KECE471 Computer Vision

Pattern Recognition Concepts

Chang-Su Kim

Chapter 4, Computer Vision by Shapiro and Stockman

Pattern Recognition

- Recognition
  - To know that  is an apple from our knowledge

rec·og·nize

rec·og·nize (rĕk′əg-nīz′) *verb, transitive*
rec·og·nized, rec·og·niz·ing, rec·og·niz·es

1. To know to be something that has been perceived before: *recognize a face.*
2. To know or identify from past experience or knowledge: *recognize hostility.*
3. To perceive or show acceptance of the validity

- Computer vision
  - To make useful decision based on sensed images
  - It depends on visual pattern recognition

Apples

Pattern Recognition

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

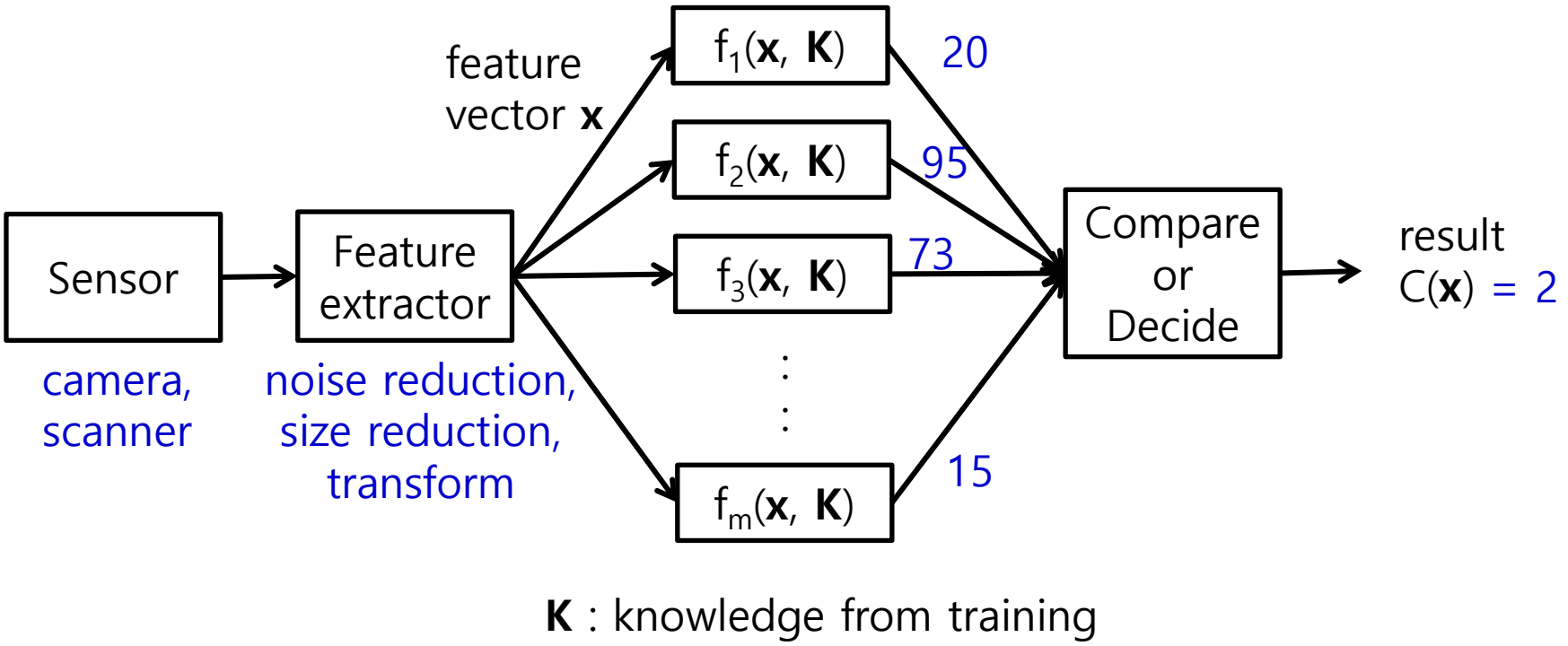
Source: www.LookupTables.com

Classes

- **Class**: a set of objects with common p
  - e.g. Apple, Orange, Grape, Reject
  - **Reject class**: a generic class for objects th placed in any other classes
- Each class is known by some description or by a set of examples
  - Apple:

  - Orange: round fruits with yellowish or reddish color
- **Classification** is a process that assigns a class label to an object
- **Classifier** is an algorithm for classification

General Diagram of Classification Systems



Evaluation of Classification Systems

- Classification error
  - It classifies an object as class C_i when the true class is C_j where $i \neq j$ and C_i is not the reject class
    - Apple -> Orange: error
    - Apple -> Reject: no error
- Empirical error rate
  - The number of errors divided by the number of classifications attempted on independent test data
- Empirical reject rate
  - The number of rejects divided by the number of classifications attempted on independent test data

Evaluation of Classification Systems

- A system that classifies all objects into the reject class
  - Empirical error rate = 0%
  - Empirical reject rate = 100%

- Independent test data
  - A set of sample objects with true class known, including objects from the reject class, that were not used in designing the feature vector extraction and classification algorithms

  - **Overfitting**: algorithm has too good performance on the training data but not on independent test data

Lena: overfitting

- For the curious: 'lena' or 'lenna' is a digitized Playboy centerfold, from November 1972. Lena Soderberg was last reported living in her native Sweden, happily married with three kids and a job with the state liquor monopoly.

- Alexander Sawchuk estimates that it was in June or July of 1973 when he, then an assistant professor of electrical engineering at the USC Signal and Image Processing Institute (SIPI), along with a graduate student and the SIPI lab manager, was hurriedly searching the lab for a good image to scan for a colleague's conference paper. They had tired of their stock of usual test images, dull stuff dating back to television standards work in the early 1960s. They wanted something glossy to ensure good output dynamic range, and they wanted a human face. Just then, somebody happened to walk in with a recent issue of Playboy. The engineers tore away the top third of the centerfold so they could wrap it around the drum of their Muirhead wirephoto scanner, which they had outfitted with analog-to-digital converters (one each for the red, green, and blue channels) and a Hewlett Packard 2100 minicomputer.

Evaluation of Classification Systems

- Two class problems

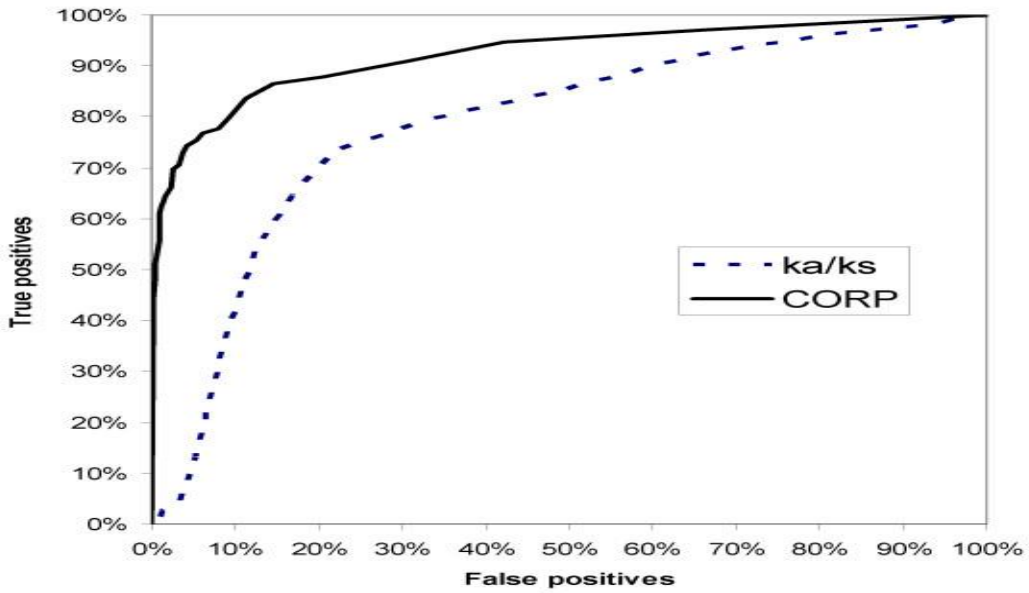
	Positive class	Negative class
ex 1) Fruit quality assessment	good	bad
ex 2) Security	Intruder present	Intruder absent
ex 3) Diagnosis	person has cancer	person has not cancer

- **False positive** or **false alarm**: classifying negative objects into the positive class
- **False negative** or **false dismissal**: classifying positive objects into the negative class

- In ex 3) false positive is OK but false negative is a critical issue

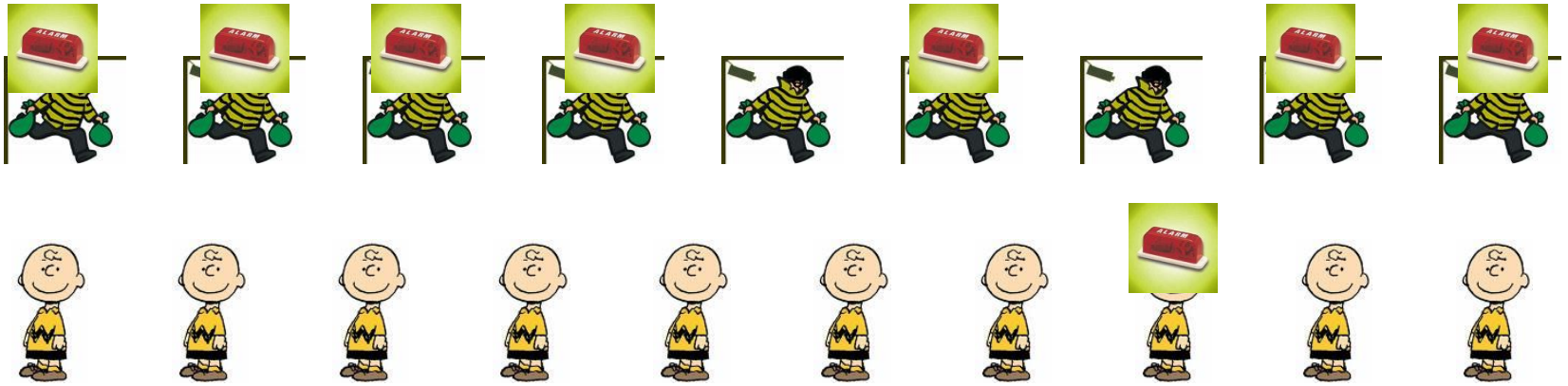
Evaluation of Classification Systems

- Receiver operating curve (**ROC**)
  - y-axis: the probability of correct detection (or true positive)
    = (the number of correctly diagnosed cancer patients)
    /(the number of all cancer patients)
  - x-axis: the probability of false positives
    = (the number of incorrectly diagnosed persons)
    /(the number of all healthy persons)



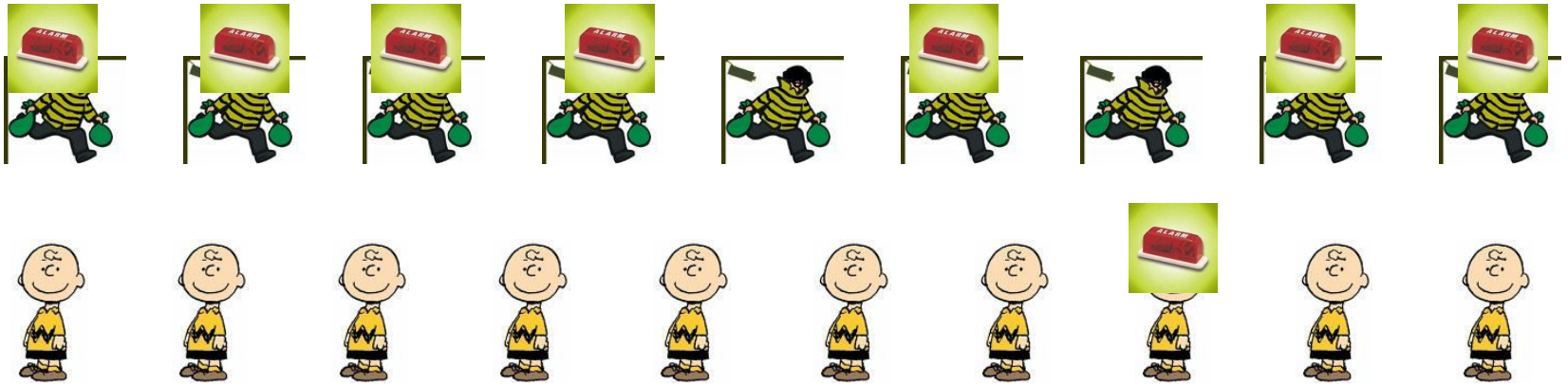
ROC Example

- Security Applications



ROC Example

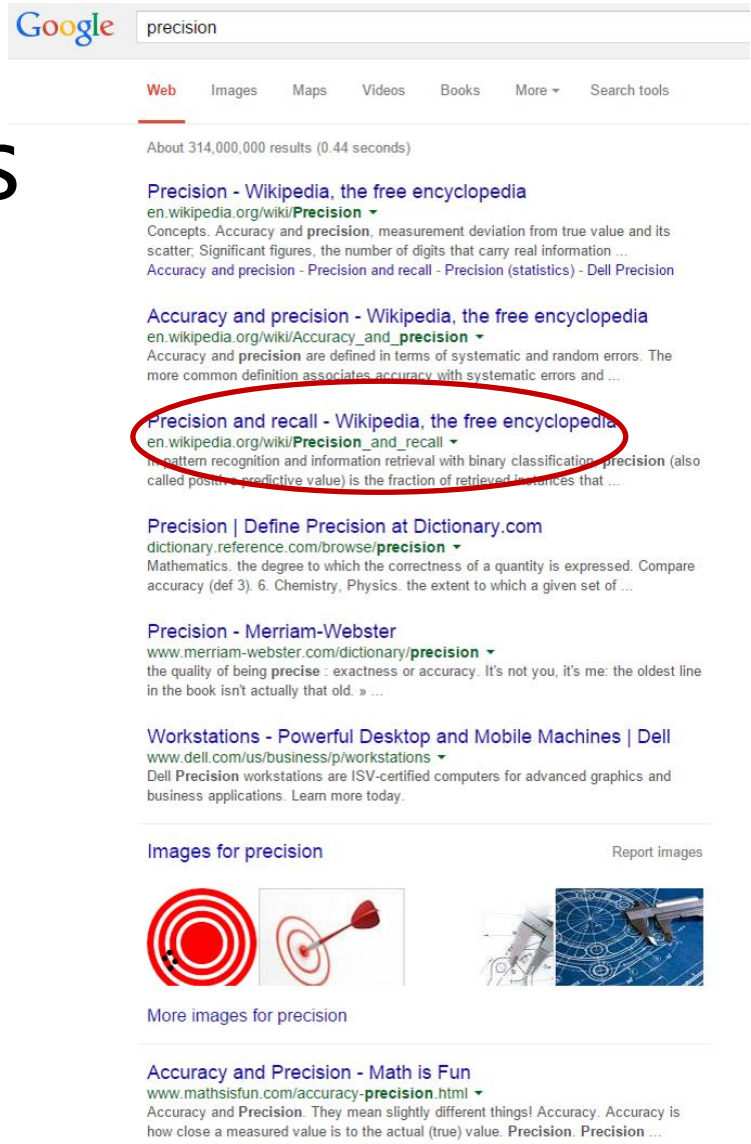
- Security Applications



  - True positive rate = 7/9
  - False positive rate = 1/10

Evaluation of Classification Systems

- Precision vs. Recall
  - In document retrieval or image retrieval system
  - Precision
    - The number of relevant documents retrieved divided by the total number of documents retrieved
  - Recall
    - The number of relevant documents retrieved divided by the total number of relevant documents in the database



Precision = 1/8
If there are about 1000 relevant documents, Recall = 1/1000

Example: Precision and Recall

Example: Precision and Recall

Precision = 4/7, Recall = 4/10

Example: Character Recognition System

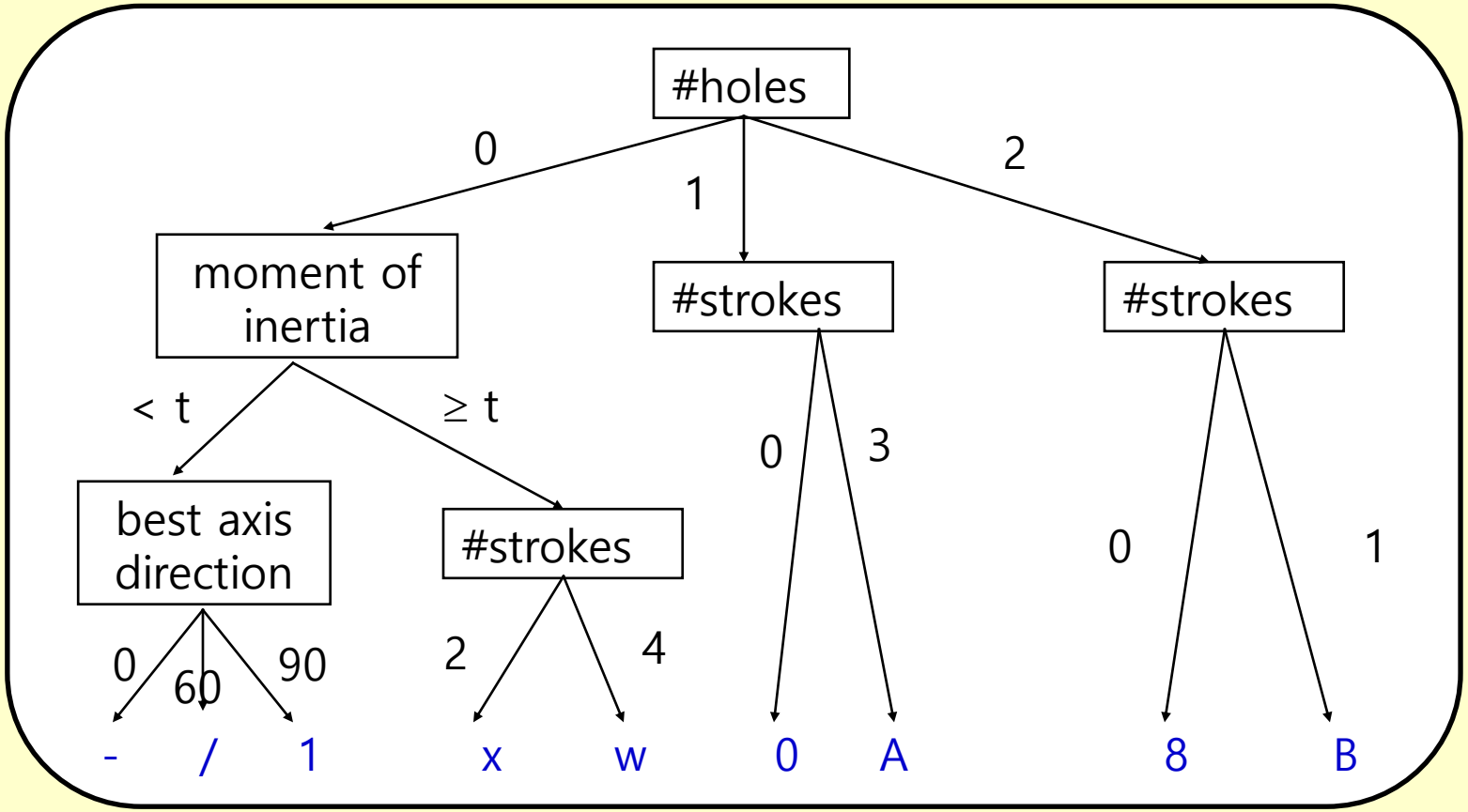
- Features
  - Properties of objects important for recognition
  - Feature extraction ~~is~~ was a key issue in designing recognition system

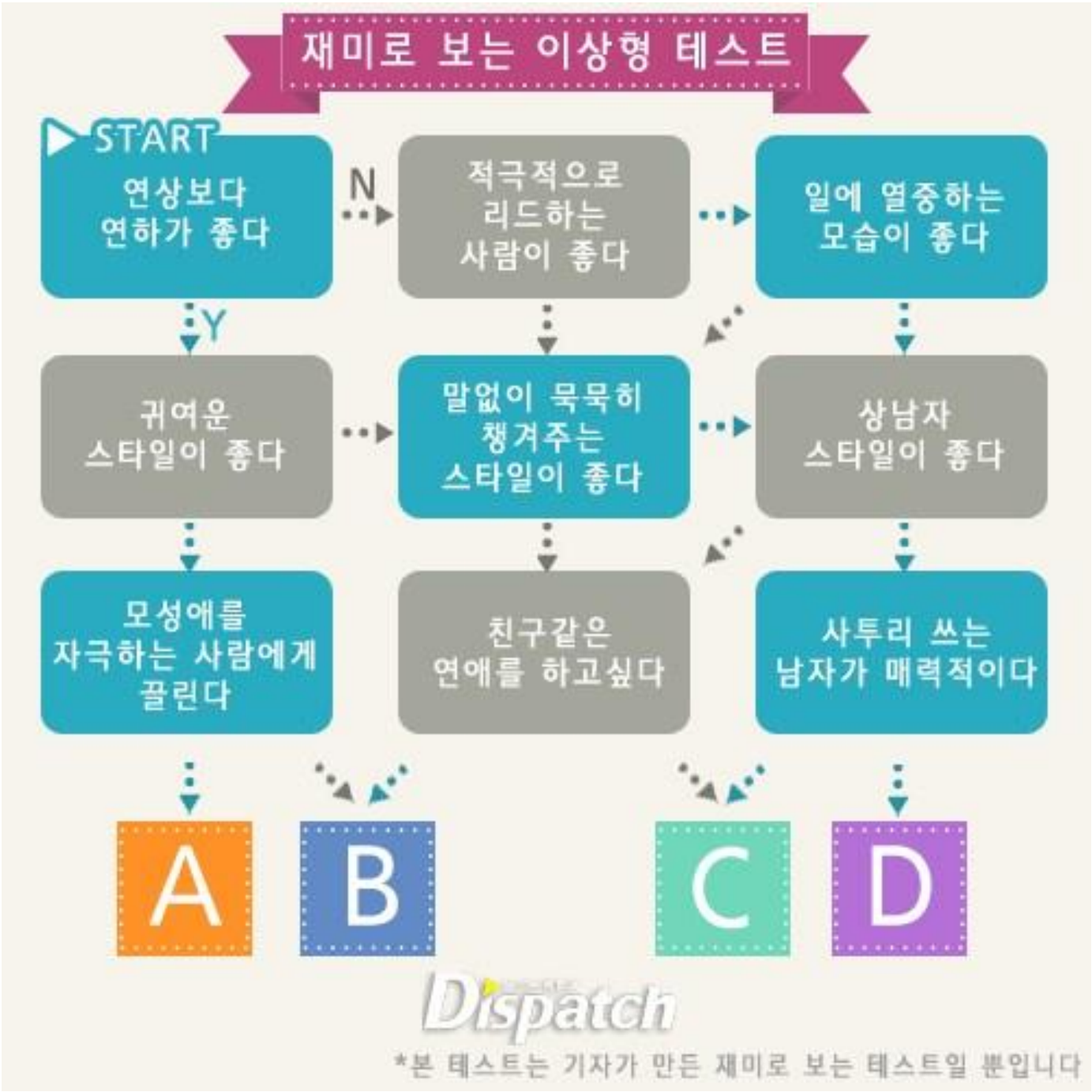
TABLE 4.1 EXAMPLE FEATURES FOR A SAMPLE CHARACTER SET.

(Class) Character	Area	Height	Width	Number Holes	Number Strokes	(cx,cy) Center	Best Axis	Least Inertia
'A'	medium	high	3/4	1	3	1/2, 2/3	90	medium
'B'	medium	high	3/4	2	1	1/3, 1/2	90	large
'8'	medium	high	2/3	2	0	1/2, 1/2	90	medium
'0'	medium	high	2/3	1	0	1/2, 1/2	90	large
'1'	low	high	1/4	0	1	1/2, 1/2	90	low
'W'	high	high	1	0	4	1/2, 2/3	90	large
'X'	high	high	3/4	0	2	1/2, 1/2	?	large
'*'	medium	low	1/2	0	0	1/2, 1/2	?	large
'-'	low	low	2/3	0	1	1/2, 1/2	0	low
'/'	low	high	2/3	0	1	1/2, 1/2	60	low

Example: Character Recognition System

- Classifier: Decision Tree Approach
  - In each node, a decision is made
  - In practice, the decision is prone to errors




재미로 보는 이상형 테스트
START
연상보다 연하가 좋다
N
적극적으로 리드하는 사람이 좋다
일에 열중하는 모습이 좋다
Y
귀여운 스타일이 좋다
말없이 묵묵히 챙겨주는 스타일이 좋다
상남자 스타일이 좋다
모성애를 자극하는 사람에게 끌린다
친구같은 연애를 하고싶다
사투리 쓰는 남자가 매력적이다
A
B
C
D
Dispatch
*본 테스트는 기자가 만든 재미로 보는 테스트일 뿐입니다

Example: Character Recognition System

- Evaluation: Confusion Matrix

		class j output by the pattern recognition system										
		'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'	'R'
	'0'	97	0	0	0	0	0	1	0	0	1	1
	'1'	0	98	0	0	1	0	0	1	0	0	0
true	'2'	0	0	96	1	0	1	0	1	0	0	1
object	'3'	0	0	2	95	0	1	0	0	1	0	1
class	'4'	0	0	0	0	98	0	0	0	0	2	0
	'5'	0	0	0	1	0	97	0	0	0	0	2
i	'6'	1	0	0	0	0	1	98	0	0	0	0
	'7'	0	0	1	0	0	0	0	98	0	0	1
	'8'	0	0	0	1	0	0	1	0	96	1	1
	'9'	1	0	0	0	3	0	0	0	1	95	0

confusion may be unavoidable between some classes
for example, between 9's and 4's, or between u's and j's
for handprinted characters

Several Approaches to Classificiation

1. Nearest Class Mean
2. Nearest Neighbor
3. Structural Techniques
4. Decision Trees
5. Bayesian Decision Making, and etc

- The combinations of these methods are also common

Feature Vectors and Distances

- Feature vector
  - Useful features are represented in numbers and gathered to form a vector
  - ex) In the character recognition system

$$\mathbf{x} = (\#\text{ of holes}, \#\text{ of strokes}, \text{moment along the best axis})$$

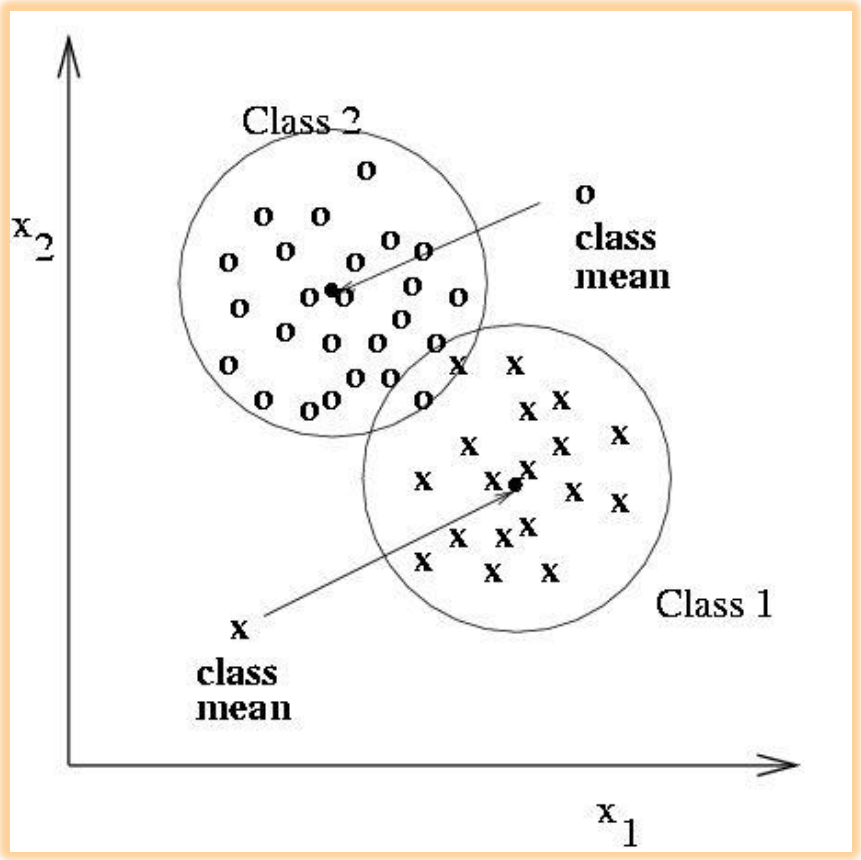
- Euclidean distance

$$\|\mathbf{x}_1 - \mathbf{x}_2\| = \sqrt{\sum_{i=1}^{d} (\mathbf{x}_1[i] - \mathbf{x}_2[i])^2}$$

1. Classification Using the Nearest Class Mean

- Assumptions
  - There are *m* classes
  - There are n_i training samples for each class

- Scheme
  - Compute the Euclidean distance between feature vector **x** and the mean of each class.
  - Choose the closest class if close enough; reject otherwise

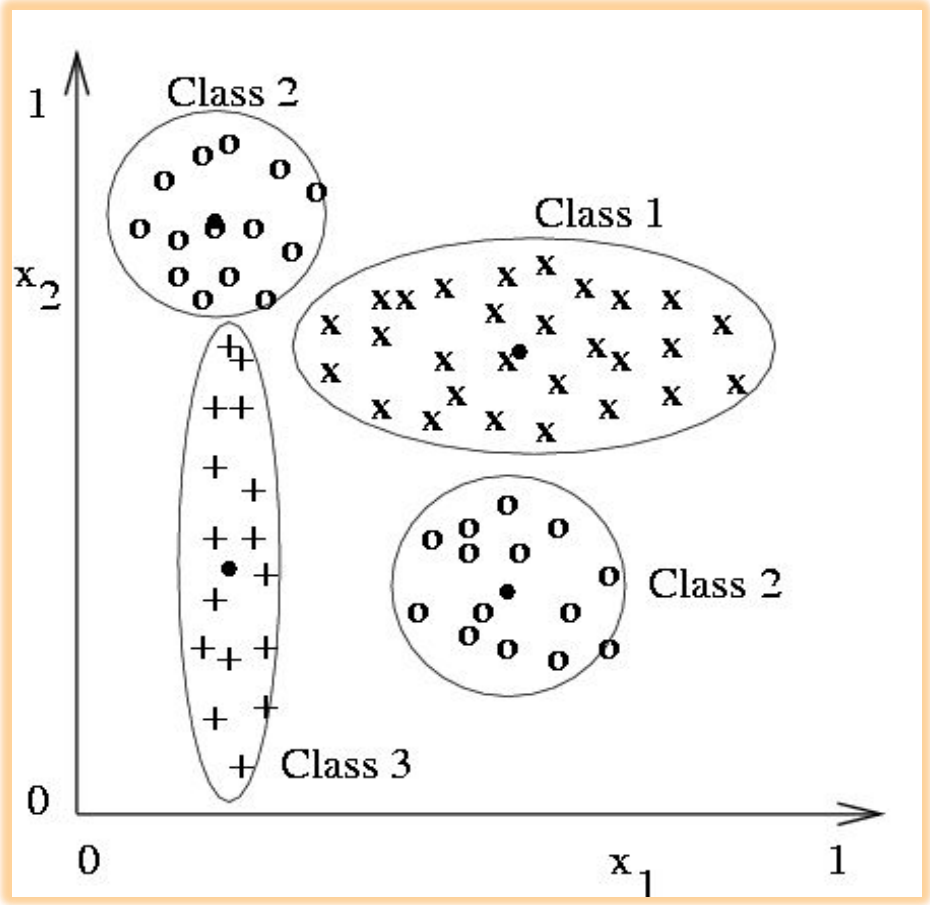


1. Classification Using the Nearest Class Mean

- It may yield poor results
- Class 2 is multi-modal
  - Use subclasses
- Classes 1 and 3 are elongated
  - Scaled Euclidean distance from **x** to class mean $\mathbf{x}_c$

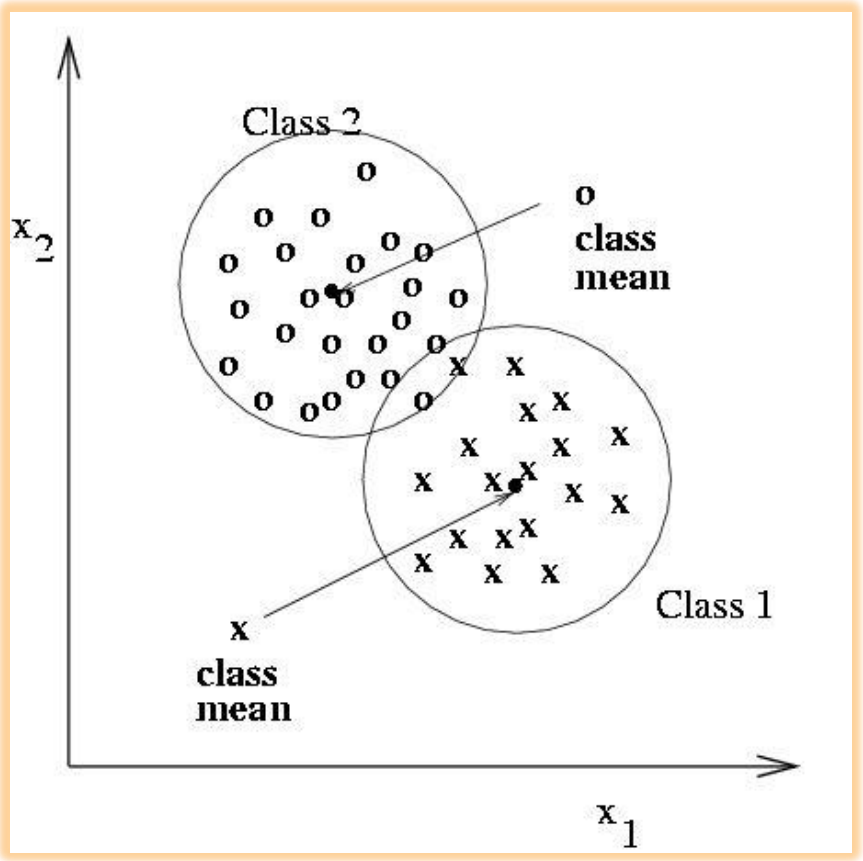
$$\|\mathbf{x}-\mathbf{x}_c\| = \sqrt{\sum_{i=1}^{d}\left\{\left(\mathbf{x}[i]-\mathbf{x}_c[i]\right)/\sigma_{c,i}\right\}^2}$$

  - Scaling is also necessary when comparing different physical quantities, such as weight and height



2. Classification Using the Nearest Neighbor

- Assumptions
  - There are *m* classes
  - There are n_i training samples for each class
- Scheme
  - Compute the distance to each sample
  - Choose the class of the closest sample
  - Computational complexity = O(# of all training samples)



2. Classification Using the Nearest Neighbor (k-NN)

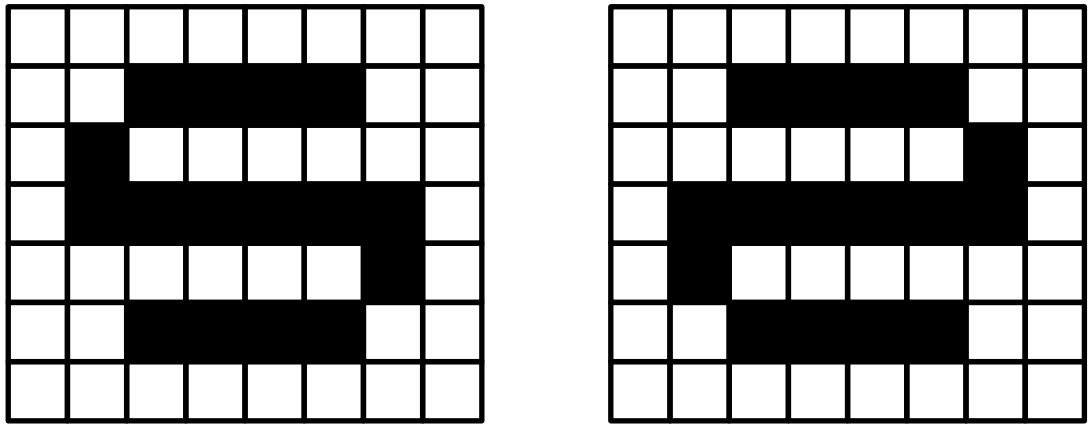
Compute the k nearest neighbors of x and return the majority class.

S is a set of n labeled class samples s_i where $s_i.\mathbf{x}$ is a feature vector and $s_i.c$ is its integer class label.
x is the unknown input feature vector to be classified.
A is an array capable of holding up to k samples in sorted order by distance d.
The value returned is a class label in the range **[1, m]**

```
procedure K_Nearest_Neighbors(x, S)
{
make A empty;
for all samples s_i in S
{
   d = Euclidean distance between s_i and x;
   if A has less than k elements then insert (d, s_i) into A;
   else if d is less than max A
      then {
               remove the max from A;
               insert (d, s_i) in A;
            }
};
assert A has k samples from S closest to x;
if a majority of the labels s_i.c from A are class c_0
   then classify x into class c_o;
   else classify x into the reject class;
return(class_of_x);
}
```

3. Structural Techniques

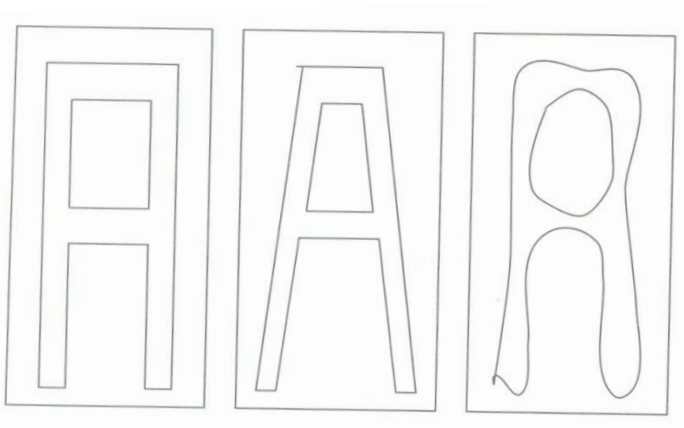
- Numeric features may not be enough



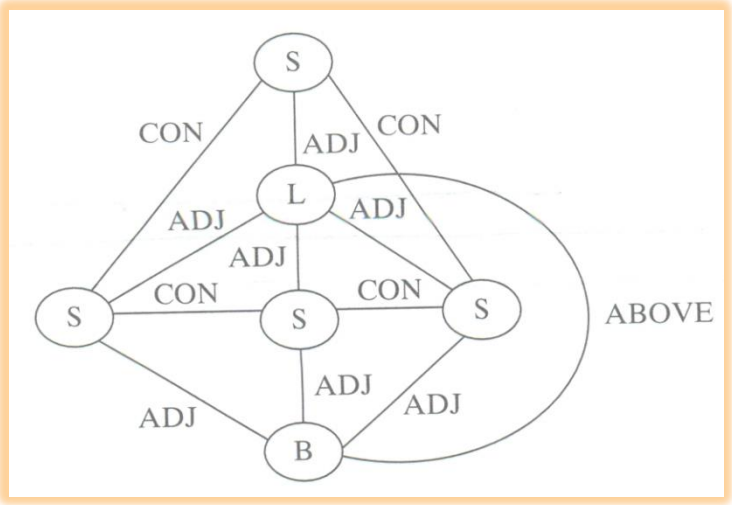
  - The same centroid
  - The same bounding box
  - The same number of strokes, etc.
  - But, they have different structures
    - The upper bays are open to the opposite directions

3. Structural Techniques

- Three A's with similar structures



- Graph representation



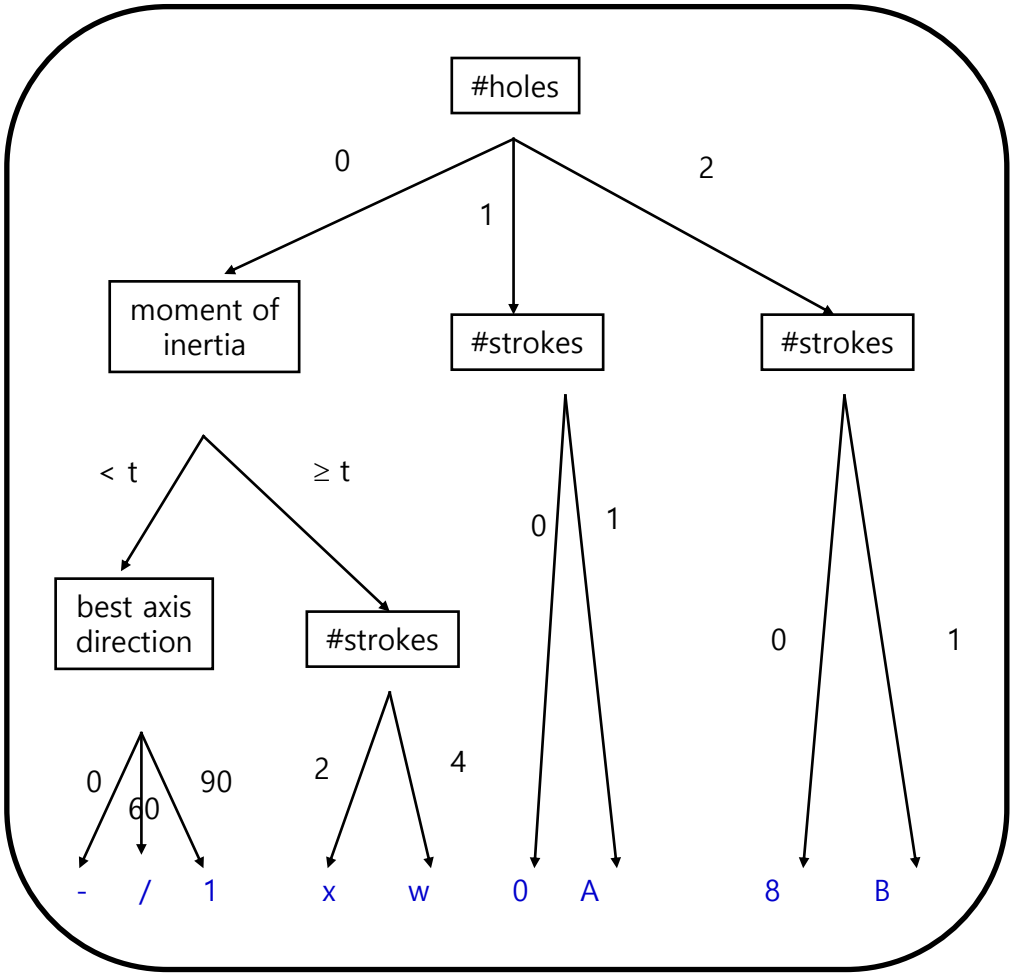
3. Structural Techniques

- Graph matching can be used for classification
  - Find the most similar graph

- Structural techniques are useful for recognition of complex patterns which involve many sub-patterns

4. Decision Trees

- Training
  - How do you construct one from training data?
  - Entropy-based Methods
- Strengths
  - Easy to Understand
- Weaknesses
  - Overtraining or overfitting

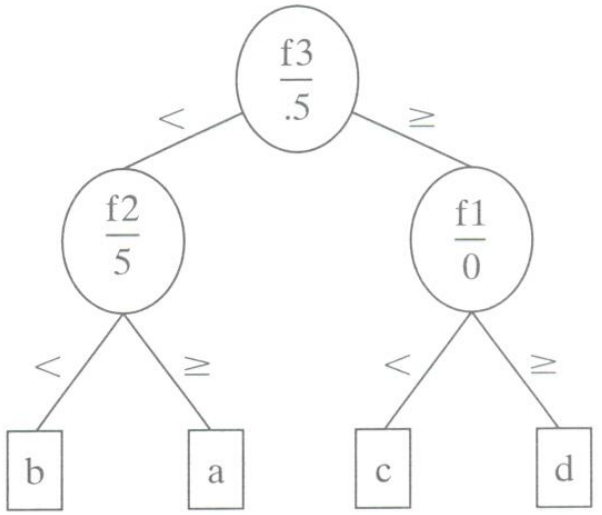


4. Decision Trees

- Example 1

f1	f2	f3	CLASS
3	6	0	a
−5	9	1	c
4	5	1	d
7	4	0	b
−1	10	0	a
2	6	1	d
−2	2	1	c
−1	3	0	b

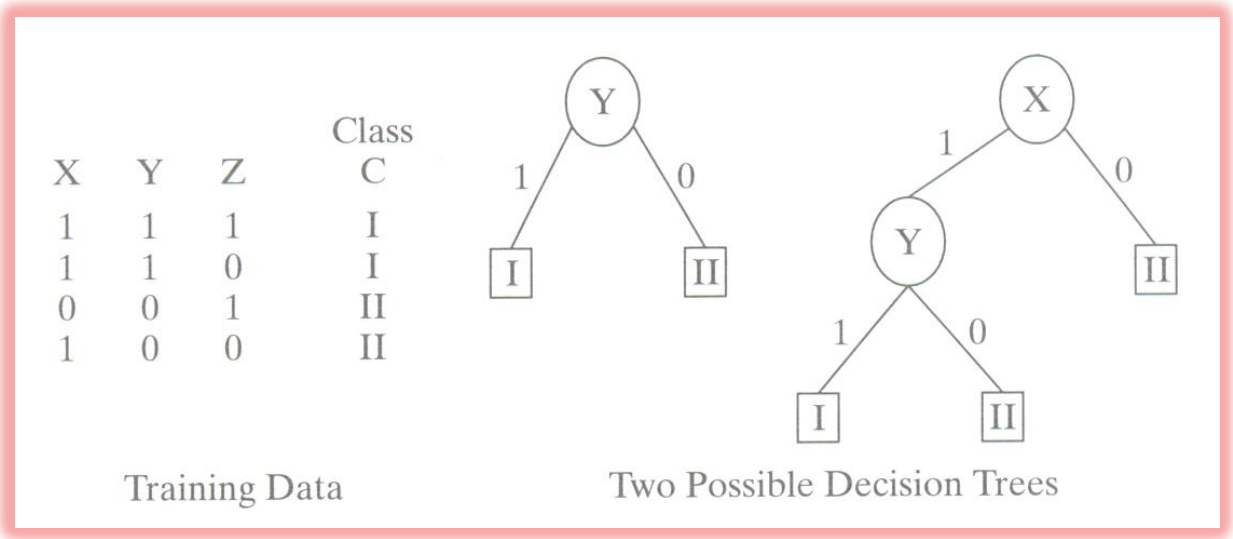
Training Data



Decision Tree

4. Decision Trees

- Example 2



X	Y	Z	Class C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Training Data

Two Possible Decision Trees

  - Which tree is better

4. Decision Trees

- Automatic construction of trees

Training Set S

$\mathbf{x}_1 = (f_1^1, f_1^2, \ldots, f_1^m)$

$\mathbf{x}_2 = (f_2^1, f_2^2, \ldots, f_2^m)$

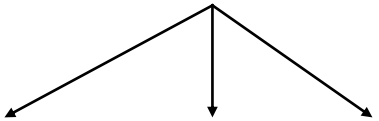
.

.

$\mathbf{x}_n = (f_n^1, f_n^2, \ldots, f_n^m)$

Node

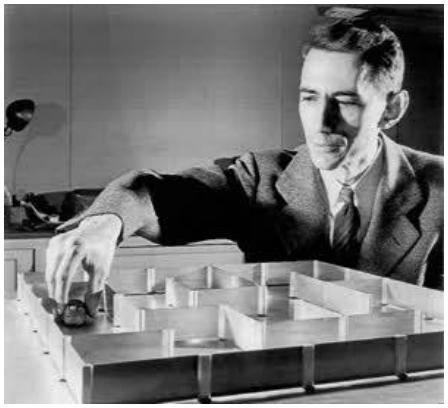
What feature should be used?



What values?

  - Entropy-based construction is one possible approach

4. Decision Trees



- Entropy-based construction
  - Entropy of a random variable *X*

$$H(X) = \sum_x p(x) \log \frac{1}{p(x)}$$

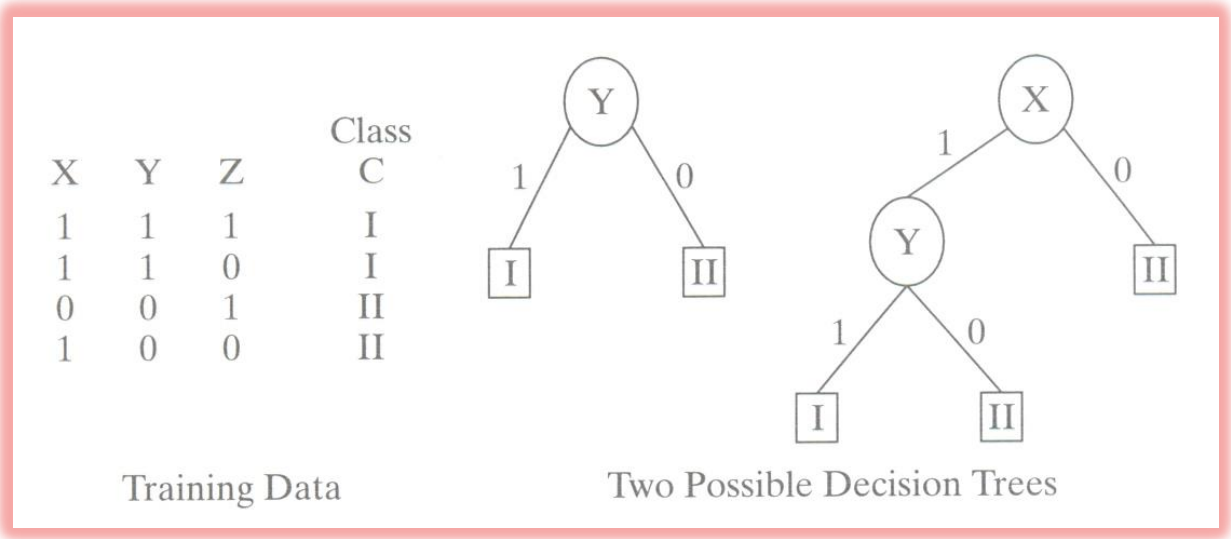
    - The amount of information of uncertainty in X

  - Mutual information between two random variables X and *Y*

$$I(X;Y) = \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

    - The amount of information that X contains about Y

4. Decision Trees



X	Y	Z	Class C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

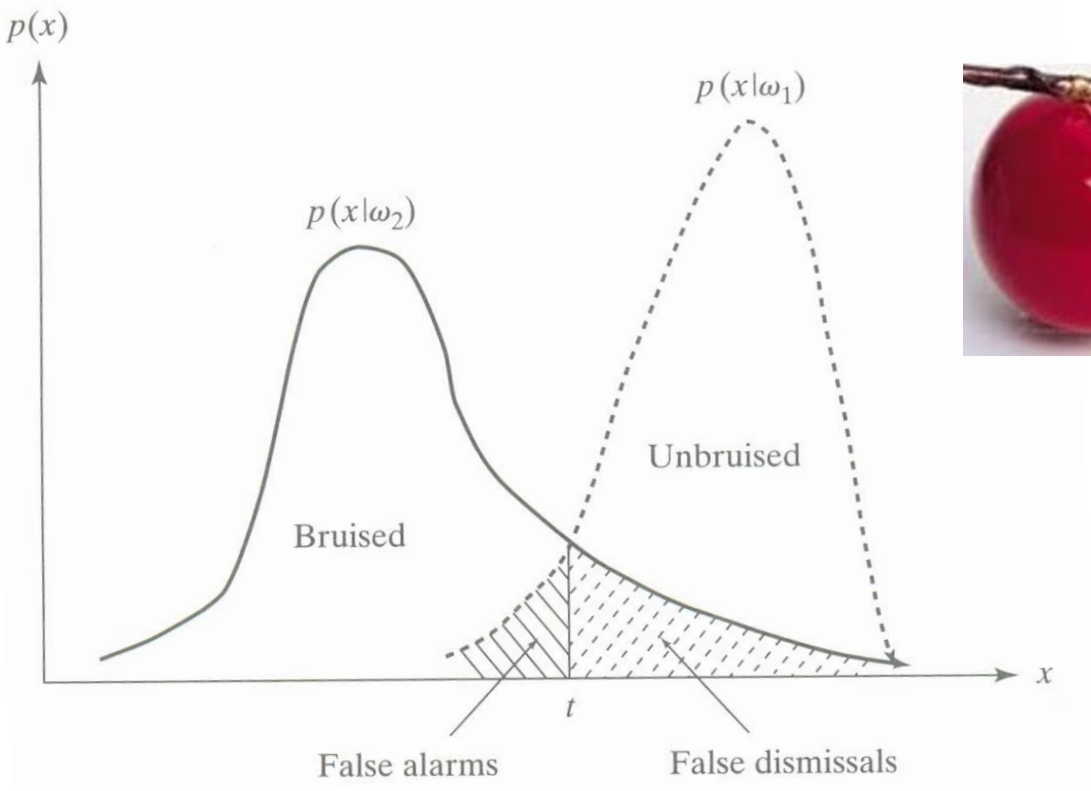
Training Data

Two Possible Decision Trees

$$I(X;C)=0.311, \quad I(Y;C)=1.0, \quad I(Z,C)=0.0$$

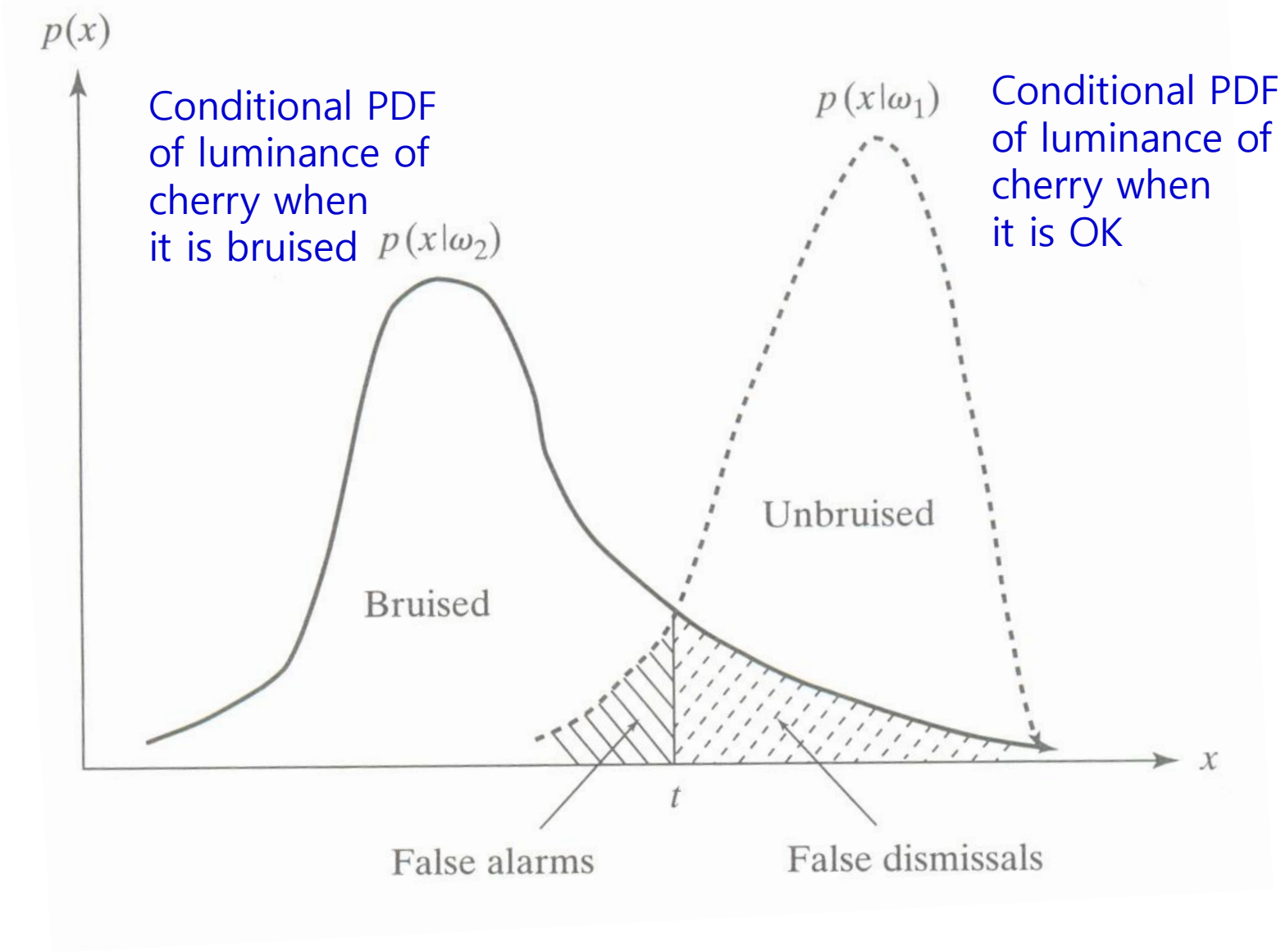
- In the above example, at the root node, I(Y;C) is maximum.
- So the computer selects Y as the first feature automatically and obtains the left tree
- If the classification is not complete, the same procedure repeats for each child nodes.

5. Bayesian Decision Making



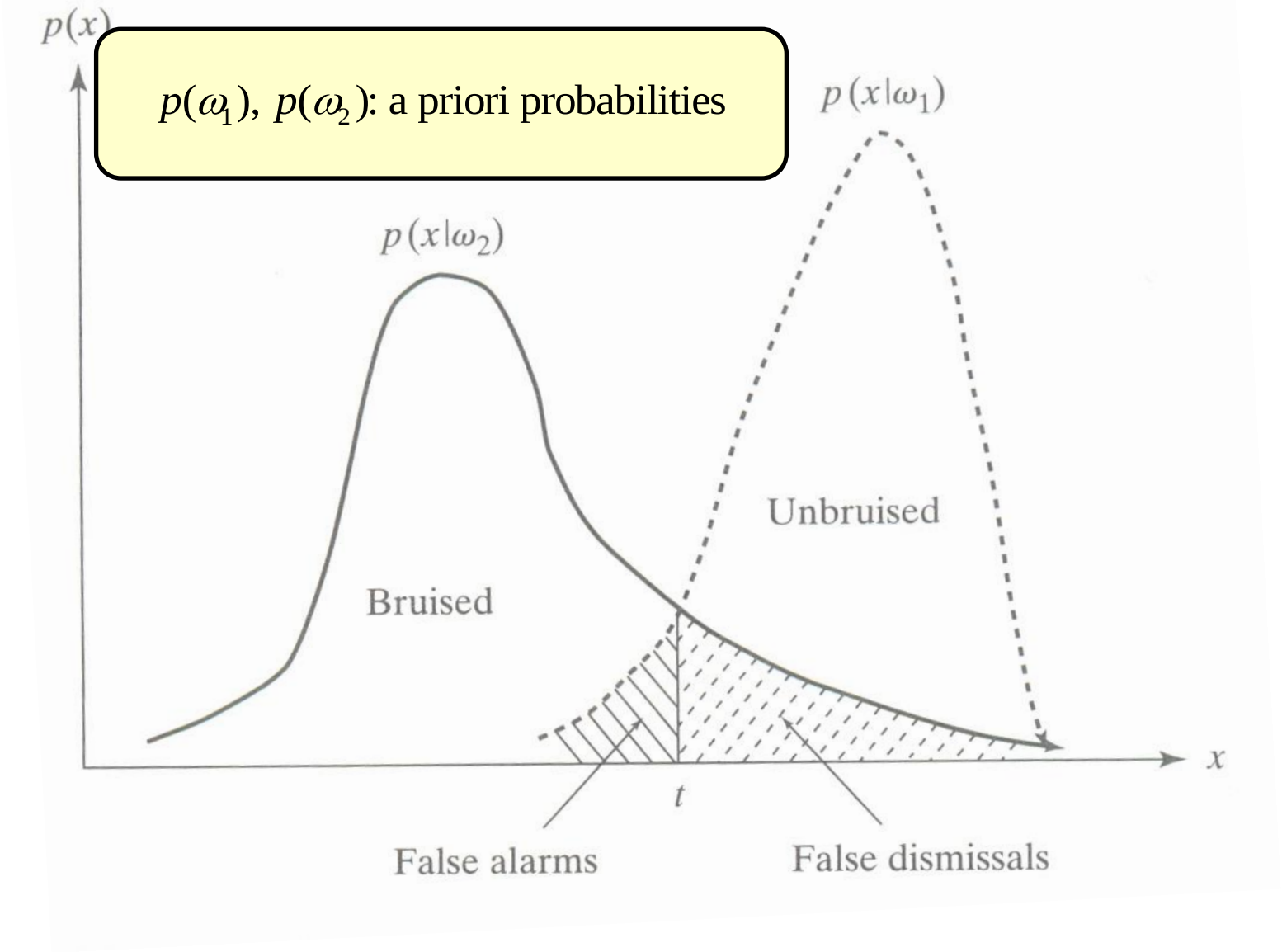
- Cherry example
  - ω_1: The cherry is OK
  - ω_2: The cherry is bruised

5. Bayesian Decision Making

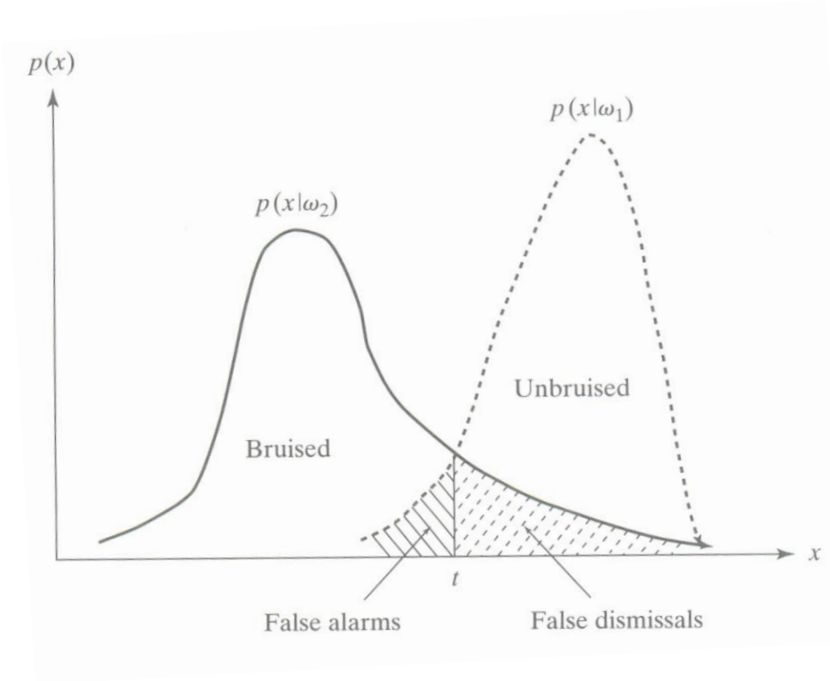


5. Bayesian Decision Making

$p(\omega_1)$, $p(\omega_2)$: a priori probabilities



5. Bayesian Decision Making



Let us assume that $p(\omega_1) = p(\omega_2) = 0.5$

We decide that the cherry is bruised if $x \le t$; OK otherwise

Probability of false dismissal = $0.5 \times$ the right area

Probability of false alarm =$0.5 \times$ the left area

Proability of error = $0.5 \times$ (the left area + the right area)

Note that the threshold t is optimal

5. Bayesian Decision Making

- A Bayesian classifier classifies an object into the class to which it belongs most likely, given the observation

- A posteriori probability

$$p(\omega_i \mid x) = \frac{p(\omega_i, x)}{p(x)} = \frac{p(x \mid \omega_i)\, p(\omega_i)}{p(x)} = \frac{p(x \mid \omega_i)\, p(\omega_i)}{\sum_k p(x \mid \omega_k)\, p(\omega_k)}$$

  - The denominator is the same for all i
  - Thus, we can compare only the numerators
  - For a priori probabilities are the same for all i, note that we can compare only the conditional probabilities $p(x|\omega_i)$. Recall the cherry example.

5. Bayesian Decision Making

- For the conditional probabilities $p(x|\omega_i)$, we often use the normal distribution

$$p(x \mid \omega_i) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$$

because of

- its simplicity
- its analytical properties

5. Bayesian Decision Making

- An apple farmer needs to classify her apples into green ones or red ones. Suppose that she has a camera that captures the reddishness (x) of an apple. The probability distribution function (PDF) of the reddishness of a red apple is given by $\mathrm{p}(\mathrm{x})$ and the PDF of the reddishness of a green apple is given by $\mathrm{q}(\mathrm{x})$.

$$\mathrm{p}(\mathrm{x}) = \begin{cases} \frac{1}{3}, & 2 \le \mathrm{x} \le 5, \\ 0, & \text{otherwise.} \end{cases} \quad \mathrm{q}(\mathrm{x}) = \begin{cases} \mathrm{x} - 1, & 1 \le \mathrm{x} \le 2, \\ 3 - \mathrm{x}, & 2 \le \mathrm{x} \le 3, \\ 0, & \text{otherwise.} \end{cases}$$

  For the last decade, she harvested 2 million green apples and 1 million red apples.

  a. Plot $\mathrm{p}(\mathrm{x})$ and $\mathrm{q}(\mathrm{x})$.
  b. Given the reddishness $(1 \le \mathrm{x} \le 5)$ of an apple, what is the Bayesian or maximum a posteriori (MAP) classification rule?
  c. In (b), what is the classification error rate? You may provide only the equation without the exact computation.
  d. What is the maximum likelihood (ML) classification rule?

NEURAL NETWORKS

Terminology

If $\mathbf{w}^T\mathbf{x} + w_0 > 0$ assign $\mathbf{x}$ to ω_1

If $\mathbf{w}^T\mathbf{x} + w_0 < 0$ assign $\mathbf{x}$ to ω_2

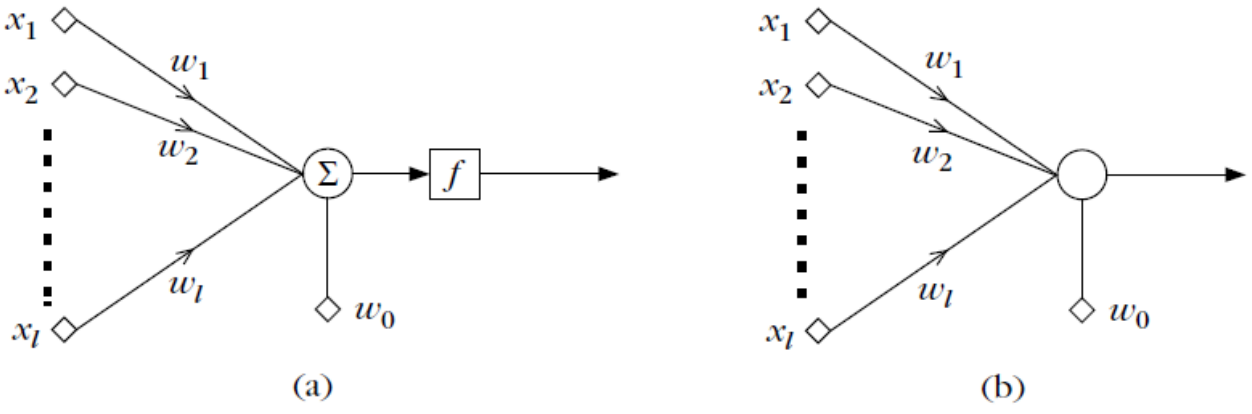


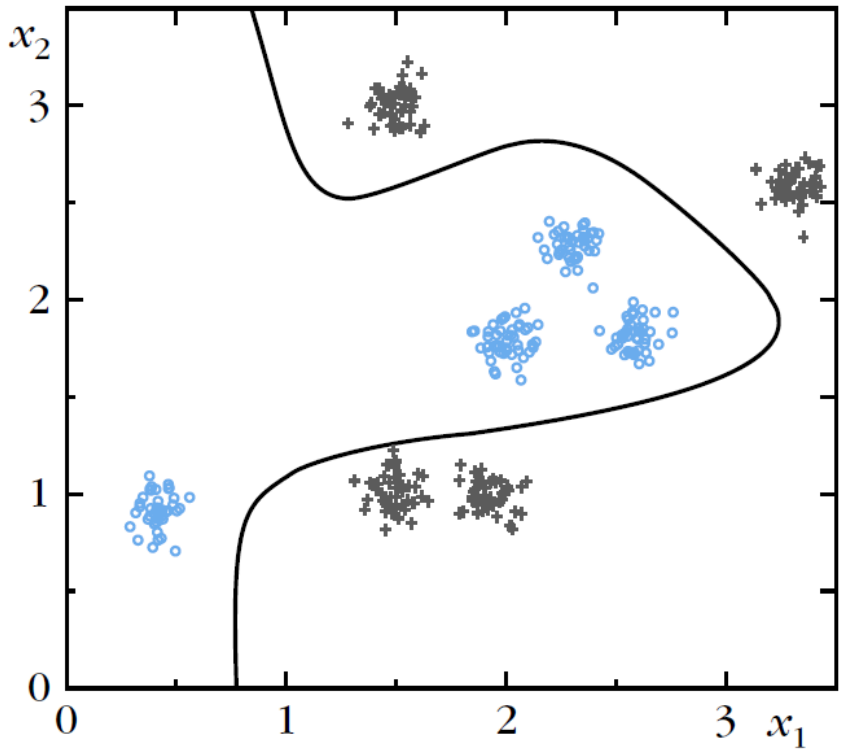
FIGURE 3.5

The basic perceptron model. (a) A linear combiner is followed by the activation function. (b) The combiner and the activation function are merged together.

- **Perceptron** or **neuron**
- **Synaptic weights** or **synapses**
- **Activation function**: *e.g.* $f(x) = s(x)$ (step function)

Nonlinear Classifiers

We deal with problems that are not linearly separable



ONE! TWO! THREE!

One-Layer Perceptron

- XOR problem is not linearly separable

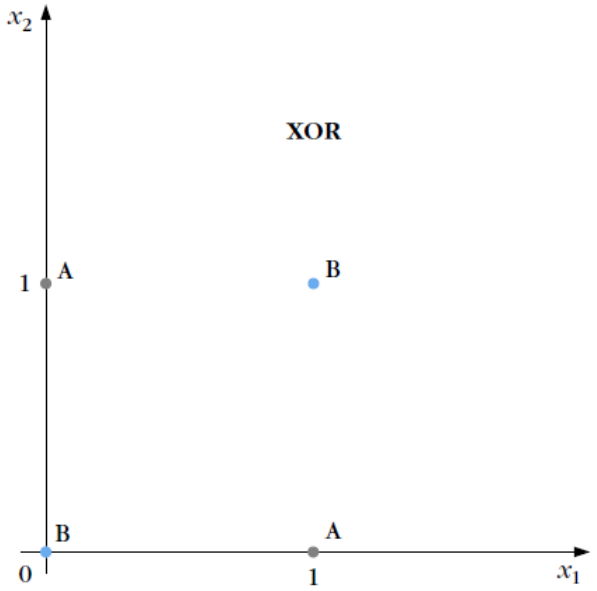
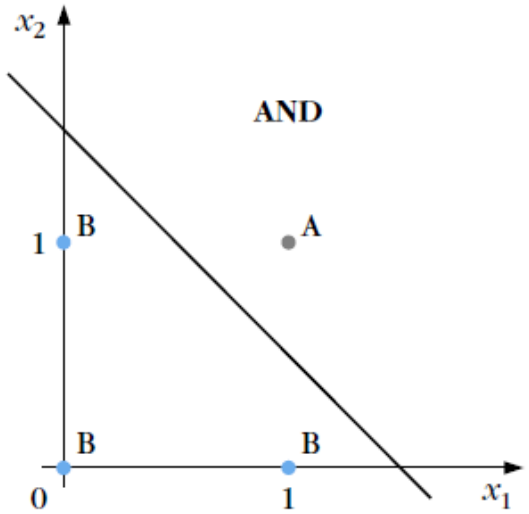


Table 4.1 Truth Table for the XOR Problem

x_1	x_2	XOR	Class
0	0	0	B
0	1	1	A
1	0	1	A
1	1	0	B

One-Layer Perceptron

- AND and OR problems are linearly separable



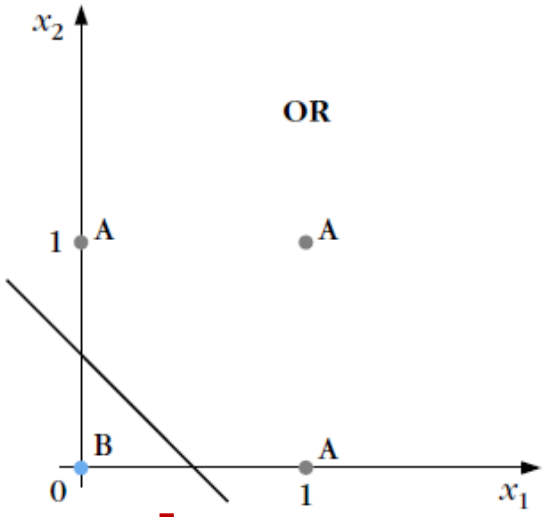
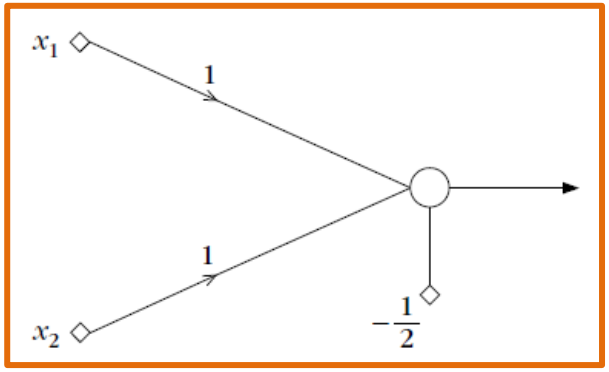


Table 4.2 Truth Table for AND and OR Problems

x_1	x_2	AND	Class	OR	Class
0	0	0	B	0	B
0	1	0	B	1	A
1	0	0	B	1	A
1	1	1	A	1	A

1-layer perceptron implementation



Two-Layer Perceptron

- XOR problem: solve it in two successive phases
  - 1st phase (or layer) uses two lines

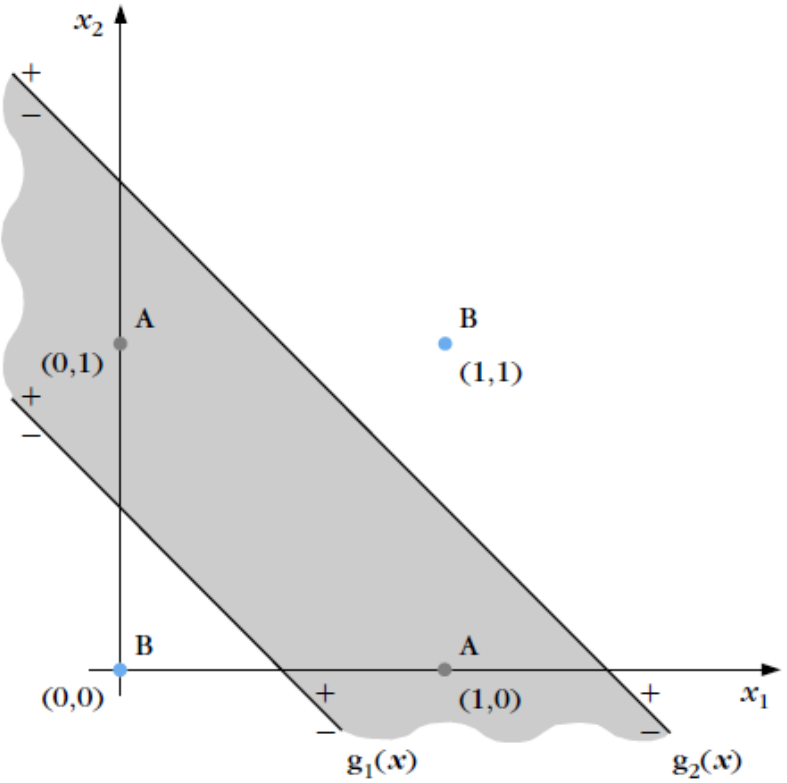


Table 4.3 Truth Table for the Two Computation Phases of the XOR Problem

1st Phase				2nd Phase
x_1	x_2	y_1	y_2	
0	0	0 (−)	0 (−)	B (0)
0	1	1 (+)	0 (−)	A (1)
1	0	1 (+)	0 (−)	A (1)
1	1	1 (+)	1 (+)	B (0)

Two-Layer Perceptron

- XOR problem: solve it in two successive phases
  - 2nd phase

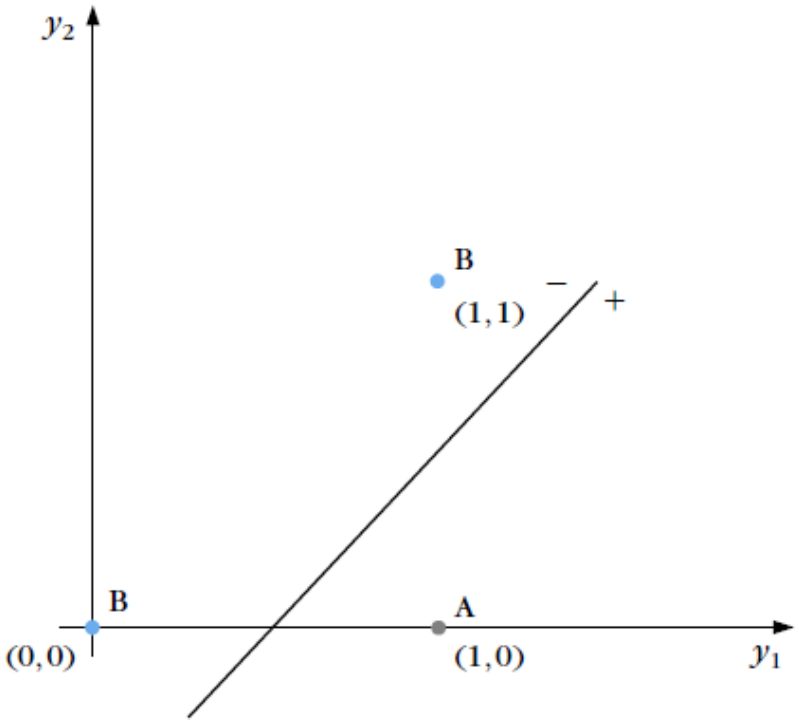
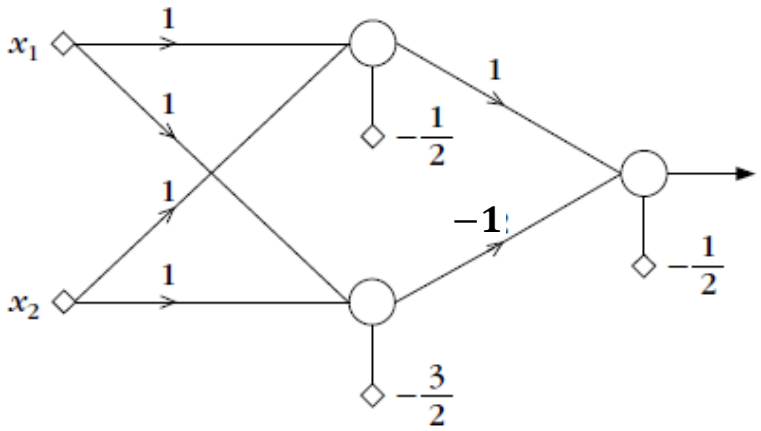


Table 4.3 Truth Table for the Two Computation Phases of the XOR Problem

1st Phase				
x_1	x_2	y_1	y_2	2nd Phase
0	0	0 (−)	0 (−)	B (0)
0	1	1 (+)	0 (−)	A (1)
1	0	1 (+)	0 (−)	A (1)
1	1	1 (+)	1 (+)	B (0)

Two-Layer Perceptron

- XOR problem: solve it in two successive phases
  - 2-layer perceptron (or 2-layer feedforward neural network)



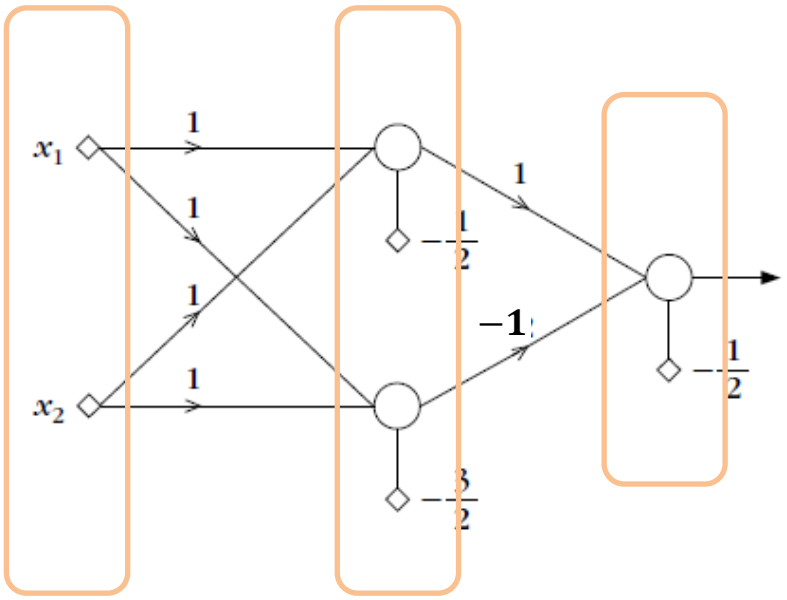
- $g_1(\mathbf{x}) = x_1 + x_2 - \frac{1}{2} = 0$
- $g_2(\mathbf{x}) = x_1 + x_2 - \frac{3}{2} = 0$
- $g(\mathbf{y}) = y_1 - y_2 - \frac{1}{2} = 0$

Table 4.3 Truth Table for the Two Computation Phases of the XOR Problem

1st Phase				
x_1	x_2	y_1	y_2	2nd Phase
0	0	0 (−)	0 (−)	B (0)
0	1	1 (+)	0 (−)	A (1)
1	0	1 (+)	0 (−)	A (1)
1	1	1 (+)	1 (+)	B (0)

Two-Layer Perceptron

- Terminology
  - 2-layer perceptron (or 2-layer feedforward neural network)



input layer (non-processing)

hidden layer

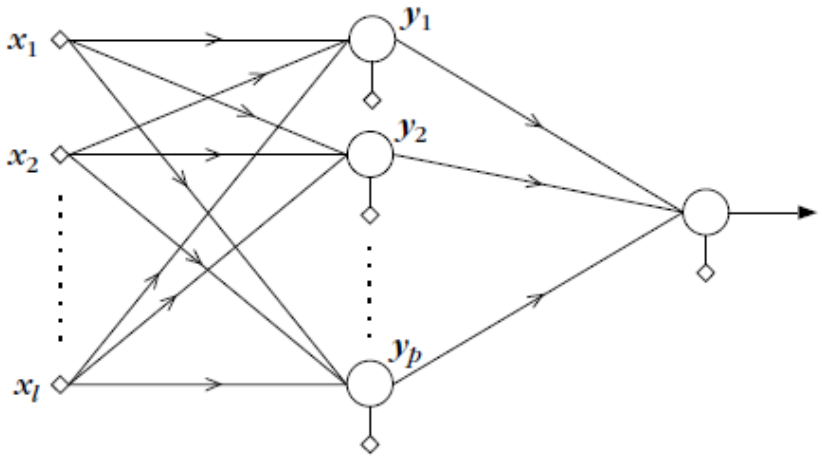
output layer

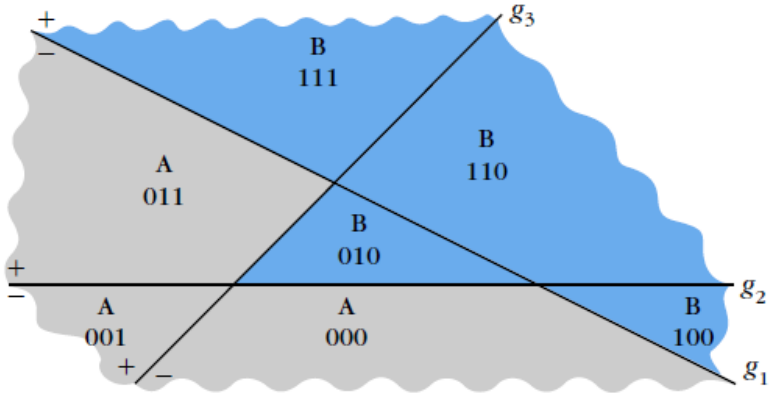
Two-Layer Perceptron

- Classification capabilities of two-layer perceptron
  - 1st layer maps input to vertices of the unit hypercube

$$H_p = \left\{ [y_1, \ldots, y_p]^T \in \mathbb{R}^p : \; y_i \in [0,1] \;\text{ for }\; 1 \le i \le p \right\}$$

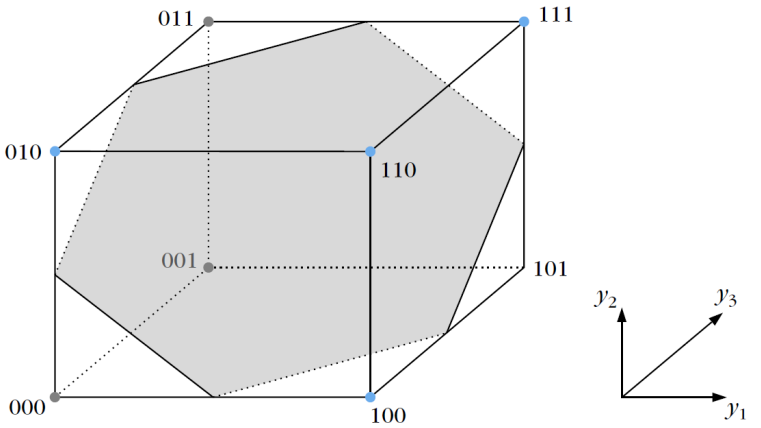
  - An output of 1st layer corresponds to a polyhedron

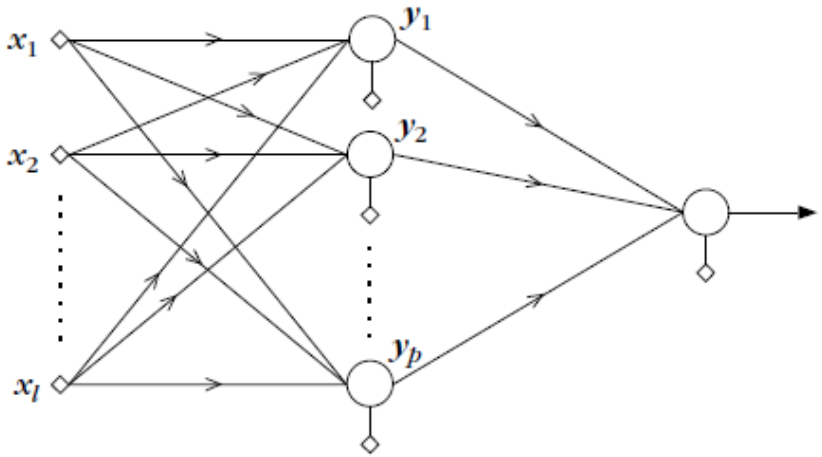


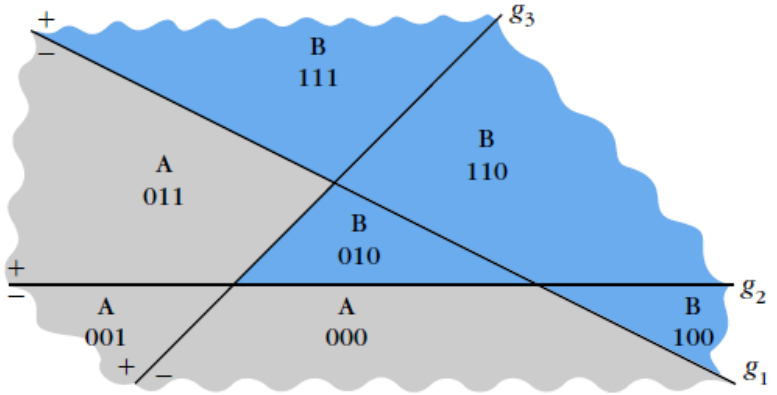


Two-Layer Perceptron

- Classification capabilities of two-layer perceptron
  - 2nd layer detects a union of selected polyhedra



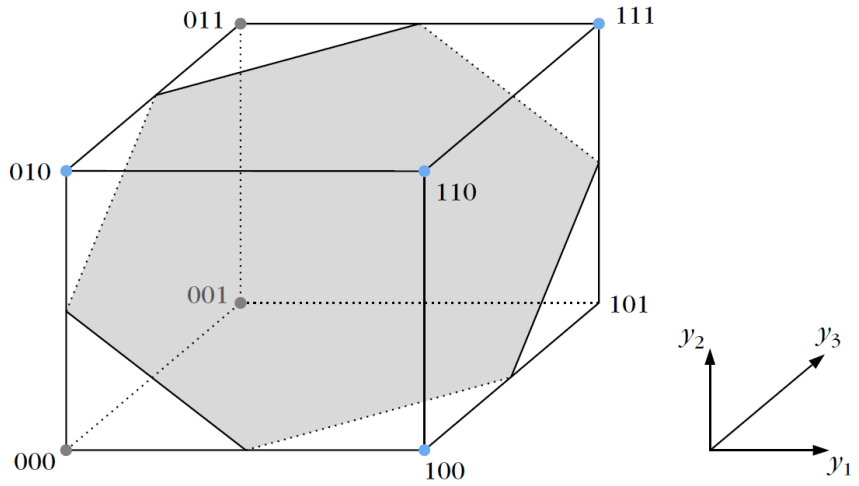




Two-Layer Perceptron

- Classification capabilities of two-layer perceptron

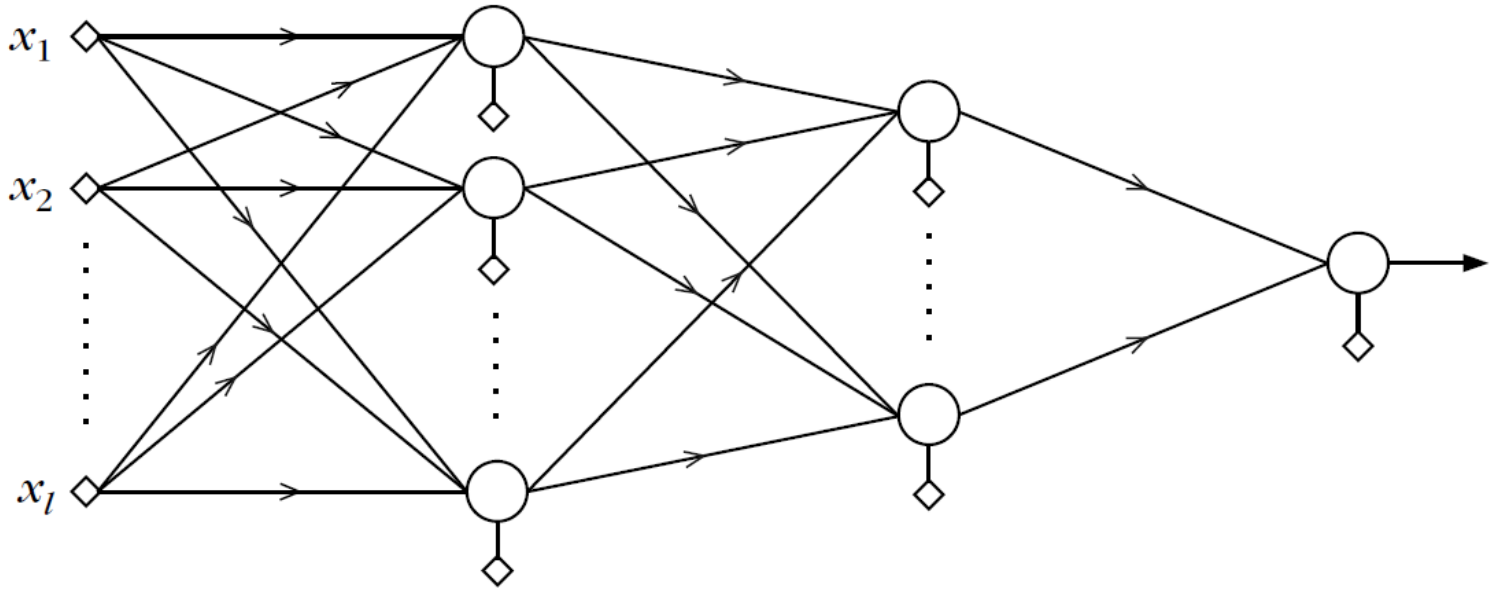
  Two-layer perceptron can detect a class, which consists of a union of polyhedral regions, but not any union of such regions



Three-Layer Perceptron

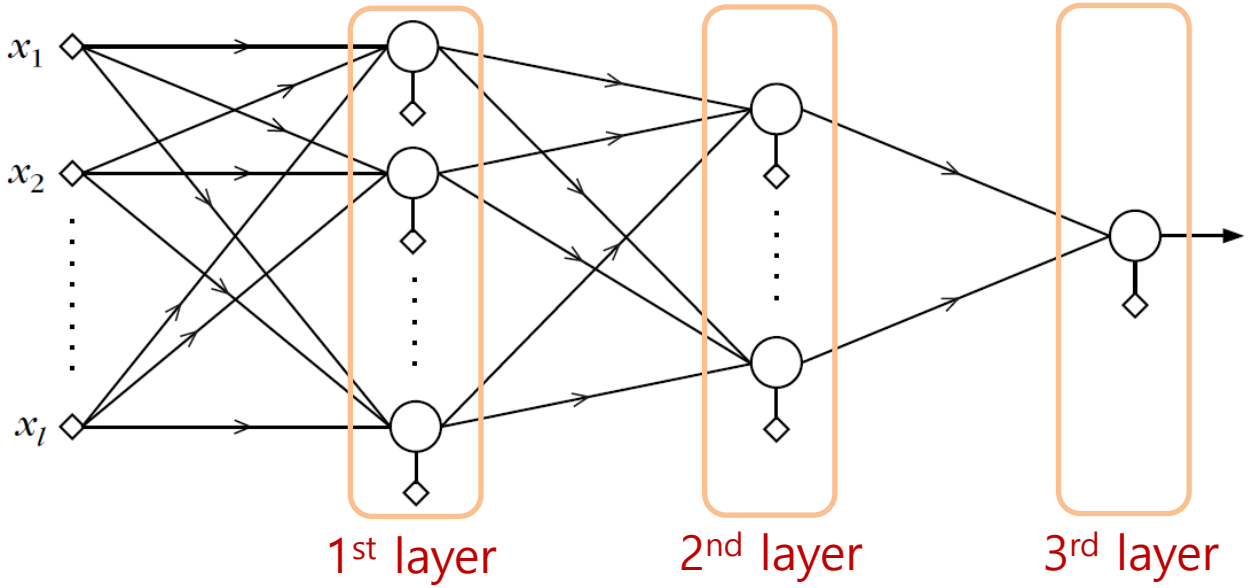
- Classification capabilities of three-layer perceptron

  Three-layer perceptron can detect a class, which consists of **any** union of polyhedral regions



Three-Layer Perceptron

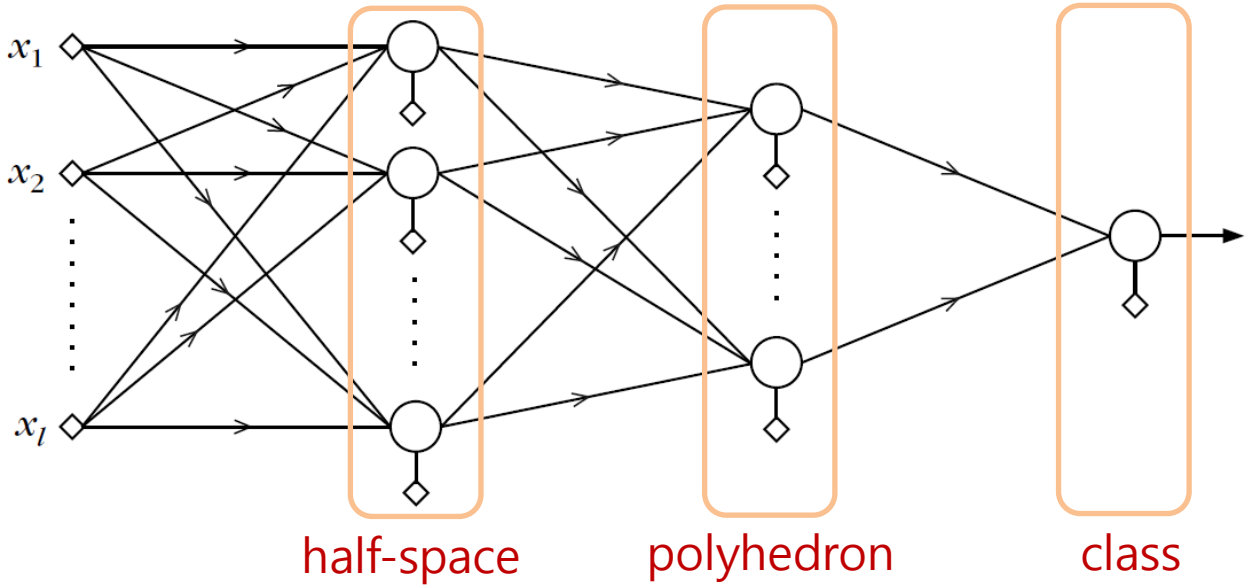
- Classification capabilities of three-layer perceptron



  - In 2nd layer, for each neuron, the synaptic weights are chosen so that the realized hyperplane leaves only one of the H_p vertices on one side and all the rest on the other
  - 3rd layer implements OR gate

Three-Layer Perceptron

- Classification capabilities of three-layer perceptron

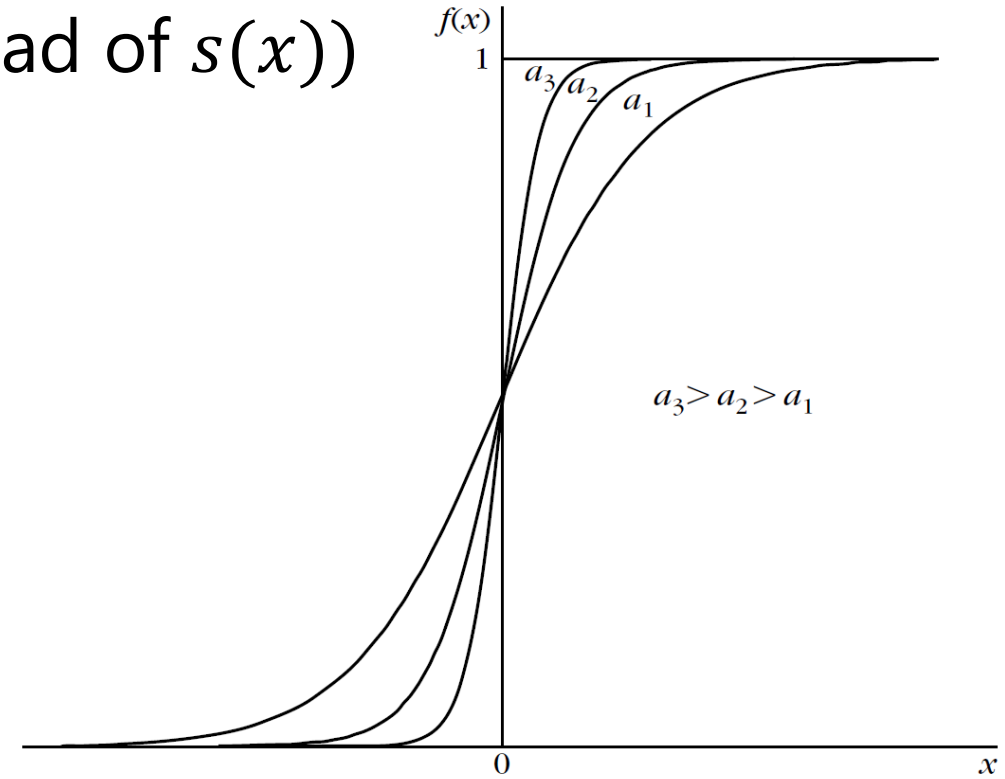


  - 1st layer detects half-spaces
  - 2nd layer detects polyhedra
  - 3rd layer detects a class, which is any union of polyhedra

BACKPROPAGATION ALGORITHM

Multilayer Perceptron Design

- Design a multilayer perceptron
  - Fix an architecture, and optimize the synaptic weights
  - To use the gradient descent scheme, we need a continuous activation function
- Logistic function (instead of $s(x)$)
  - $f(x) = \frac{1}{1+\exp(-ax)}$



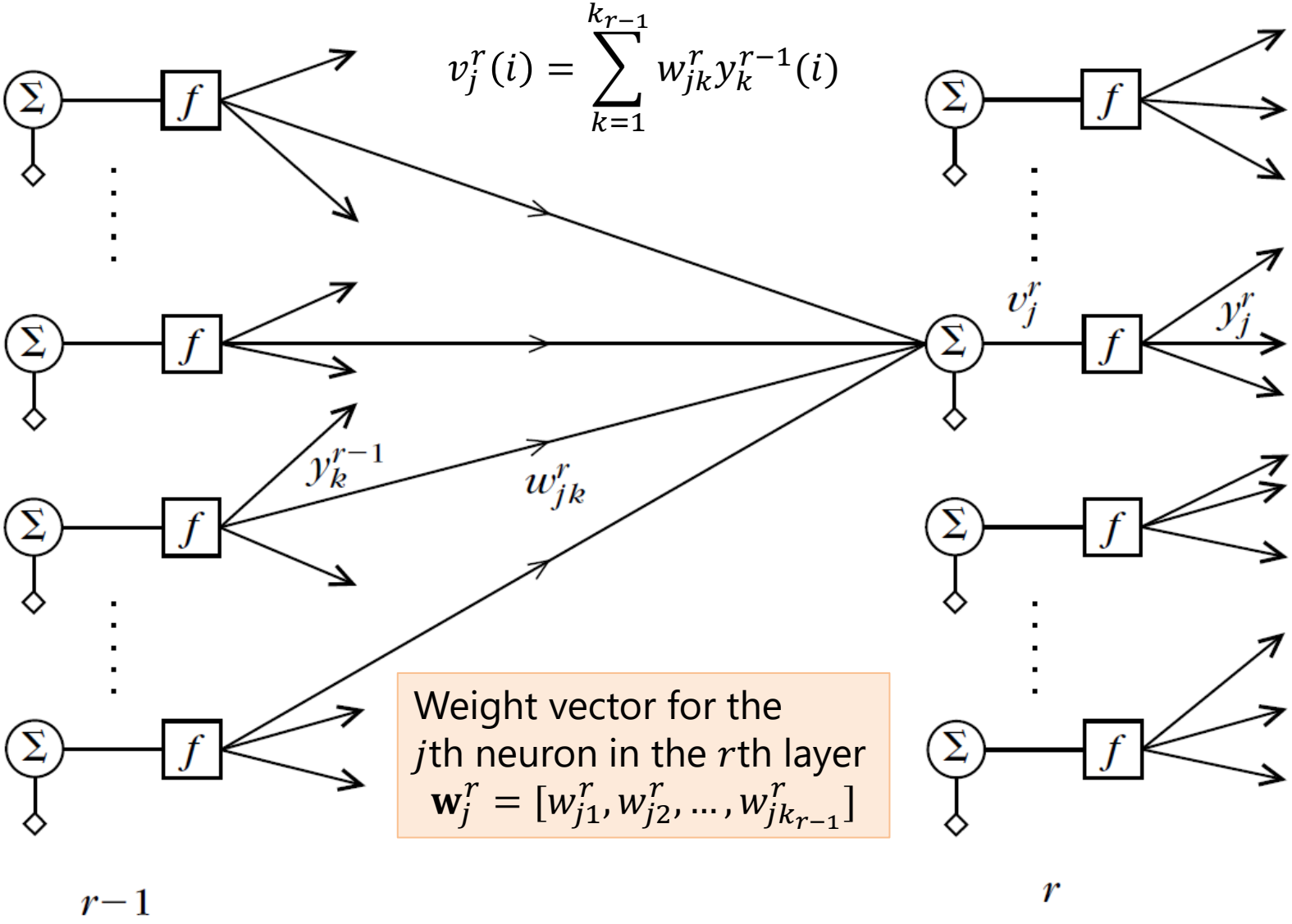
Architecture and Formulation

- L layers and k_r neurons in the rth layer ($r = 1, \dots, L$)
  - $k_0 = l$ nodes in the input layer
  - k_L output neurons
- N training pairs, $(\mathbf{y}(i), \mathbf{x}(i))$, $i = 1, \dots, N$, are available
  - $\mathbf{y}(i) = \left[y_1(i), \dots, y_{k_L}(i)\right]^T$
  - $\mathbf{x}(i) = \left[x_1(i), \dots, x_{k_0}(i)\right]^T$
- During training, the actual output $\hat{\mathbf{y}}(i)$ is different from the desired one $\mathbf{y}(i)$
- Compute the synaptic weights to minimize

$$J = \sum_{i=1}^{N} \mathcal{E}(i)$$

$$\mathcal{E}(i) = \frac{1}{2} \sum_{m=1}^{k_L} e_m^2(i) \equiv \frac{1}{2} \sum_{m=1}^{k_L} \left(\hat{y}_m(i) - y_m(i)\right)^2$$

Definition of Variables



Gradient Descent

$$\mathbf{w}_j^r(\text{new}) = \mathbf{w}_j^r(\text{old}) + \Delta\mathbf{w}_j^r$$

$$\Delta\mathbf{w}_j^r = -\mu\frac{\partial J}{\partial \mathbf{w}_j^r}$$

- Details for subsequent steps are omitted