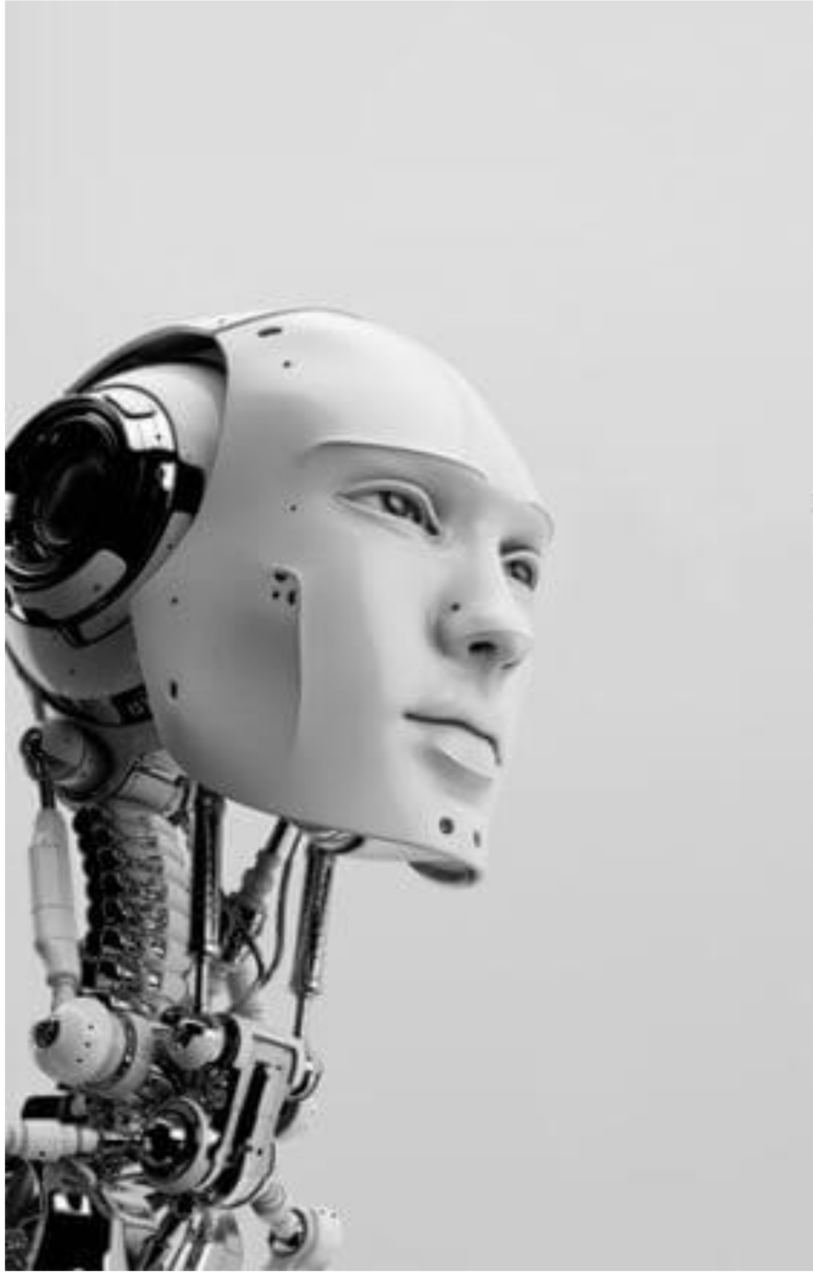


Artificial Intelligence

18 Learning from examples

Russell & Norvig, AI: A Modern Approach, 3rd Ed

Lec Zinia Sultana
CSE Dept, MIST

Outline

- ❖ Learning
- ❖ Forms of Learning
- ❖ Supervised Learning
- ❖ Leaning Decision Trees
- ❖ Artificial Neural Network
- ❖ Support Vector Machine

Learning

An agent is **learning** if it improves its performance on future tasks after making observations about the world.

In which we describe agents that can improve their behavior through diligent study of their own experiences.

Forms of Learning

Any component of an agent can be improved by learning from **data**.

The improvements depend on four major factors:

- ✓ Which *component* is to be improved.
- ✓ What prior knowledge the agent already has.
- ✓ What representation is used for the data and the component.
- ✓ What feedback is available to learn from.

Forms of Learning .. feedback

There are three types of feedback that determine the three main types of learning:

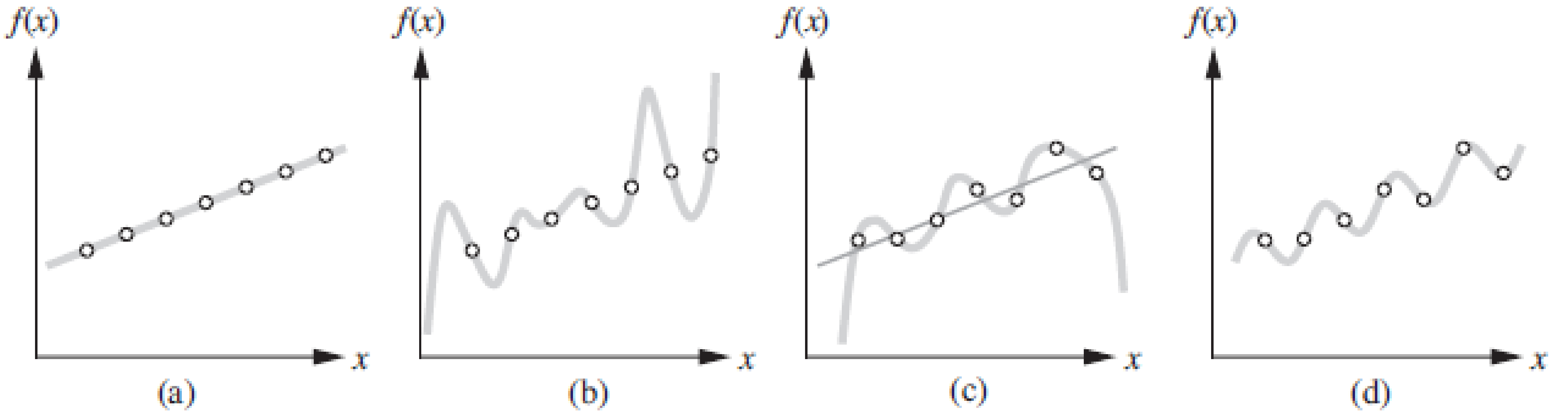
- Unsupervised learning
  - ✓ Agent learns patterns in the input even though no explicit feedback is supplied
  - ✓ Clustering – concept of “good traffic days” and “bad traffic days
- Reinforcement learning
  - ✓ Agent learns from a series of reinforcements—rewards or punishments
  - ✓ Getting tips for taxi agent, wining point for chess game
- Supervised learning
  - ✓ Agent observes some example input–output pairs and learns a function that maps from input to output
  - ✓ Semi-supervised learning - a few labeled examples and a large collection of unlabeled examples

Supervised learning

- **Training Set –** a set of N example input–output pairs
- **Hypothesis –** a possible function for generating output
- **Test Set –** a set of examples that are distinct from the training set
- **Learning –** a search through the space of possible hypotheses for one that will perform well

Supervised learning

❖ **Hypothesis** – a possible function for generating output



Supervised learning . .

- **Classification** – When the output y is one of a finite set of values (such as sunny, cloudy or rainy), the learning problem is called classification, and is called Boolean or binary classification if there are only two values.

- **Regression** – When y is a number (such as tomorrow's temperature), the learning problem is called regression.

Learning decision trees

- One of the simplest and most successful forms of machine learning.

- A decision tree represents a function that takes as input a vector of attribute values and returns a "decision"—a single output value

- The input and output values can be discrete or continuous

Learning decision trees . .

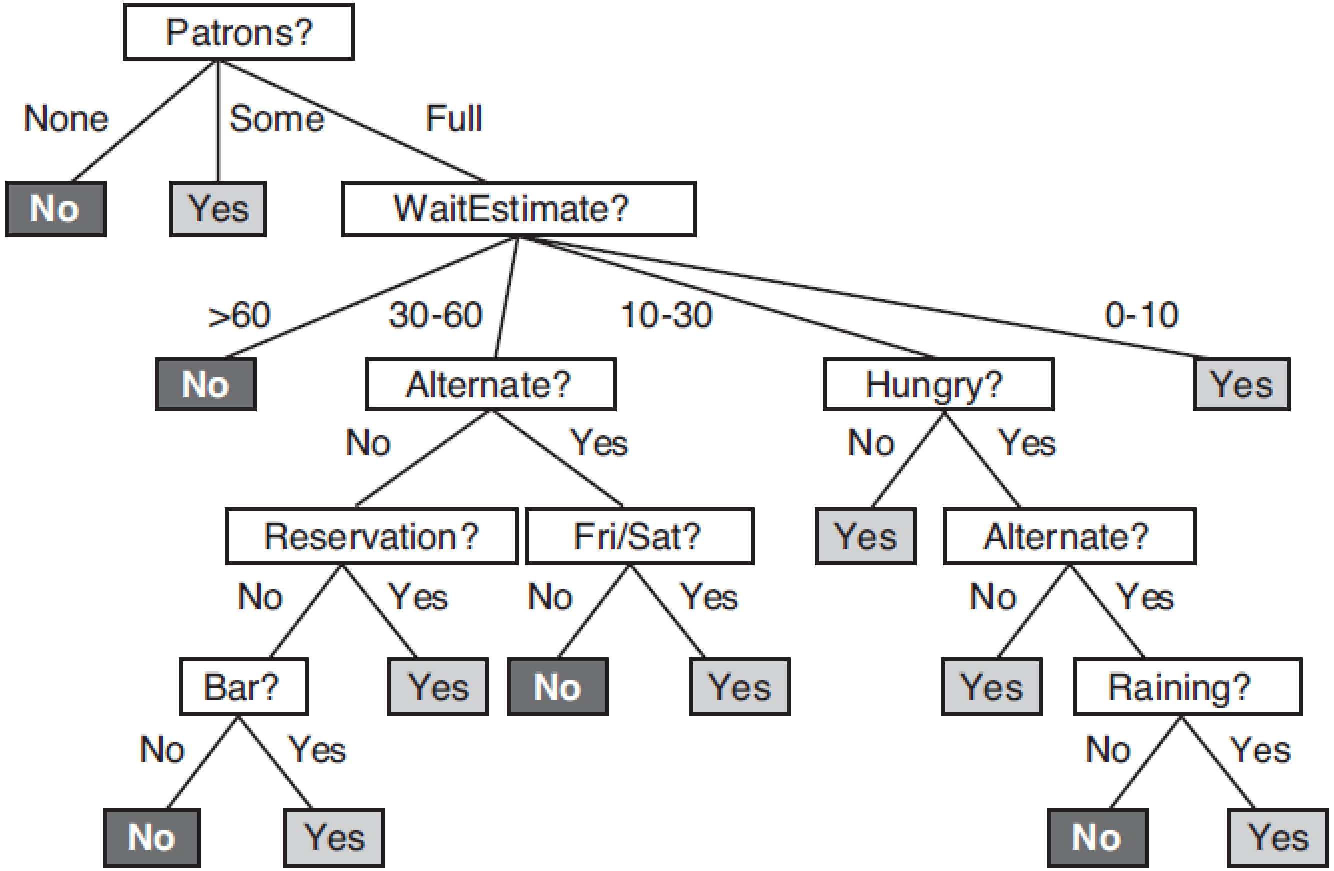
- A decision tree reaches its decision by performing a sequence of tests.
- Each internal node in the tree corresponds to a test of the value of one of the input attributes and
- the branches from the node are labeled with the possible values of the attribute

Learning decision trees . . .

❖ A decision tree to decide whether to wait for a table at a restaurant or not.

Boolean classification – the inputs have discrete values and the output has exactly two possible values.

– Each example input will be classified as true (a **positive** example) or false (a **negative** example).



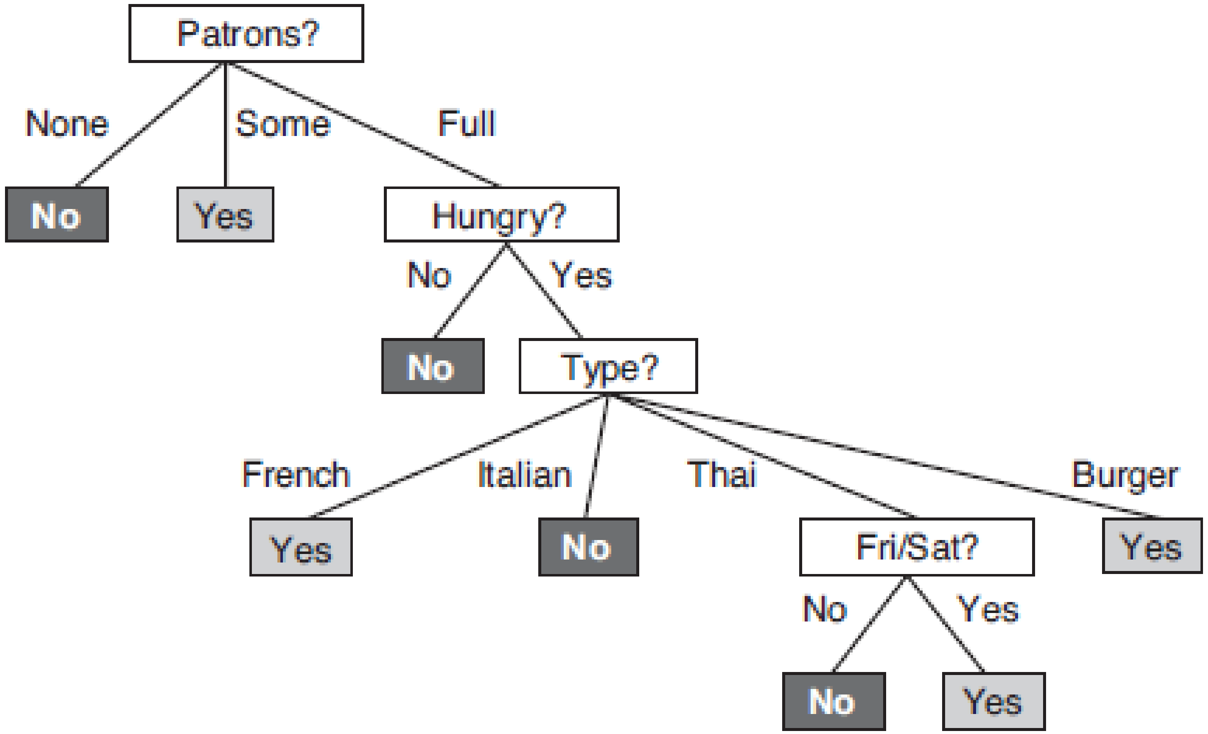
Learning decision trees . . . .

❖ A decision tree to decide whether to wait for a table at a restaurant or not.

Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$

Learning decision trees . . . . .

❖ A decision tree to decide whether to wait for a table at a restaurant or not.



Learning decision trees . . .

There are four cases to consider:

1. If the remaining examples are all positive (or all negative), then we are done: we can answer Yes or No.
2. If there are some positive and some negative examples, then choose the best attribute to split them.
3. If there are no examples left, it means that no example has been observed for this combination of attribute values, and we return a default value calculated from the plurality classification of all the examples that were used in constructing the node's parent. These are passed along in the variable parent examples.
4. If there are no attributes left, but both positive and negative examples, it means that these examples have exactly the same description, but different classifications. The best we can do is return the plurality classification of the remaining examples.

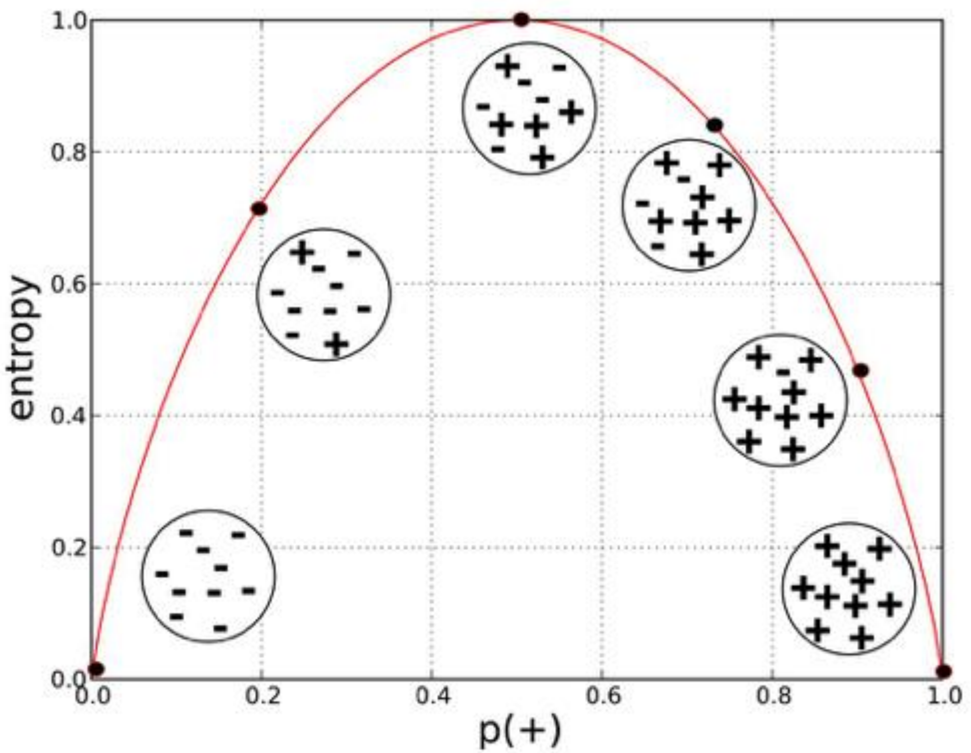
Decision-Tree Learning Algorithm

function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** a tree

if *examples* is empty **then return** PLURALITY-VALUE(*parent_examples*)
else if all *examples* have the same classification **then return** the classification
else if *attributes* is empty **then return** PLURALITY-VALUE(*examples*)
else
    $A \leftarrow \mathrm{argmax}_{a \in attributes}$ IMPORTANCE(a, *examples*)
    tree ← a new decision tree with root test A
    for each value v_k of A **do**
        $exs \leftarrow \{e : e \in examples$ **and** $e.A = v_k\}$
        subtree ← DECISION-TREE-LEARNING(*exs*, *attributes* − A, *examples*)
        add a branch to *tree* with label ($A = v_k$) and subtree *subtree*
    return *tree*

Choosing Best Attribute: Entropy

- Entropy is a measure of the **uncertainty** of a random variable
- Entropy is nothing but the measure of **disorder**
- In general, the entropy of a random variable V with values v_k, each with probability $P(v_k)$, is defined as

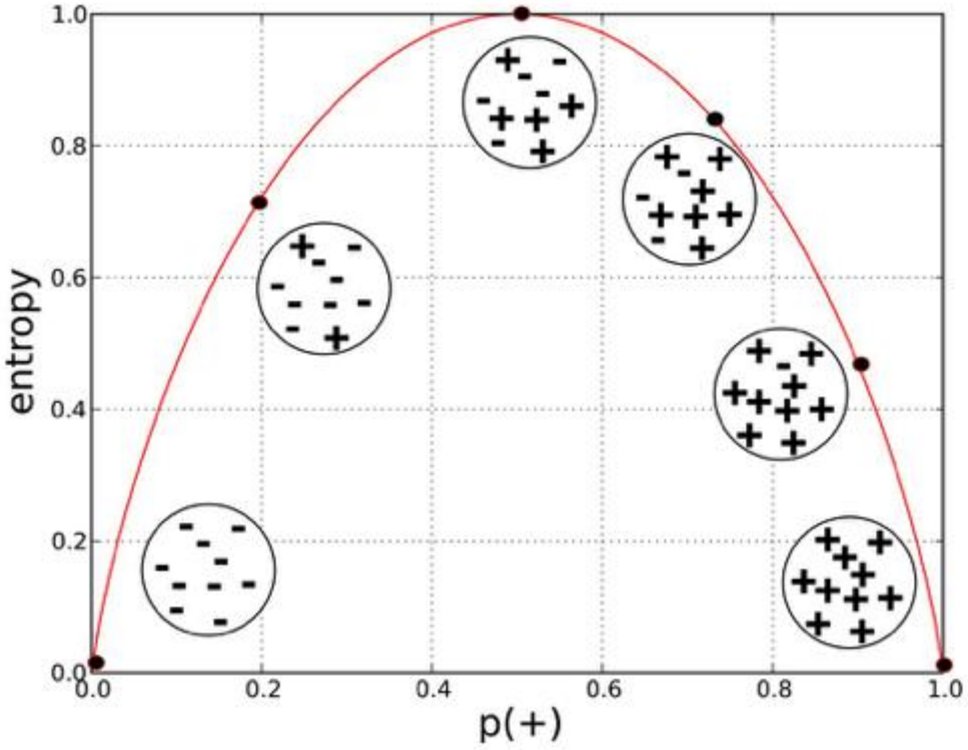


- Entropy:

$$\mathbf{H(V) = E(V)} = \sum_k P(v_k)\, log_2 \frac{1}{P(v_k)} = -\sum_k P(v_k)\, log_2\, P(v_k)$$

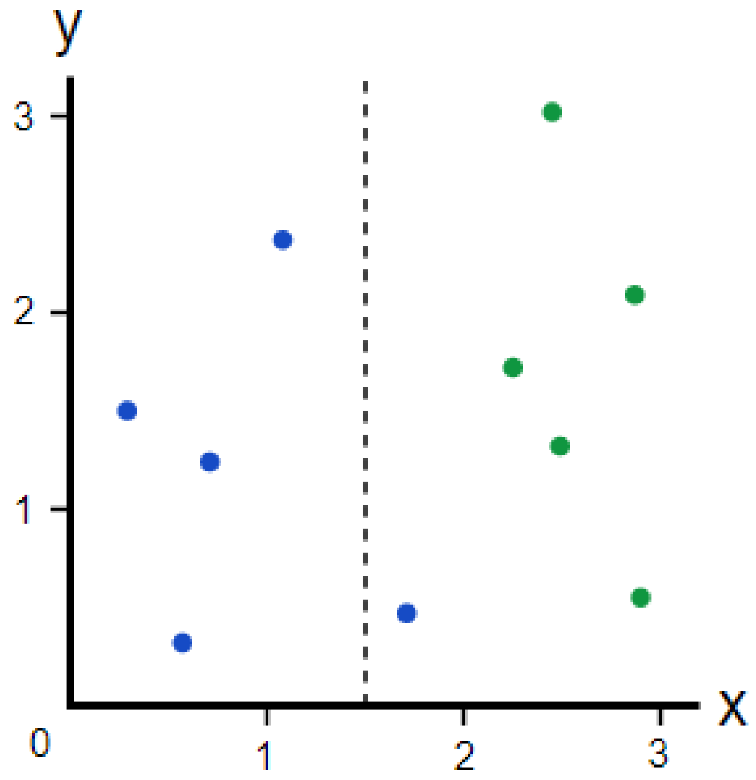
Choosing Best Attribute: Entropy

$$\mathrm{H(V)} = \mathrm{E(V)} = \sum_k P(v_k)\, log_2 \frac{1}{P(v_k)} = -\sum_k P(v_k)\, log_2\, P(v_k)$$



Choosing Best Attribute: Information Gain

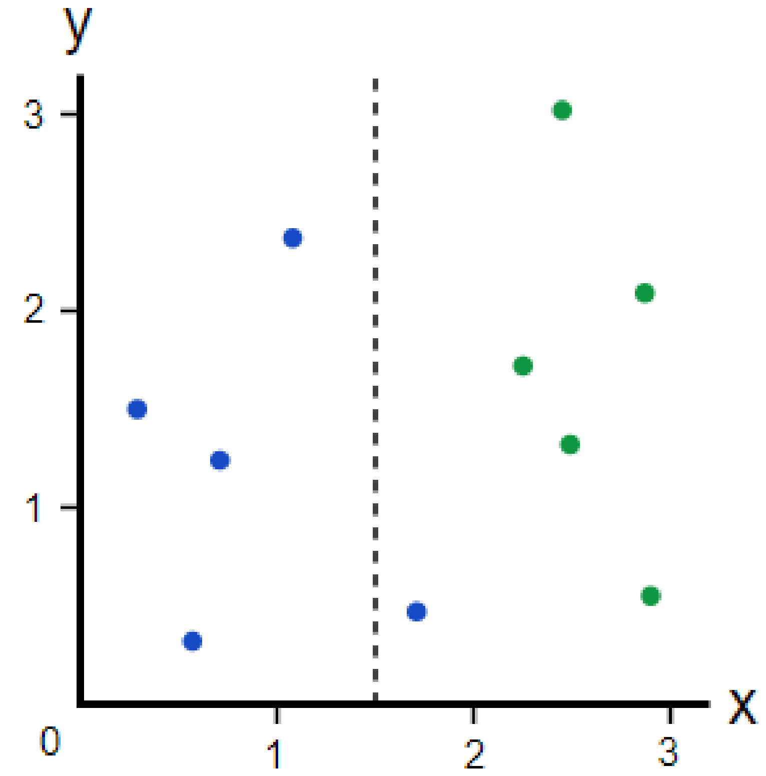
- The metrics to measure the quality of a split
- Reduction/decrease of entropy after a dataset is split on an attribute
- Information Gain is calculated for a split by the difference between the entropy of parent node and weighted average entropy of child nodes
- When training a Decision Tree using these metrics, the best split is chosen by maximizing Information Gain.



Information Gain: IG(S, A) = E(S) – E(S, A) = E(S) – $\sum_i P(A_i) * E(A_i)$

Choosing Best Attribute: Information Gain

Information Gain: IG(S, A) = E(S) – E(S, A) = E(S) – $\sum_i P(A_i) * E(A_i)$



Simulation – Decision Tree

Information Gain: IG(S, A) = E(S) – E(S, A) = E(S) – $\sum_i P(A_i) * E(A_i)$

Entropy: $\mathbf{E(V) = -\sum_k P(v_k)\, log_2\, P(v_k)}$

Sample	Status	Department	Room Size	Place a Bin?
D1	Faculty	CSE	Large	Yes
D2	Faculty	CSE	Large	Yes
D3	Staff	EECE	Medium	No
D4	Staff	CSE	Small	No
D5	Student	CSE	Small	Yes
D6	Student	EECE	Medium	Yes
D7	Faculty	EECE	Large	No
D8	Staff	CSE	Medium	No

Yes = 4
No = 4

Total= 8

Simulation – Decision Tree

Sample	Status	Department	Room Size	Bin?
D1	**Faculty**	CSE	Large	**Yes**
D2	**Faculty**	CSE	Large	**Yes**
D3	**Staff**	EECE	Medium	No
D4	**Staff**	CSE	Small	No
D5	**Student**	CSE	Small	**Yes**
D6	**Student**	EECE	Medium	**Yes**
D7	**Faculty**	EECE	Large	No
D8	**Staff**	CSE	Medium	No

Yes = 4+
No = 4-

Total= 8

Status	Faculty	Yes: 2+ No: 1-
	Staff	Yes: 0+ No: 3-
	Student	Yes: 2+ No: 0-

$$E(Faculty) = -\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3} = 0.92$$

$$E(Staff) = -\frac{0}{3}\log_2\frac{0}{3} - \frac{3}{3}\log_2\frac{3}{3} = 0$$

$$E(Student) = -\frac{2}{2}\log_2\frac{2}{2} - \frac{0}{2}\log_2\frac{0}{2} = 0$$

$$Weighted\ Avg\ E(Status) = \frac{3}{8}\times 0.92 + \frac{3}{8}\times 0 + \frac{2}{8}\times 0 = 0.35$$

Simulation – Decision Tree

Sample	Status	Department	Room Size	Bin?
D1	Faculty	**CSE**	Large	**Yes**
D2	Faculty	**CSE**	Large	**Yes**
D3	Staff	**EECE**	Medium	No
D4	Staff	**CSE**	Small	No
D5	Student	**CSE**	Small	**Yes**
D6	Student	**EECE**	Medium	**Yes**
D7	Faculty	**EECE**	Large	No
D8	Staff	**CSE**	Medium	No

Yes = 4+
No = 4-

Total= 8

Department	CSE	Yes: 3+ No: 2-
	EECE	Yes: 1+ No: 2-

$$E(CSE) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.97$$

$$E(EECE) = -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} = 0.92$$

$$Weighted\ Avg\ E(Department) = \frac{5}{8}\times 0.97 + \frac{3}{8}\times 0.92 = 0.95$$

Simulation – Decision Tree

Sample	Status	Department	Room Size	Bin?
D1	Faculty	CSE	**Large**	**Yes**
D2	Faculty	CSE	**Large**	**Yes**
D3	Staff	EECE	**Medium**	No
D4	Staff	CSE	**Small**	No
D5	Student	CSE	**Small**	**Yes**
D6	Student	EECE	**Medium**	**Yes**
D7	Faculty	EECE	**Large**	No
D8	Staff	CSE	**Medium**	No

Yes = 4+
No = 4-

Total= 8

Room Size	Large	Yes: 2+ No: 1-
	Medium	Yes: 1+ No: 2-
	Small	Yes: 1+ No: 1-

$$E(Large) = -\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3} = 0.92$$

$$E(Medium) = -\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3} = 0.92$$

$$E(Small) = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1$$

$$Weighted\ Avg\ E(RoomSize) = \frac{3}{8}\times 0.92 + \frac{3}{8}\times 0.92 + \frac{2}{8}\times 1 = 0.94$$

Simulation – Decision Tree

Sample	Status	Department	Room Size	Bin?
D1	Faculty	CSE	Large	Yes
D2	Faculty	CSE	Large	Yes
D3	Staff	EECE	Medium	No
D4	Staff	CSE	Small	No
D5	Student	CSE	Small	Yes
D6	Student	EECE	Medium	Yes
D7	Faculty	EECE	Large	No
D8	Staff	CSE	Medium	No

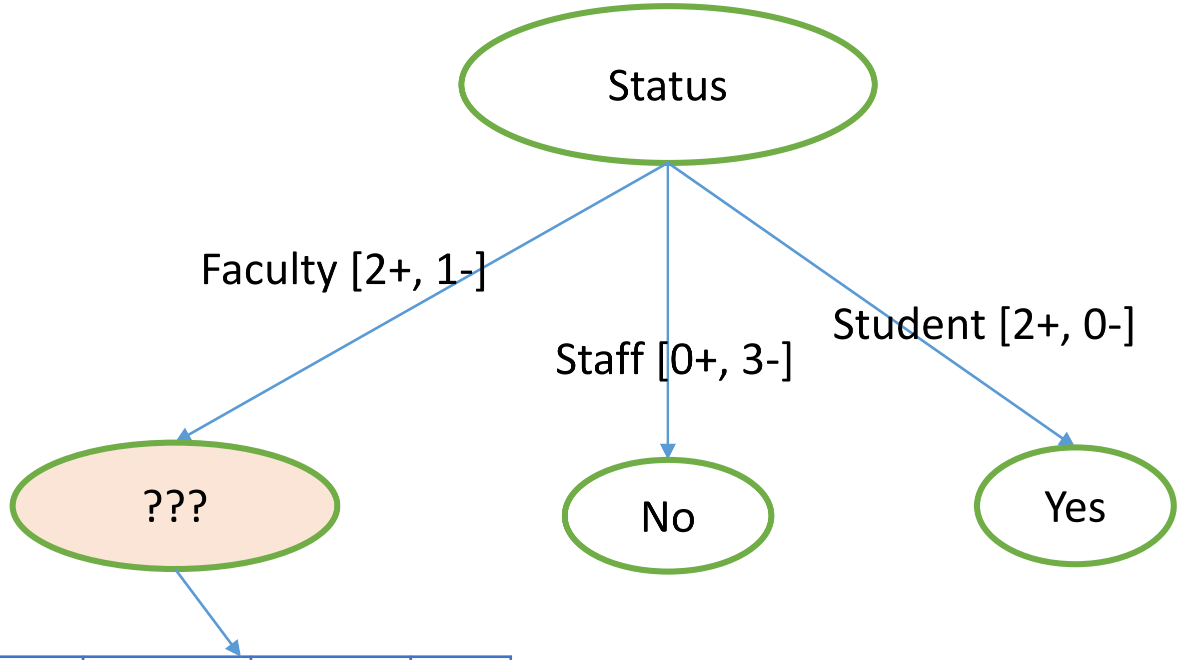
Attribute	Entropy	Information Gain
Status	.35	1-0.35=0.65
Department	0.95	1-0.95=0.05
Room Size	0.94	1-0.94=0.06

Yes = 4+
No = 4-

Total= 8

Best Attribute

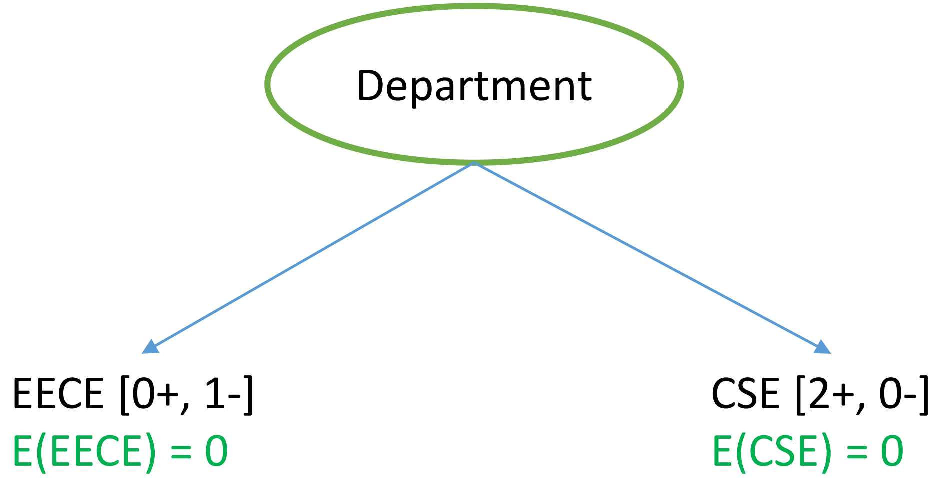
Simulation – Decision Tree



Sample	Department	Room Size	Bin ?
D1	CSE	Large	Yes
D2	CSE	Large	Yes
D7	EECE	Large	No

Sample	Status	Department	Room Size	Bin?
D1	Faculty	CSE	Large	Yes
D2	Faculty	CSE	Large	Yes
D3	Staff	EECE	Medium	No
D4	Staff	CSE	Small	No
D5	Student	CSE	Small	Yes
D6	Student	EECE	Medium	Yes
D7	Faculty	EECE	Large	No
D8	Staff	CSE	Medium	No

Simulation – Decision Tree

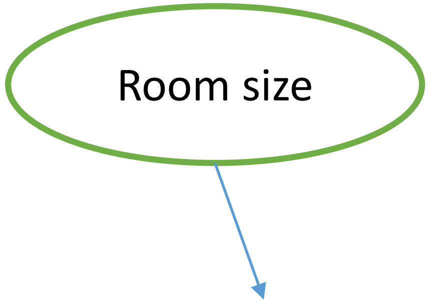


$Weighted\ Avg$ E(Department) = 0
Gain = 0.92 – 0 = 0.92

Sample	Department	Room Size	Bin ?
D1	CSE	Large	Yes
D2	CSE	Large	Yes
D7	EECE	Large	No

Yes = 2
No = 1
Entropy = 0.92

Simulation – Decision Tree



Large[2+, 1-]
E(Large) = 0.92

Sample	Department	Room Size	Bin ?
D1	CSE	Large	Yes
D2	CSE	Large	Yes
D7	EECE	Large	No

Yes = 2
No = 1
Entropy = 0.92

$Weighted\ Avg$ E(Room Size) = 0.92
Gain = 0.92 – 0.92 = 0

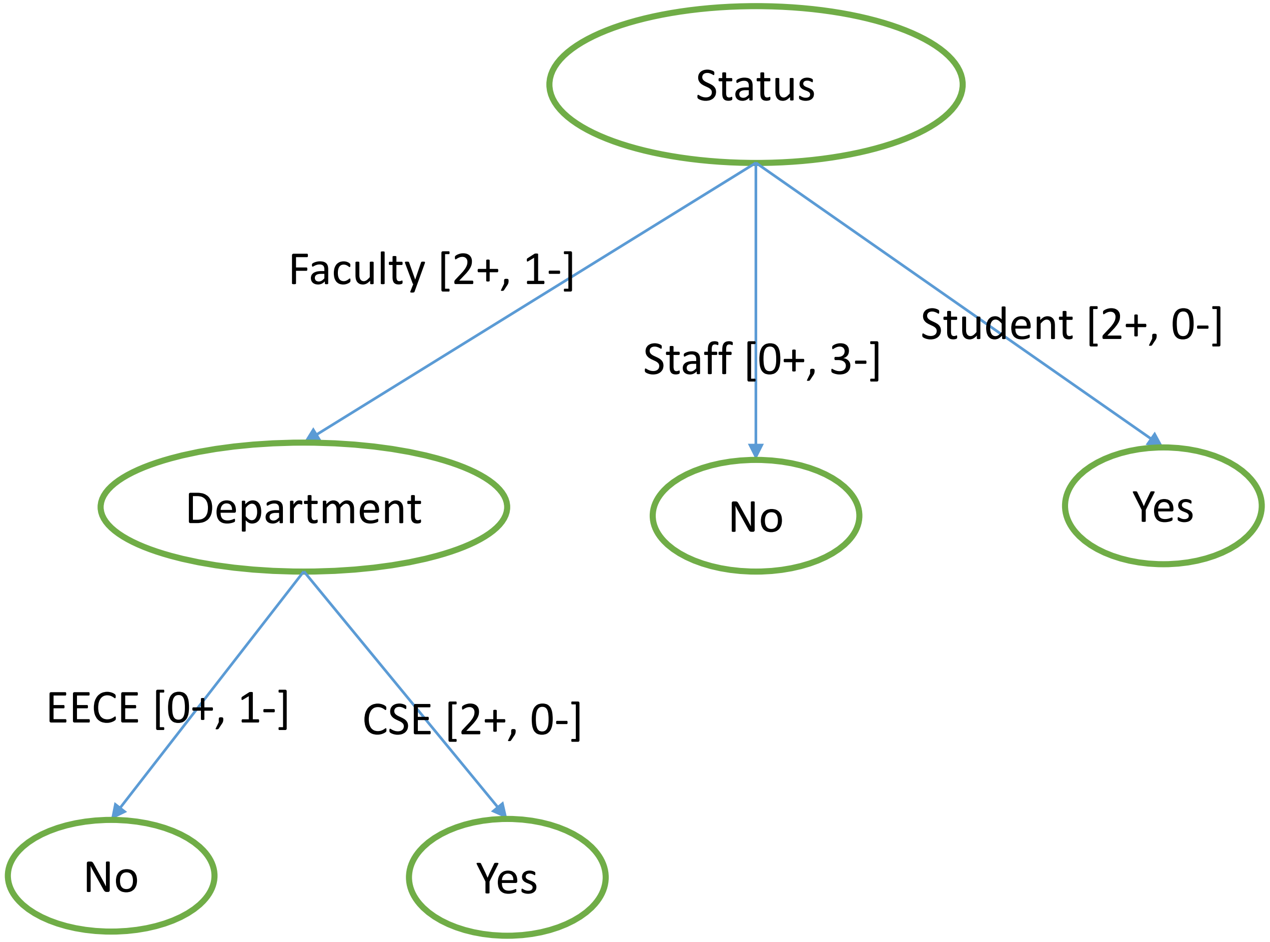
Simulation – Decision Tree

Sample	Department	Room Size	Bin?
D1	CSE	Large	Yes
D2	CSE	Large	Yes
D7	EECE	Large	No

Attribute	Entropy	Information Gain
Department	0	0.92-0=0.92
Room Size	0.92	0.92-0.92=0

Best Attribute

Simulation – Decision Tree



Sample	Status	Department	Room Size	Bin?
D1	Faculty	CSE	Large	Yes
D2	Faculty	CSE	Large	Yes
D3	Staff	EECE	Medium	No
D4	Staff	CSE	Small	No
D5	Student	CSE	Small	Yes
D6	Student	EECE	Medium	Yes
D7	Faculty	EECE	Large	No
D8	Staff	CSE	Medium	No

Broadening the applicability of decision trees

- Missing data
- Multivalued attributes
- Continuous and integer-valued input attributes
- Continuous-valued output attributes

Evaluating and Choosing The Best Hypothesis

- Holdout cross-validation
- K-fold cross-validation
- Leave-one-out cross-validation (LOOCV)

Artificial Neural Network

Input layer: Number of neurons in this layer corresponds to the number of inputs to the neuronal network.

Hidden layer: This layer has arbitrary number of layers with arbitrary number of neurons.

Output layer: The number of neurons in the output layer corresponds to the number of the output values of the neural network.

Artificial Neural Network

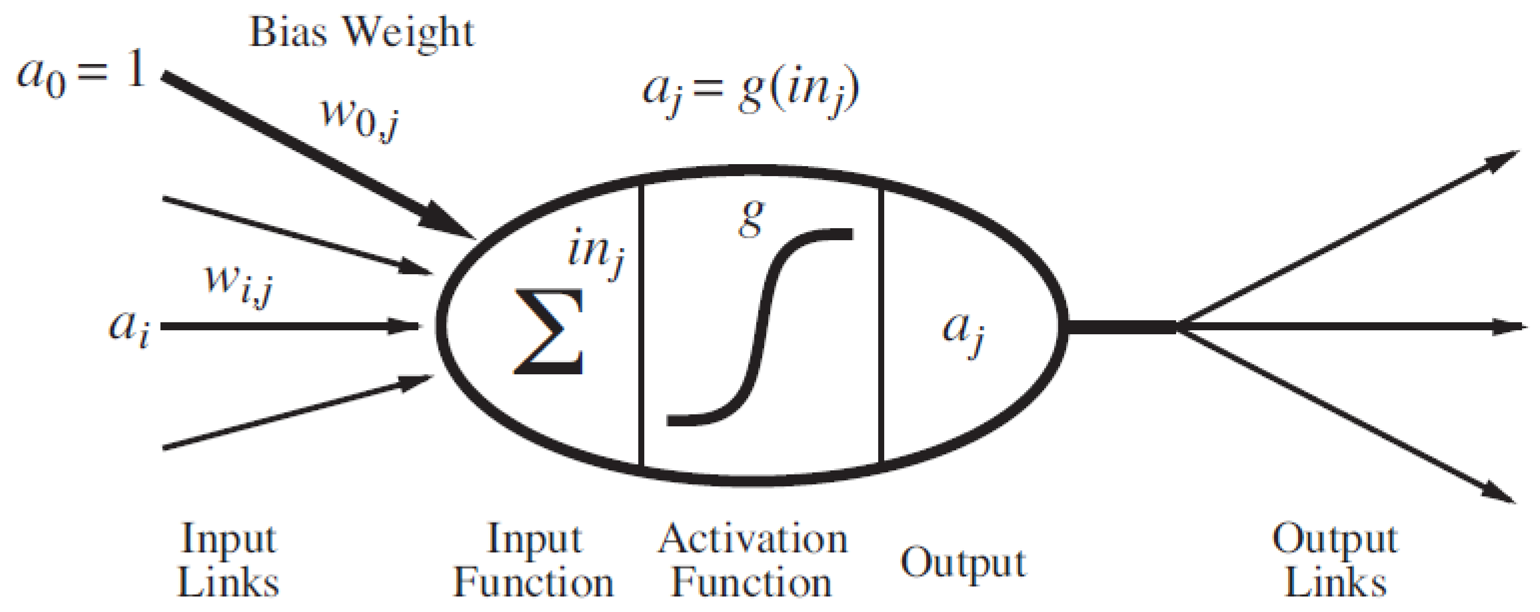
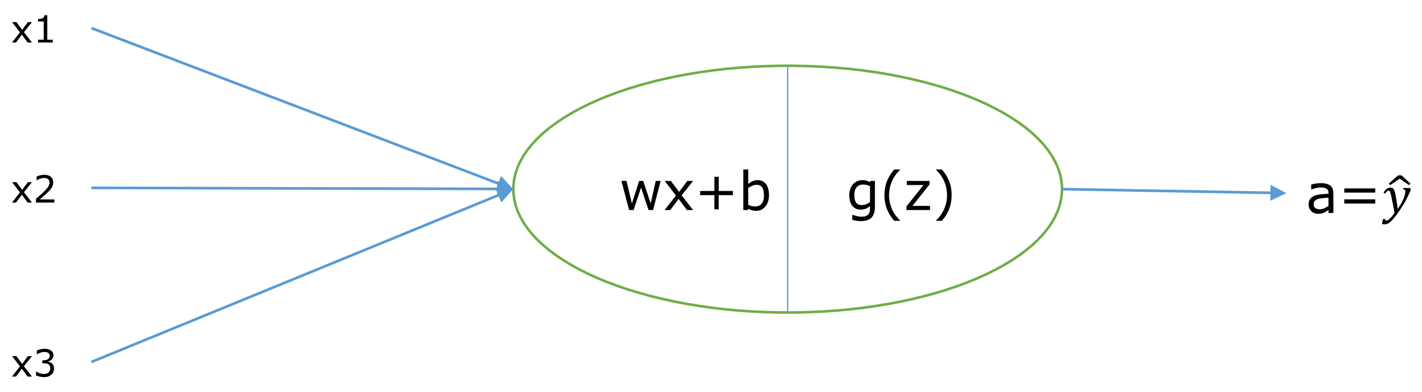


Figure 18.19 A simple mathematical model for a neuron. The unit's output activation is $a_j = g(\sum_{i=0}^{n} w_{i,j} a_i)$, where a_i is the output activation of unit i and $w_{i,j}$ is the weight on the link from unit i to this unit.

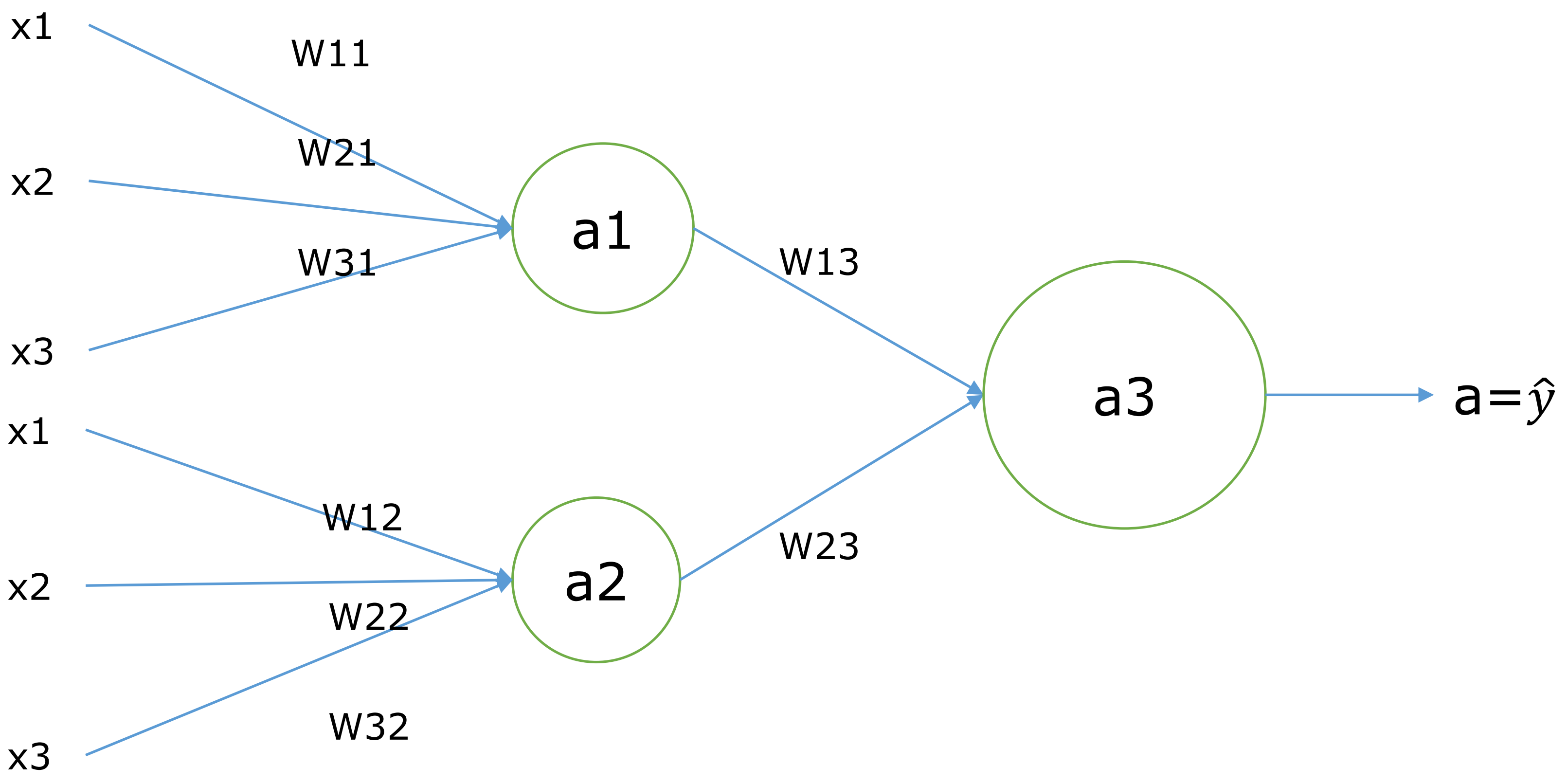
Artificial Neural Network

- Units
- Links
- Weight
- Activation
- Activation function

Artificial Neural Network



Artificial Neural Network



Artificial Neural Network – Activation Function

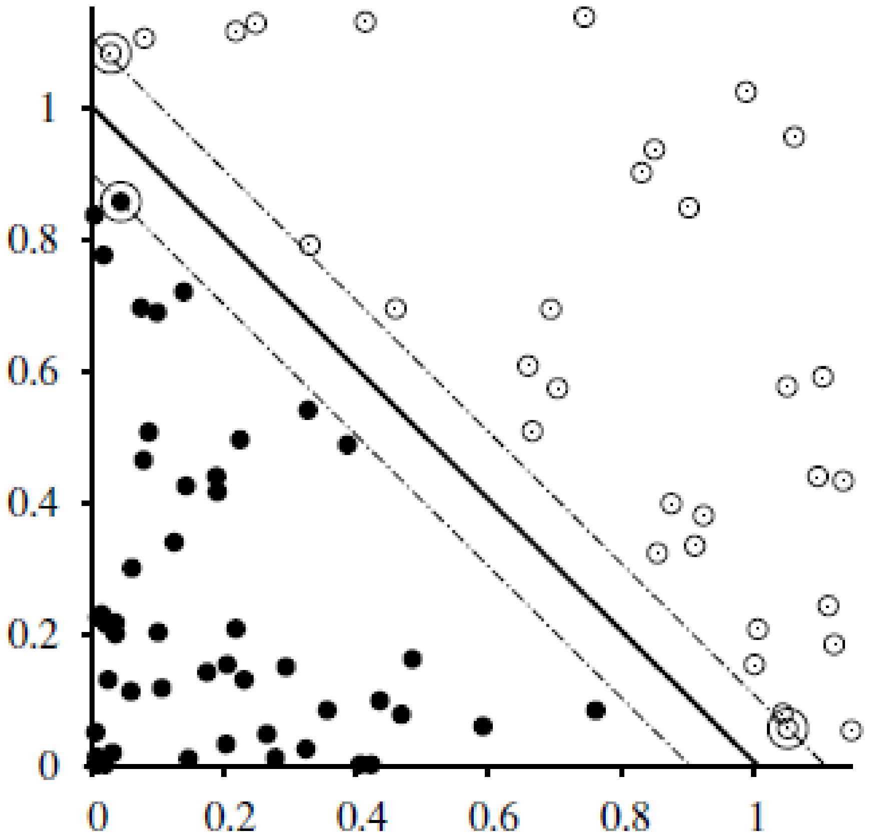
1. tanh
2. Sigmoid
3. Relu
4. Leaky Relu
5. Softmax
6. Identity

Algorithm

```
function BACK-PROP-LEARNING(examples, network) returns a neural network
  inputs: examples, a set of examples, each with input vector x and output vector y
          network, a multilayer network with L layers, weights w_{i,j}, activation function g
  local variables: Δ, a vector of errors, indexed by network node

  repeat
     for each weight w_{i,j} in network do
        w_{i,j} ← a small random number
     for each example (x, y) in examples do
        /* Propagate the inputs forward to compute the outputs */
        for each node i in the input layer do
           a_i ← x_i
        for ℓ = 2 to L do
           for each node j in layer ℓ do
              in_j ← Σ_i w_{i,j} a_i
              a_j ← g(in_j)
        /* Propagate deltas backward from output layer to input layer */
        for each node j in the output layer do
           Δ[j] ← g′(in_j) × (y_j − a_j)
        for ℓ = L − 1 to 1 do
           for each node i in layer ℓ do
              Δ[i] ← g′(in_i) Σ_j w_{i,j} Δ[j]
        /* Update every weight in network using deltas */
        for each weight w_{i,j} in network do
           w_{i,j} ← w_{i,j} + α × a_i × Δ[j]
  until some stopping criterion is satisfied
  return network
```

Support Vector Machine (SVM)



Thank you ☺