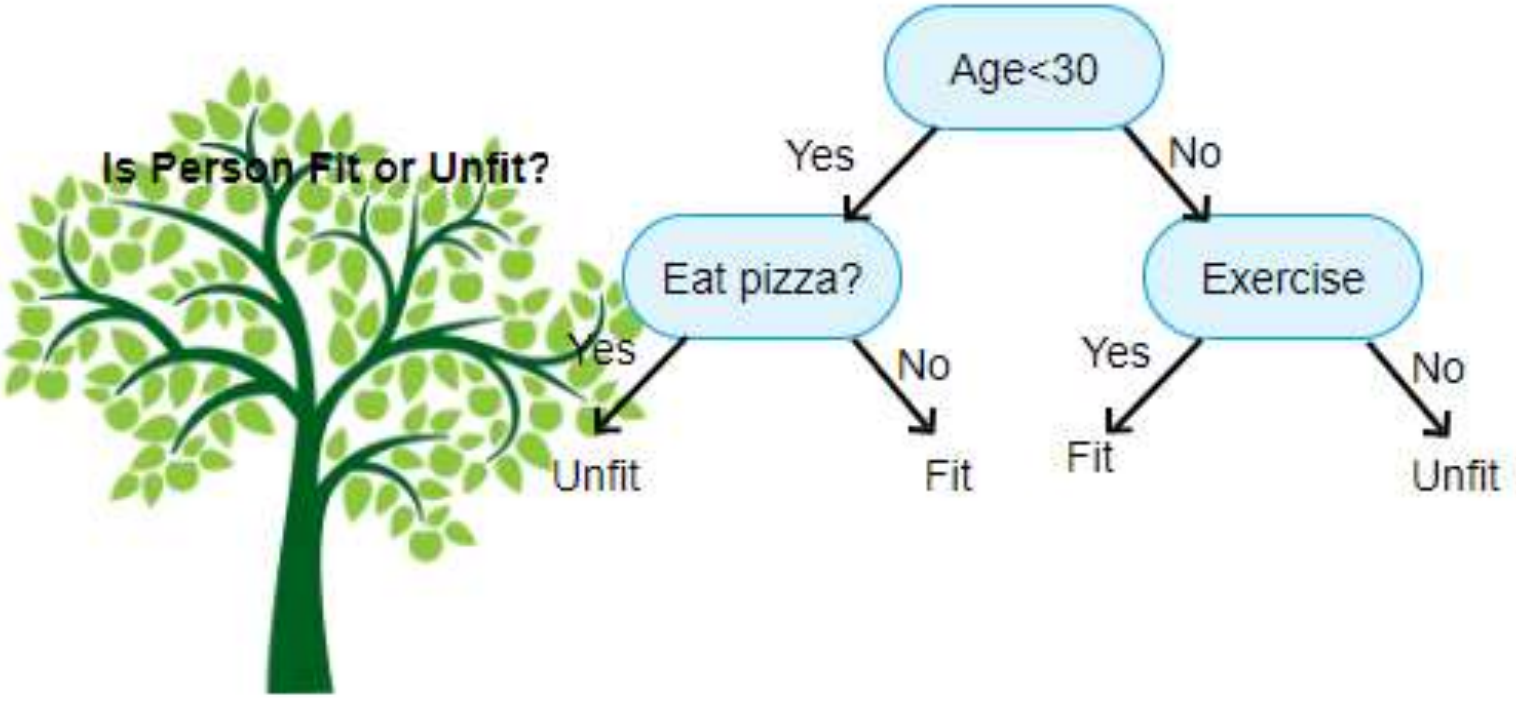
Module 3

Decision Tree Learning

Decision tree Model

- Decision Tree is a tree like predictive model used for regression and classification.
- It is a supervised learning model
- Commonly used decision tree algorithm are C5.0,IR and Ripper Algorithms.
- Decision trees start with a root node , then it passed through decision nodes. The splits into branches indicate decision’s choices. The tree terminated by leaf node that denote the result following a combinations of decisions.

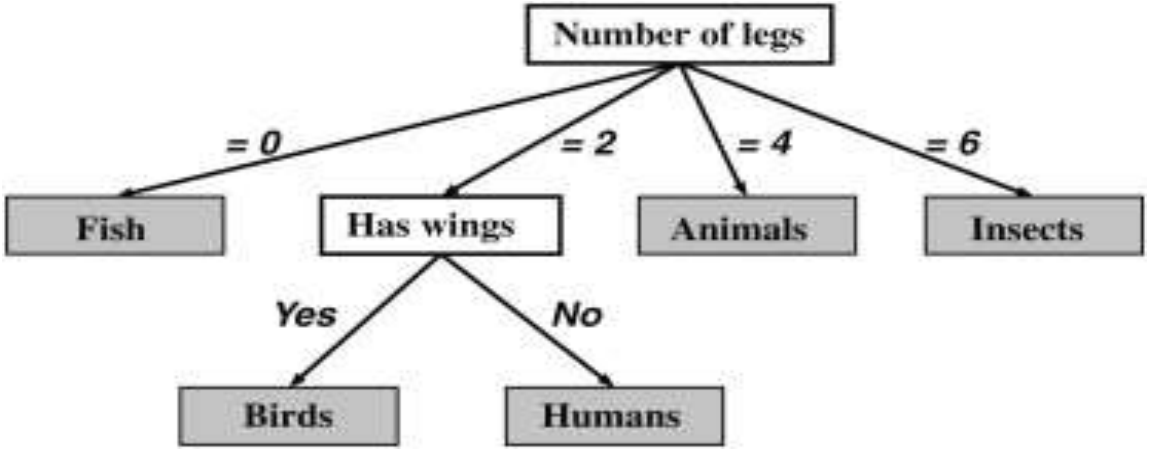

Is Person Fit or Unfit?
Age<30
Yes
No
Eat pizza?
Exercise
Yes
No
Yes
No
Unfit
Fit
Fit
Unfit

Applications

- **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 **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**

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.

Strengths:
An all purpose classifier that does well on most problems.
Highly automatic learning process, which can handle numeric or nominal features as well as missing data.
Excludes unimportant features.
Can be used on both small and large data sets.
Results in a model that can be interpreted with a mathematical background.
More efficient than other complex models.

Weakness:
Decision tree models are often biased towards split on features influences number of levels.
It is easy to overfit and underfit the model.
Small changes in the training data can result in large changes in decision logic.
Large tress can be difficult to interpret.

Choosing the best split

- At each level of decision tree construction an attribute in the dataset consider as a splitting attribute.
- Splitting is performed based on the value of this attribute.
- While constructing the decision tree main challenge is to find out this splitting attribute.
- Random selection is not a good idea, it may give bad results with low accuracy.
- Best splitting attribute is determine using information gain.

Entropy & Information gain

Entropy is the measure of impurity in a data set.

In a subset, if all the samples are in the same class, then it is called pure.

C5.0 uses entropy to quantifies the randomness within a set of data.

Decision tree hopes to find splits that reduce entropy, ultimately increasing homogeneity within the groups.

If there are only two possible classes entropy values can range from 0 to 1

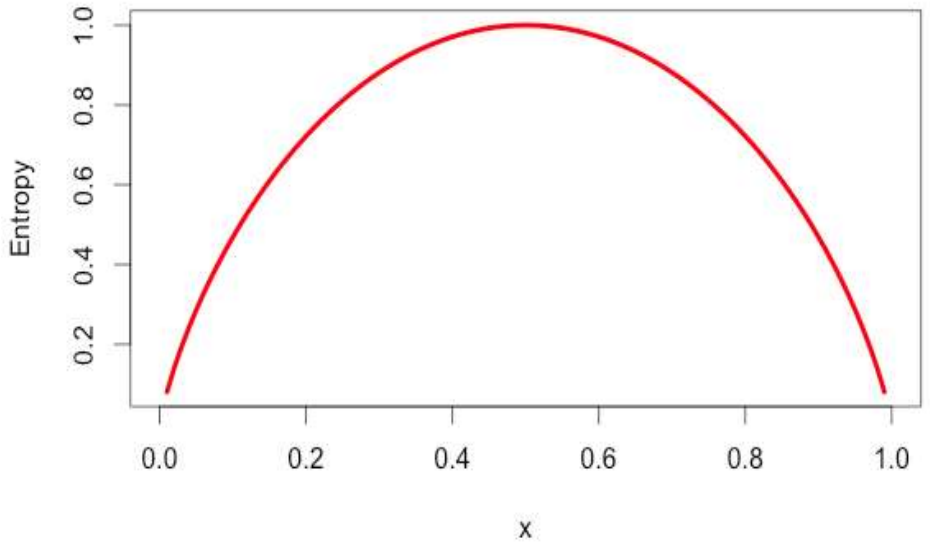
For n classes entropy ranges from 0 to $\log_2(n)$.

Minimum value indicate that the sample is completely homogeneous.

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

Entropy can be calculated as:
C-number of class levels
Pi-Proportion of values falling into class level I
S-subset of a dataset

X-Proportion of samples

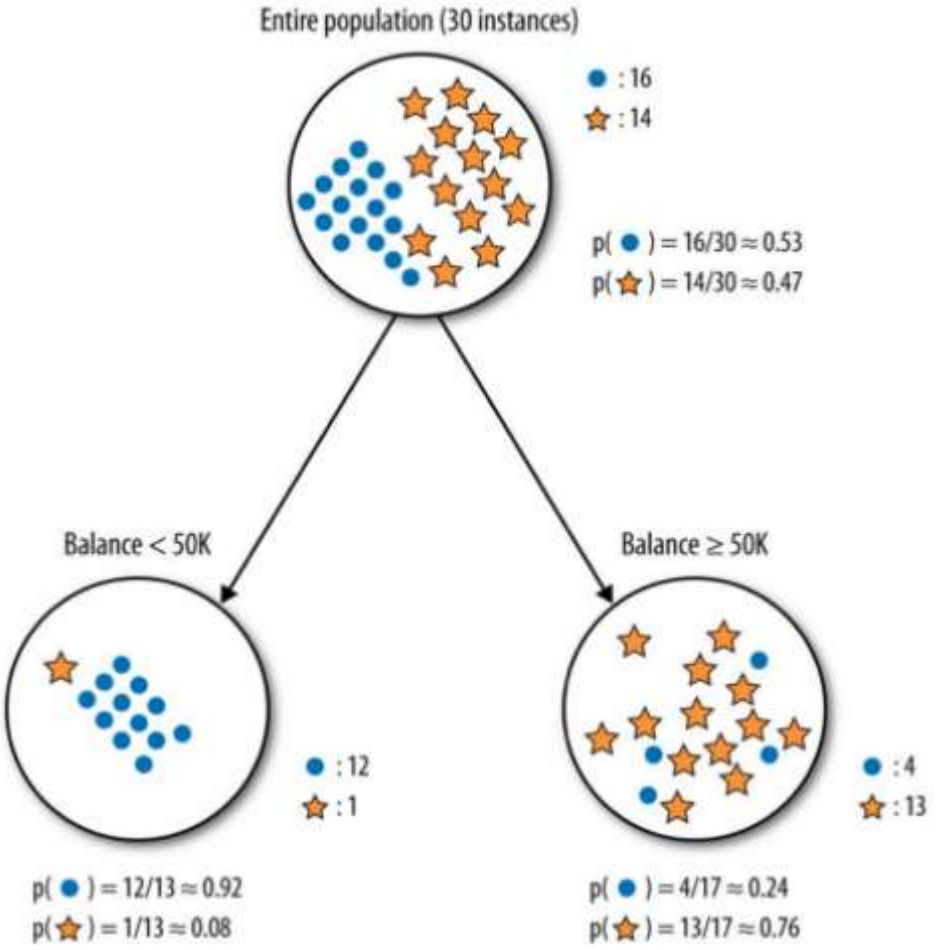


- As illustrated by the peak in entropy at $x = 0.50$, a 50-50 split results in the maximum entropy.
- Information gain is the measure of change in homogeneity or it is a measure of reduction in entropy.
- The information gain for a feature F is calculated as the difference between the entropy in the segment before the split ($S\ 1$), and the partitions resulting from the split ($S\ 2$):
- $\text{InfoGain}\ (\ F\ ) = \text{Entropy}\ (\ S\ 1\ ) - \text{Entropy}\ (\ S\ 2\ )$
- The higher the information gain the better a feature is at creating homogeneous group.

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



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

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

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

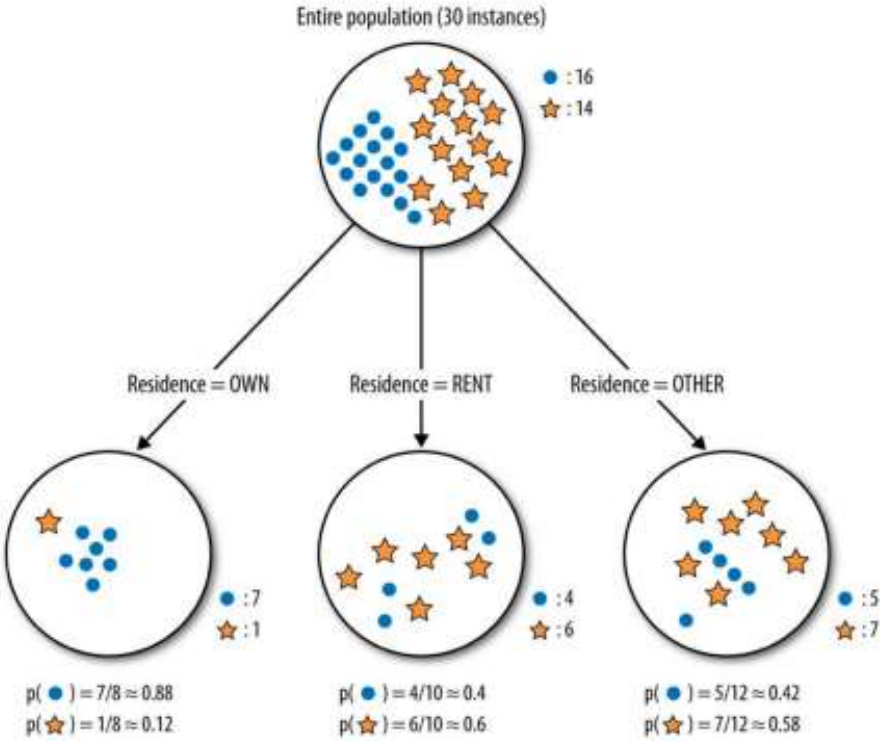
Weighted Average of entropy for each node:

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

Information Gain:

$$\begin{aligned} IG(Parent, Balance) &= E(Parent) - E(Balance) \\ &= 0.99 - 0.62 \\ &= \mathbf{0.37} \end{aligned}$$

Feature 2-Type of residence



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

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

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

Weighted Average of entropies for each node:

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

Information Gain:

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

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

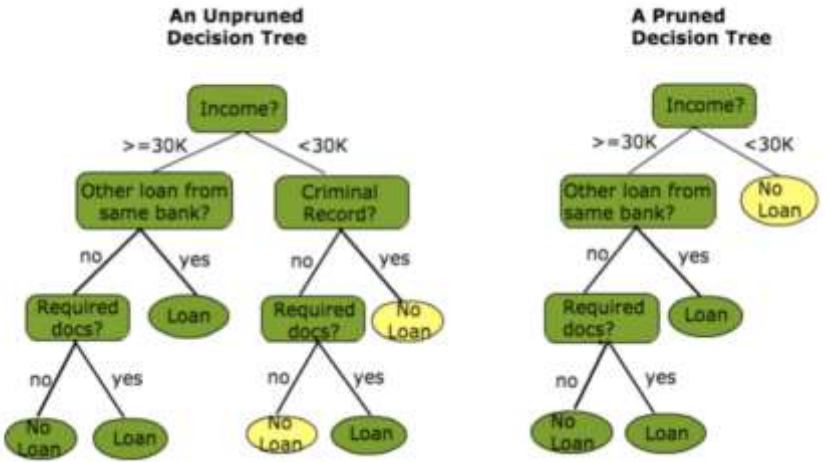
The main problem in decision tree is that ,if it has too many branches can result in overfitting of the training data.

Pruning a decision tree helps to prevent overfitting the training data so that model generalize well to unseen data.

Two types of pruning

-pre-pruning

-post pruning



Pre-pruning

Subtree construction is halted at a particular node after evaluation of some measures like information gain.

Pruning condition set based on some measures.

If the condition is satisfied prune the subtree ie replace the decision node with a leaf node.

Post pruning

Prune after the tree built.

Built the tree entirely using decision tree algorithm, then prune the subtree in the tree in a bottom up fashion.

- C5.0 follow post pruning strategy.
- 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

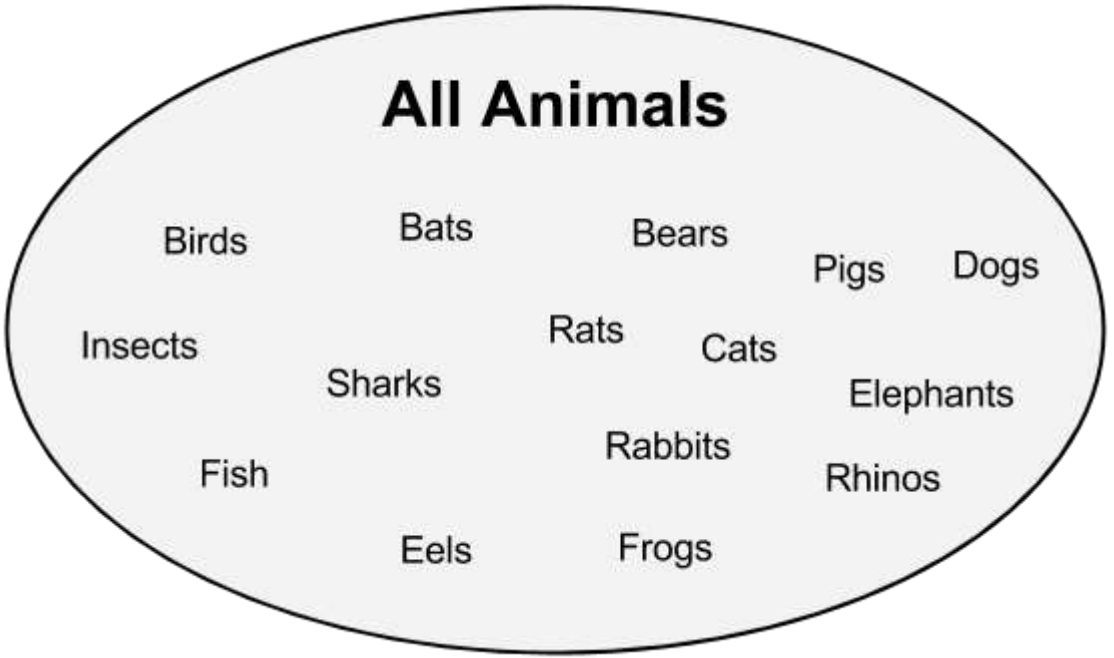
Classification Rule learning

- Classification rules represent knowledge in the form of logical if-else statements that assign a class to unlabelled 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.

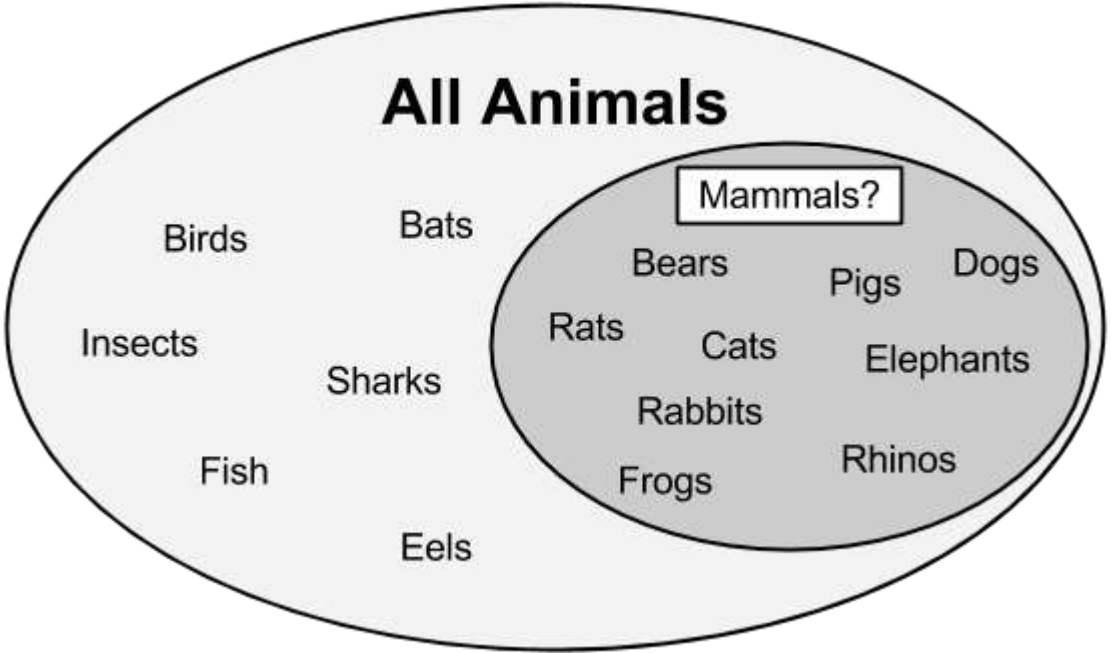
Separate and conquer

- 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.

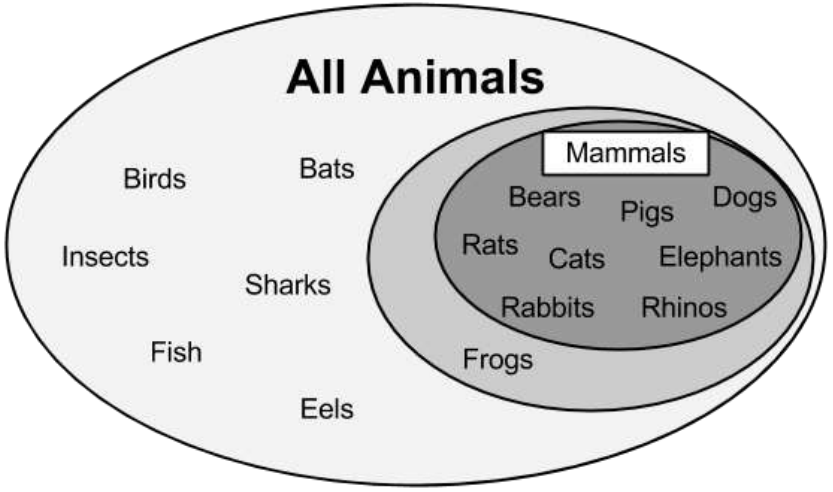
- Suppose creating rules for identifying whether or not an animal is a mammal. Depict the set of all animals as a large space, as shown in the following diagram:
-



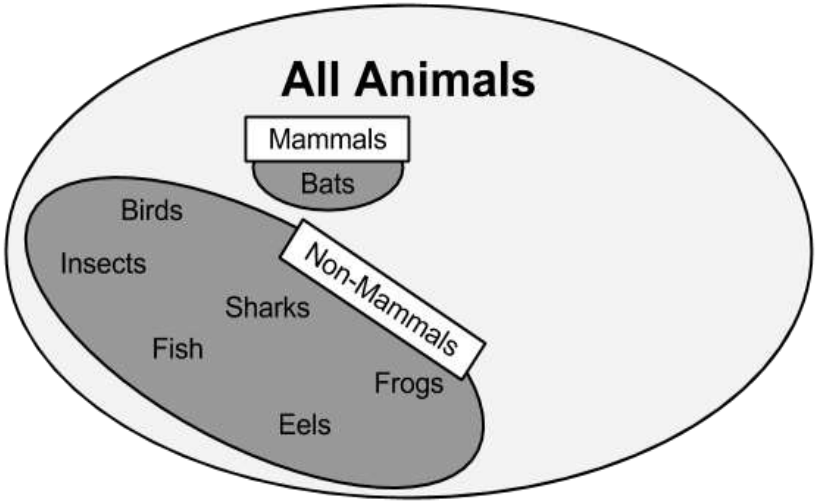
- A rule learner begins by using the available features to find homogeneous groups. For example, using a feature that measured whether the species travels via land, sea,or air, the first rule might suggest that any land-based animals are mammals:
-



- 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:



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. 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

Animal	Travels By	Has Fur	Mammal
Bats	Air	Yes	Yes
Bears	Land	Yes	Yes
Birds	Air	No	No
Cats	Land	Yes	Yes
Dogs	Land	Yes	Yes
Eels	Sea	No	No
Elephants	Land	No	Yes
Fish	Sea	No	No
Frogs	Land	No	No
Insects	Air	No	No
Pigs	Land	No	Yes
Rabbits	Land	Yes	Yes
Rats	Land	Yes	Yes
Rhinos	Land	No	Yes
Sharks	Sea	No	No

Full Dataset

Travels By	Predicted	Mammal	
Air	No	Yes	✖
Air	No	No	
Air	No	No	
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	No	✖
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	Yes	
Sea	No	No	
Sea	No	No	
Sea	No	No	

Rule for "Travels By"

Error Rate = 2 / 15

Has Fur	Predicted	Mammal	
No	No	No	
No	No	No	
No	No	Yes	✖
No	No	No	
No	No	No	
No	No	No	
No	No	Yes	✖
No	No	Yes	✖
No	No	No	
Yes	Yes	Yes	
Yes	Yes	Yes	
Yes	Yes	Yes	
Yes	Yes	Yes	
Yes	Yes	Yes	
Yes	Yes	Yes	

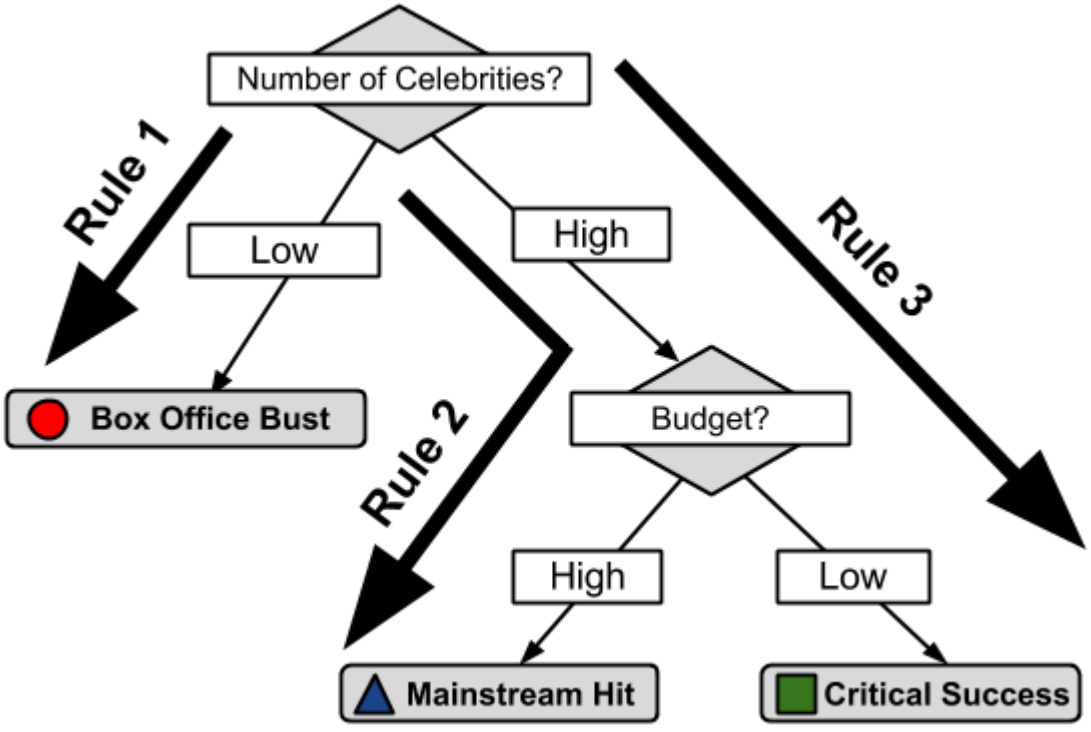
Rule for "Has Fur"

Error Rate = 3 / 15

- 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:


Number of Celebrities?
Rule 1
Low
High
Rule 3
Box Office Bust
Rule 2
Budget?
High
Low
Mainstream Hit
Critical 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.

Exercises

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:

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

$$= 1 - 0.625$$

$$= 0.375$$