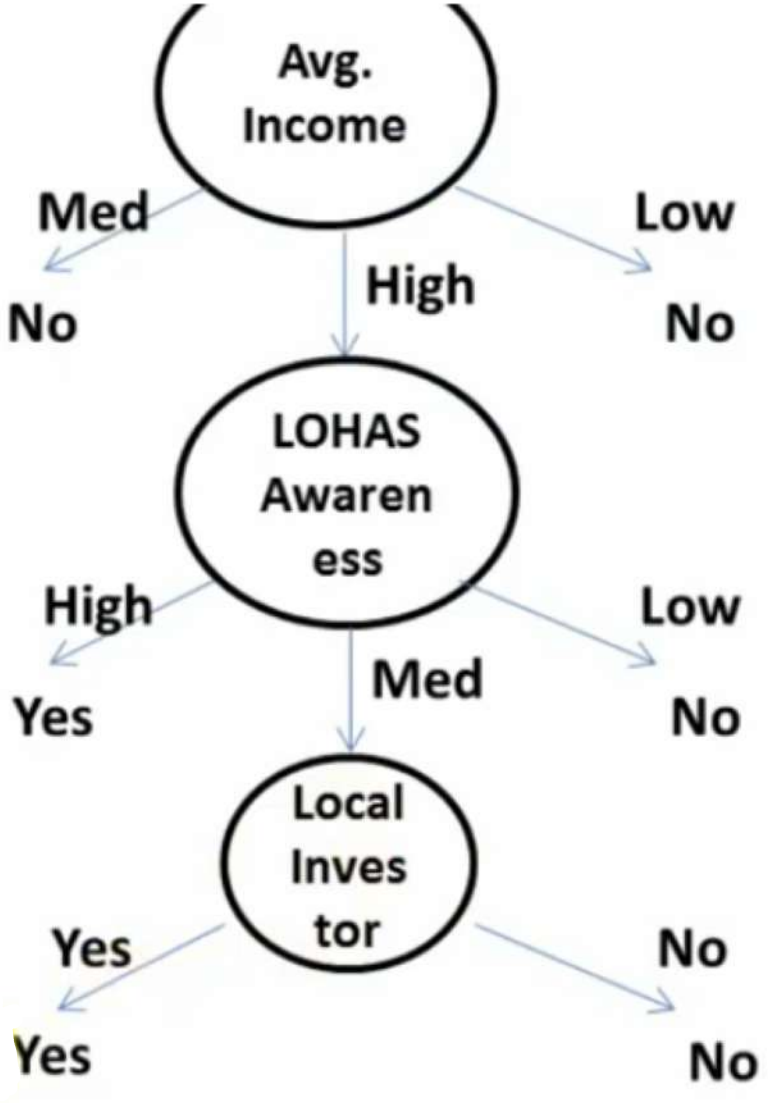
Attribute	Rules	Error	Total Error
City Size	Med->Yes/No	1/2	**1/2**

Local Investor

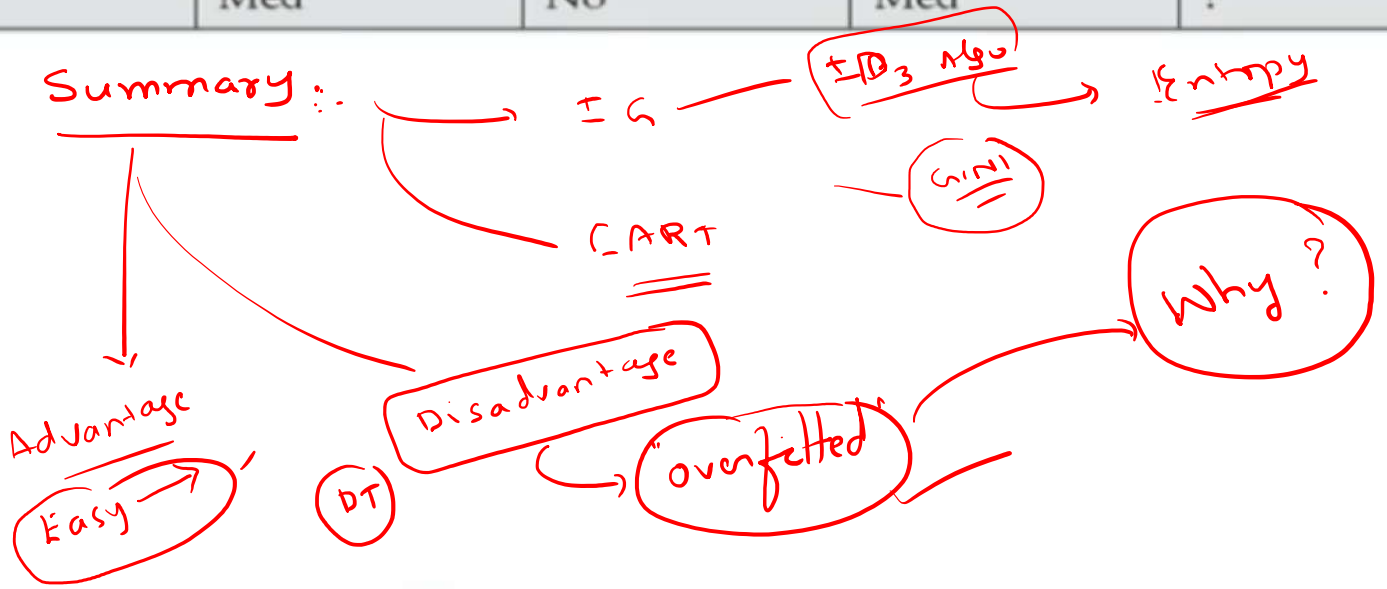
Yes	1	Yes	1
		No	0
No	1	Yes	0
		No	1

Attribute	Rules	Error	Total Error
Local Investor	Yes->Yes	0/2	**0/2**
	No->No	0/1	

Example 3

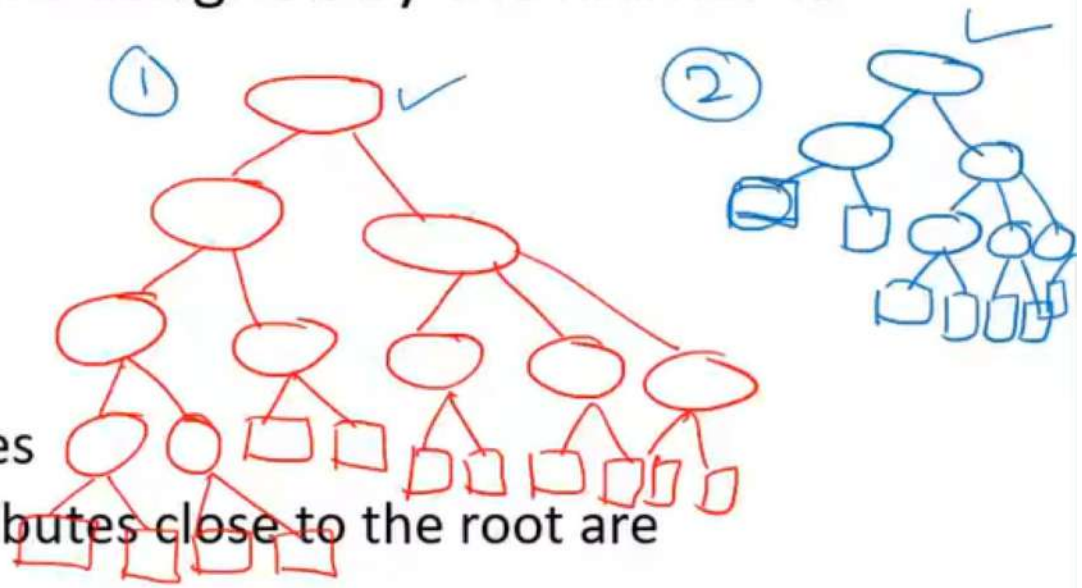


City Size	Avg Income	Local Investors	LOHAS Awareness	Decision
Med	Med	No	Med	?



Inductive Bias in ID3 Decision Tree Learning

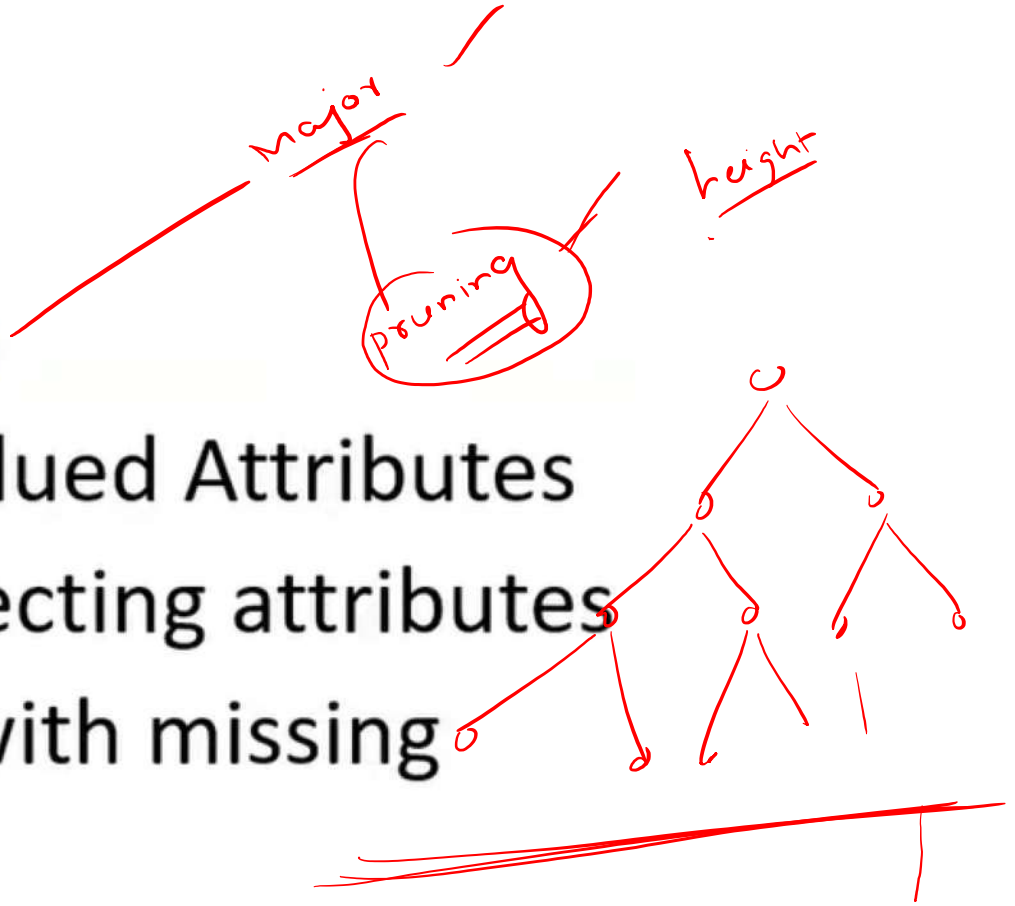
- Inductive bias is a set of assumptions that, together with the training data, deductively justify the classifications assigned by the learner to future instances.



- Many decision trees may be consistent
- How to select the one hypothesis?
  - Shorter trees are preferred over larger trees
  - Trees that place high information gain attributes close to the root are preferred over those that do not.

Issues in Decision Tree Learning

- Avoiding Overfitting the data
- Incorporating Continuous-Valued Attributes
- Alternative measures for Selecting attributes
- Handling training examples with missing attribute values
- Handling attributes with differing costs



Issues in Decision Tree Learning

- Avoiding Overfitting the data
  - Approaches that stop growing the tree earlier, before it reaches the point where it perfectly classifies the training data
  - Approaches that allow the tree to overfit the data, and then post-prune the tree

Issues in Decision Tree Learning

- Avoiding Overfitting the data
  - What criterion to be used to determine the correct final tree size?
    - Training and validation set approach
    - Use all the available data for training, but apply a statistical test to estimate whether expanding (or pruning) a particular node is likely to produce improvement beyond the training dataset. (e.g. use of chi-square test)
    - Use of Minimum Description Length principle

Issues in Decision Tree Learning

- Avoiding Overfitting the data
  - Reduced-Error Pruning
    - Consider each of the decision nodes in the tree to be candidates for pruning
    - Pruning
      - removing the subtree rooted at that node, making it a leaf node, and assigning it the most common classification label of the training examples affiliated with that node.
      - Nodes are removed only if the resulting pruned tree performs no worse than the original over the validation set
    - Limitation – need enough data to be partitioned into training and validation sets

Issues in Decision Tree Learning

- Avoiding Overfitting the data
- Incorporating Continuous-Valued Attributes
- Alternative measures for Selecting attributes
- Handling training examples with missing attribute values
- Handling attributes with differing costs

Issues in Decision Tree Learning

- Incorporating Continuous-Valued Attributes
  - The attributes tested in the decision nodes can be continuous-valued
  - Dynamically define new discrete valued attributes that partition the continuous attribute value into a discrete set of intervals using threshold
  - How to select threshold ?
    - Threshold that produces the greatest information gain

Issues in Decision Tree Learning

- Avoiding Overfitting the data
- Incorporating Continuous-Valued Attributes
- Alternative measures for Selecting attributes
- Handling training examples with missing attribute values
- Handling attributes with differing costs

Issues in Decision Tree Learning

- Alternative measures for Selecting attributes

$$GainRatio(S,A) = \frac{Gain(S,A)}{SplitInformation(S,A)}$$

$$SplitInformation(S,A) = -\sum_{i=1}^{c} \frac{|S_v|}{|S|} \log_2 \left(\frac{|S_v|}{|S|}\right)$$

$$Gini(S) = 1 - \sum_{i=1}^{c} p_i^{\,2}$$

Attribute with lower Gini Index should be preferred

$$GiniIndex(S,A) = \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Gini(S_v)$$

Issues in Decision Tree Learning

- Avoiding Overfitting the data
- Incorporating Continuous-Valued Attributes
- Alternative measures for Selecting attributes
- Handling training examples with missing attribute values
- Handling attributes with differing costs

Issues in Decision Tree Learning

- Handling training examples with missing attribute values
  - If node n tests A, assign the most common value of A among the other examples at node n
  - Assign the most common value of A among the other examples with same target value (classification)

Issues in Decision Tree Learning

- Avoiding Overfitting the data
- Incorporating Continuous-Valued Attributes
- Alternative measures for Selecting attributes
- Handling training examples with missing attribute values
- Handling attributes with differing costs

Issues in Decision Tree Learning

- Handling attributes with differing costs
  - Replace Gain by

$$\frac{Gain^2(S,A)}{Cost(A)}$$

Tan and Schlimmer, 1990

$$\frac{2^{Gain(S,A)}-1}{(Cost(A)+1)^w}$$

Nunez, 1988

where w ∈ [0, 1] determines importance of cost

Prefer shorter hypotheses: Occam's razor

Why?

- Occam's razor says that when presented with competing hypotheses that make the **same** predictions, one should select the solution which is simple“
- Arguments in favor
  - There are fewer short hypotheses than long ones
  - If a short hypothesis fits data unlikely to be a coincidence
  - Elegance and aesthetics
- Arguments against
  - Not every short hypothesis is a reasonable one.

Prefer shorter hypotheses: Occam's razor

Why?

Idea: The simplest consistent explanation is the best

- Therefore, the smallest decision tree that correctly classifies all of the training examples is best
  - Finding the provably smallest decision tree is NP-hard
  - ...So instead of constructing the absolute smallest tree consistent with the training examples, construct one that is pretty small

Avoid overfitting in Decision Trees

- Two strategies:
  - Stop growing the tree earlier, before perfect classification
    - Leaf nodes are impure
  - Allow the tree to *overfit* the data, and then *post-prune* the tree

- Training and validation set
  - split the training in two parts (training and validation) and use validation to assess the utility of *post-pruning*
    - Reduced error pruning
    - Rule post pruning

Reduced-error pruning

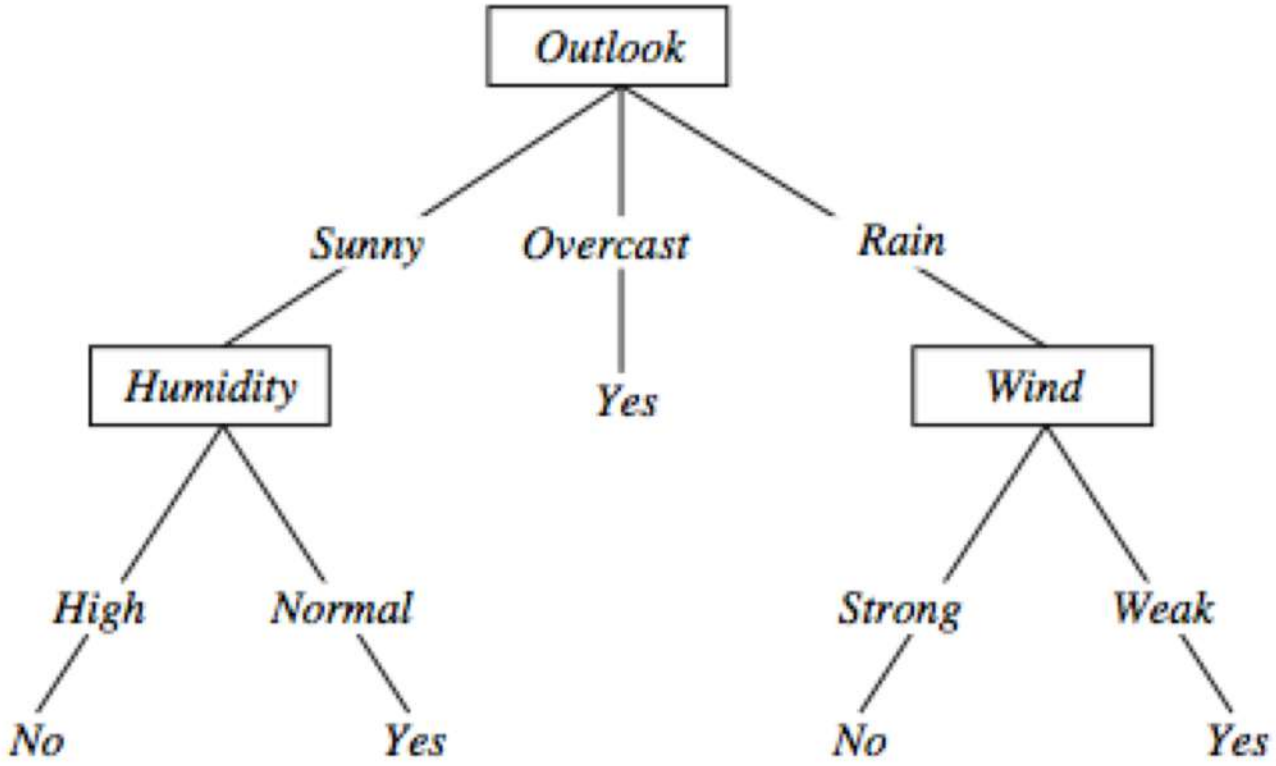
1. Partition training data in “train” and “validation” sets.
2. Build a complete tree from the “train” data
3. While pruning doesn’t harm:

   For each non leaf node, n in the tree do:

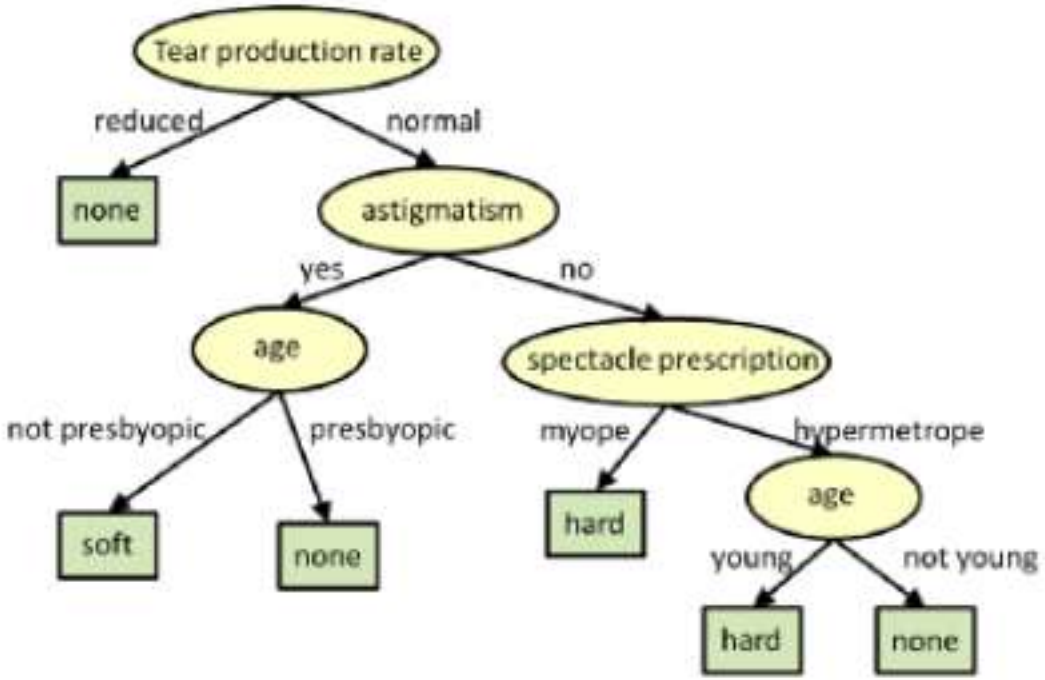
   - Temporarily prune the subtree below n and replace it with a leaf labelled with the current majority class at that node.
   - Measure and record the accuracy of the pruned tree on the validation set

4. Permantenly prune the node that in the greatest increase in the accuracy on the validation set.

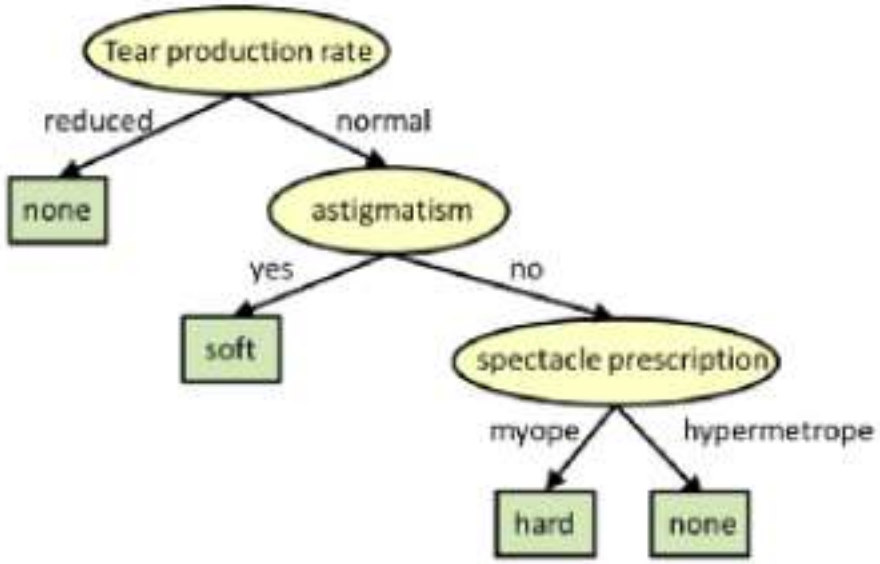
Reduced-error pruning



Reduced-error pruning



Original Tree



Pruned Tree

Reduced-error pruning

Drawbacks:

What happens if we have a small dataset?, so that we cannot have enough training set and validation set

Is there a solution?

-Rule post pruning appraoch

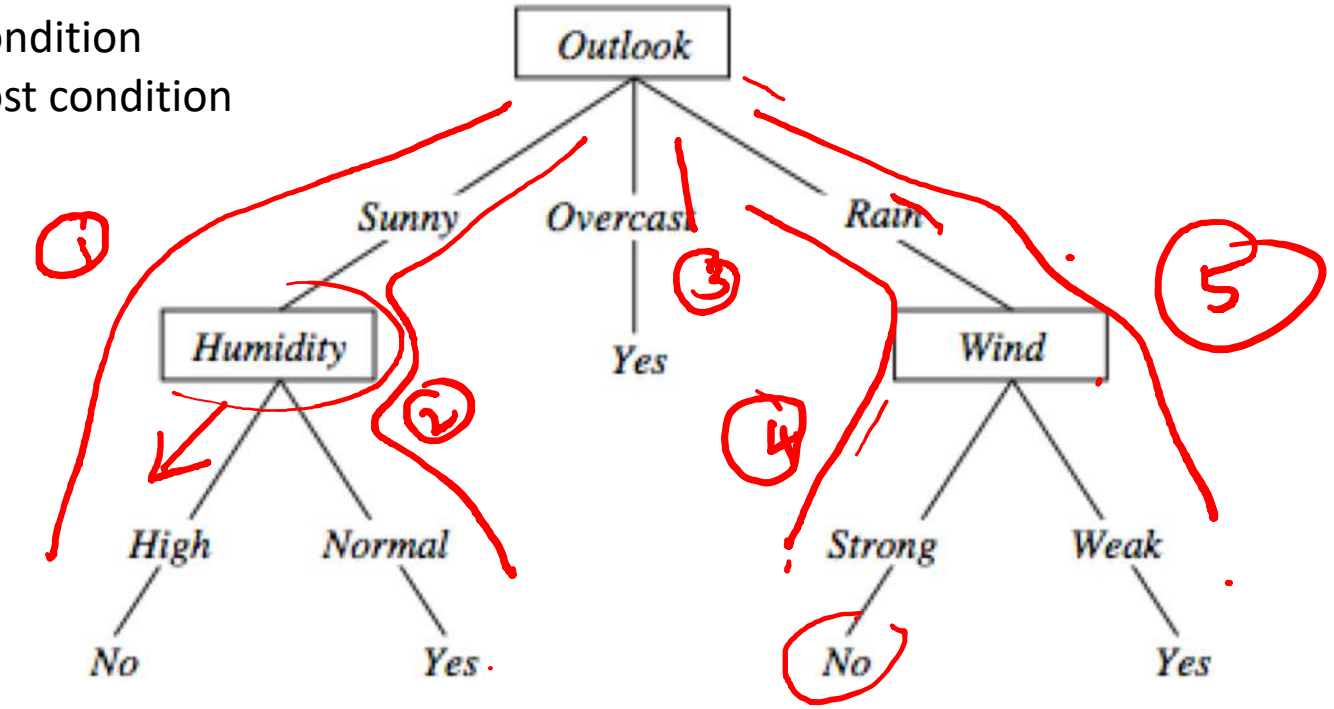
https://www.cloudyml.com/blog/decision-tree-pruning-techniques-in-python/

Rule post-pruning

- Create the decision tree from the training set
- Convert the tree into an equivalent set of rules
  - Each path corresponds to a rule
  - Each node along a path corresponds to a pre-condition
  - Each leaf classification to the post-condition
- Prune (generalize) each rule by removing those preconditions whose removal improves accuracy
  - over validation set
- Sort the rules in estimated order of accuracy, and consider them in sequence when classifying new instances

Converting to rules

5 paths, hence 5 rules
Each node gives precondition
Classification is the post condition

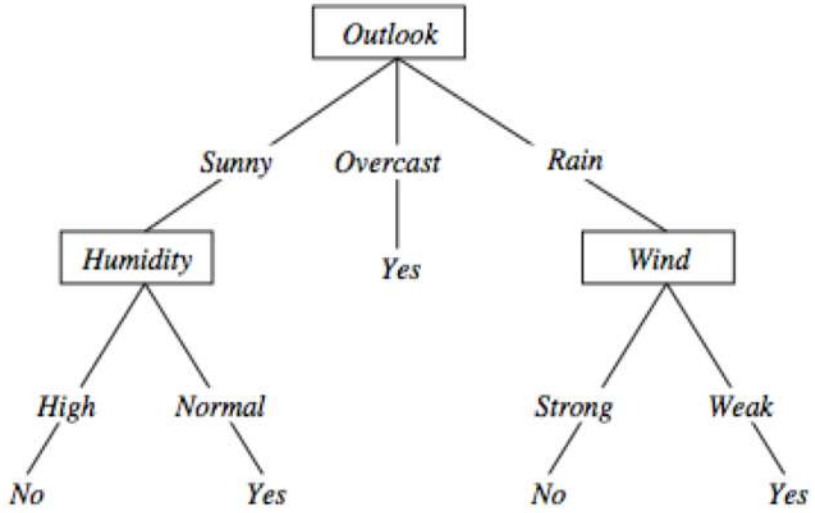


$$(Outlook=Sunny) \wedge (Humidity=High) \Rightarrow (PlayTennis=No)$$

(Outlook = Sunny) ∧ (Humidity = Normal) ⇒ (Playtennis = yes)
(Outlook = overcast) ⇒ (play tennis = yes)

Rule Post-Pruning

- Convert tree to rules (one for each path from root to a leaf)
- For each antecedent in a rule, remove it if error rate on validation set does not decrease
- Sort final rule set by accuracy




```
Outlook=sunny ^ humidity=high -> No
Outlook=sunny ^ humidity=normal -> Yes
Outlook=overcast -> Yes
Outlook=rain ^ wind=strong -> No
Outlook=rain ^ wind=weak -> Yes
```

- Compare first rule to:
  - `Outlook=sunny->No`
  - `Humidity=high->No`
- Calculate accuracy of 3 rules based on validation set and pick best version.

Why converting to rules?

- Each distinct path produces a different rule
  - a condition removal may be based on a local (contextual) criterion.
  - Node pruning is global and affects all the rules
- Provides flexibility of not removing entire node
- In rule form, tests are not ordered and there is no book-keeping involved when conditions (nodes) are removed
- Converting to rules improves readability for humans

Problems with information gain

- Natural bias of information gain: it favors attributes with many possible values.
- Consider the attribute *Date* in the *PlayTennis* example.
  - *Date* would have the highest information gain since it perfectly separates the training data.
  - It would be selected at the root resulting in a very broad tree
  - Very good on the training, this tree would perform poorly in predicting unknown instances.
    - Leading to overfitting.
- The problem is that the partition is too specific, too many small classes are generated.
- We need to look at alternative measures …

An alternative measure: Gain Ratio

- $$SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$
- S_i are the sets obtained by partitioning on value i of A
- *SplitInformation* measures the entropy of S with respect to the values of A.
- The more uniformly dispersed the data the higher it is.

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

- $GainRatio$ penalizes attributes that split examples in many small classes such as *Date.*
- Let $|S|=n$, $Date$ splits examples in n classes
- $SplitInformation(S, Date) = -[(1/n \log_2 1/n)+\ldots+ (1/n \log_2 1/n)] = -\log_2 1/n = \log_2 n$
- Compare with A, which splits data in two even classes:
  - $SplitInformation(S, A) = -[(1/2 \log_2 1/2)+ (1/2 \log_2 1/2)] = -[-1/2 -1/2] = 1$

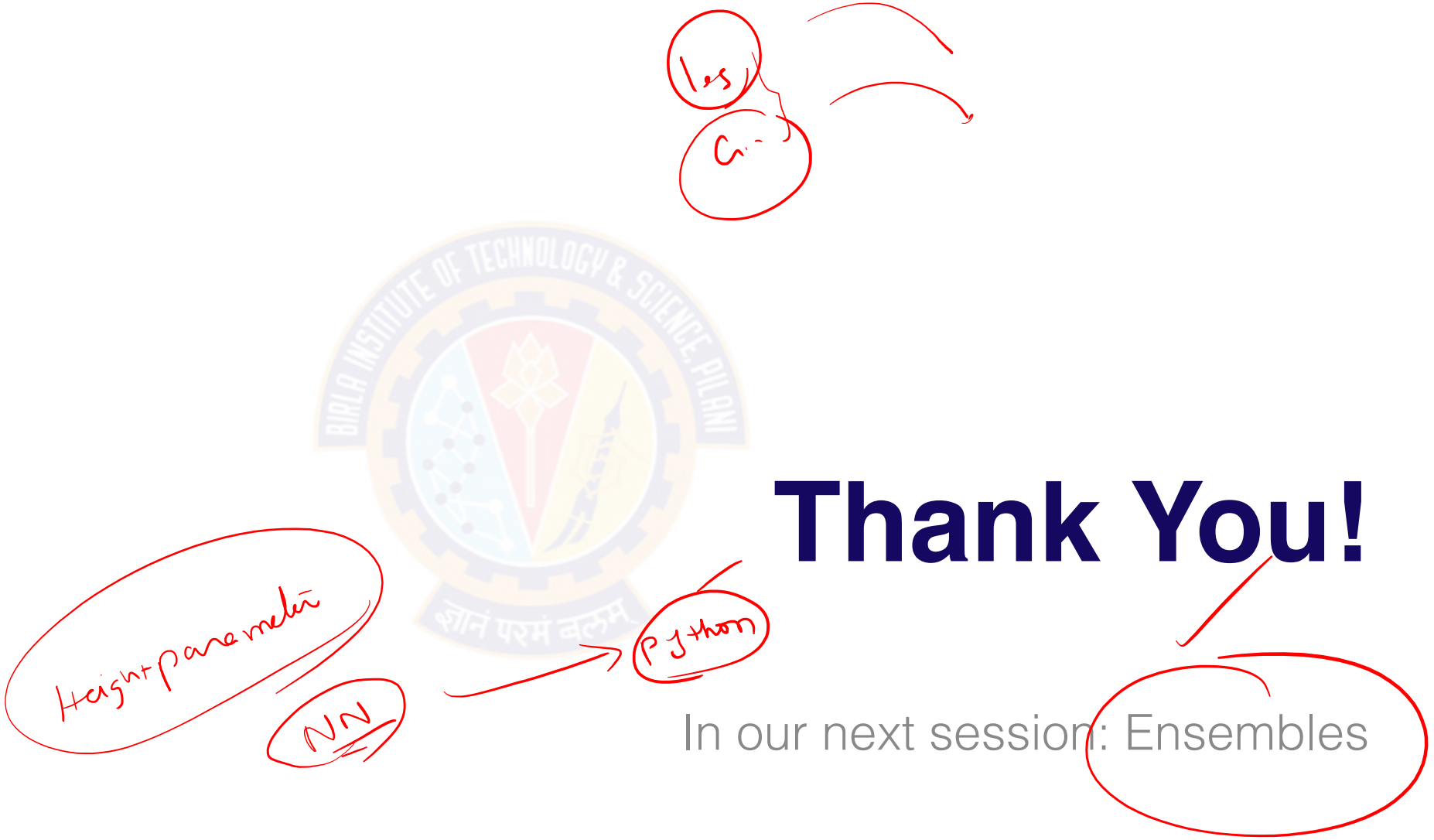
Dealing with continuous-valued attributes

- Given a continuous-valued attribute *A*, dynamically create a new attribute A_c
  - A_c = True *if* $A < c$, False *otherwise*
- How to determine threshold value *c* ?
- Example. *Temperature* in the *PlayTennis* example
  - Sort the examples according to *Temperature*
  - *Temperature* 40 48 | 60 72 80 | 90
  - *PlayTennis* No No ***54*** Yes Yes Yes ***85*** No
- Determine candidate thresholds by averaging consecutive values where there is a change in classification: (48+60)/2=54 and (80+90)/2=85

Applications

- Suited for following classification problems:
  - Applications whose Instances are represented by attribute-value pairs.
  - The target function has discrete output values
  - Disjunctive descriptions may be required
  - The training data may contain missing attribute values

- Real world applications
  - Biomedical applications
  - Manufacturing
  - Banking sector
  - Make-Buy decisions



Thank You!

In our next session: Ensembles