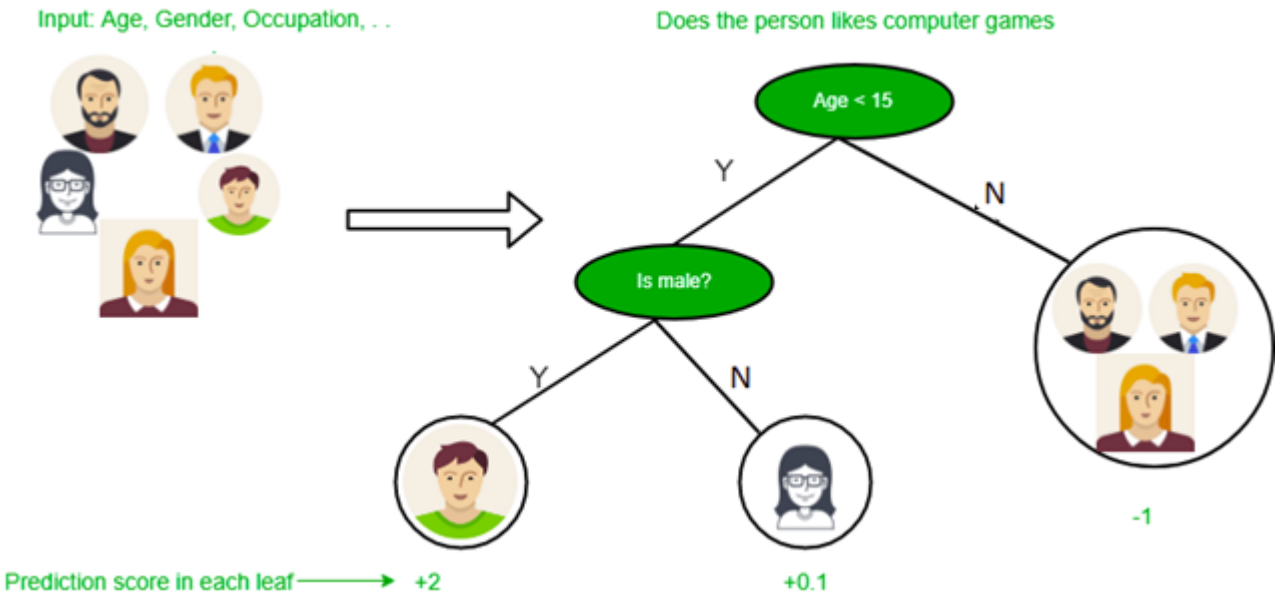
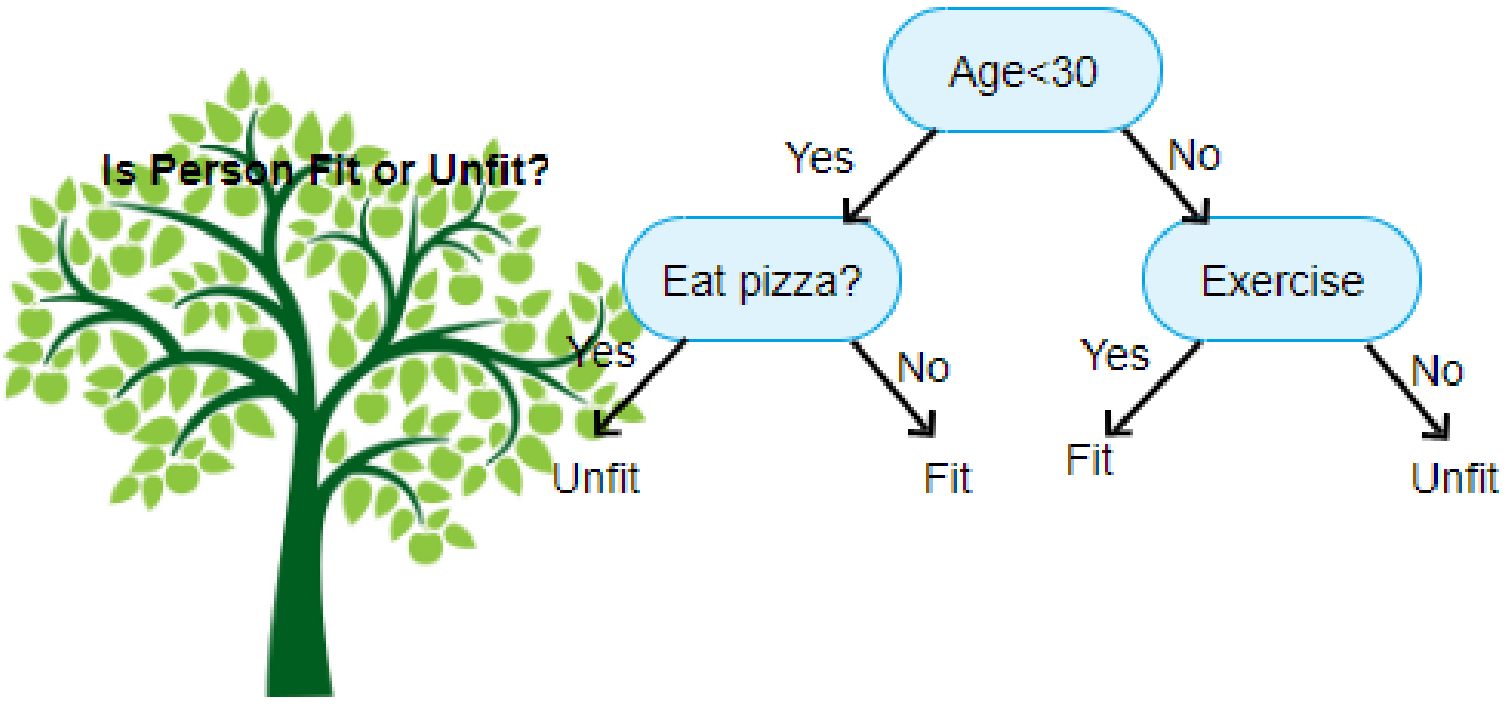
Module 3

Topic 1 :Decision Tree Learning



Decision tree Model

- The model itself comprises a series of logical decisions, similar to a flowchart, with decision nodes that indicate a decision to be made on an attribute.
- These split into branches that indicate the decision's choices. The tree is terminated by leaf nodes (also known as terminal nodes) that denote the result of following a combination of decisions.

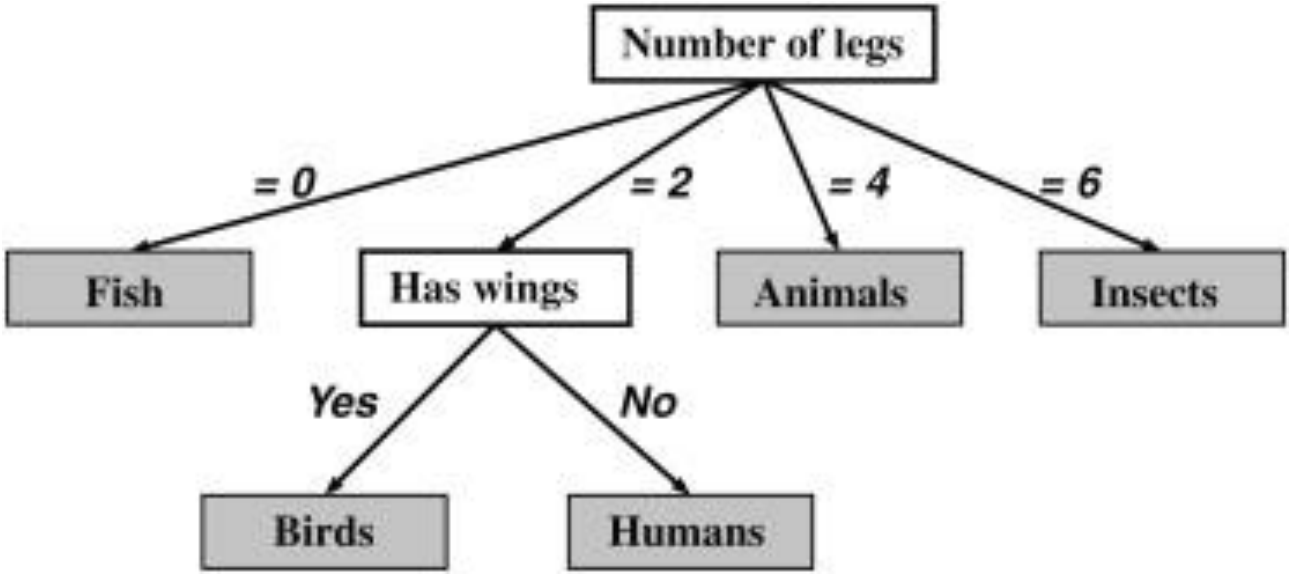

Is Person Fit or Unfit?
Age<30
Yes
No
Eat pizza?
Exercise
Yes
No
Yes
No
Unfit
Fit
Fit
Unfit

- Data that is to be classified begin at the root node where it is passed through the various decisions in the tree according to the values of its features.
- The path that the data takes from each record into a leaf node, assigns it to a predicted class.

Application Areas

- **Credit scoring models** in which the criteria that causes an **applicant to be rejected** need to be well-specified
- **Marketing studies of customer churn or customer satisfaction** that will be shared with management or advertising agencies
- **Diagnosis of medical conditions based on laboratory measurements,symptoms, or rate of disease progression**

- **Divide and conquer**
- Decision trees are built using a heuristic called **recursive partitioning.** This approach is generally known as divide and conquer because it uses the feature values to split the data into smaller and smaller subsets of similar classes.

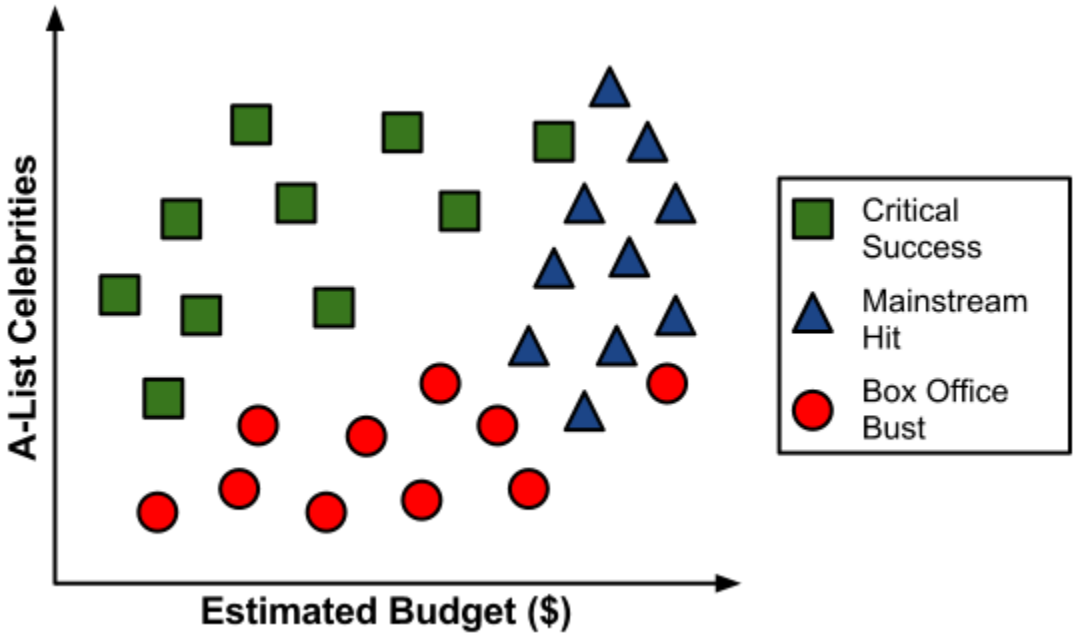


- Beginning at the **root node, which represents the entire dataset, the algorithm chooses a feature that is the most predictive of the target class.**
- The examples are then partitioned into groups of distinct values of this feature; this decision forms the first set of tree branches.
- The algorithm continues to divide-and-conquer the nodes,choosing the best candidate feature each time until a stopping criterion is reached.This might occur at a node if:
- **All (or nearly all) of the examples at the node have the same class**
- **There are no remaining features to distinguish among examples**
- **The tree has grown to a predefined size limit**

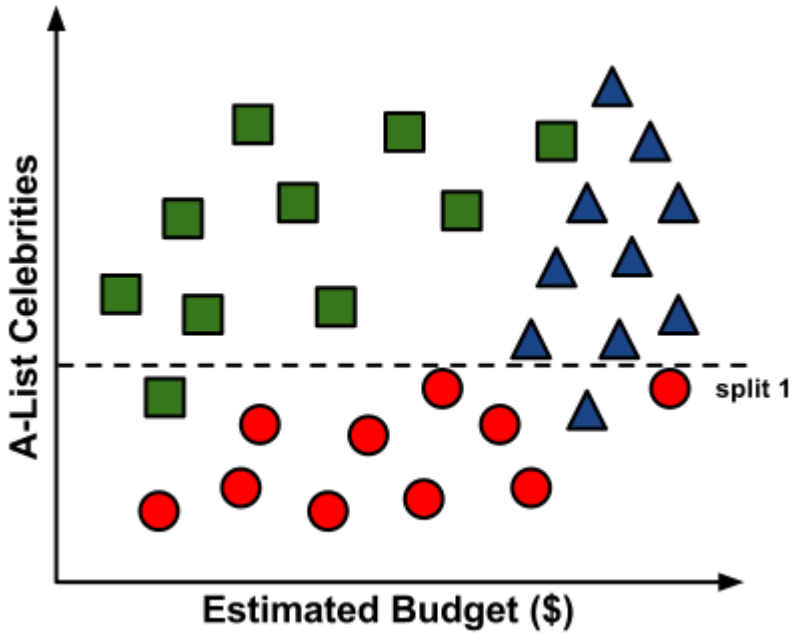
Example

- Imagine that you are working for a Hollywood film studio, and your desk is piled high with screenplays.
- Rather than read each one cover-to-cover, you decide to develop a decision tree algorithm to predict **whether a potential movie** would fall into one of three categories: **mainstream hit, critic's choice, or box office bust.**

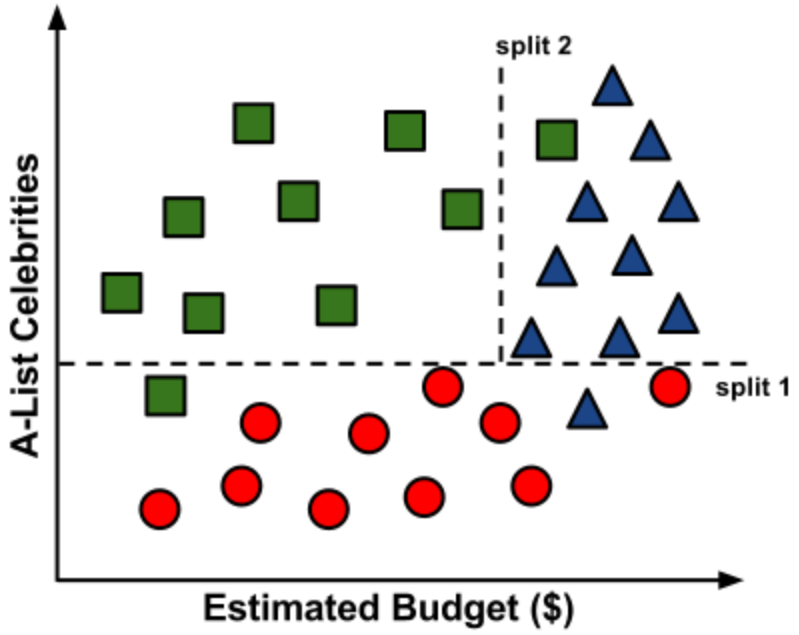
- To gather data for your model, you turn to the studio archives to examine the previous ten years of movie releases.
- After reviewing the data for 30 different movie scripts, a pattern emerges. There seems to be a relationship between the film's **proposed shooting budget, the number of A-list celebrities lined up for starring roles, and the categories of success**. A scatter plot of this data might look something like the following diagram:

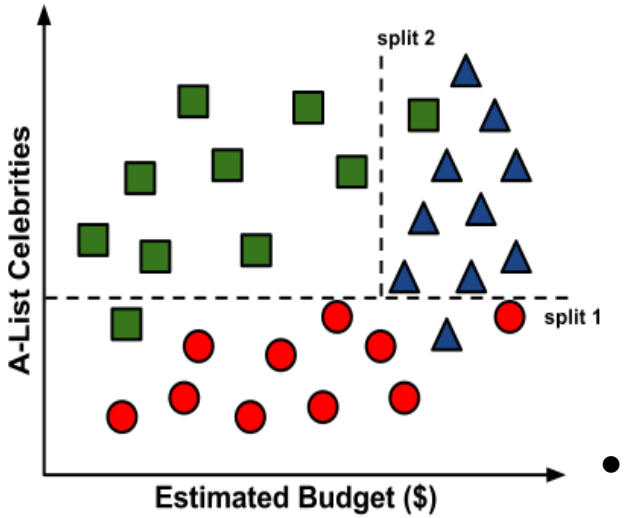

A-List Celebrities
Estimated Budget ($)
Critical Success
Mainstream Hit
Box Office Bust

- To build a simple decision tree using this data, we can apply a divide-and-conquer strategy.
- Let's first split the feature indicating the number of celebrities, partitioning the movies into groups **with and without a low number of A-list stars:**


A-List Celebrities
Estimated Budget ($)
split 1

- Next, among the group of movies with a larger number of celebrities, we can make another split between movies with and without a high budget:

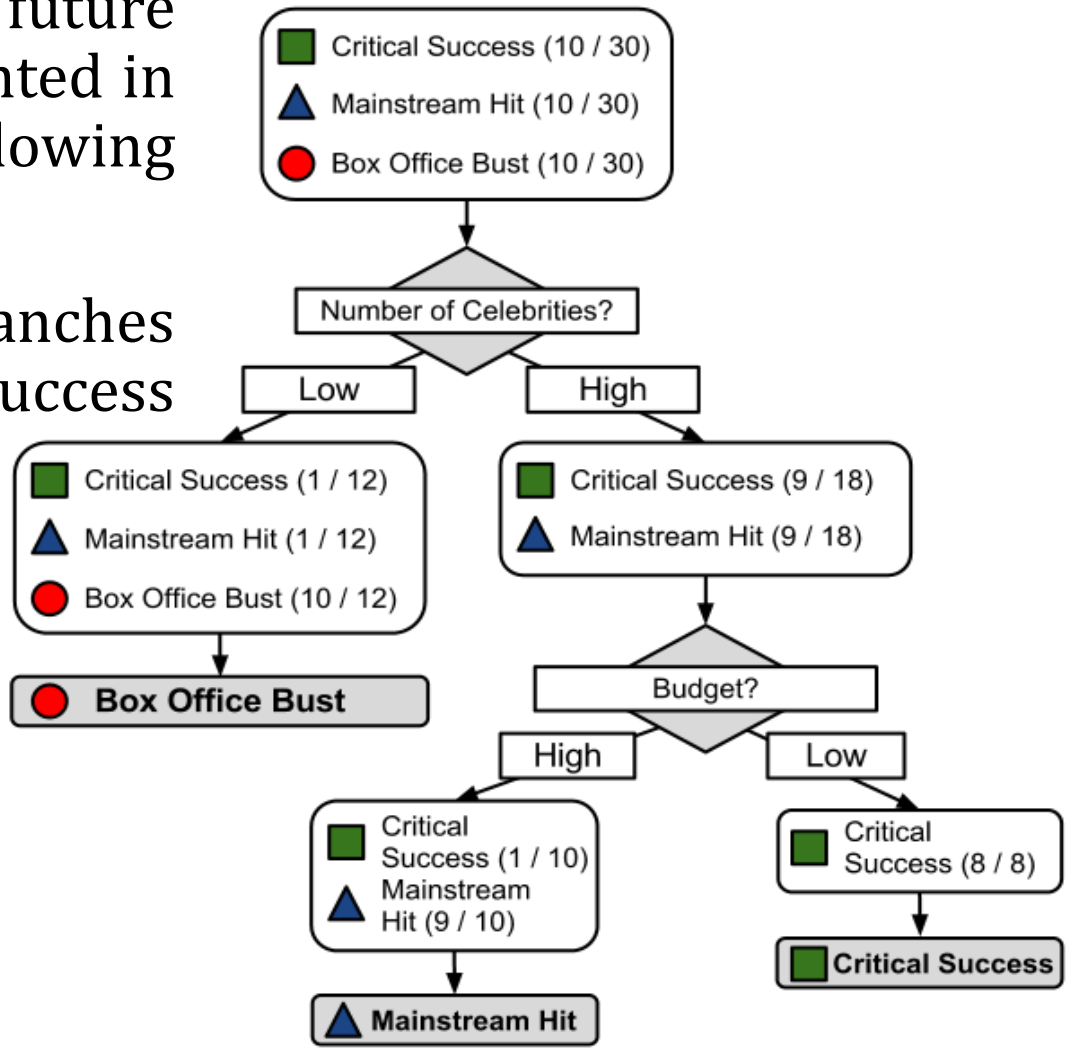




- The group at the top-left corner of the diagram is composed entirely of critically-acclaimed films. This group is distinguished by a high number of celebrities and a relatively low budget.

- At the top-right corner, the majority of movies are box office hits, with high budgets and a large number of celebrities. The final group, which has little star power but budgets ranging from small to large, contains the flops.

- Our model for predicting the future success of movies can be represented in a simple tree as shown in the following diagram.
- To evaluate a script, follow the branches through each decision until its success or failure has been predicted



- Since real-world data contains more than two features, decision trees quickly become far more complex than this, with many more nodes, branches, and leaves.

The C5.0 decision tree algorithm

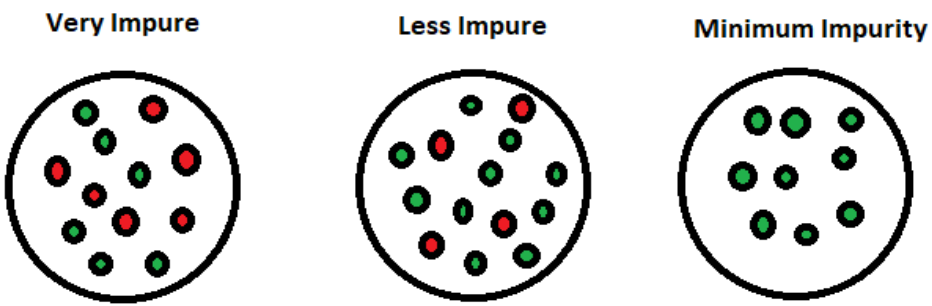
- This algorithm was developed by computer scientist J.Ross Quinlan as an improved version of his prior algorithm, C4.5, which itself is an improvement over his ID3 (Iterative Dichotomiser 3) algorithm.

Choosing the best split

- The first challenge that a decision tree will face is to identify which **feature to split upon.**
- In the process, we looked for feature values that split the data in such a way that the partitions contained examples belong to a single class.
- **If the segments of data contain only a single class, they are considered pure.**

Entropy

- Entropy basically tells us how impure a collection of data is. The term impure here defines non-homogeneity.
- In other words, “Entropy is the measurement of homogeneity. It returns the information about how impure/non-homogeneous the data set is.”



- C5.0 uses ***entropy*** **for measuring purity. The entropy of a sample of data indicates how mixed the class values are**;
- Its minimum value is 0, indicates that the sample is completely homogenous. Its value 1 indicates the maximum amount of disorder.

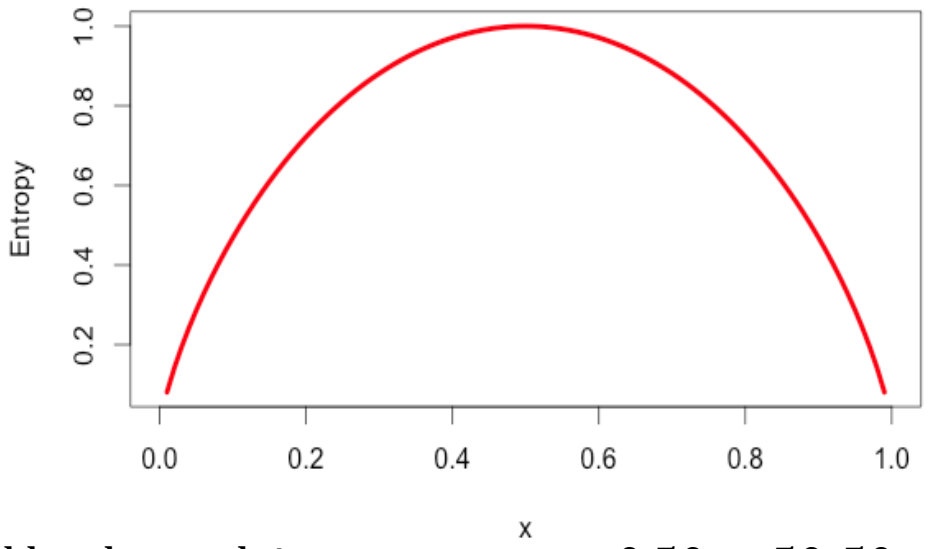
- The definition of entropy is specified by:

$$\text{Entropy}(S) = \sum -p_i \log_2(p_i)$$, i ranges from 1 to c (Assume that the dataset contains ‘c’ no. of classes)

- Entropy

  - For example, suppose we have a partition of data with two classes: red (60 percent), and white (40 percent). We can calculate the entropy as:
  - > $-0.60 * \log_2(0.60) - 0.40 * \log_2(0.40)$
  - = 0.9709506

- We can examine the entropy for all possible two-class arrangements. If we know the proportion of examples in one class is x , then the proportion in the other class is 1-x .
- Using the curve() function, we can then plot the entropy for all possible values of x :
- > curve(-x * $\log_2(x)$ - (1 - x) * $\log_2(1 - x)$)



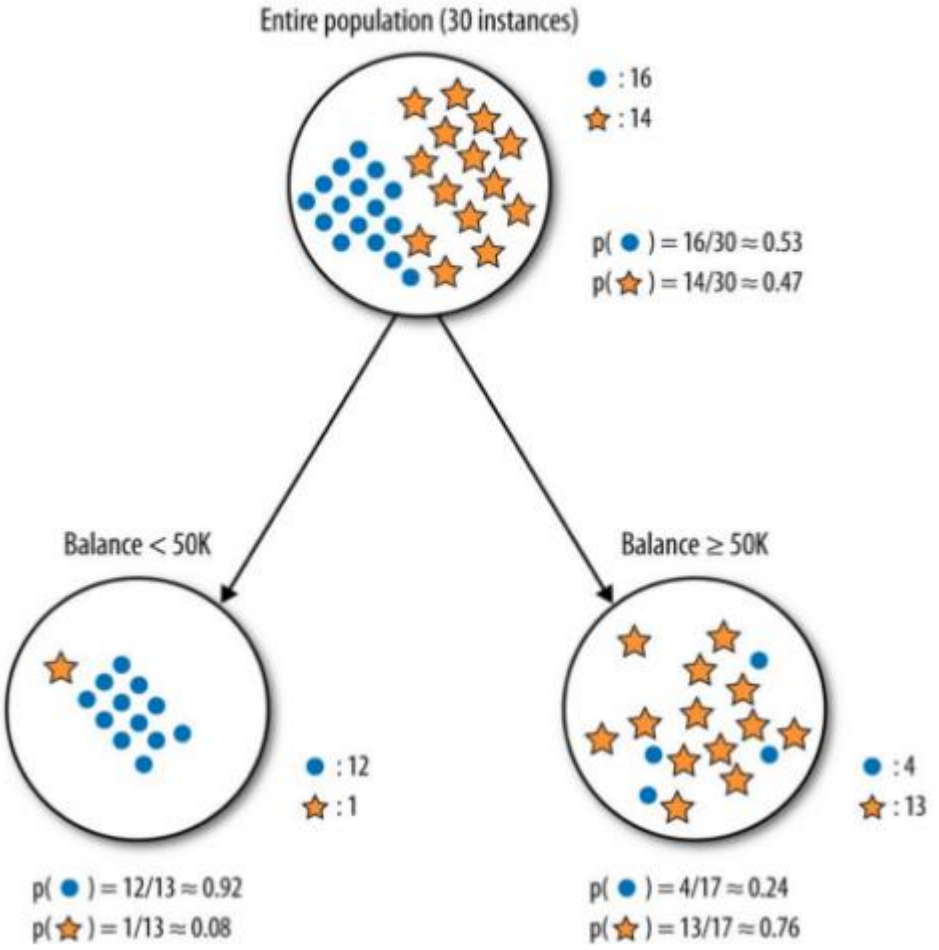
- As illustrated by the peak in entropy at x = 0.50 , a 50-50 split results in the maximum entropy.

- Information Gain
  - The algorithm uses entropy to calculate the change in homogeneity resulting from a split on each possible feature. The calculation is known as **information gain.**
  - **The information gain for a feature F is calculated as the difference between the entropy in the segment before the split (S 1), and the partitions resulting from the split (S 2):**
  - **InfoGain (F) = Entropy (S 1) – Entropy (S 2)**

Information gain using entropy calculation-Example

- Consider an example where we are **building a decision tree to predict whether a loan given to a person** would result in a write-off or not.
- Our entire population consists of 30 instances. 16 belong to the write-off class and the other 14 belong to the non-write-off class. We have two features, namely “Balance” that can take on two values -> “< 50K” or “>50K” and “Residence” that can take on three values -> “OWN”, “RENT” or “OTHER”.

Feature 1: Balance



$$E(Parent) = -\frac{16}{30}\log_2\left(\frac{16}{30}\right) - \frac{14}{30}\log_2\left(\frac{14}{30}\right) \approx 0.99$$

$$E(Balance < 50K) = -\frac{12}{13}\log_2\left(\frac{12}{13}\right) - \frac{1}{13}\log_2\left(\frac{1}{13}\right) \approx 0.39$$

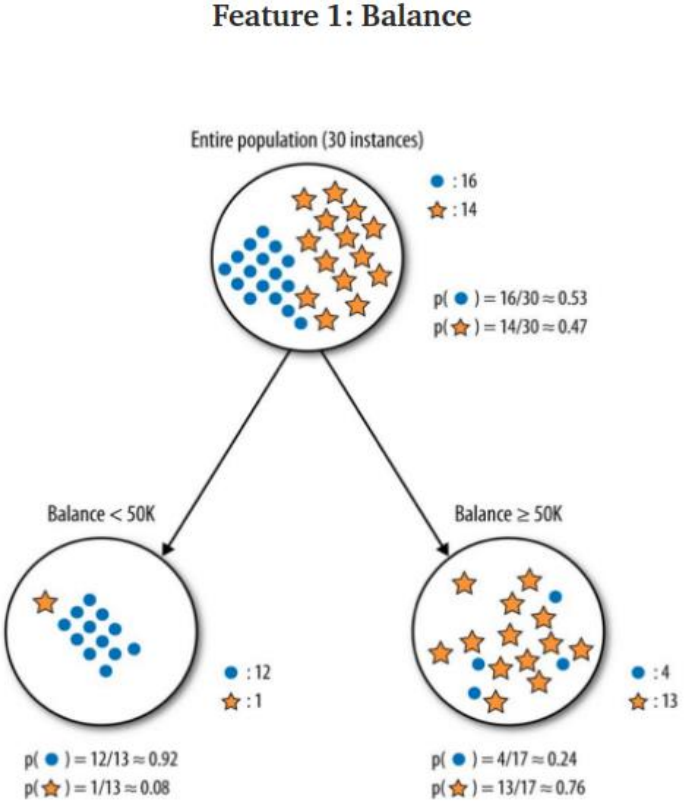
$$E(Balance > 50K) = -\frac{4}{17}\log_2\left(\frac{4}{17}\right) - \frac{13}{17}\log_2\left(\frac{13}{17}\right) \approx 0.79$$

Weighted Average of entropy for each node:

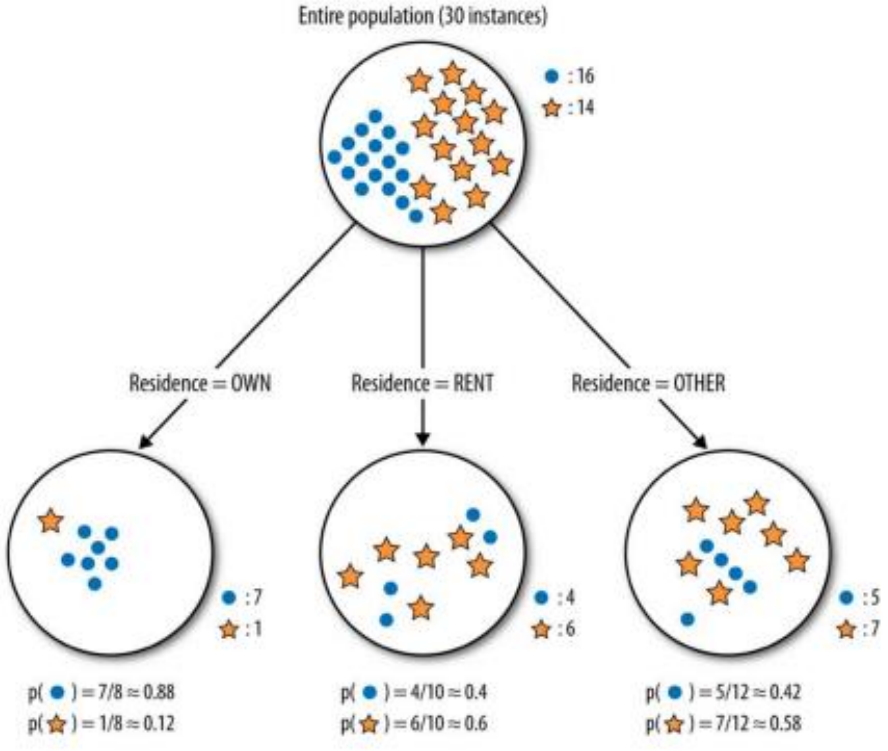
$$E(Balance) = \frac{13}{30} \times 0.39 + \frac{17}{30} \times 0.79$$
$$= 0.62$$

Information Gain:

$$IG(Parent, Balance) = E(Parent) - E(Balance)$$
$$= 0.99 - 0.62$$
$$= 0.37$$



Feature 2-Type of residence



$$E(Residence = OWN) = -\frac{7}{8}\log_2\left(\frac{7}{8}\right) - \frac{1}{8}\log_2\left(\frac{1}{8}\right) \approx 0.54$$

$$E(Residence = RENT) = -\frac{4}{10}\log_2\left(\frac{4}{10}\right) - \frac{6}{10}\log_2\left(\frac{6}{10}\right) \approx 0.97$$

$$E(Residence = OTHER) = -\frac{5}{12}\log_2\left(\frac{5}{12}\right) - \frac{7}{12}\log_2\left(\frac{7}{12}\right) \approx 0.98$$

Weighted Average of entropies for each node:

$$E(Residence) = \frac{8}{30} \times 0.54 + \frac{10}{30} \times 0.97 + \frac{12}{30} \times 0.98 = 0.86$$

Information Gain:

$$\begin{aligned} IG(Parent, Residence) &= E(Parent) - E(Residence) \\ &= 0.99 - 0.86 \\ &= 0.13 \end{aligned}$$

Entropy calculation based on dataset

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

Example -Entropy calculation

- let's use this equation and measure the **information gain of attribute Wind** from the dataset of Figure 1.
- The dataset has 14 instances, so the sample space is 14 where the sample has 9 positive and 5 negative instances.
- The Attribute Wind can have the values **Weak or Strong.** Therefore,

  Values(Wind) = Weak, Strong

$S = [9+, 5-]$
$S_{\text{weak}} = [6+, 2-]$
$S_{\text{strong}} = [3+, 3-]$
$Entropy(S) = 0.940$ from eqtn 1.2

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

$$Gain(S, Wind) = Entropy(S) - (\frac{8}{14} Entropy(S_{\text{weak}}) + \frac{6}{14} Entropy(S_{\text{strong}})) \tag{1.6}$$

$$Entropy(S_{\text{weak}}) = -(\frac{6}{8} log_2 \frac{6}{8} + \frac{2}{8} log_2 \frac{2}{8}) = 0.811 \tag{1.7}$$

$$Entropy(S_{\text{strong}}) = -(\frac{3}{3} log_2 \frac{3}{3} + \frac{3}{3} log_2 \frac{3}{3}) = 1.00 \tag{1.8}$$

Put the values of $Entropy(S_{\text{weak}})$ and $Entropy(S_{\text{strong}})$ in eqtn 1.6

$$\begin{aligned} Gain(S, Wind) &= Entropy(S) - (\frac{8}{14} 0.811 + \frac{6}{14} 1.00) \\ &= 0.940 - (0.463 + 0.429) \\ &= 0.048 \end{aligned} \tag{1.9}$$

- So, the information gain by the Wind attribute is 0.048. Let’s calculate the information gain by the Outlook attribute.
-

$Values(Outlook) = Sunny, Overcast, Rain$
$S = [9+, 5-]$
$S_{\text{sunny}} = [2+, 3-]$
$S_{\text{overcast}} = [4+, 0-]$
$S_{\text{rain}} = [3+, 2-]$

$$G(S, Outlook) = Entropy(S) - (\frac{5}{14} Entropy(S_{\text{sunny}}) + \frac{4}{14} Entropy(S_{\text{overcast}}) + \frac{5}{14} Entropy(S_{\text{rain}})) \tag{1.10}$$

$$Entropy(S_{\text{sunny}}) = -(\frac{2}{5} log_2 \frac{2}{5} + \frac{3}{5} log_2 \frac{3}{5}) = 0.971 \tag{1.11}$$

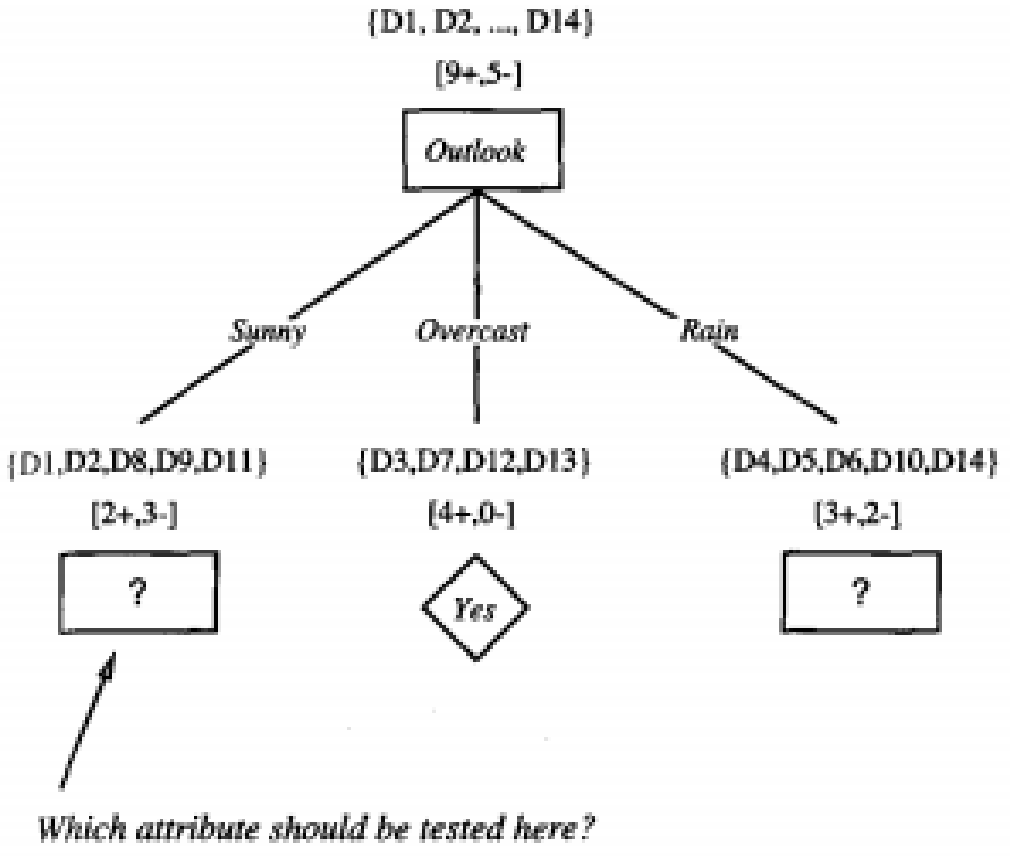
$$Entropy(S_{\text{overcast}}) = -(\frac{4}{4} log_2 \frac{4}{4} + \frac{0}{4} log_2 \frac{0}{4}) = 0 \tag{1.12}$$

$$Entropy(S_{\text{rain}}) = -(\frac{3}{5} log_2 \frac{3}{5} + \frac{2}{5} log_2 \frac{2}{5}) = 0.971 \tag{1.13}$$

Put the values of eqtn 1.11, 1.12, 1.13 in 1.10.

$$Gain(S, Outlook) = 0.246$$
$$Gain(S, Humidity) = 0.151$$
$$Gain(S, Wind) = 0.048$$
$$Gain(S, Temperature) = 0.029$$

- As far as we calculated, the most useful attribute is “Outlook” as it is giving us more information than others. So, “Outlook” will be the root of our tree.


{D1, D2, ..., D14}
[9+,5-]
Outlook
Sunny
Overcast
Rain
{D1,D2,D8,D9,D11}
[2+,3-]
{D3,D7,D12,D13}
[4+,0-]
{D4,D5,D6,D10,D14}
[3+,2-]
?
Yes
?
Which attribute should be tested here?

- The Overcast descendant has only positive instances and therefore becomes a leaf node with classification Yes.
-

$S_{\text{suuny}} = 5 = S$
$Humidity = High, Normal$
$Humidity_{\text{high}} = [0+, 3-]$
$Humidity_{\text{normal}} = [2+, 0-]$
$Gain(S, Humidity) = ?$

$$Gain(S_{\text{sunny}}, Humidity) = Entropy(S) - (\frac{3}{5}Entropy(Humidity_{\text{high}}) + \frac{2}{5}Humidity_{\text{normal}} \tag{1.15}$$

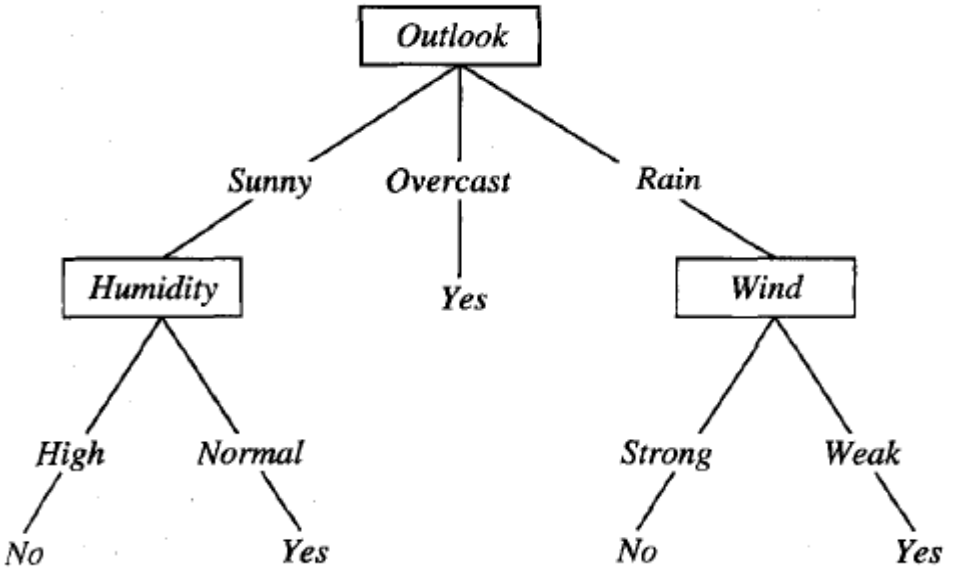
$$Entropy(Humidity_{\text{high}}) = -(\frac{0}{3}log_2\frac{0}{3} + \frac{3}{3}log_2\frac{3}{3}) = 0 \tag{1.16}$$

$$Entropy(Humidity_{\text{normal}}) = -(\frac{2}{2}log_2\frac{2}{2} + \frac{0}{2}log_2\frac{0}{2}) = 0 \tag{1.17}$$

Put the values in eqtn 1.15

$$\begin{aligned} Gain(S_{\text{sunny}}, Humidity) &= Entropy(S) - (\frac{3}{5}0 + \frac{2}{5}0) \\ &= 0.970 - 0 \\ &= 0.970 \end{aligned} \tag{1.18}$$

$Gain(S, Humidity) = 0.970$
$Gain(S, Temperature) = 0.570$
$Gain(S, Wind) = 0.019$



- If we expand the Rain descendant by the same procedure we will see that the Wind attribute is providing most information

C5.0 -steps

- A C5.0 model works by splitting the sample based **on the field that provides the maximum information gain.**
- Each sub-sample defined by the first split is then split again, usually based on a different field, and the **process repeats until the subsamples cannot be split any further.**
- Finally, the **lowest-level splits are re-examined, and those that do not contribute significantly to the value of the model are removed or pruned.**

Pruning the decision tree

- A decision tree can continue to grow indefinitely, choosing splitting features and dividing into smaller and smaller partitions until each example is perfectly classified or the algorithm runs out of features to split on.
- However, **if the tree grows overly large, many of the decisions it makes will be overly specific** and the model will have been overfitted to the training data.
- The process of **pruning a decision tree involves reducing its size such that it generalizes better to unseen data.**

Pre/post pruning

- One solution to this problem is t**o stop the tree from growing once it reaches a certain number of decisions** or if the decision nodes contain only a small number of examples. This is called **early stopping or pre-pruning** the decision tree
- An alternative, called post-pruning **involves growing a tree that is too large, then using pruning criteria based on the error rates at the nodes to reduce the size of the tree to a more appropriate level.**
- This is often a more effective approach than **pre-pruning because it is quite difficult to determine the optimal depth of a decision tree without growing it first.** Pruning the tree later on allows the algorithm **to be certain that all important data structures** were discovered.

C5.0 strategy

- Its overall strategy is to postprune the tree. It first **grows a large tree that overfits the training data**. Later, **nodes and branches that have little effect** on the classification errors are **removed.**
- In some cases, **entire branches are moved further up the tree or replaced by simpler decisions.** These processes of grafting branches are known as **subtree raising and subtree replacement**, respectively.

Classification Rule learning

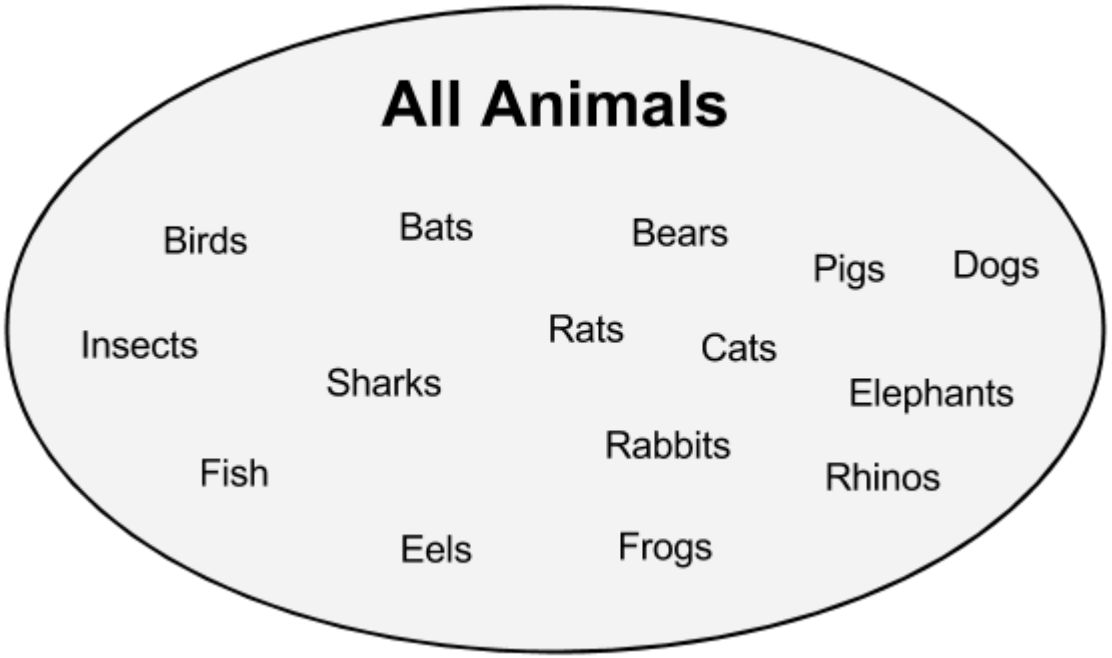
- Classification rules represent knowledge in the form of logical if-else statements that assign a class to unlabeled examples.
- They are specified in terms of **an antecedent and a consequent**; these form a hypothesis stating that "**if this happens, then that happens**.

- The antecedent comprises certain combinations of feature values, while the consequent specifies the class value to assign if the rule's conditions are met.

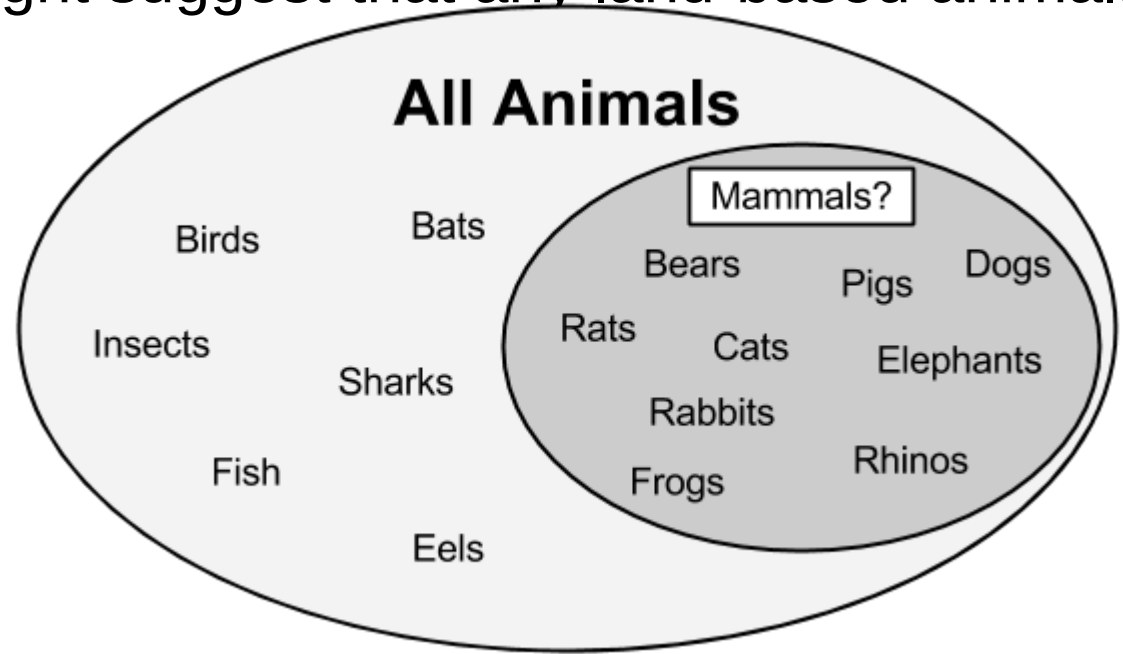
Separate and conquer

- Classification rule learning algorithms utilize a heuristic known as separate and conquer.
- The process involves identifying **a rule that covers a subset of examples in the training data, and then separating this partition from the remaining data.**
- **As rules are added, additional subsets of data are separated until the entire dataset has been covered and no more examples remain.**

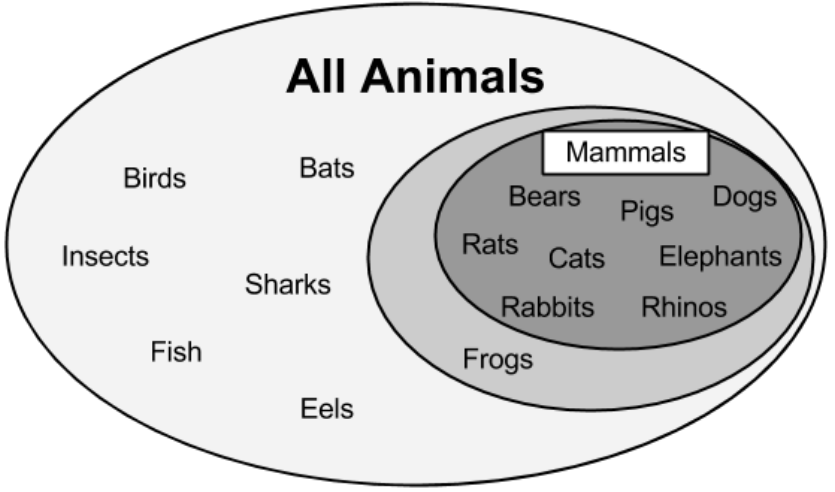
- Suppose you were tasked with creating rules for identifying whether or not an animal is a mammal. You could depict the set of all animals as a large space, as shown in the following diagram:
-



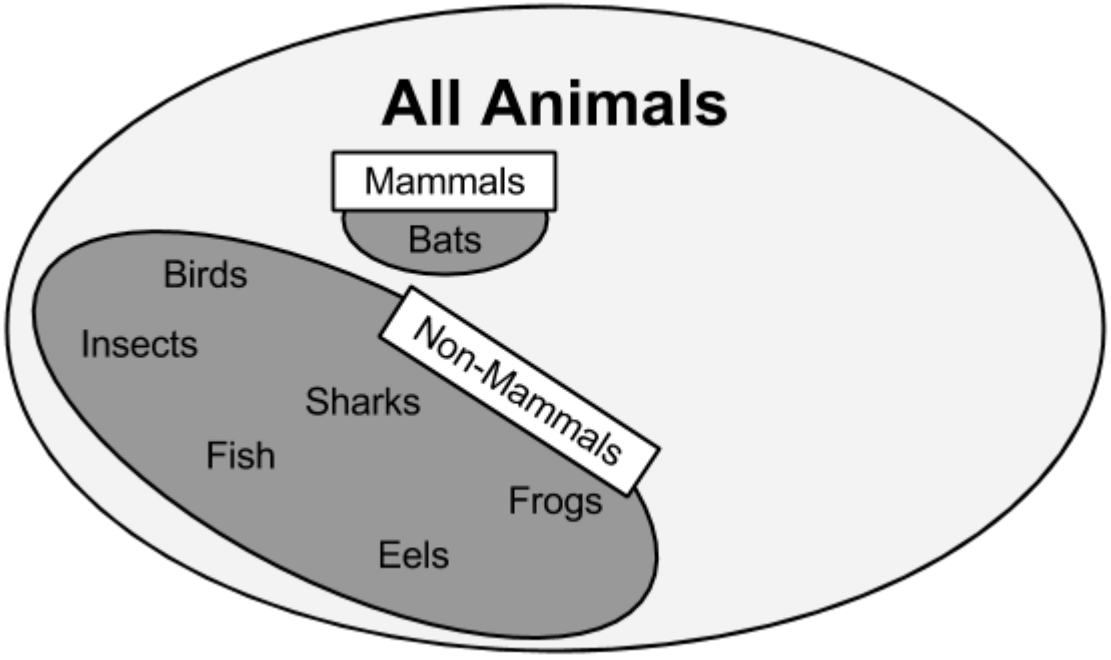
- A rule learner begins by using the available features to find homogeneous groups.For example, using a feature that measured whether the species travels via land, sea,or air, the first rule might suggest that any land-based animals are mammals:
-



- If you look carefully, you might note that frogs are amphibians, not mammals. Therefore, our rule needs to be a bit more specific. Let's drill down further by suggesting that mammals walk on land and have a tail:
-



- Thus, this subset can be separated from the other data and additional rules can be defined to identify the remaining **mammal bats**.
- A potential feature distinguishing bats from the other remaining animals would be the **presence of fur.**
- Using a rule built around this feature, we have then correctly identified all the animals:


All Animals
Mammals
Bats
Birds
Insects
Non-Mammals
Sharks
Fish
Frogs
Eels

- At this point, since all of the training instances have been classified, the rule learning process would stop. We learned a total of **three rules**:
- Animals that walk on land and have tails are mammals
- If the animal has fur, it is a mammal
- Otherwise, the animal is not a mammal

The One Rule algorithm(1R Algorithm)

- ZeroR, a rule learner that literally learns no rules (hence the name).
- For every unlabeled example, regardless of the values of its features, it predicts the most common class.The **One Rule algorithm** (1R or OneR), improves over ZeroR by selecting a single rule.

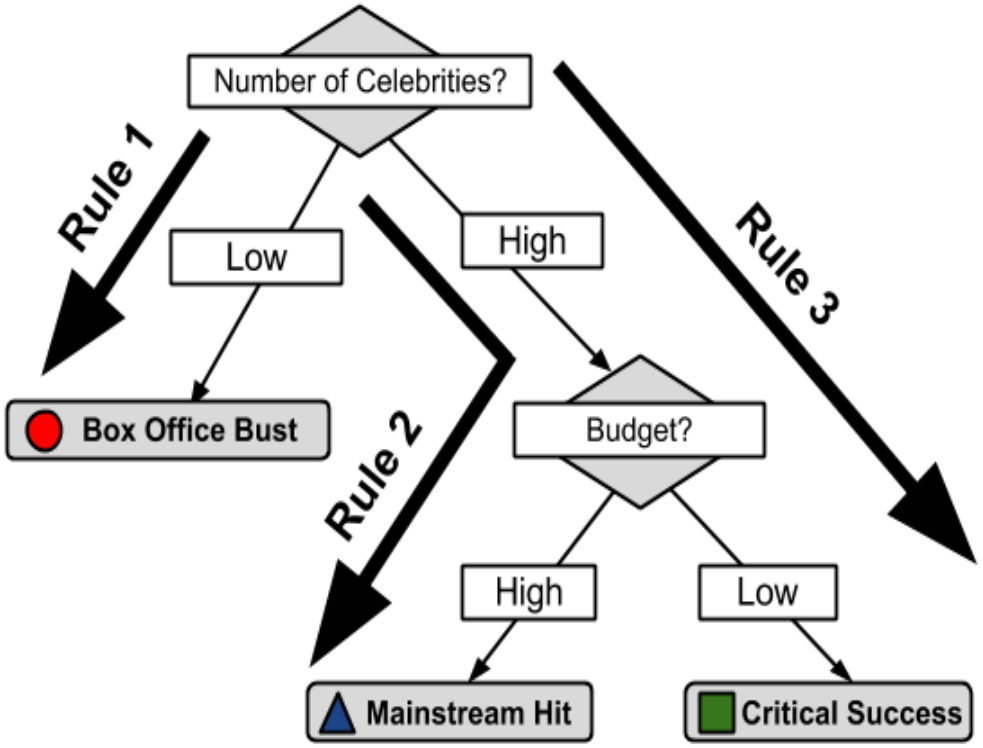
- For each feature, **1R divides the data into groups based on similar values of the feature**. Then, for each segment, the algorithm predicts the majority class.
- The **error rate for the rule based on each feature is calculated, and the rule with the fewest errors is chosen as the one rule.**

- For the Travels By feature, the data was divided into three groups: **Air, Land, and Sea.**
- Animals in the **Air and Sea groups were predicted to be non-mammal**, while
- animals in the **Land group were predicted to be mammals**. This resulted in two
- **errors: bats and frogs.** The **Has Fur feature divided animals into two groups. Those**
- **with fur were predicted to be mammals, while those without were not**. **Three errors**
- were counted: **pigs, elephants, and rhinos.** As the Travels By feature resulted in
- fewer errors, the 1R algorithm would return the following **"one rule" based on**
- **Travels By:**

- If the animal travels by air, it is not a mammal
- If the animal travels by land, it is a mammal
- If the animal travels by sea, it is not a mammal
- The algorithm stops here, **having found the single most important rule.**

Rules from decision trees

- Classification rules can also be obtained directly from decision trees. Beginning at a leaf node and following the branches back to the root, you will have obtained a series of decisions.
- These can be combined into a single rule. The following figure shows how rules could be constructed from the decision tree for predicting movie success:



- Following the paths from the root to each leaf, the rules would be:
- 1. If the number of celebrities is low, then the movie will be a Box Office Bust.
- 2. If the number of celebrities is high and the budget is high, then the movie will be a Mainstream Hit.
- 3. If the number of celebrities is high and the budget is low, then the movie will be a Critical Success.

3.2 Consider the following set of training examples:

(a) What is the entropy of this collection of training examples with respect to the target function classification?

(b) What is the information gain of ***a2*** relative to these training examples?

Instance	Classification	a_1	a_2
1	+	T	T
2	+	T	T
3	−	T	F
4	+	F	F
5	−	F	T
6	−	F	T

Ans.

(a) Entropy = 1

(b) Gain(a2) = 1-4/6*1-2/6*1 = 0

- Entropy(s)=-3/6 log(3/6)-3/6(log(3/6)=1
- InfoGain(a2)=Entropy(s)-4/6(Entropy(T)-

2/6(Entropy=F)

= 1- 4/6- 2/6 =0

Examples -Entropy /Information Gain computations

- Problem 1
-

Credit Rating	Liability		
	Normal	High	Total
Excellent	3	1	4
Good	4	2	6
Poor	0	4	4
Total	7	7	14

$$E(\textit{Liability}) = -\frac{7}{14}\log_2\left(\frac{7}{14}\right) - \frac{7}{14}\log_2\left(\frac{7}{14}\right)$$

$$= -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)$$

$$= 1$$

$$E(\textit{Liability} \mid \textit{CR} = \textit{Excellent}) = -\frac{3}{4}\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) \approx 0.811$$

$$E(\textit{Liability} \mid \textit{CR} = \textit{Good}) = -\frac{4}{6}\log_2\left(\frac{4}{6}\right) - \frac{2}{6}\log_2\left(\frac{2}{6}\right) \approx 0.918$$

$$E(\textit{Liability} \mid \textit{CR} = \textit{Poor}) = -0\log_2(0) - \frac{4}{4}\log_2\left(\frac{4}{4}\right) = 0$$

Weighted Average:

$$E(\textit{Liability} \mid \textit{CR}) = \frac{4}{14} \times 0.811 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0$$

$$= 0.625$$

Inf ormation Gain:

$$IG(\mathit{Liability}, \mathit{CR}) = E(\mathit{Liability}) - E(\mathit{Liability} \mid \mathit{CR})$$

$$= 1 - 0.625$$

$$= 0.375$$