Learning

By learning, agents improve their performance at future task. A weather forecast program may learn from past weather data to predict weather in future.

- Agents may learn by teacher, discover, and by examples.
- We will focus on learning by *example (inductive learning)*.

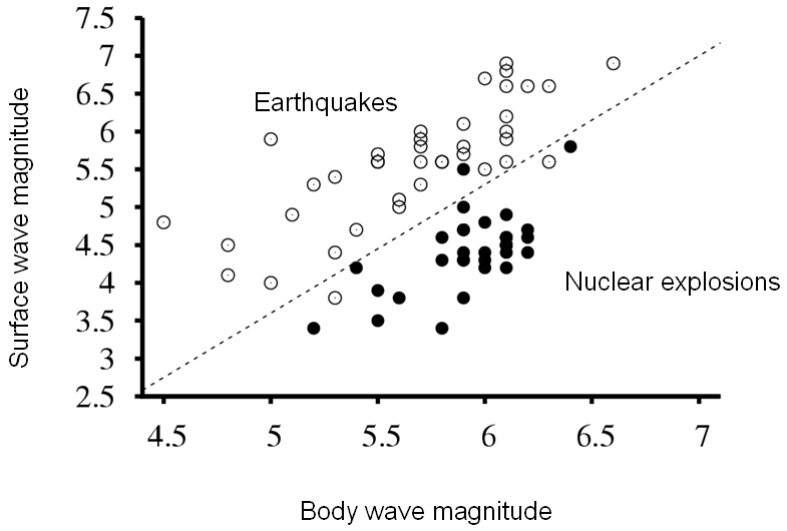
Learning types
• *Supervised learning*: correct answers for each example are given.
• *Unsupervised learning*: correct answers is not given
• *Reinforcement learning*: occasional positive and/or negative rewards.

Supervised Learning

Given training examples of inputs and corresponding outputs, produce the "correct" outputs for new inputs.

Two main scenarios:

1- *Classification*: outputs are *discrete* variables (category labels). Learn a decision boundary that separates one class from the other.

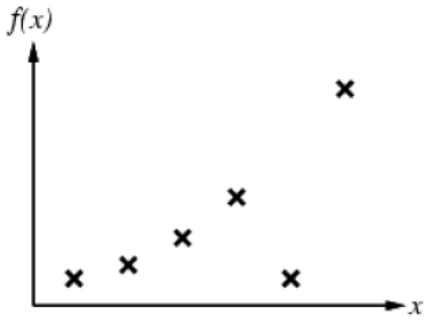


2- *Regression*: also known as "curve fitting" or "function approximation." Learn a *continuous* input-output mapping from examples (possibly noisy)

$$y = f(x)$$

Learning: given a *training set* of example pairs $\{(\boldsymbol{x}_1, f(y_1)), \ldots, (\boldsymbol{x}_N, f(y_N))\}$, estimate hypothesis H such that $H \approx f$. i.e. recover the unknown function f.

Inference: apply H to a never before seen *test example* **x** and output the predicted value y = f(**x**)



Key challenge: *generalization* to unseen examples.

Example*: prediction:* predict future climate from current and past data

Classifiers

Learning Decision Trees (LDT)

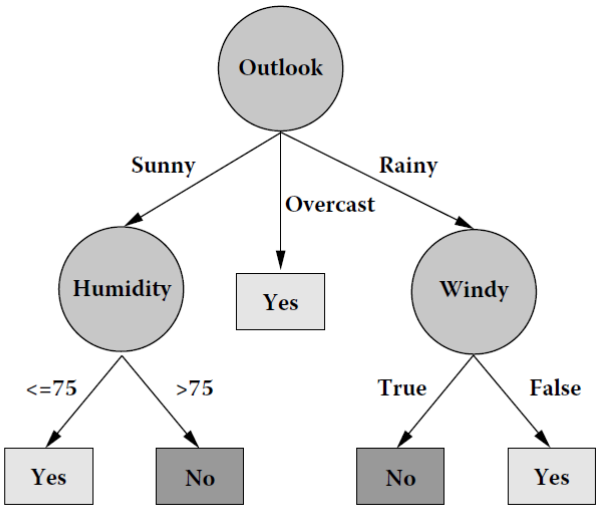
It is a supervised learning. Given an attribute-valued dataset where *examples* are described by collections of *attributes* and belong to one of a set of *classes*, LDT learns a mapping from attribute values to classes that can be applied to classify new, unseen instances.

Example: decide whether to play golf, based on the following attributes:

1- Outlook (a ternary-valued random variable),
2- Temperature (continuous-valued),
3- Humidity (also continuous-valued), and
4- Windy (binary).
   - Classification is: Yes or NO.

Day	Outlook	Temperature	Humidity	Windy	Play Golf?
1	Sunny	85	85	False	No
2	Sunny	80	90	True	No
3	Overcast	83	78	False	Yes
4	Rainy	70	96	False	Yes
5	Rainy	68	80	False	Yes
6	Rainy	65	70	True	No
7	Overcast	64	65	True	Yes
8	Sunny	72	95	False	No
9	Sunny	69	70	False	Yes
10	Rainy	75	80	False	Yes
11	Sunny	75	70	True	Yes
12	Overcast	72	90	True	Yes
13	Overcast	81	75	False	Yes
14	Rainy	71	80	True	No

The decision tree of the above data is:

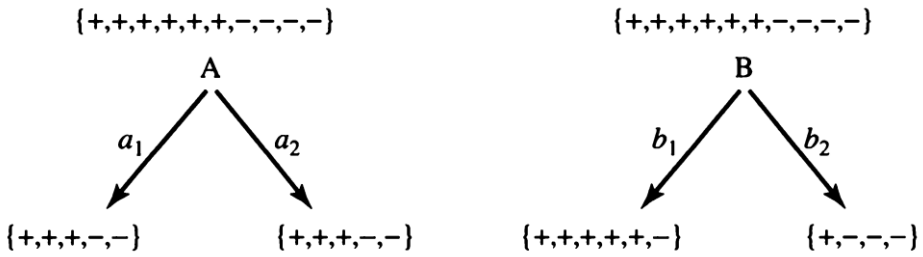


Basic algorithm for learning decision trees

1. starting with whole training data
2. select attribute that gives "best" split
3. create child nodes based on split
4. repeat on each child using child data until a stopping criterion is reached
   - all examples have same class
   - amount of data is too small
   - tree too large

Central problem: How do we choose the "best" attribute?

The following example shows that attribute B is better than A. because B splits the examples set into much purer subsets than A.



We select the attribute of the maximum Gain(attribute) to be the best attribute.

$$Gain(attribute) = entropy(Class\ in\ D) - \sum_{v} \frac{|D_v|}{D} entropy(Class\ in\ D_v)$$

Where *entropy(Class in D)* is the entropy of the class *Class* in the dataset *D*, |.| is the size of dataset, *v* is the values of the attribute.

Class *PlayGolf*? variable takes two values with probability 9/14 (for "Yes") and 5/14 (for "No"). The entropy of a class random variable that takes on *c* values with probabilities *p1, p2, . . . , pc* is given by:

$$\sum_{i=1}^{c} - p_i \log(p_i)$$

The entropy of *PlayGolf*? thus is:

$$-(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.940$$

Gain(*Outlook*) = 0.940−0.694 = 0.246.

Gain(*Windy*) = 0.940−0.892 = 0.048

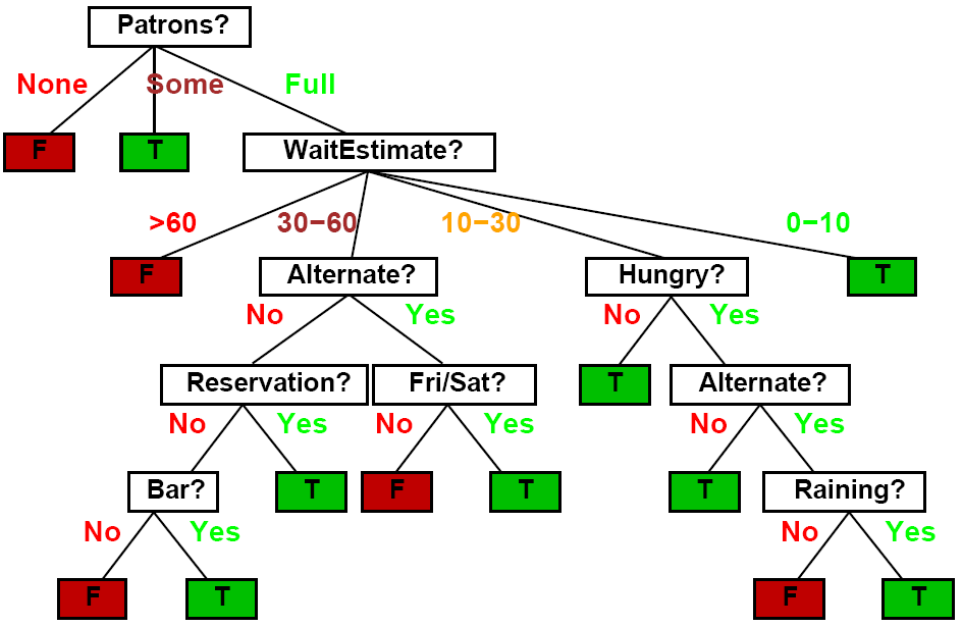
Thus we choose (*Outlook*) first to split the tree.

For the example (Outlook= "sunny", Temperature="70", Humidity="63", Windy="true") the class PlayGolf?= "Yes".

Another example: decide whether to wait for a table at a restaurant, based on the following attributes:

1. **Alternate:** is there an alternative restaurant nearby?
2. **Bar:** is there a comfortable bar area to wait in?
3. **Fri/Sat:** is today Friday or Saturday?
4. **Hungry:** are we hungry?
5. **Patrons:** number of people in the restaurant (None, Some, Full)
6. **Price:** price range ($, $$, $$$)
7. **Raining:** is it raining outside?
8. **Reservation:** have we made a reservation?
9. **Type:** kind of restaurant (French, Italian, Thai, Burger)
10. **WaitEstimate:** estimated waiting time (0-10, 10-30, 30-60, >60)

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T



Na¨ıve Bayes classifier

- The naive Bayes classifier assigns an instance *Sk* with attribute values (*A*1=v1, *A*2=v2, …, *Am*=v*m*) to class *Ci* with maximum P(*Ci*| (v1, v2, …, v*m*)) for all *i*.
- The naive Bayes classifier exploits the *Bayes's rule* and assumes *independence* of attributes.
- Likelihood of *Sk* belonging to *Ci*

$$= \mathrm{P}(C_i \mid (\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_m)) = \frac{P((\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_m) \mid C_i) P(C_i)}{P((\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_m))}$$

- Likelihood of *sk* belonging to *Cj*

$$= \mathrm{P}(C_j \mid (\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_m)) = \frac{P((\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_m) \mid C_j) P(C_j)}{P((\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_m))}$$

- Therefore, when comparing P(*Ci*| (v1, v2, …, v*m*)) and P(*Cj* |(v1, v2, …, v*m*)), we only need to compute P((v1, v2, …, v*m*)|*Ci*)P(*Ci*) and P((v1, v2, …, v*m*)|*Cj*)P(*Cj*)
- Under the assumption of independent attributes

$$P((\mathrm{v}_1, \mathrm{v}_2, ..., \mathrm{v}_m)|C_j)$$
$$= P(A_1 = \mathrm{v}_1 | C_j) \cdot P(A_2 = \mathrm{v}_2 | C_j) \cdot \ldots \cdot P(A_m = \mathrm{v}_m | C_j)$$
$$= \prod_{h=1}^{m} P(A_h = \mathrm{v}_h | C_j)$$

- Furthermore, *P*(*Cj*) can be computed by

$$\frac{\text{number of training samples belonging to } C_j}{\text{total number of training samples}}$$

Example:

The weather data, with counts and probabilities												
outlook			temperature			humidity			windy		play	
	yes	no		yes	no		yes	no		yes	no	yes	no
sunny	2	3	hot	2	2	high	3	4	false	6	2	9	5
overcast	4	0	mild	4	2	normal	6	1	true	3	3		
rainy	3	2	cool	3	1								
sunny	2/9	3/5	hot	2/9	2/5	high	3/9	4/5	false	6/9	2/5	9/14	5/14
overcast	4/9	0/5	mild	4/9	2/5	normal	6/9	1/5	true	3/9	3/5		
rainy	3/9	2/5	cool	3/9	1/5								

Decide if we play or not for the following day:

outlook	temperature	humidity	windy	play
sunny	cool	high	true	?

- Likelihood of yes

$$= \frac{2}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{9}{14} = 0.0053$$

- Likelihood of no

$$= \frac{3}{5} \times \frac{1}{5} \times \frac{4}{5} \times \frac{3}{5} \times \frac{5}{14} = 0.0206$$

- Therefore, the prediction is No