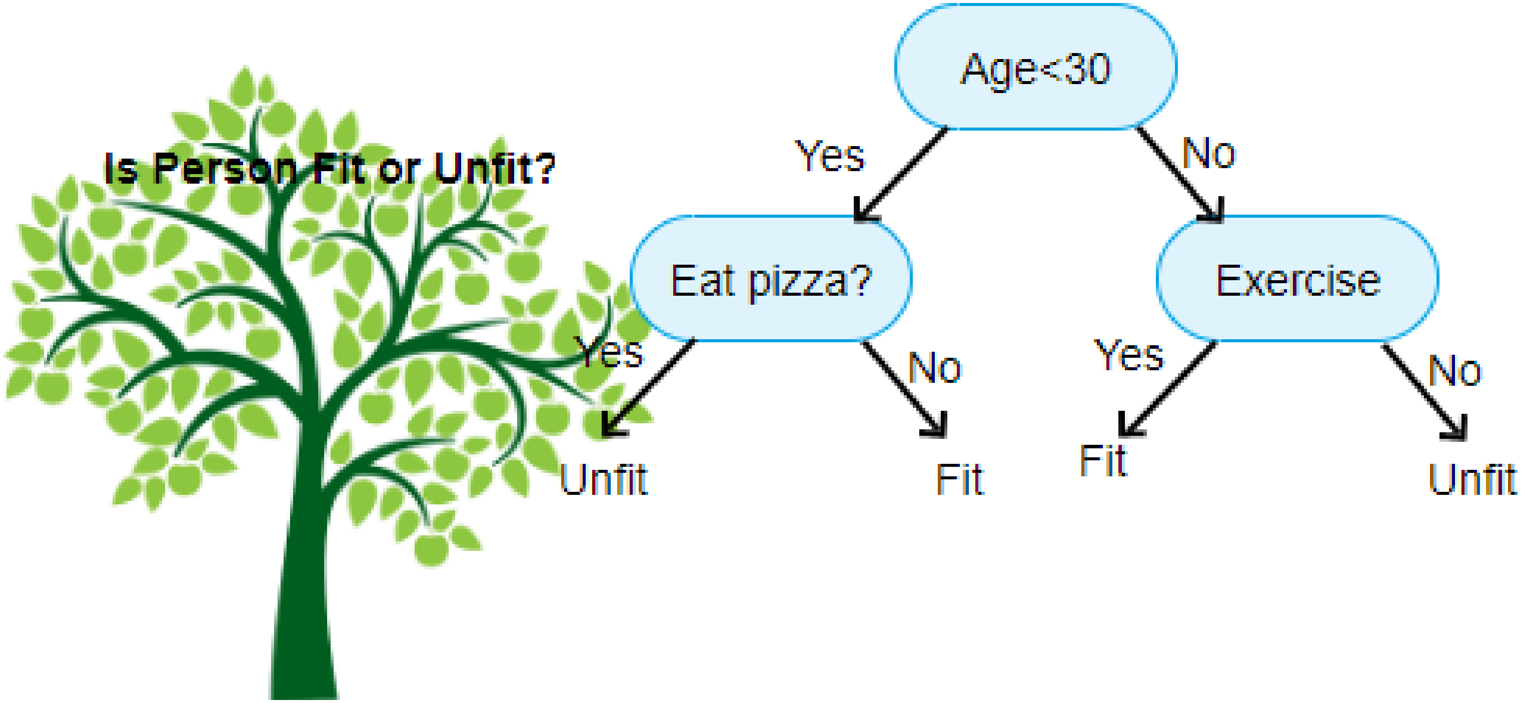
Decision Trees

Introduction

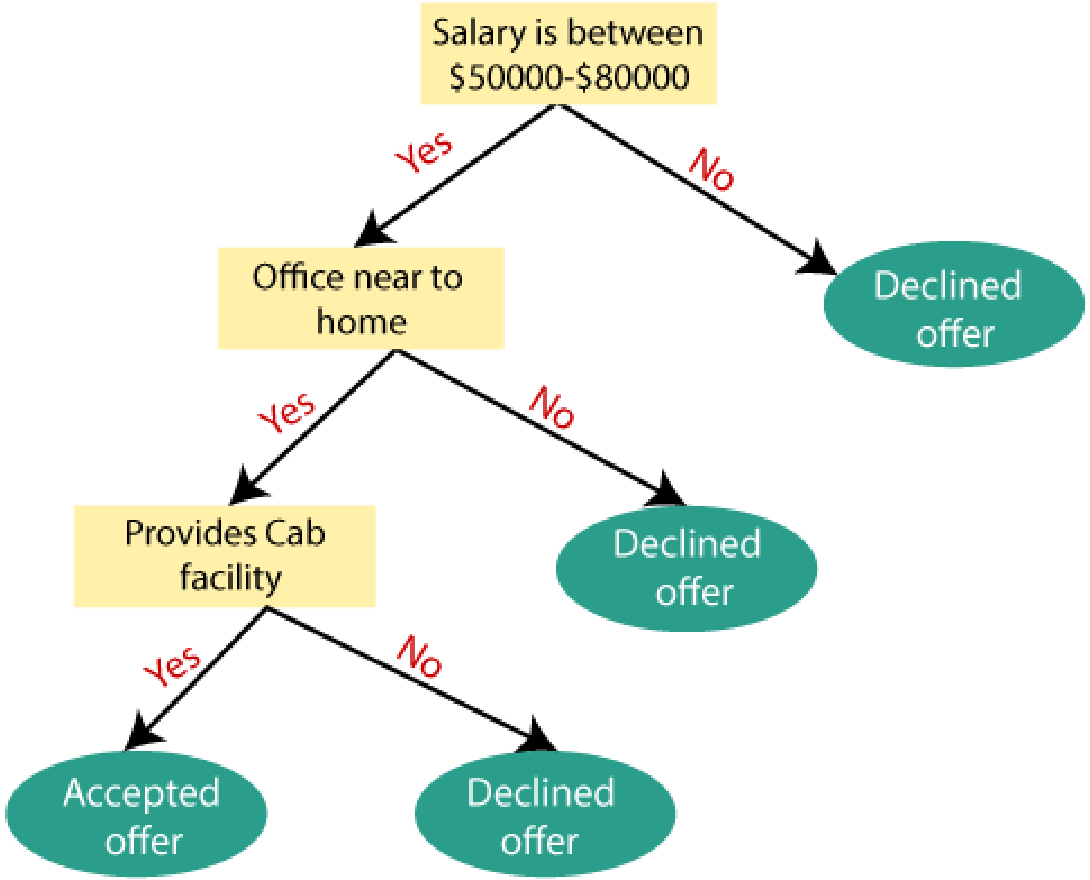
- Decision tree is one of the most widely used and practical methods for supervised learning.
- It can be used for both classification and regression tasks.
- The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.
- **The output is a set of rules**
- The decision rules are generally in form of **if-then-else** statements.
- It is a tree-like graph with
  - **Nodes** representing the place where we pick an attribute and ask a question;
  - **Edges** represent the answers the to the question;
  - **Leaves** represent the actual output or class label.
- The deeper the tree, the more complex the rules and fitter the model.

Overview


Is Person Fit or Unfit?
Age<30
Yes
No
Eat pizza?
Exercise
Yes
No
Yes
No
Unfit
Fit
Fit
Unfit

- Goal: Classify the response to a JOB offer as “Accept JOB offer” or “Reject JOB offer”, based on ‘Compensation Offered’ and ‘Location’
- Rule might be
    - “IF (**Offer** >= 10 Lakh) AND (**Location** = Bangalore)
        - THEN **Accept Offer**
    - “If (**Offer** < 10 Lakh) AND (**Location** = Bangalore)
        - THEN **Reject Offer**
  - …..
- Such rules are best captured by tree diagrams

Overview



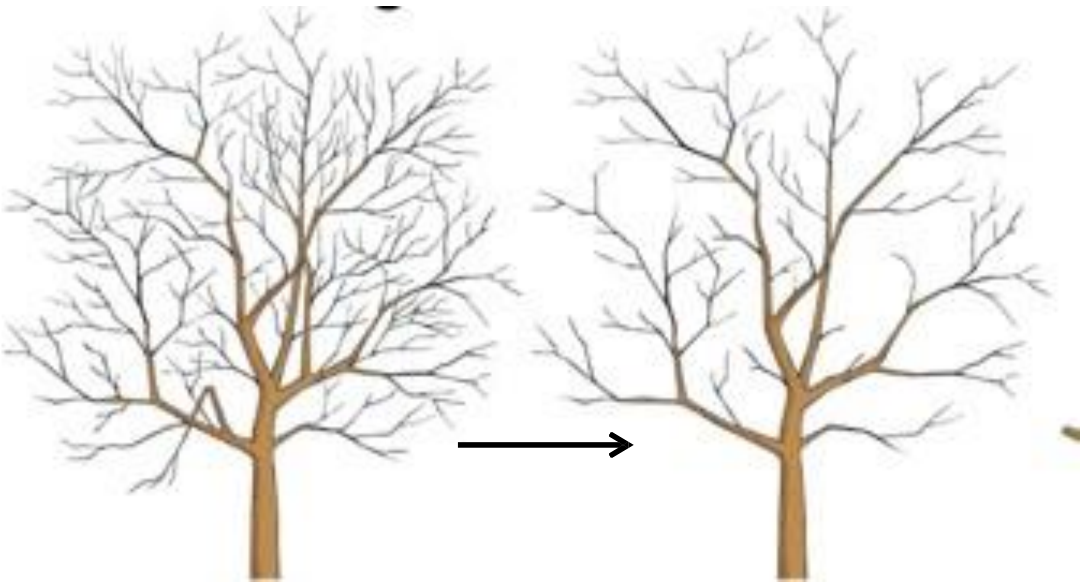
Salary is between $50000-$80000
Yes
No
Office near to home
Declined offer
Yes
No
Provides Cab facility
Declined offer
Yes
No
Accepted offer
Declined offer

Terminologies

- **Root Node**: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node**: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting**: Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree**: A tree formed by splitting the tree.
- **Pruning**: Pruning is the process of removing the unwanted branches from the tree which reduces the impact of overfitting.
- **Parent/Child node**: The root node of the tree is called the parent node, and other nodes are called the child nodes.

Pruning

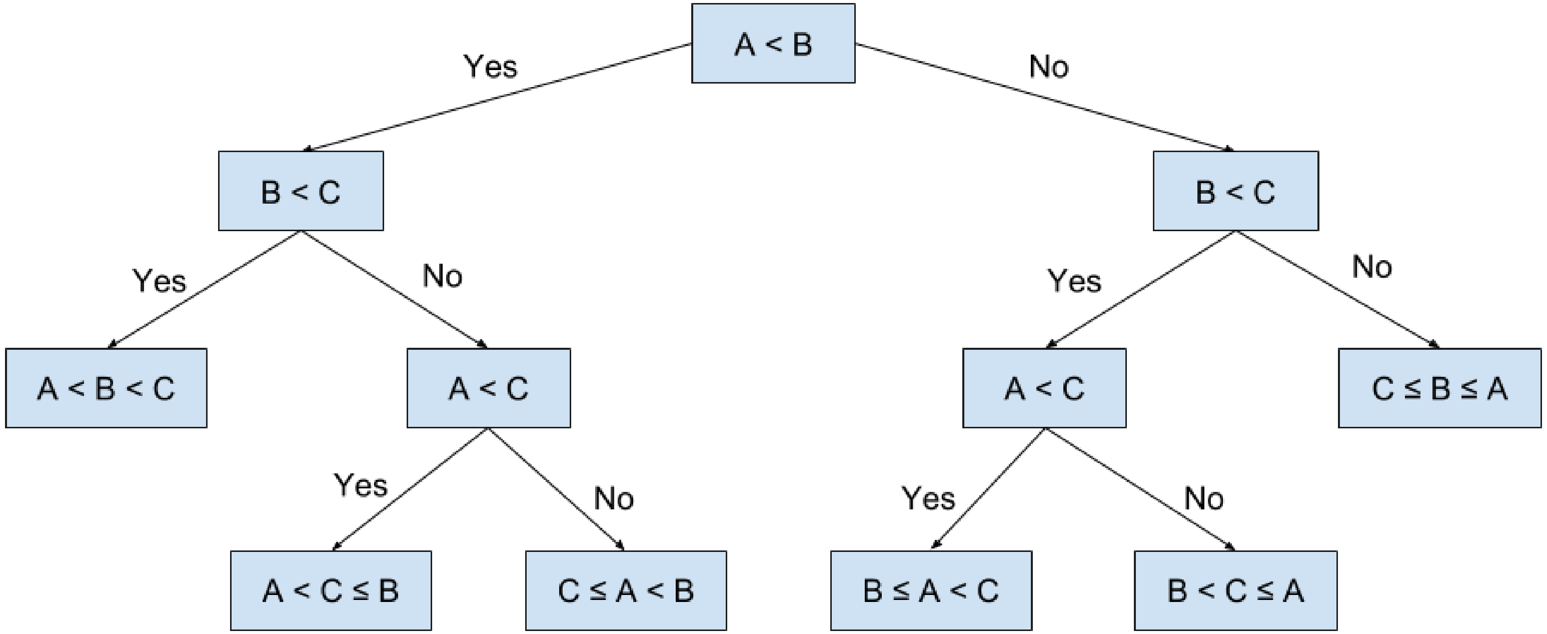
- Pruning reduces the impact of overfitting
- Allowing the tree to grow to the full extent, then prune it back
- Generate successively smaller trees by pruning leaves
- At each pruning stage, multiple trees are possible
- Use *cost complexity* to choose the best tree at that stage



Divide and conquer

- Decision trees classify the examples by sorting them down the tree from the root to some leaf node, with the leaf node providing the classification to the example.
- Each node in the tree acts as a test case for some attribute, and each edge descending from that node corresponds to one of the possible answers to the test case.
- This process is recursive in nature and is repeated for every subtree rooted at the new nodes.
- **Recursive partitioning:** Repeatedly split the records into two parts so as to achieve maximum homogeneity of outcome within each new part

Programming view point



Real life example

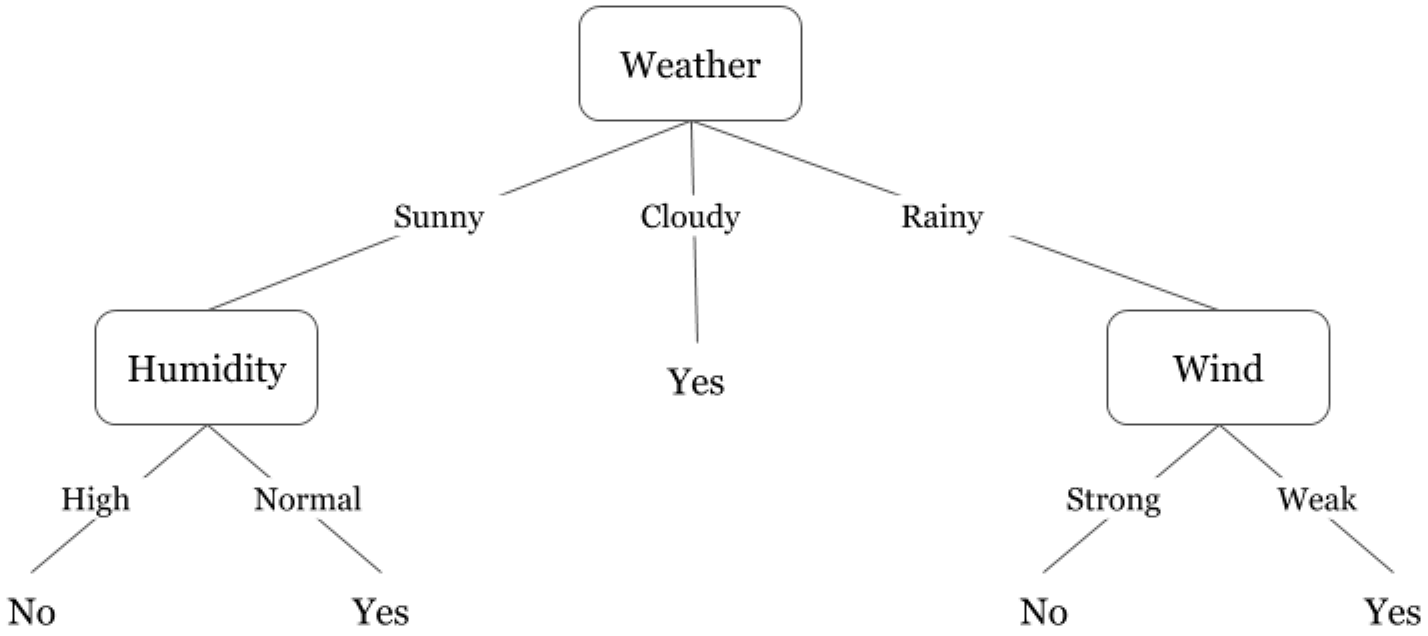
Assume that you want to play badminton on a particular day. How will you decide whether to play or not?

Factors influencing the decision – Weather, temperature, humidity, wind.

Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Cloudy	Hot	High	Weak	Yes
3	Sunny	Mild	Normal	Strong	Yes
4	Cloudy	Mild	High	Strong	Yes
5	Rainy	Mild	High	Strong	No
6	Rainy	Cool	Normal	Strong	No
7	Rainy	Mild	High	Weak	Yes
8	Sunny	Hot	High	Strong	No
9	Cloudy	Hot	Normal	Weak	Yes
10	Rainy	Mild	High	Strong	No

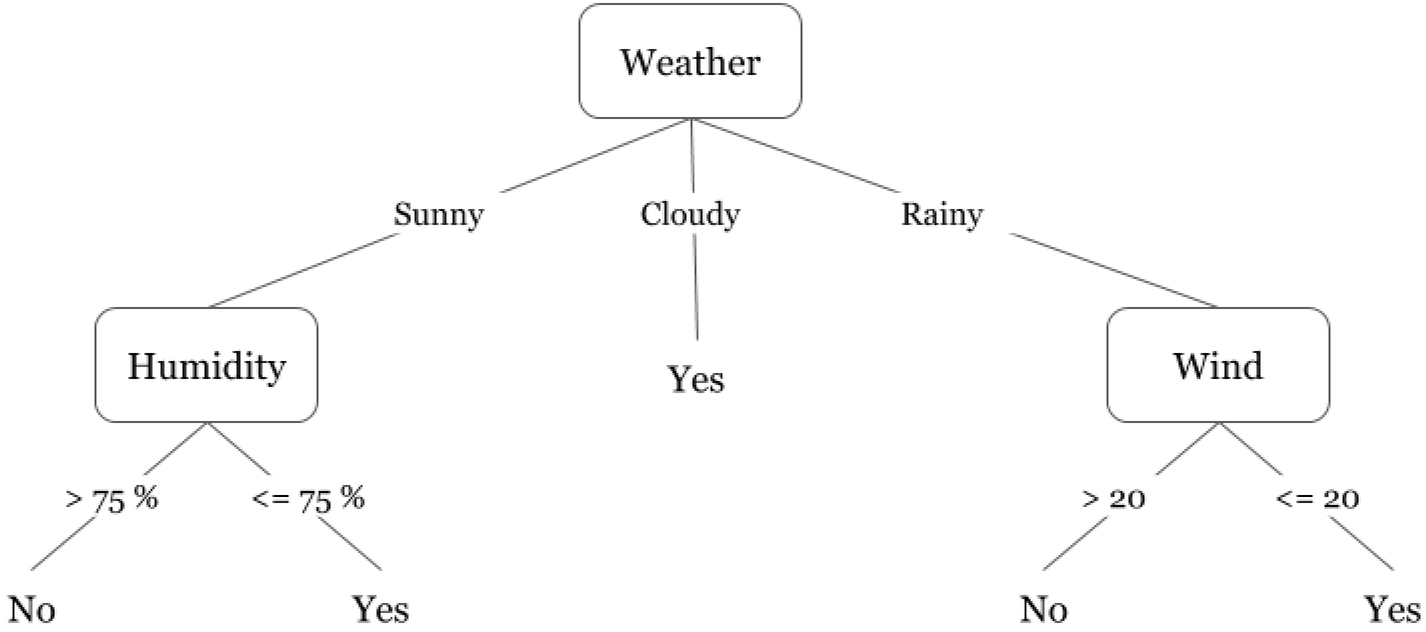
Real life example

Now, you may use this table to decide whether to play or not.



Real life example

We can see that each node represents an attribute or feature and the branch from each node represents the outcome of that node. Finally, its the leaves of the tree where the final decision is made.



Algorithm

A general algorithm for a decision tree can be described as follows:

Steps

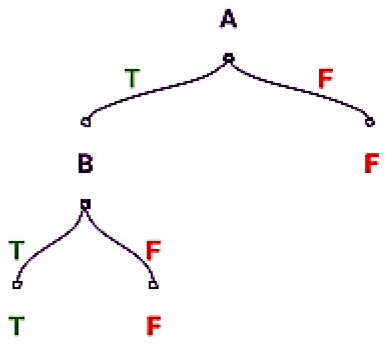
1. Pick the best attribute/feature. The best attribute is one which best splits or separates the data.
2. Ask the relevant question.
3. Follow the answer path.
4. Go to step 1 until you arrive to the answer.

The best split is one which separates two different labels into two sets.

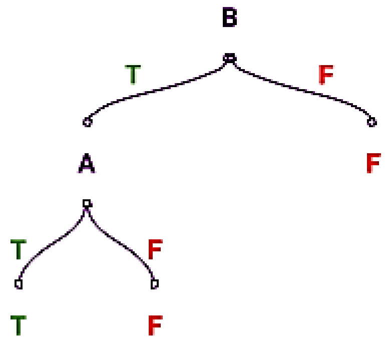
Expressiveness of decision trees

Let’s use decision trees to perform the function of three boolean gates AND, OR and XOR.

A	B	A AND B
F	F	F
F	T	F
T	F	F
T	T	T

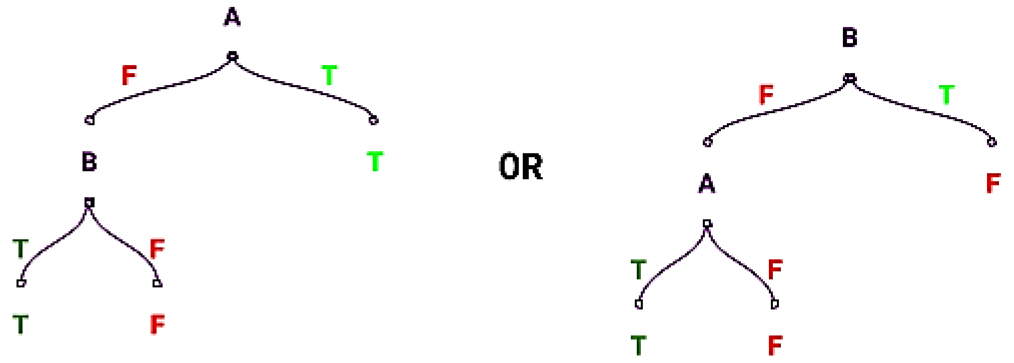


OR

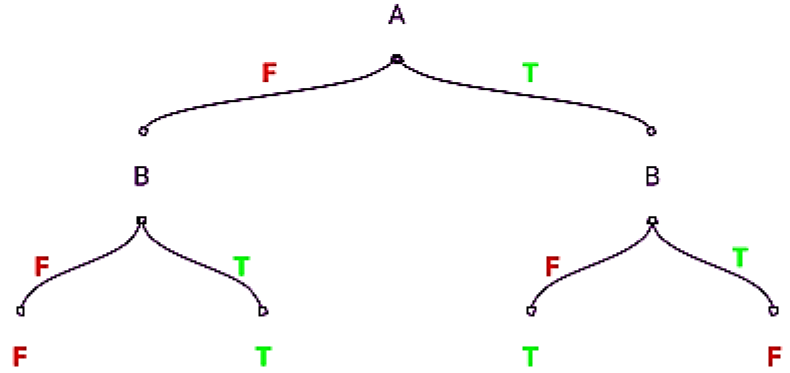


We can see that there are two candidate concepts for producing the decision tree that performs the AND operation.

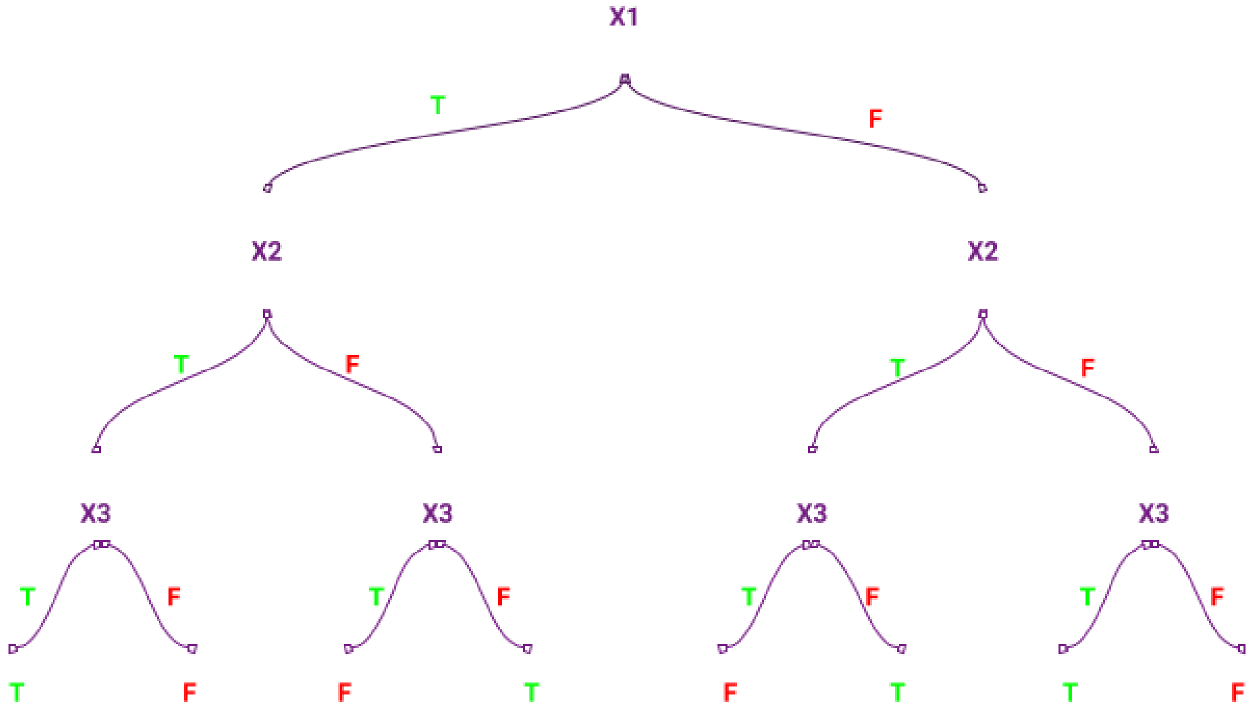
A	B	A OR B
F	F	F
F	T	T
T	F	T
T	T	T

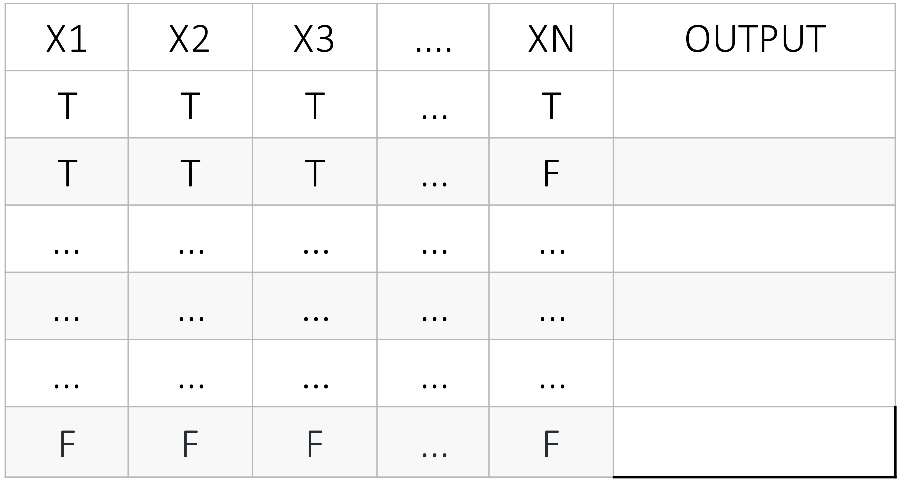


A	B	A XOR B
F	F	F
F	T	T
T	F	T
T	T	F



Let’s produce a decision tree performing XOR functionality using 3 attributes:



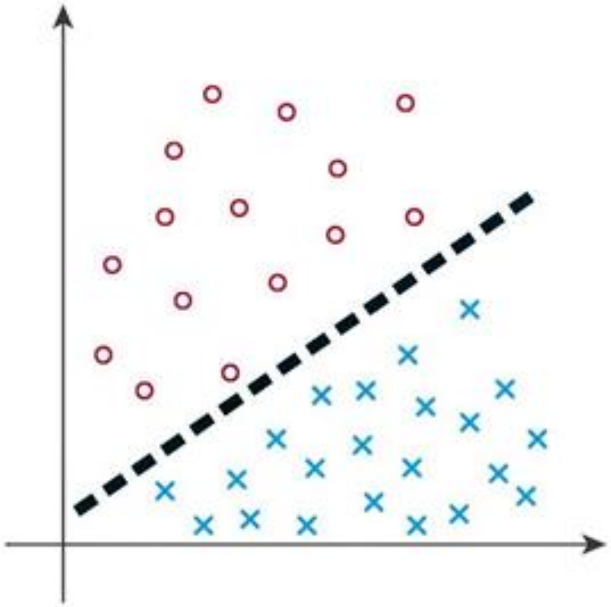


X1	X2	X3	....	XN	OUTPUT
T	T	T	...	T	
T	T	T	...	F	
...	...	...	...	...	
...	...	...	...	...	
...	...	...	...	...	
F	F	F	...	F	

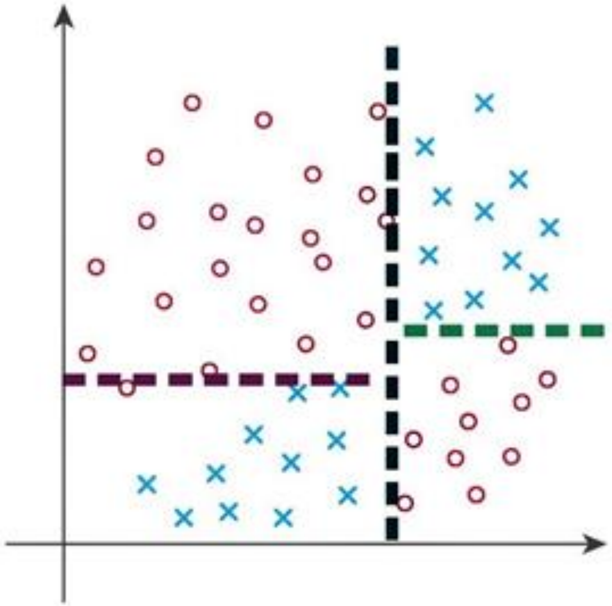
The above truth table has 2^n rows (i.e. the number of nodes in the decision tree), which represents the possible combinations of the input attributes, and since each node can a hold a binary value, the number of ways to fill the values in the decision tree is 2^{2^n}.

Thus, the space of decision trees, i.e, the hypothesis space of the decision tree is very expressive because there are a lot of different functions it can represent. But, it also means one needs to have a clever way to search the best tree among them.

Separability

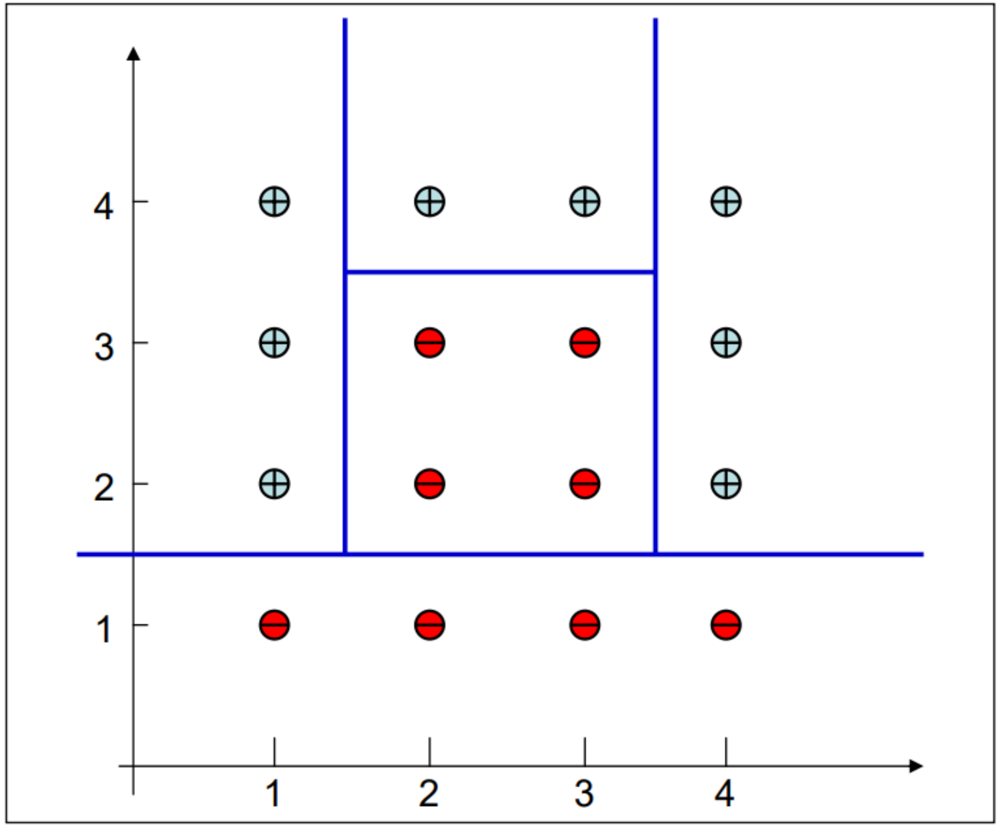


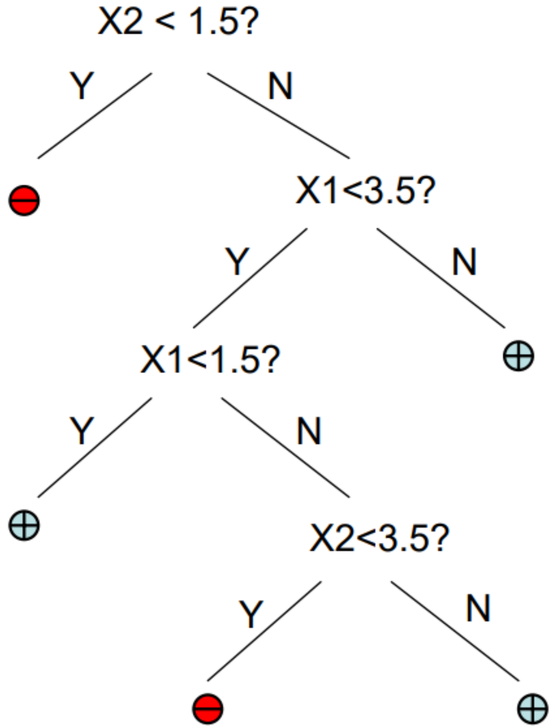
Linearly separable dataset



Linearly inseparable dataset

Decision tree boundary





Learning algorithm

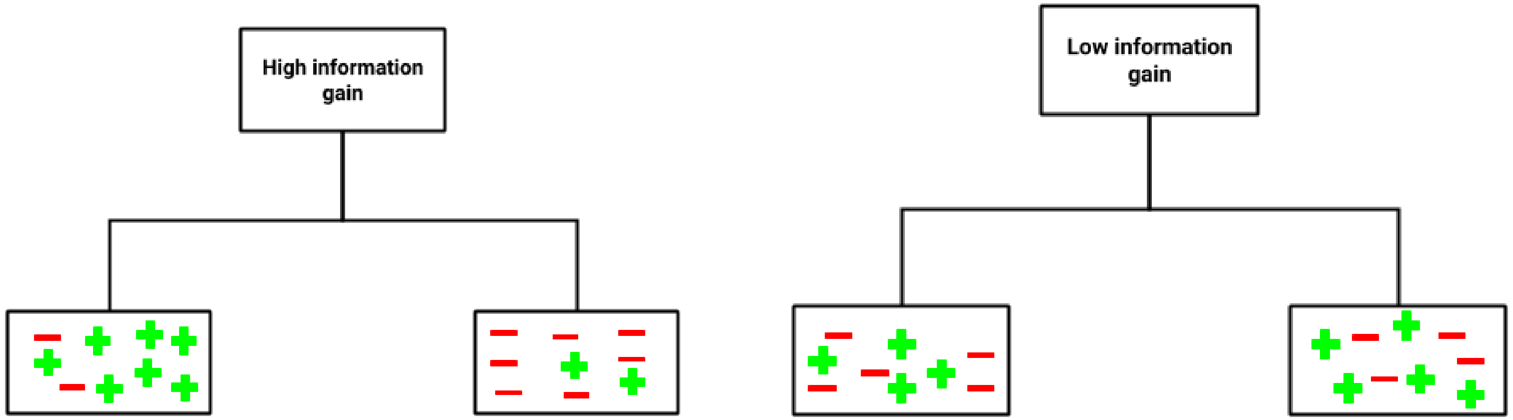
- The basic algorithm used in decision trees is known as the ID3 algorithm.
- The ID3 algorithm builds decision trees using a top-down, greedy approach.

Steps

1. Select the best attribute → A
2. Assign A as the decision attribute (test case) for the **NODE**.
3. For each value of A, create a new descendant of the **NODE**.
4. Sort the training examples to the appropriate descendant node leaf.
5. If examples are perfectly classified, then STOP else iterate over the new leaf nodes.

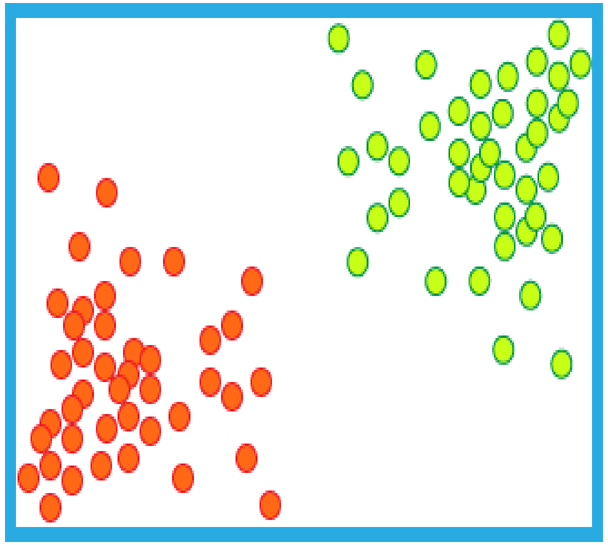
Selecting the best attribute

- The best attribute is the one with the highest information gain.
- **Information gain:** A measure that expresses how well an attribute splits that data into groups based on classification.
- An attribute with low information gain splits the data relatively evenly and as a result doesn’t bring us any closer to a decision.
- Whereas, an attribute with high information gain splits the data into groups with an uneven number of positives and negatives.

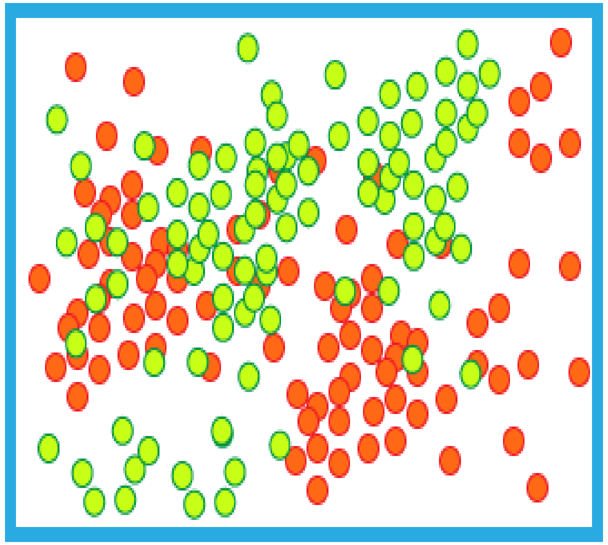


Information Gain

- To define information gain precisely, we need to define a measure commonly used in information theory called *entropy* that measures the level of *impurity/disorder* in a group of examples.



Low Entropy



High Entropy

- Mathematically, entropy is defined as:

$$H[X] = -\sum_{i=1}^{n} P(x_i) \log P(x_i)$$

- Since, the basic version of the ID3 algorithm deal with the case where classification are either positive or negative, we can define entropy as :

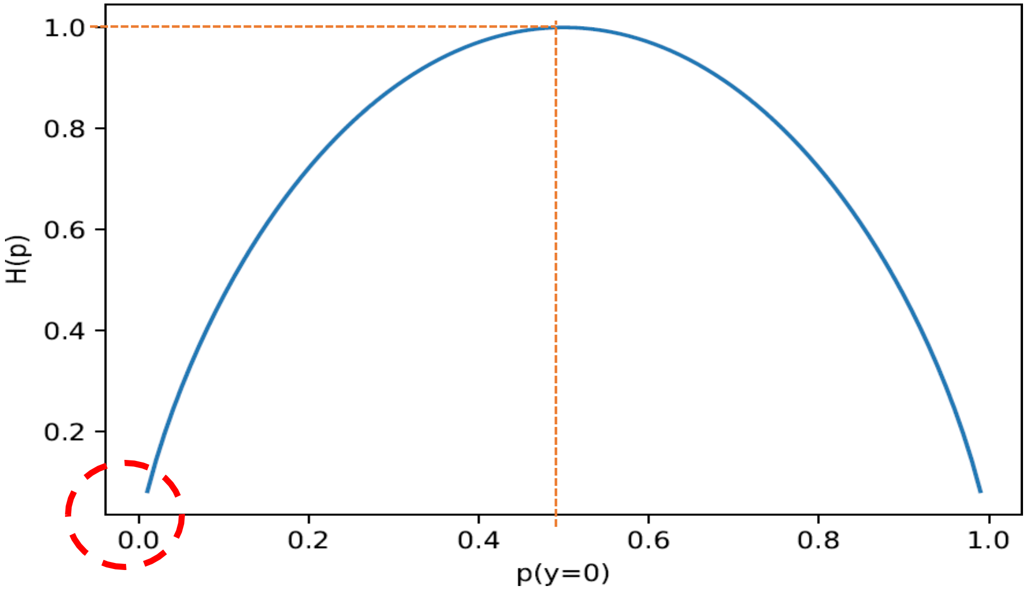
where,

$$Entropy(S) = -p_+ log_2 p_+ - p_- log_2 p_-$$

S is a sample of training examples

p_+ is the proportion of positive examples in S

p_- is the proportion of negative examples in S



- Entropy ranges between 0 (most pure) and $\log_2(m)$ (equal representation of classes)
- Implying for a 2 class scenario (m=2) with equal representation, entropy would be $\log_2(2) = 1$

Example

- Suppose *S* is a sample containing 14 boolean examples, with 9 positive (+) and 5 negative (-) examples.
- Then, the entropy of *S* relative to this boolean classification is:

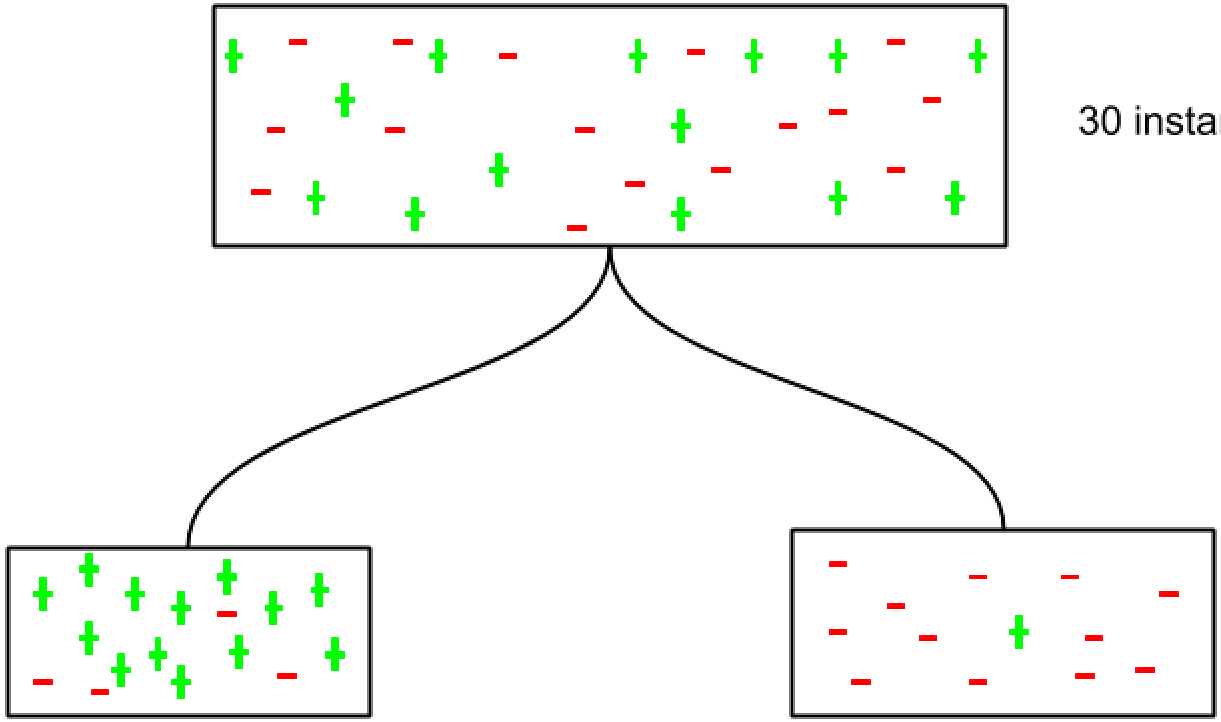
$$\text{Entropy (S)} = -(9/14) * \log_2(9/14) - (5/14) * \log_2(5/14) = 0.283$$

- Note that entropy is 0 if all the members of S belong to the same class.
- For example, if all members are positive ($p_+=1$), then p_- is 0, and $\text{Entropy(S)} = -1 * \log_2(1) -0 * \log_2(0) = 0$.
- Entropy is 1 when the sample contains an equal number of positive and negative examples.
- If the sample contains unequal number of positive and negative examples, entropy is between 0 and 1.

- Now, given entropy as a measure of the impurity in a sample of training examples, we can now define *information gain* as a measure of the effectiveness of an attribute in classifying the training data.
- Information gain, *Gain (S, A)* of an attribute A, relative to a sample of examples *S*, is defined as:

$$Gain\ (S, A) = Entropy(parent) - AverageEntropy(children)$$

- Suppose a sample (S) has 30 instances (14 positive and 16 negative labels) and an attribute A divides the samples into two subsamples of 17 instances (4 negative and 13 positive labels) and 13 instances (1 positive and 12 negative labels)



30 instances

17 instances

13 instances

Let’s calculate the information gain of the attribute *A*.

Entropy (parent) = -14/30 * log214/30 – 16/30 * log216/30 = 0.996

Entropy (child with 17 instances) = -13/17 * $\log_2$13/17 – 4/17 * $\log_2$4/17
= 0.787

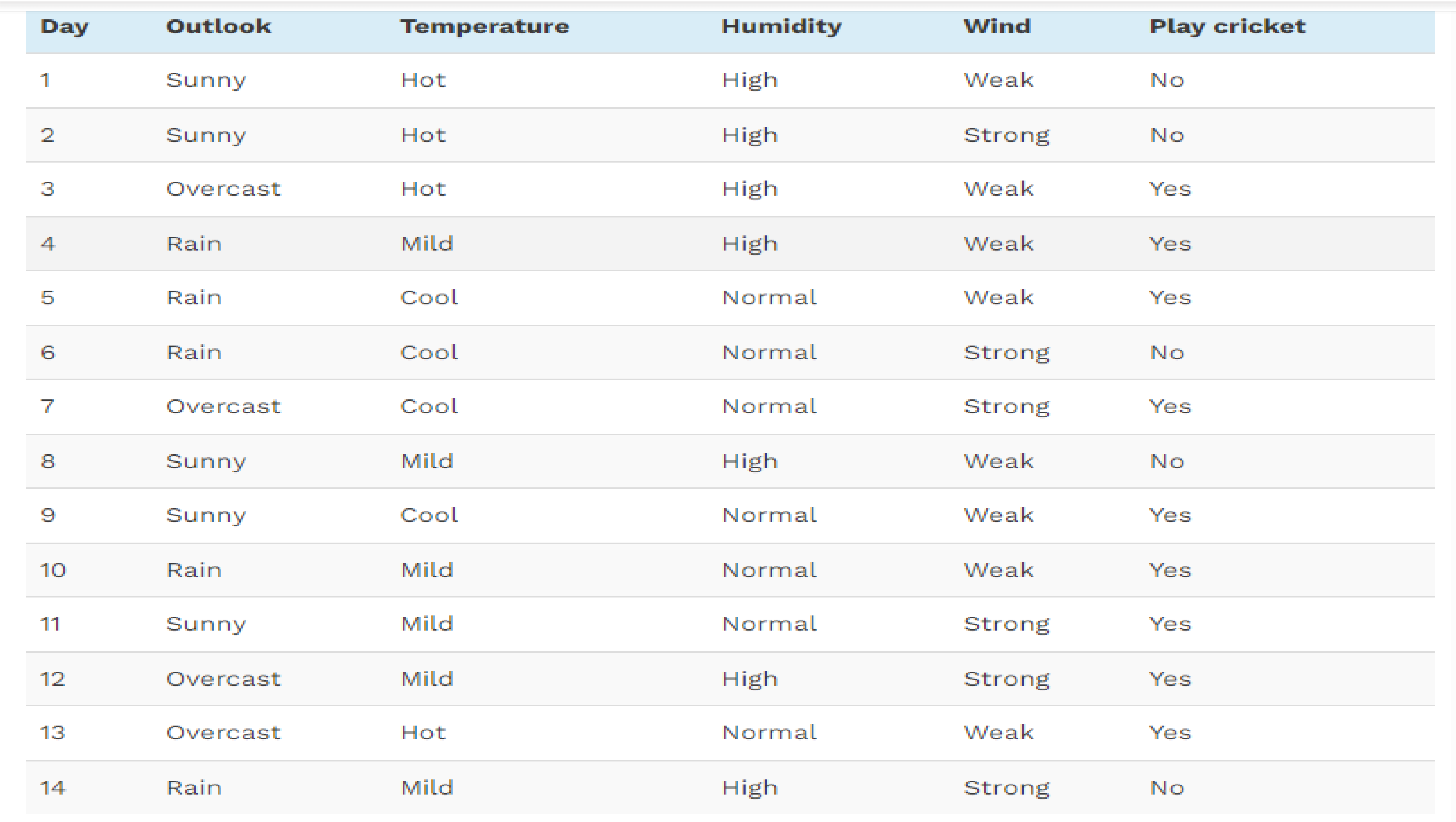
Entropy (child with 13 instances) = -1/13 * $\log_2$1/13 – 12/13 * $\log_2$12/13
= 0.391

Average Entropy of children = 17/30 * 0.787 + 13/30 * 0.391
= 0.615

Information Gain = G(S, A) = 0.996 - 0.615 = 0.38

Splitting Criteria

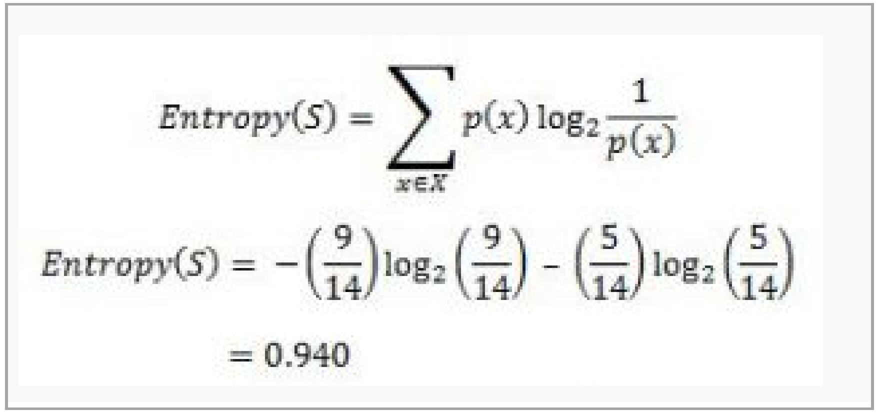
- Obtain overall impurity measure (weighted avg. of individual rectangles)
  - Entropy
  - Gini Index
  - Misclassification Error
- At each successive stage, compare this measure across all possible splits in all variables
- Choose the split that reduces impurity the most
- Chosen split points become nodes on the tree



Day	Outlook	Temperature	Humidity	Wind	Play cricket
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Calculation of Entropy:

Yes	No
9	5



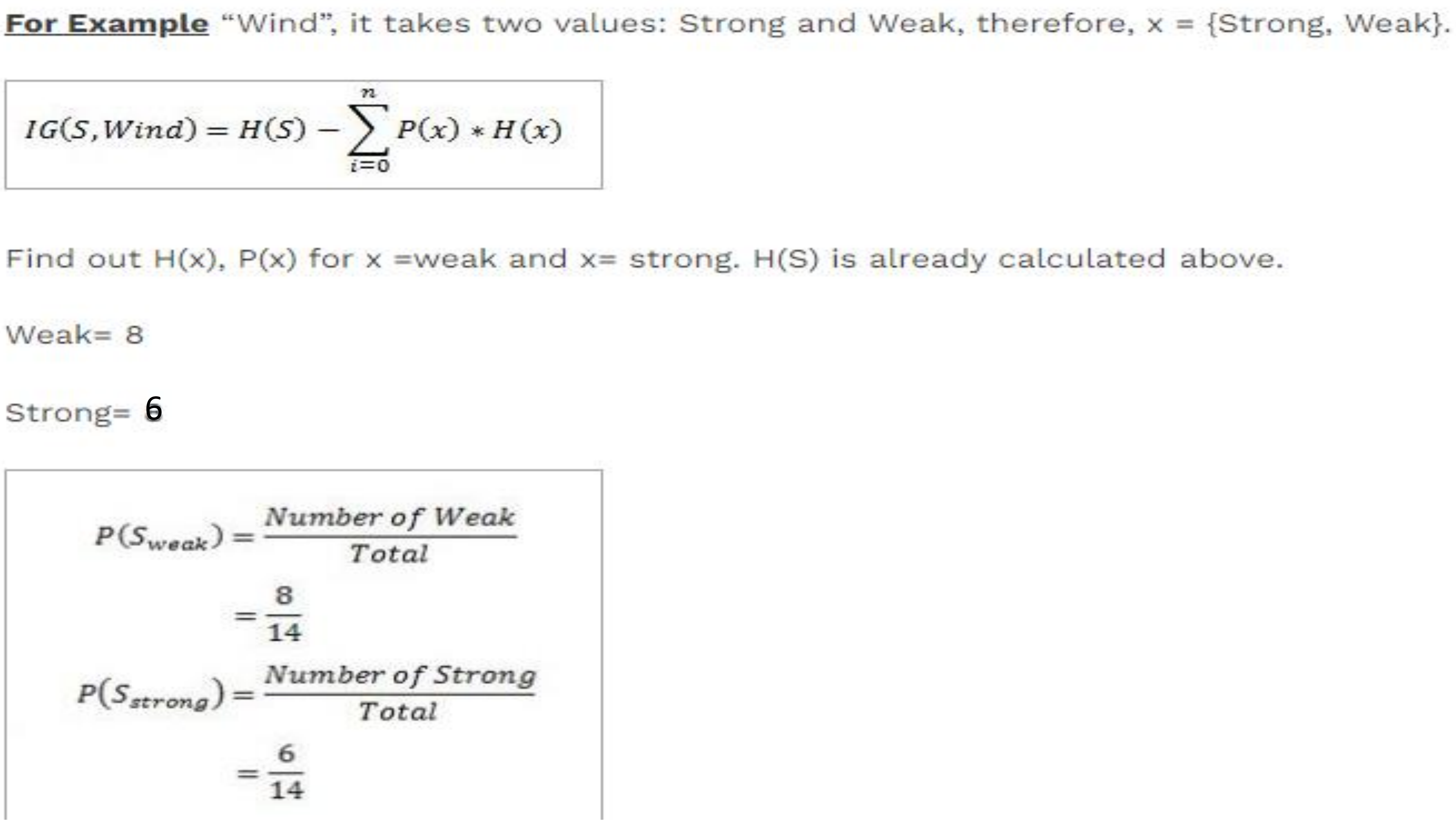
$$Entropy(S) = \sum_{x \in X} p(x) \log_2 \frac{1}{p(x)}$$

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

$$= 0.940$$

If entropy is zero, it means that all members belong to the same class and if entropy is one then it means that half of the tuples belong to one class and one of them belong to other class. 0.94 means fair distribution.

Find the information gain attribute which gives maximum information gain.



For Example “Wind”, it takes two values: Strong and Weak, therefore, x = {Strong, Weak}.

$$IG(S, Wind) = H(S) - \sum_{i=0}^{n} P(x) * H(x)$$

Find out H(x), P(x) for x =weak and x= strong. H(S) is already calculated above.

Weak= 8

Strong= 6

$$P(S_{weak}) = \frac{Number\ of\ Weak}{Total}$$
$$= \frac{8}{14}$$
$$P(S_{strong}) = \frac{Number\ of\ Strong}{Total}$$
$$= \frac{6}{14}$$

For “weak” wind, 6 of them say “Yes” to play cricket and 2 of them say “No”. So entropy will be:

$$
\begin{aligned}
Entropy(S_{weak}) &= -\left(\frac{6}{8}\right)\log_2\left(\frac{6}{8}\right) - \left(\frac{2}{8}\right)\log_2\left(\frac{2}{8}\right) \\
&= 0.811
\end{aligned}
$$

For “strong” wind, 3 said “No” to play cricket and 3 said “Yes”.

$$
\begin{aligned}
Entropy(S_{strong}) &= -\left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right) - \left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right) \\
&= 1.000
\end{aligned}
$$

This shows perfect randomness as half items belong to one class and the remaining half belong to others.

Calculate the information gain,

$$IG(S,Wind) = H(S) - \sum_{i=0}^{n} P(x) * H(x)$$

$$
\begin{aligned}
IG(S,Wind) &= H(S) - P(S_{weak}) * H(S_{weak}) - P(S_{strong}) * H(S_{strong}) \\
&= 0.940 - \left(\frac{8}{14}\right)(0.811) - \left(\frac{6}{14}\right)(1.00) \\
&= 0.048
\end{aligned}
$$

Information gain of outlook

- $IG(s, outlook) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$
- X= {sunny=5,overcast=4,rain=5}
- $P_{sunny} \quad = \frac{5}{14}$
- $P_{overcast} = \frac{4}{14}$
- $P_{rain} \quad = \frac{5}{14}$
- For sunny , 2 yes and 3 no.
- For overcast, 4 yes and 0 no.
- For rain, 3 yes and 2 no.
- $Entropy(sunny) = -\frac{2}{5} log_2(\frac{2}{5})$ - $\frac{3}{5} log_2(\frac{3}{5})$=0.97
- Entropy(overcast)= $-\frac{4}{4} log_2(\frac{4}{4})$=0
- Entropy(rain)= $-\frac{3}{5} log_2(\frac{3}{5})$ - $\frac{2}{5} log_2(\frac{2}{5})$=.97

- $IG(s, outlook) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$

  $= 0.94 - (\frac{5}{14} * 0.97) - (\frac{4}{14} * 0) - (\frac{5}{14} * 0.97)$

  $= 0.247$

Information gain of Temperature

- $IG(s, temperature) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$
- X= {Hot=4,Mild=6,cool=4}
- $P_{Hot} \quad = \frac{4}{14}$
- $P_{Mild} = \frac{6}{14}$
- $P_{cool} \quad = \frac{4}{14}$
- For Hot , 2 yes and 2 no.
- For Mild, 4 yes and 2 no.
- For Cool, 3 yes and 1 no.
- $Entropy(Hot) = -\frac{2}{4} log_2(\frac{2}{4})$ - $\frac{2}{4} log_2(\frac{2}{4})$=1
- Entropy(mild)= $-\frac{4}{6} log_2(\frac{4}{6}) - \frac{2}{6} log_2(\frac{2}{6})$=0.91
- Entropy(cool)= $-\frac{3}{4} log_2(\frac{3}{4})$ - $\frac{1}{4} log_2(\frac{1}{4})$=.81

- $IG(s, temperature) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$

$$= 0.94 - \left(\frac{4}{14} * 1\right) - \left(\frac{6}{14} * 0.91\right) - \left(\frac{4}{14} * 0.81\right)$$

$$= 0.047$$

Information gain of Humidity

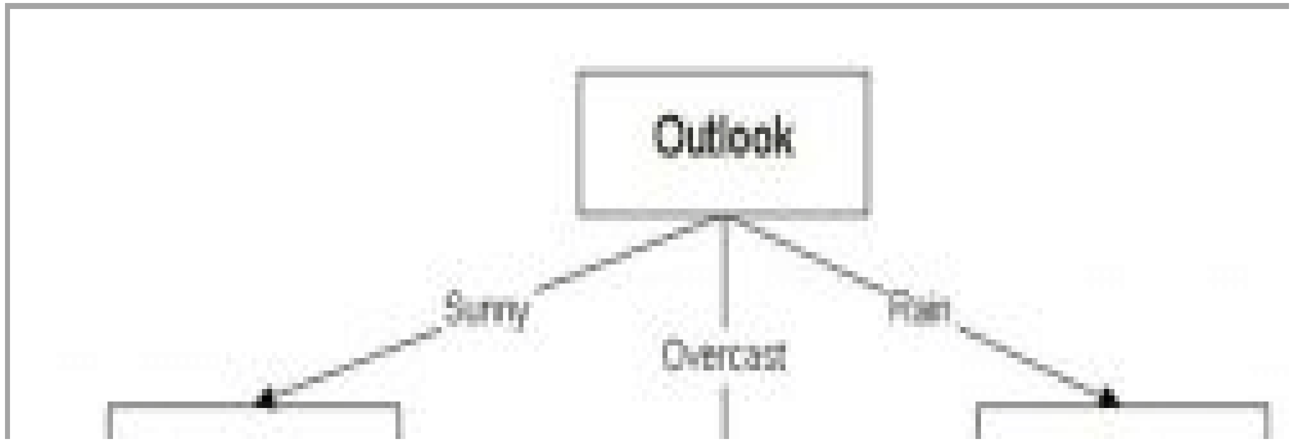
- $IG(s, Humidity) = \mathrm{H}(s) - \sum_{i=0}^{n} p(x) * H(x)$
- X= {High=7,normal=7}
- $P_{High} \quad = \frac{7}{14}$
- $P_{normal} = \frac{7}{14}$
- For High , 3 yes and 4 no.
- For Normal, 5 yes and 2 no
- $Entropy(High) = -\frac{3}{7} log_2(\frac{3}{7})$ - $\frac{4}{7} log_2(\frac{4}{7})$=0.98
- Entropy(normal)= $-\frac{5}{7} log_2(\frac{5}{7}) - \frac{2}{7} log_2(\frac{2}{7}$)=0.86

- $IG(s, Humidity) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$

  $= 0.94 - (\frac{7}{14} * .98) - (\frac{7}{14} * 0.86)$

  = 0.02

- The Attribute outlook has the highest information gain , so we choose outlook as root node.



- Outlook has three attributes. Sunny, Overcast and Rain.
- Overcast with play cricket is always “Yes”. So it ends up with a leaf node, “yes”.
- For the other values “Sunny” and “Rain”,we need to find the entropy and information gain.

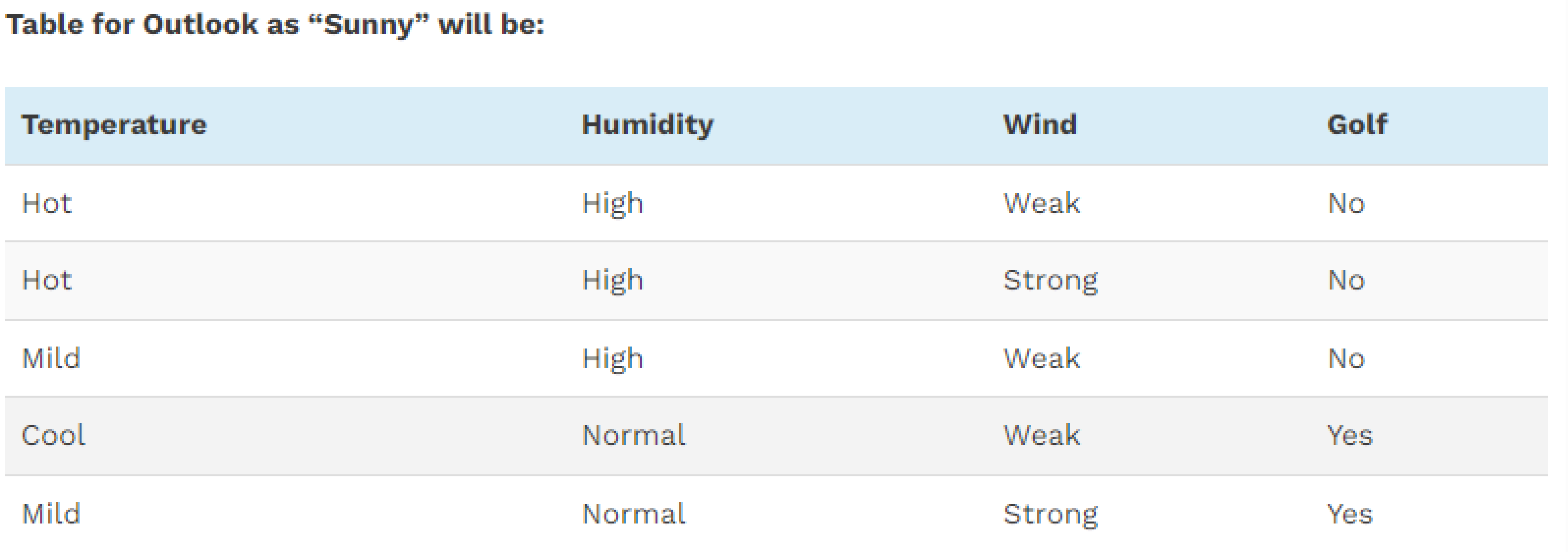
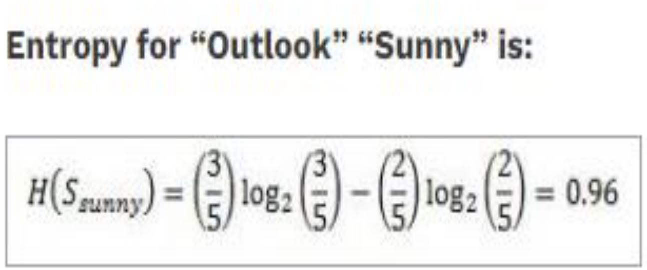


Table for Outlook as "Sunny" will be:

Temperature	Humidity	Wind	Golf
Hot	High	Weak	No
Hot	High	Strong	No
Mild	High	Weak	No
Cool	Normal	Weak	Yes
Mild	Normal	Strong	Yes



Entropy for "Outlook" "Sunny" is:

$$H(S_{sunny}) = \left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) = 0.96$$

Information gain of Temperature

- $IG(sunny, temperature) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$
- X= {Hot=2,Mild=2,cool=1}
- $P_{Hot} \quad = \frac{2}{5}$
- $P_{Mild} = \frac{2}{5}$
- $P_{cool} \quad = \frac{1}{5}$
- For Hot , 0 yes and 2 no.
- For Mild, 1 yes and 1 no.
- For Cool, 1 yes and 0 no.
- $Entropy(Hot) = -\frac{2}{2} log_2 (\frac{2}{2})$ =0
- Entropy(mild)= $-\frac{1}{2} log_2 (\frac{1}{2}) - \frac{1}{2} log_2 (\frac{1}{2})$=1
- Entropy(cool)= $-\frac{1}{1} log_2 (1)$ =0

- $IG(sunny, temperature) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$

  =0.57

Information gain of Humidity

- $IG(sunny, Humidity) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$
- X= {High=3,normal=2}
- $P_{High} \quad = \frac{3}{5}$
- $P_{normal} = \frac{2}{5}$
- For High , 0 yes and 3 no.
- For Normal, 2 yes and 0 no
- $Entropy(High) = 0$
- Entropy(normal)= 0

- $IG(sunny, Humidity) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$

  =0.96

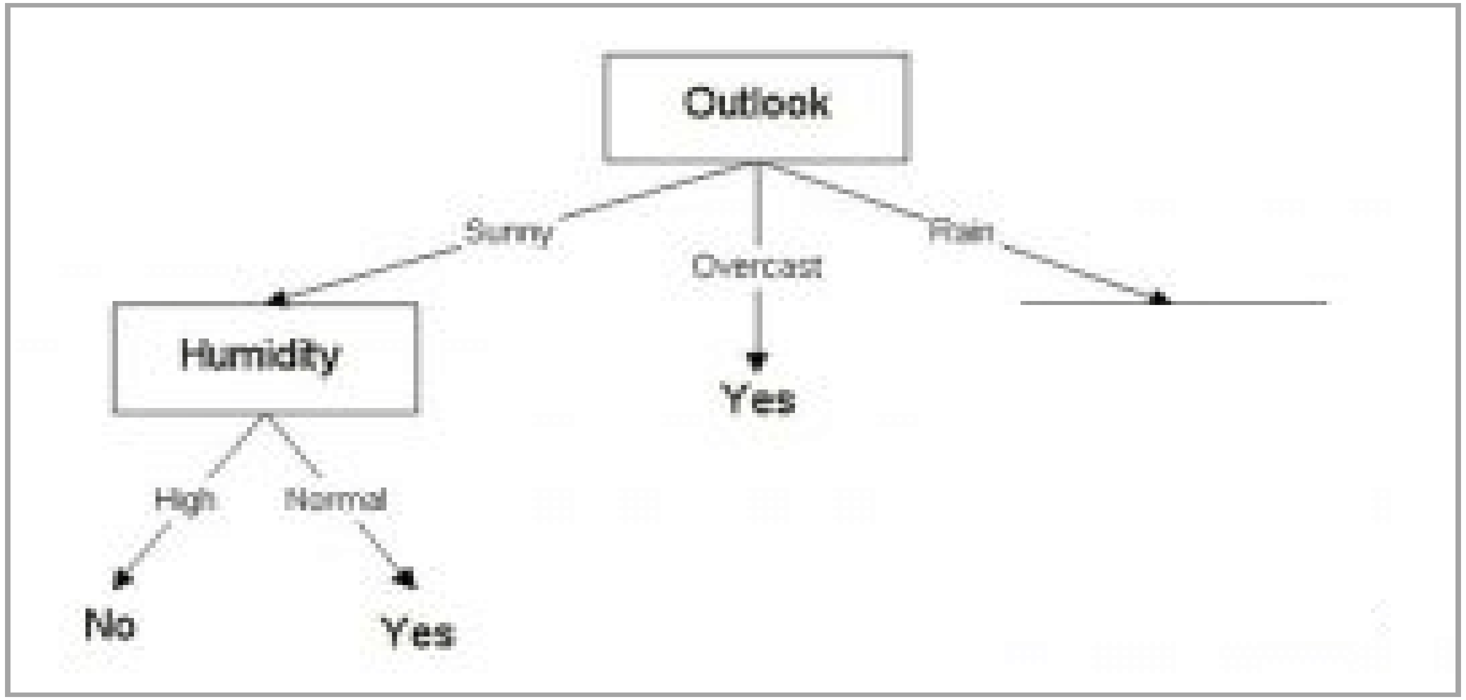
Information gain of wind

- $IG(sunny, wind) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$
- X= {weak=3,Strong=2}
- $P_{weak} \quad = \frac{3}{5}$
- $P_{strong} = \frac{2}{5}$
- For weak , 1 yes and 2 no.
- For strong, 1 yes and 1 no
- $Entropy(weak) = -\frac{1}{3} log_2(\frac{1}{3})$ - $\frac{2}{3} log_2(\frac{2}{3})$=0.91
- Entropy(strong)= $-\frac{1}{2} log_2(\frac{1}{2}) - \frac{1}{2} log_2(\frac{1}{2}$)=1

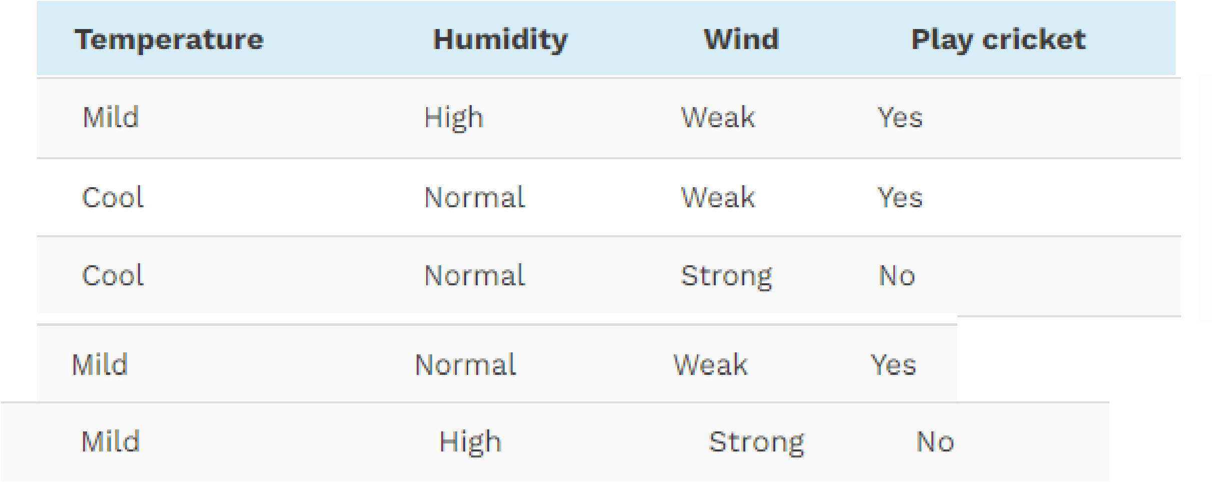
- $IG(sunny, wind) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$

  $=0.014$

The information gain for humidity is highest, therefore it is chosen as the next node



For Rain, find entropy and information gain



Temperature	Humidity	Wind	Play cricket
Mild	High	Weak	Yes
Cool	Normal	Weak	Yes
Cool	Normal	Strong	No
Mild	Normal	Weak	Yes
Mild	High	Strong	No

Entropy of outlook,rainy = $-\frac{3}{5}log_2\frac{3}{5}-\frac{2}{5}log_2\frac{2}{5}$= 0.97

Information gain of Humidity

- $IG(Rainy, Humidity) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$
- X= {High=2,normal=3}
- $P_{High} \quad = \frac{2}{5}$
- $P_{normal} = \frac{3}{5}$
- For High , 1 yes and 1 no.
- For Normal, 2 yes and 1 no
- $Entropy(High) = 1$
- Entropy(normal)= $-\frac{2}{3} log_2(\frac{2}{3}) - \frac{1}{3} log_2(\frac{1}{3}$)=0.91

- $IG(rainy, Humidity) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$

  =0.024

Information gain of temperature

- $IG(Rainy, Temperature) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$
- X= {Mild=3,cool=2}
- $P_{Mild} \quad = \frac{3}{5}$
- $P_{cool} = \frac{2}{5}$
- For mild, 2 yes and 1 no.
- For cool, 1 yes and 1 no
- $Entropy(cool) = 1$
- Entropy(mild)= $-\frac{2}{3} log_2(\frac{2}{3}) - \frac{1}{3} log_2(\frac{1}{3}$)=0.91

- $IG(rainy, temperature) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$

=0.024

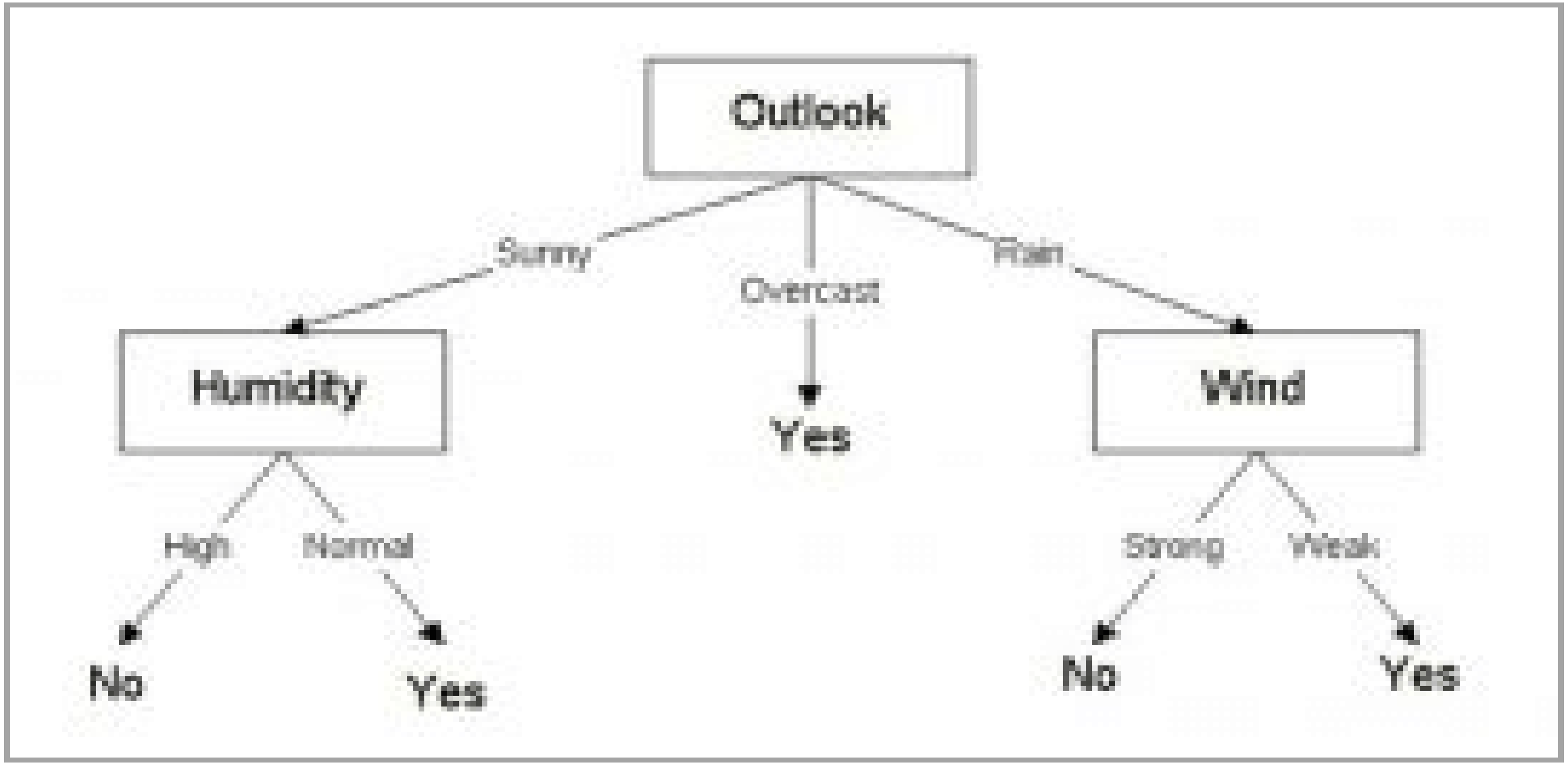
Information gain of wind

- $IG(rainy, wind) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$
- X= {weak=3,Strong=2}
- $P_{weak} \quad = \frac{3}{5}$
- $P_{strong} = \frac{2}{5}$
- For weak , 3 yes and 0 no.
- For strong, 0 yes and 2 no
- $Entropy(weak) = 0$
- Entropy(strong)= 0

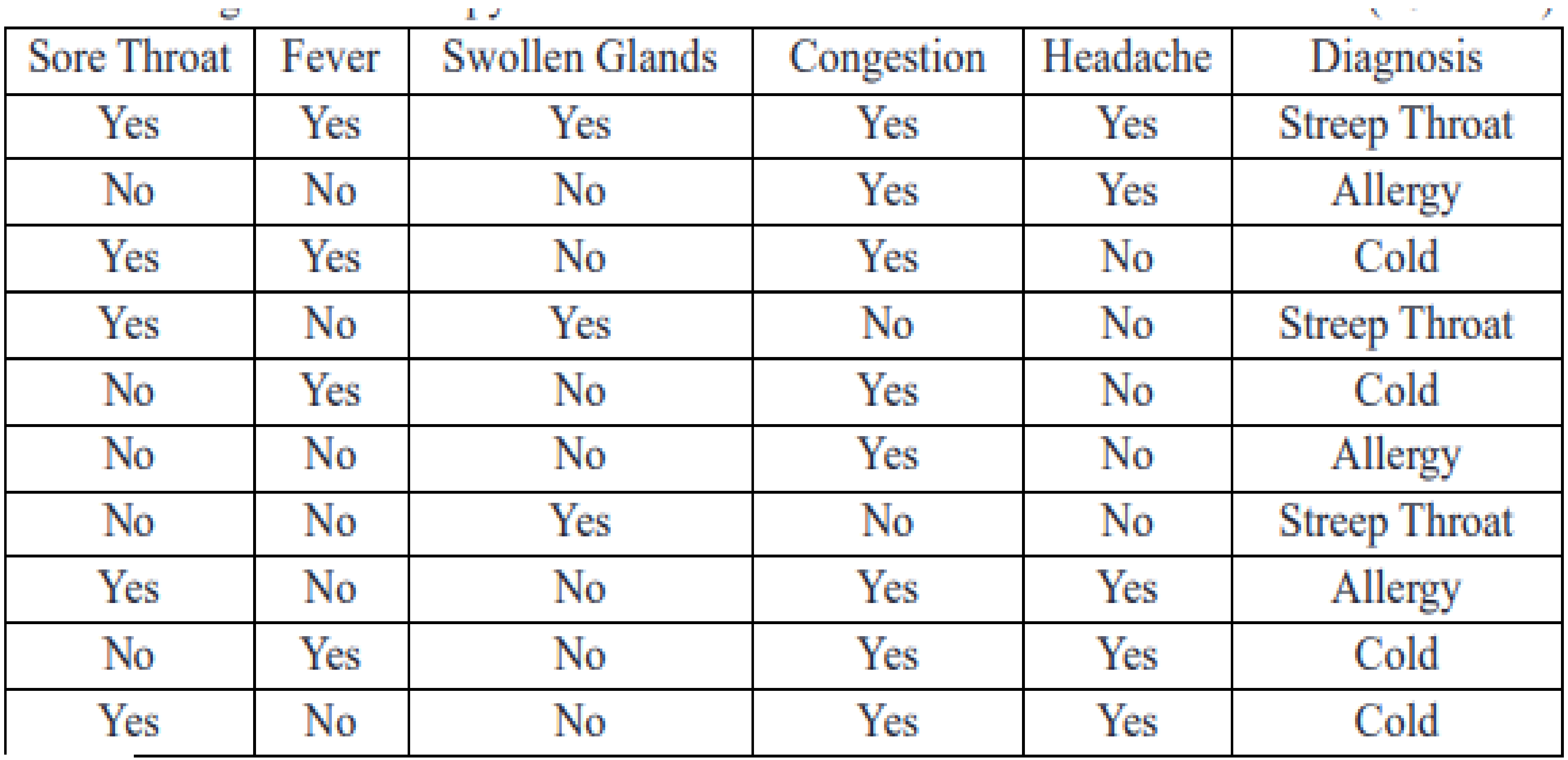
- $IG(rainy, wind) = \mathrm{H(s)} - \sum_{i=0}^{n} p(x) * H(x)$

  =0.97

Wind gives the highest information gain.so we select wind as next node.


Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

Draw the decision tree of the given problem.



Sore Throat	Fever	Swollen Glands	Congestion	Headache	Diagnosis
Yes	Yes	Yes	Yes	Yes	Streep Throat
No	No	No	Yes	Yes	Allergy
Yes	Yes	No	Yes	No	Cold
Yes	No	Yes	No	No	Streep Throat
No	Yes	No	Yes	No	Cold
No	No	No	Yes	No	Allergy
No	No	Yes	No	No	Streep Throat
Yes	No	No	Yes	Yes	Allergy
No	Yes	No	Yes	Yes	Cold
Yes	No	No	Yes	Yes	Cold

Advantages and Disadvantages of Decision trees

- Easy to use, understand
- Produce rules that are easy to interpret & implement
- Variable selection & reduction is automatic
- Does not require the assumptions of statistical models
- Can work without extensive handling of missing data

Disadvantage of single trees:

- instability and poor predictive performance
- Easy to overfit
- Require elaborate pruning
- Output range is bounded (in regression trees)

Gini Index

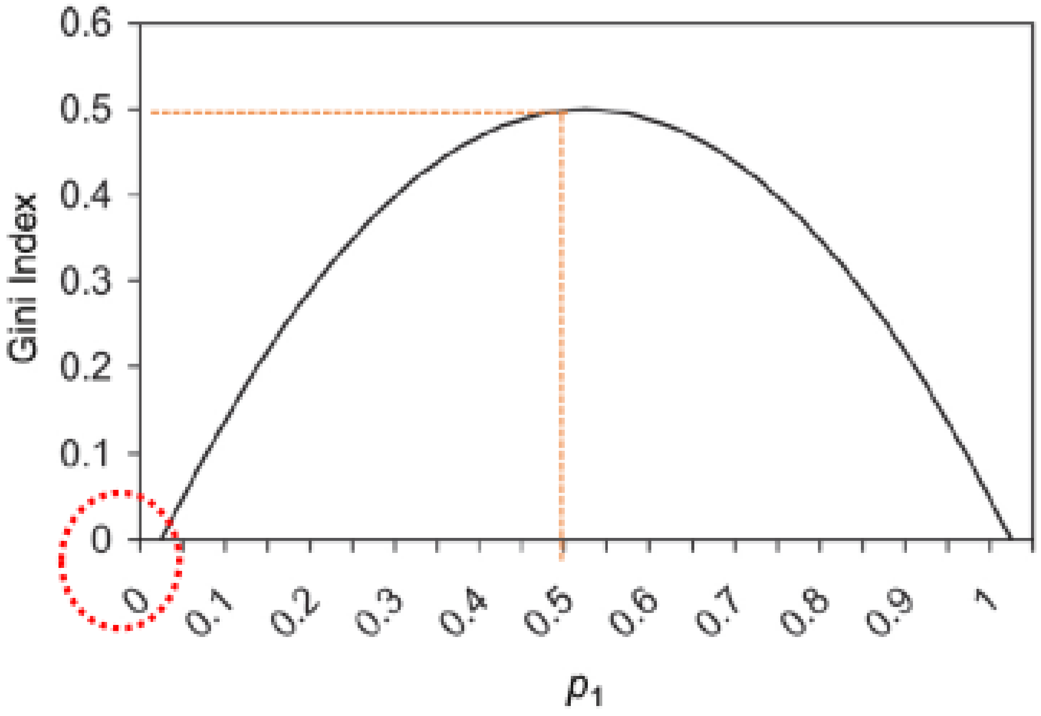
Gini Impurity Index for rectangle *A*

$$I(A) = 1 - \sum_{k=1}^{m} p_k^2$$

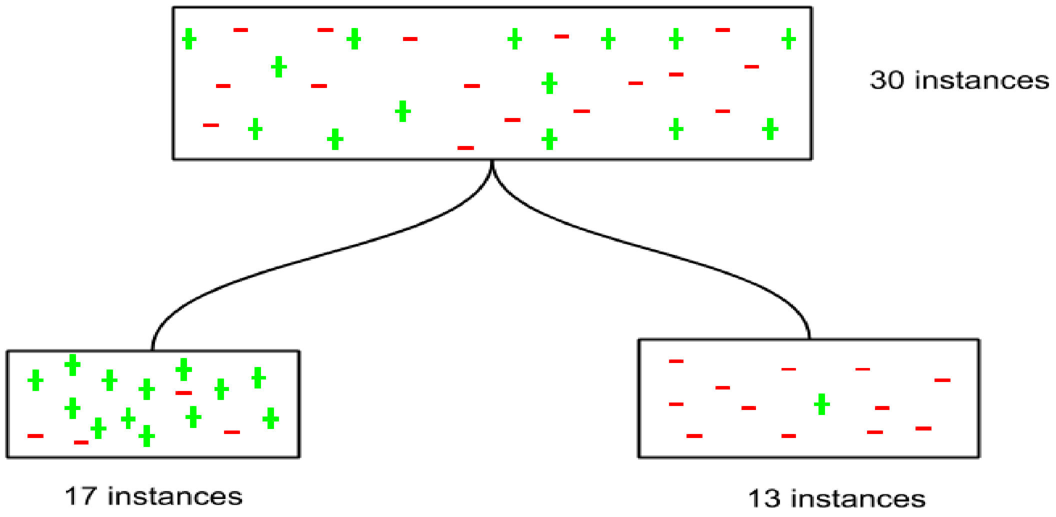
= proportion of cases in rectangle *A* belonging to class *k* (out of *m* classes)

- I(A) = 0 when all cases belong to same class
- Max value when all classes are equally represented (= 0.50 in binary case)

Gini is computationally more efficient to compute than entropy (due to the lack of the log), which could make code negligibly more efficient in terms of computational performance.



VALUES OF THE GINI INDEX FOR A TWO-CLASS CASE AS A FUNCTION OF THE PROPORTION OF RECORDS IN CLASS 1 (p_1)



Gini index(parent)= 1 - $(\frac{14}{30})^2 - \left(\frac{16}{30}\right)^2 = 0.497$

Gini index(child with 17 instances)= 1 - $(\frac{13}{17})^2 - \left(\frac{4}{17}\right)^2 = 0.359$

Gini index(child with 13 instances)= 1 - $(\frac{1}{13})^2 - \left(\frac{12}{13}\right)^2 = 0.142$

Average Gini index child = $(\frac{17}{30})$*0.359+ $\left(\frac{13}{30}\right) * 0.142 = 0.264$

Thus Gini score measuring Impurity has dropped from 0.497 to 0.264 after this split indicating better homegeneity

Misclassification Error – Another Splitting Criteria

- Measures the impurity error
- Instead of using Entropy as an impurity measure, the misclassification error ERR is used,

$$GAIN(\mathcal{D}, xj) = ERR(\mathcal{D}) - \sum_{v \in Values(x_j)} \frac{|\mathcal{D}_v|}{|\mathcal{D}|} ERR(\mathcal{D}_v)$$

$$ERR(\mathcal{D}) = \frac{1}{n} \sum_{i=1}^{n} L(\hat{y}^{[i]}, y^{[i]})$$

with the 0-1 Loss,

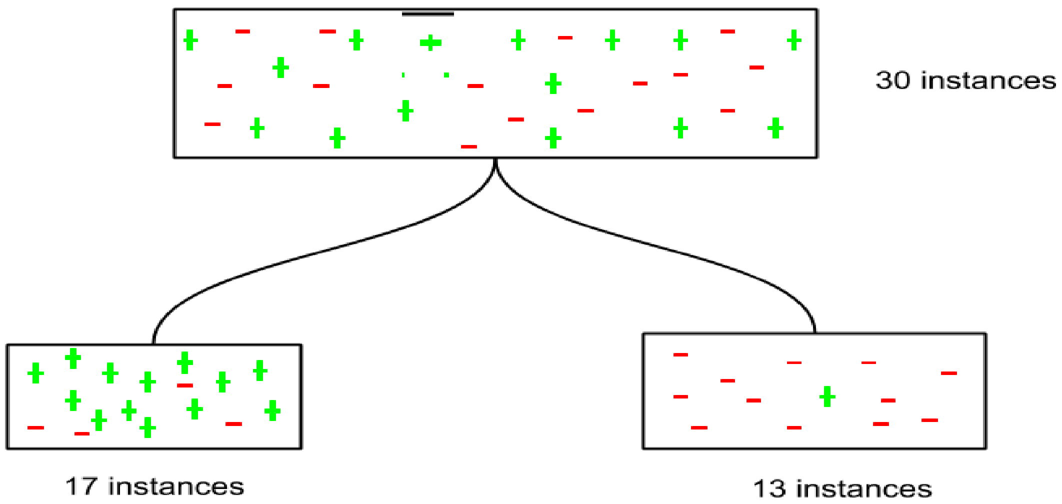
$$L(\hat{y}^{[i]}, y^{[i]}) = \begin{cases} 0 \; if \; \hat{y} = y \\ 1 \; otherwise \end{cases}$$

This in case of the training set is equal to

$$ERR(p) = 1 – \max((p(i / x_j))$$

for a given node if **we use majority voting at this node**

Reference: Sebastian Raschka, STAT 451 Machine Learning Notes, http://stat.wisc.edu/~sraschka/teaching/stat451-fs2020/



ERR(D)= $1 - \max\left(\left(\frac{15}{30}\right), \left(\frac{15}{30}\right)\right) = 0.5$

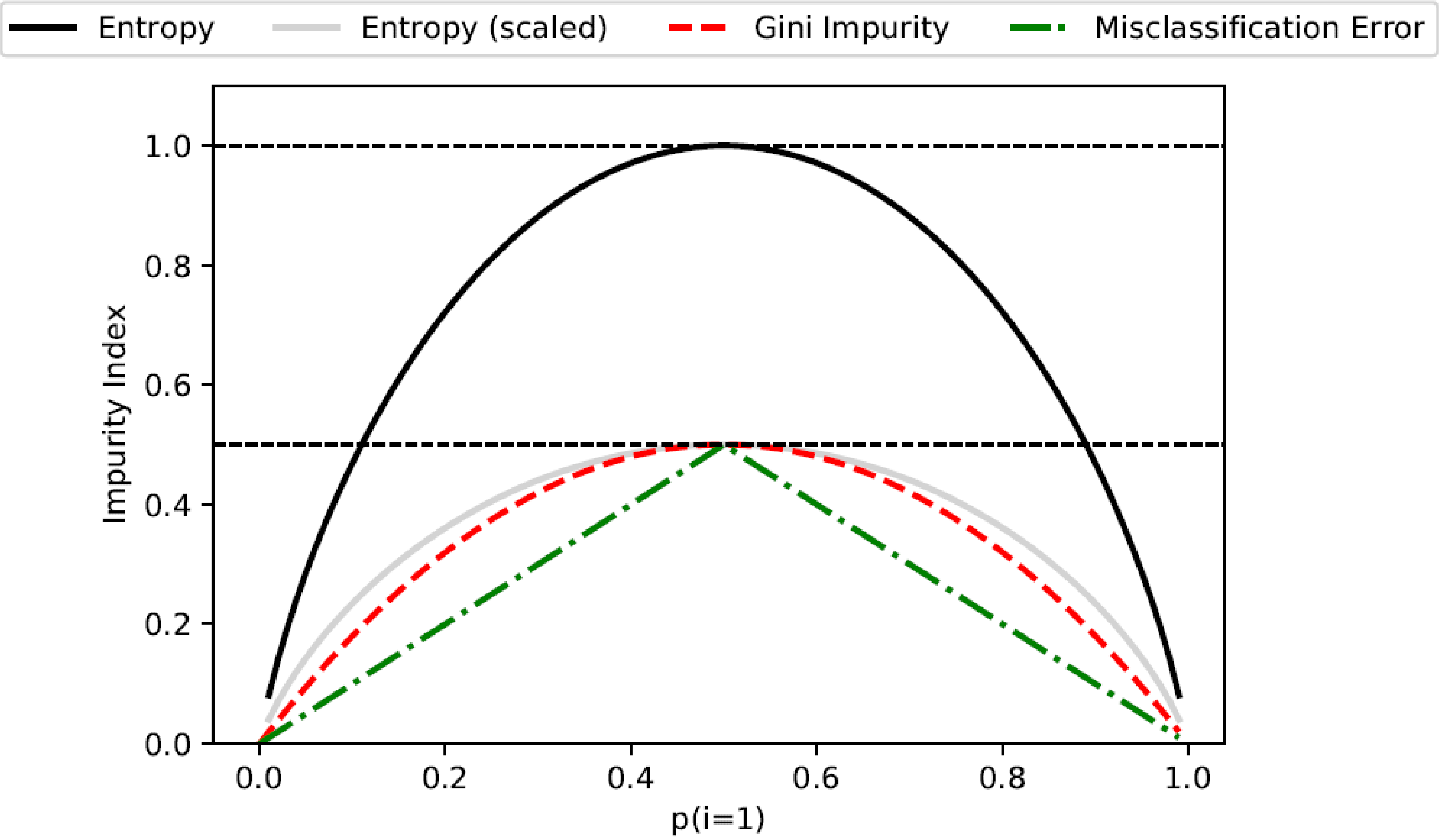
ERR(child with 17 instances)= $1 - \max\left(\left(\frac{13}{17}\right), \left(\frac{4}{17}\right)\right) = 0.235$

ERR(child with 13 instances)= $1 - \max\left(\left(\frac{12}{13}\right), \left(\frac{1}{13}\right)\right) = 0.077$

Average ERR child = $\left(\frac{17}{30}\right)$*0.235+ $\left(\frac{13}{30}\right) * 0.077 = 0.167$

Gain=0.5-0.167=0.333

Comparison of different impurity measures.


Entropy
Entropy (scaled)
Gini Impurity
Misclassification Error
Impurity Index
p(i=1)