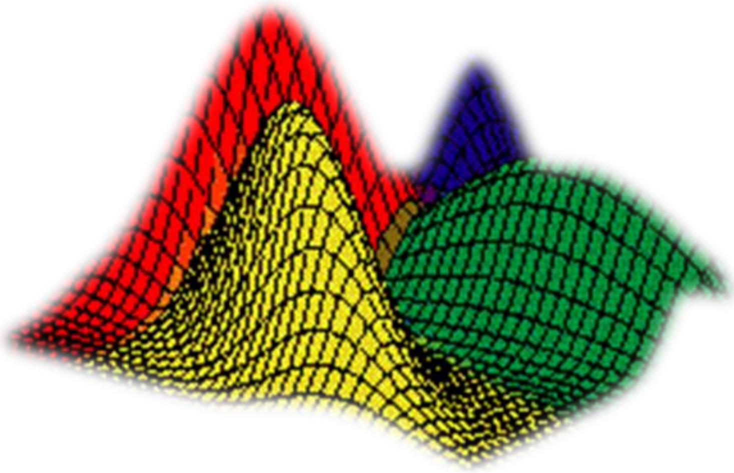
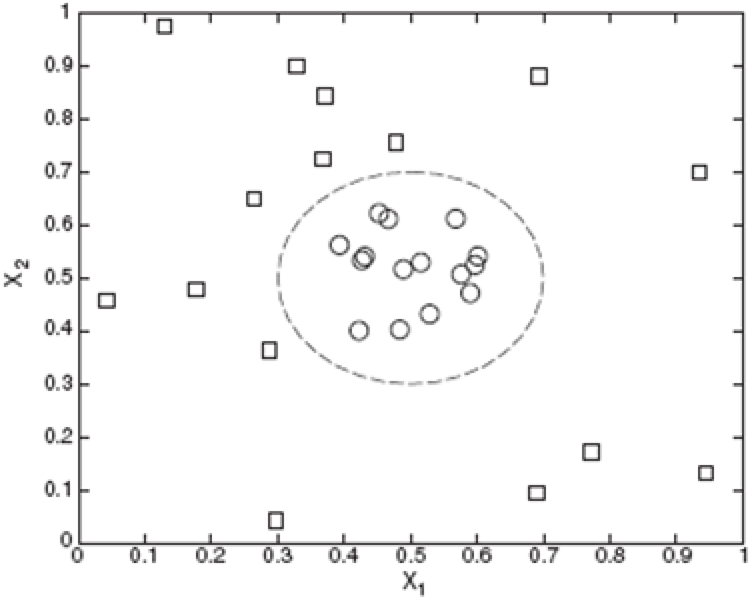


CSE 473
Pattern Recognition



Nonlinear Classifier

Review of Perceptron’s Capability

Recall the AND or OR functions

x_1	x_2	AND		OR	
0	0	0		0	
0	1	0		1	
1	0	0		1	
1	1	1		1	

Review of Perceptron’s Capability

Recall the AND or OR functions

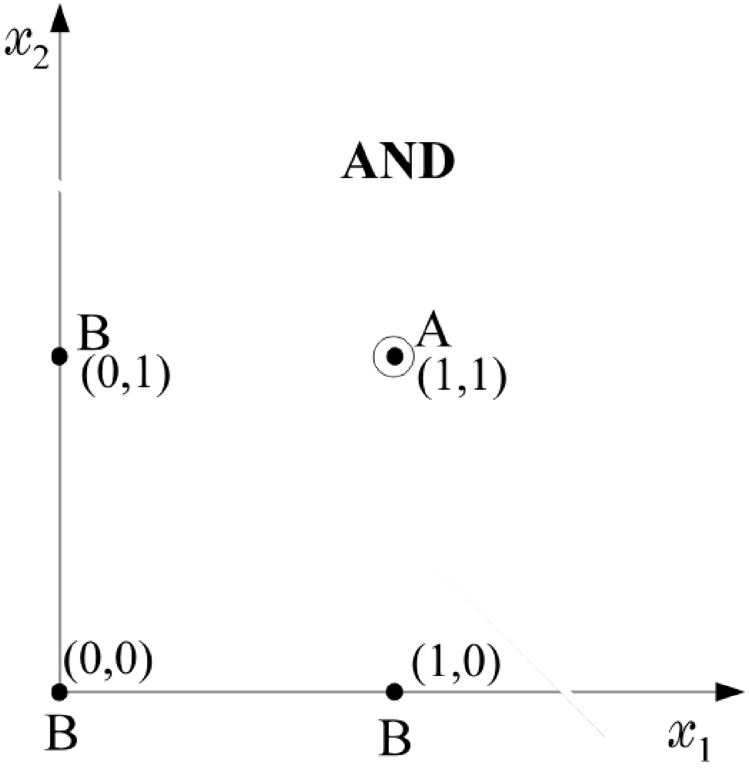
x_1	x_2	AND	Class	OR	Class
0	0	0	B	0	B
0	1	0	B	1	A
1	0	0	B	1	A
1	1	1	A	1	A

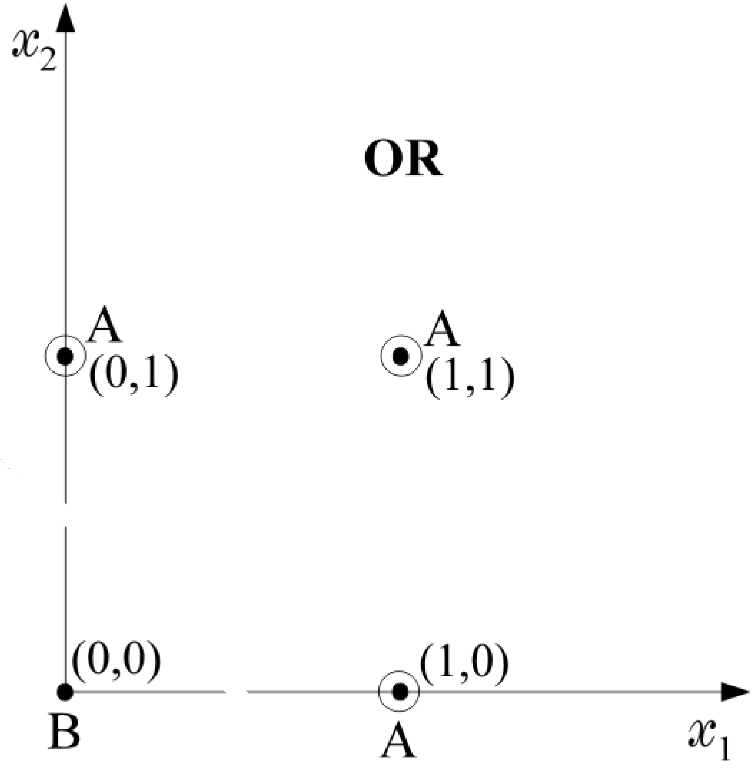
Review of Perceptron’s Capability

Can you remember the perceptron’s capability to separate them?

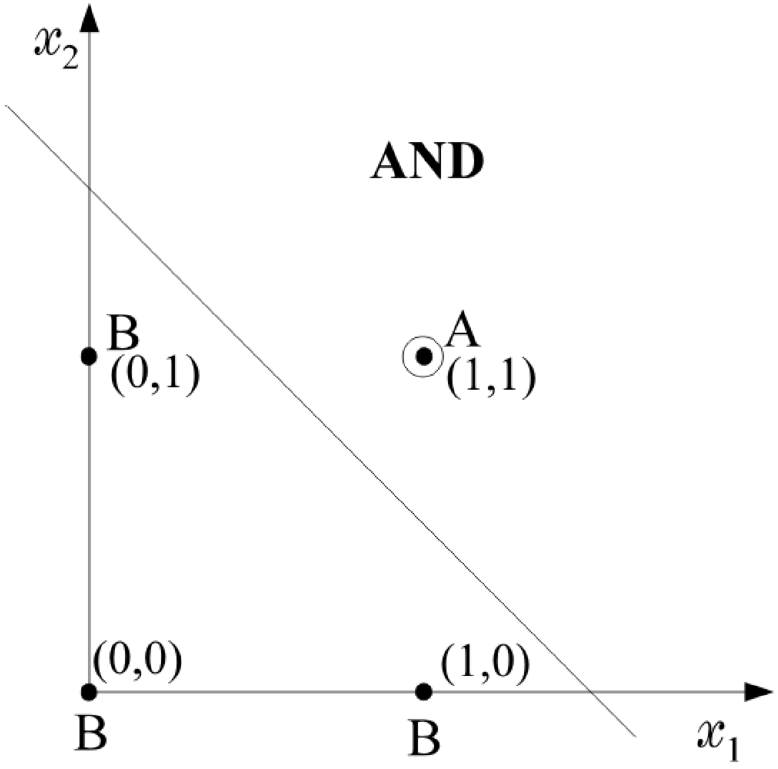
x_1	x_2	AND	Class	OR	Class
0	0	0	B	0	B
0	1	0	B	1	A
1	0	0	B	1	A
1	1	1	A	1	A

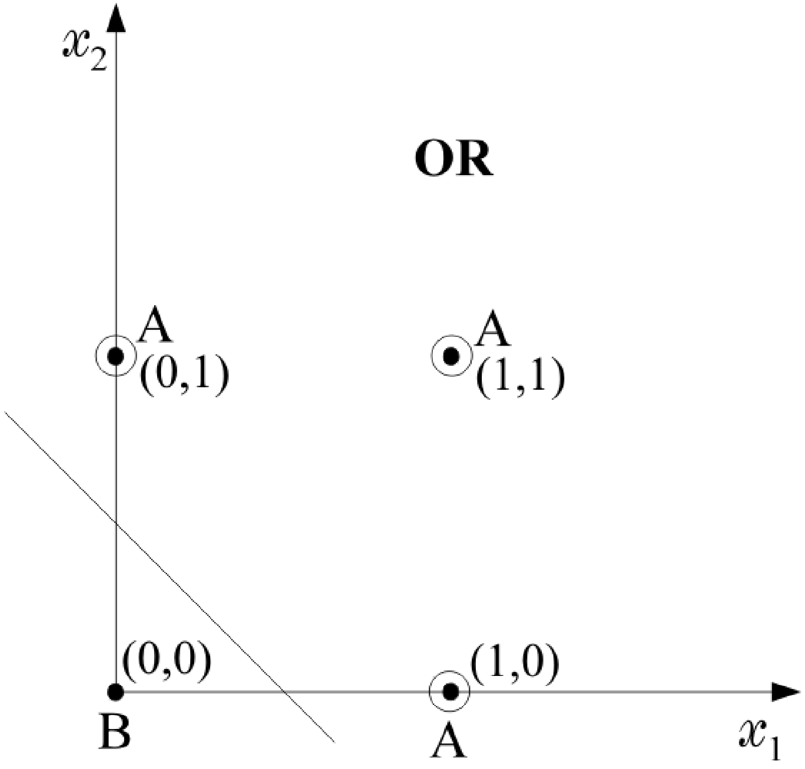
Review of Perceptron's Capability





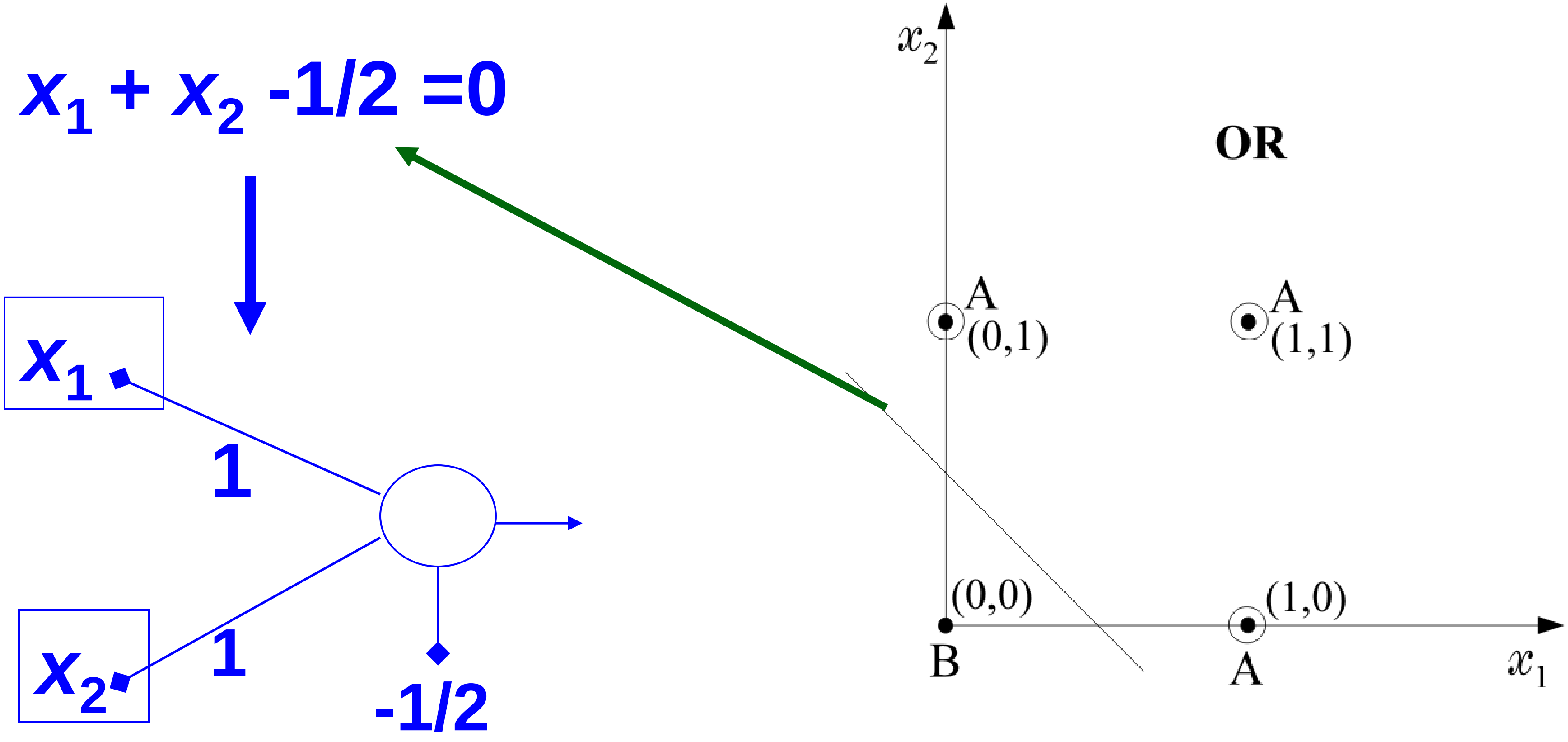
Review of Perceptron’s Capability



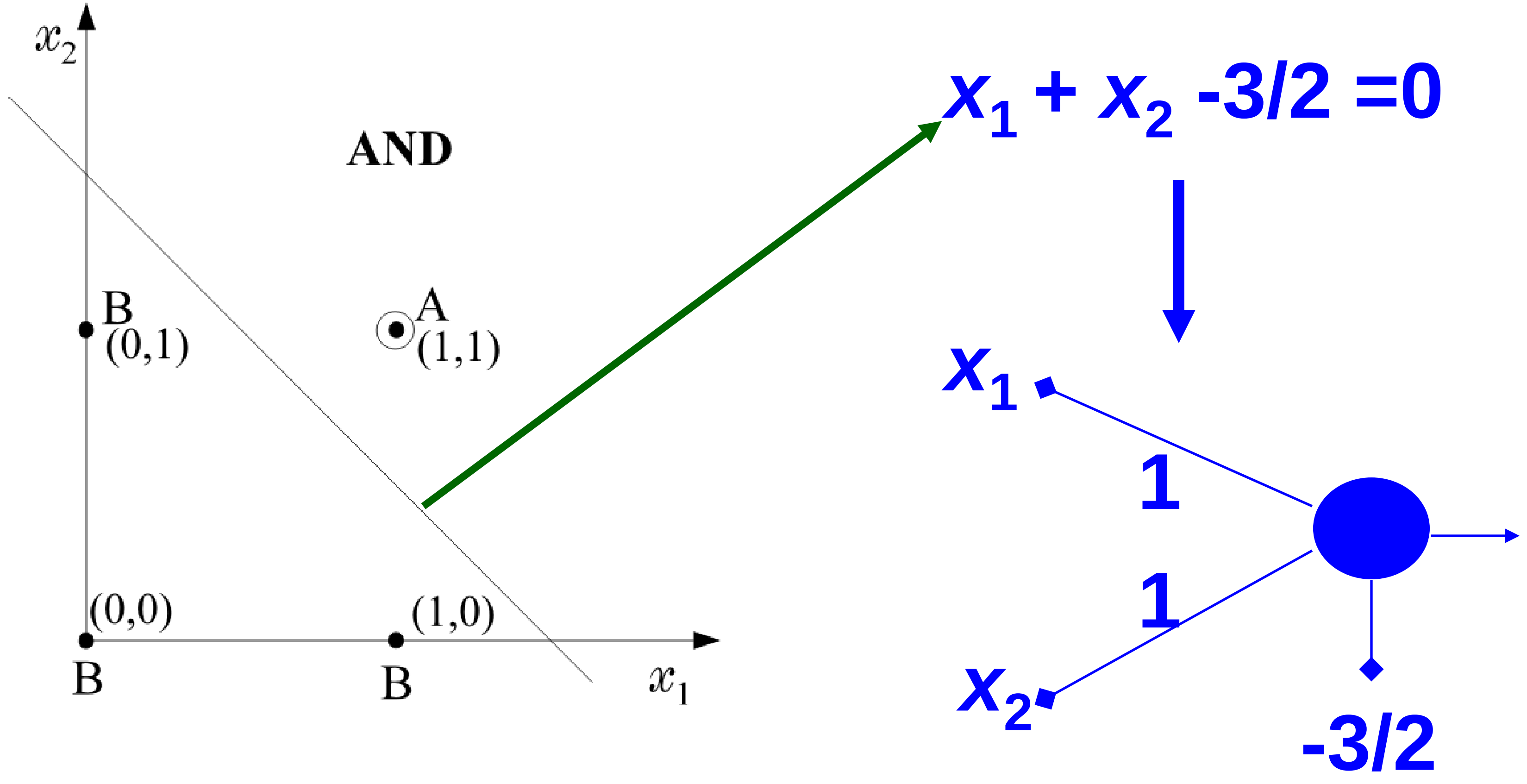


Review of Perceptron’s Capability

$$x_1 + x_2 - 1/2 = 0$$



Review of Perceptron’s Capability



$$x_1 + x_2 -3/2 =0$$

Review of Perceptron's Capability

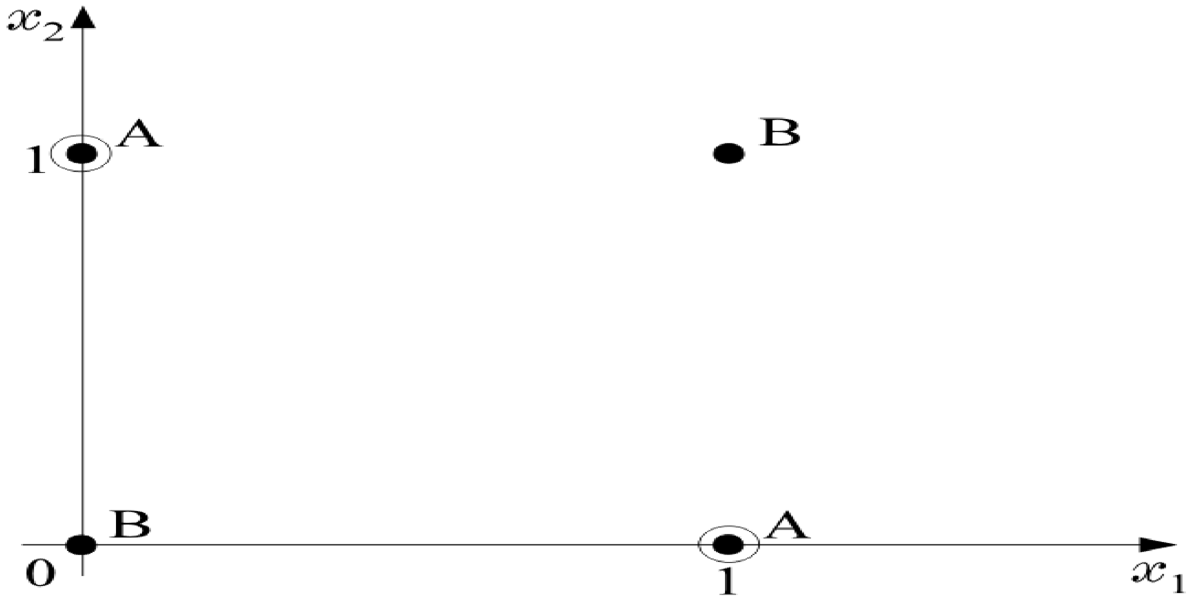
Now recall the XOR function

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

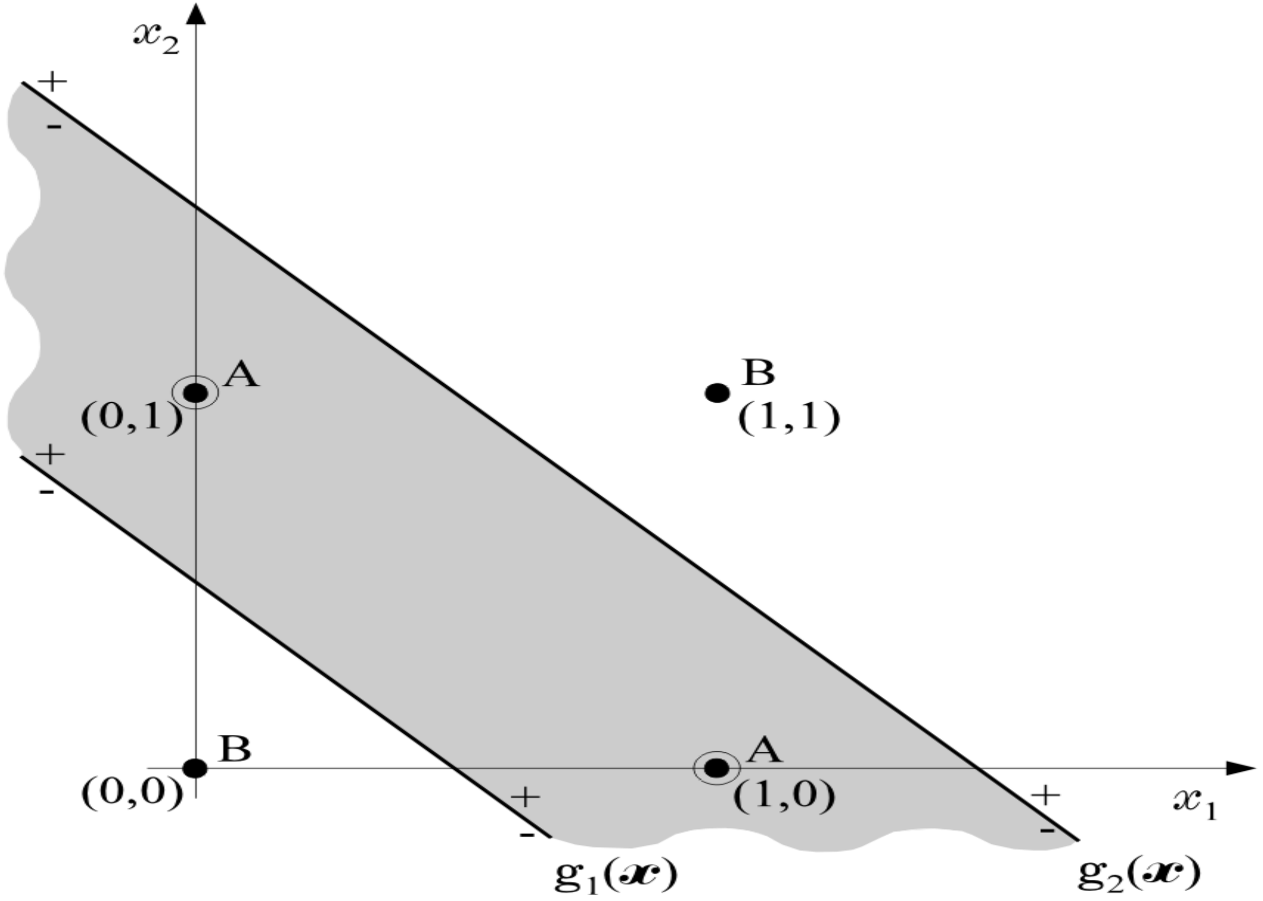
Review of Perceptron’s Capability

Now recall the XOR function

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

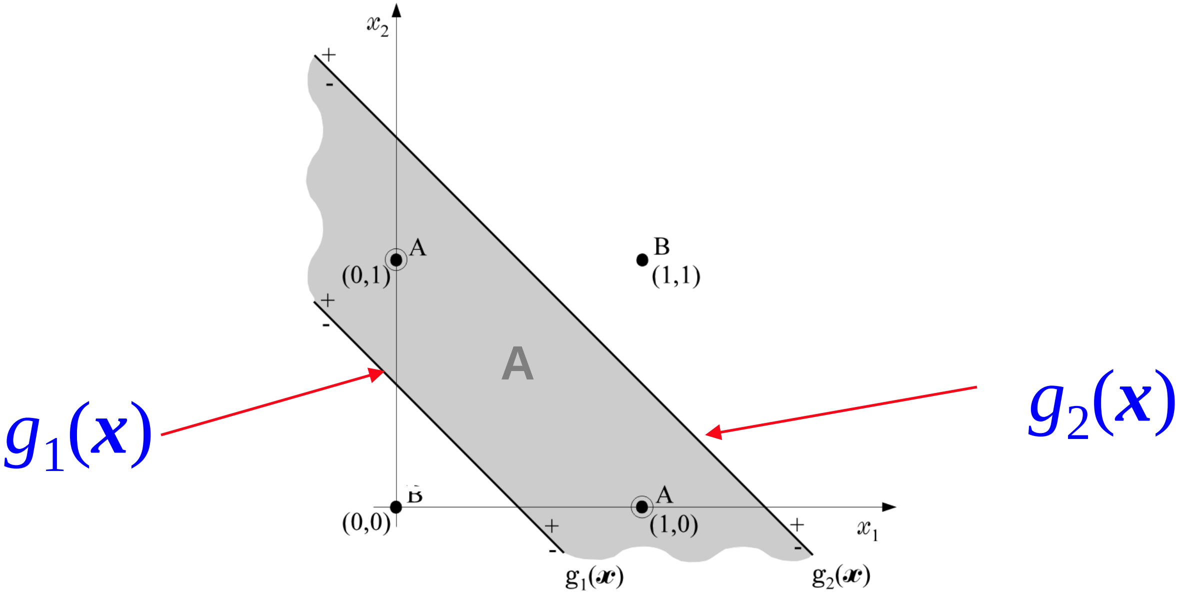


Review of Perceptron’s Capability



- For the XOR problem, draw two lines instead of one

Review of Perceptron’s Capability



- Each of them is realized by a perceptron.

$$y_i = f(g_i(\underline{x})) = \begin{cases} 0 \\ 1 \end{cases} \quad i = 1, 2$$

- Find the position of $\underline{x}$ *w.r.t.* both lines, based on the values of y_1, y_2.

Review of Perceptron’s Capability

1st phase			
x_1	x_2	y_1	y_2
0	0	-	-
0	1	+	-
1	0	+	-
1	1	+	+

- Equivalently: The computations of the first phase perform a mapping $\underline{x} \rightarrow \underline{y} = [y_1, y_2]^T$

Review of Perceptron's Capability

1st phase			
x_1	x_2	y_1	y_2
0	0	0(-)	0(-)
0	1	1(+)	0(-)
1	0	1(+)	0(-)
1	1	1(+)	1(+)

- Equivalently: The computations of the first phase perform a mapping $\underline{x} \rightarrow \underline{y} = [y_1, y_2]^T$

Review of Perceptron's Capability

1st phase			
x_1	x_2	y_1	y_2
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

- Equivalently: The computations of the first phase perform a mapping $\underline{x} \rightarrow \underline{y} = [y_1, y_2]^T$

Review of Perceptron’s Capability

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)

- Equivalently: The computations of the first phase perform a mapping $\underline{x} \rightarrow \underline{y} = [y_1, y_2]^T$

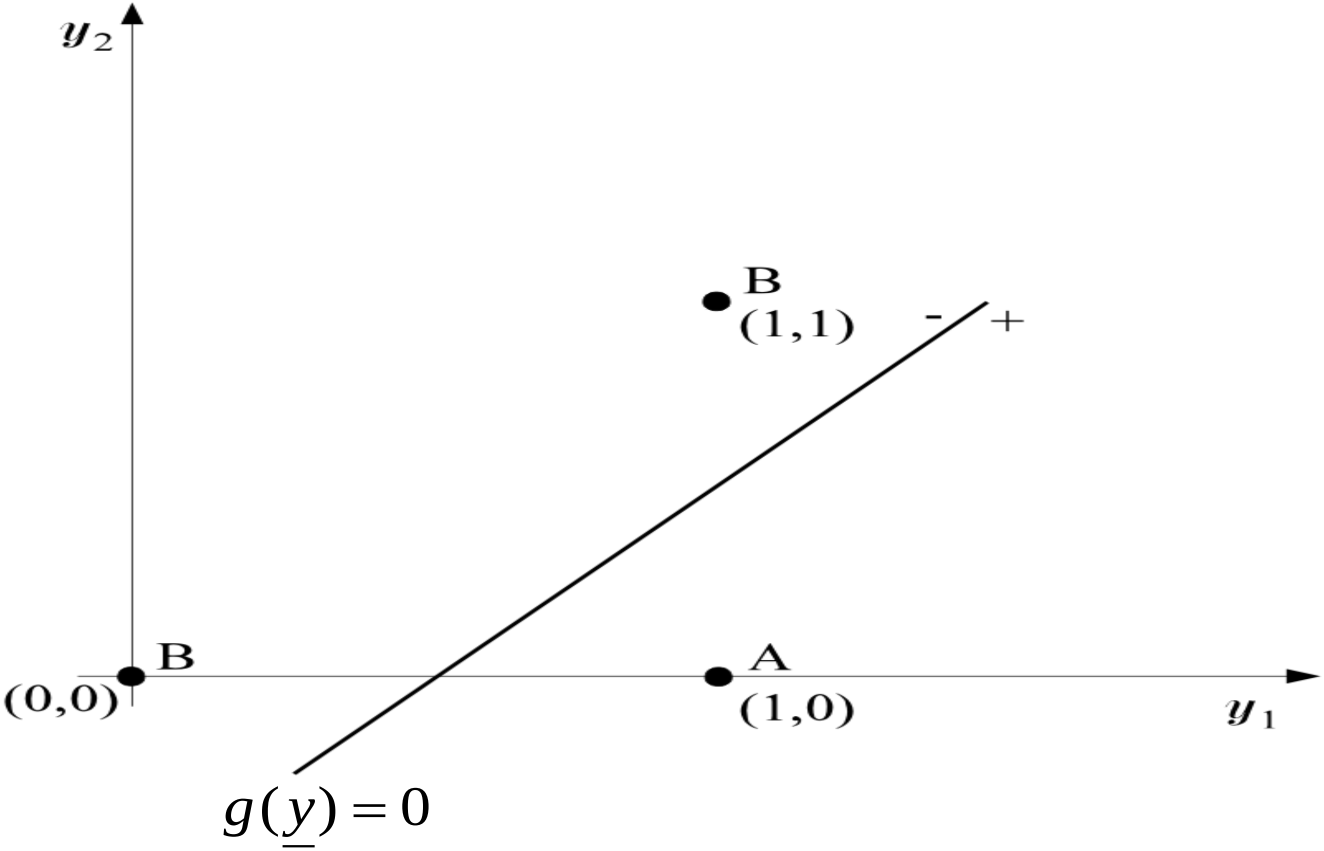
Review of Perceptron’s Capability

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)

Now classify based on $[y_1, y_2]$

Review of Perceptron’s Capability

The decision is now performed on the transformed $\underline{y}$ data.

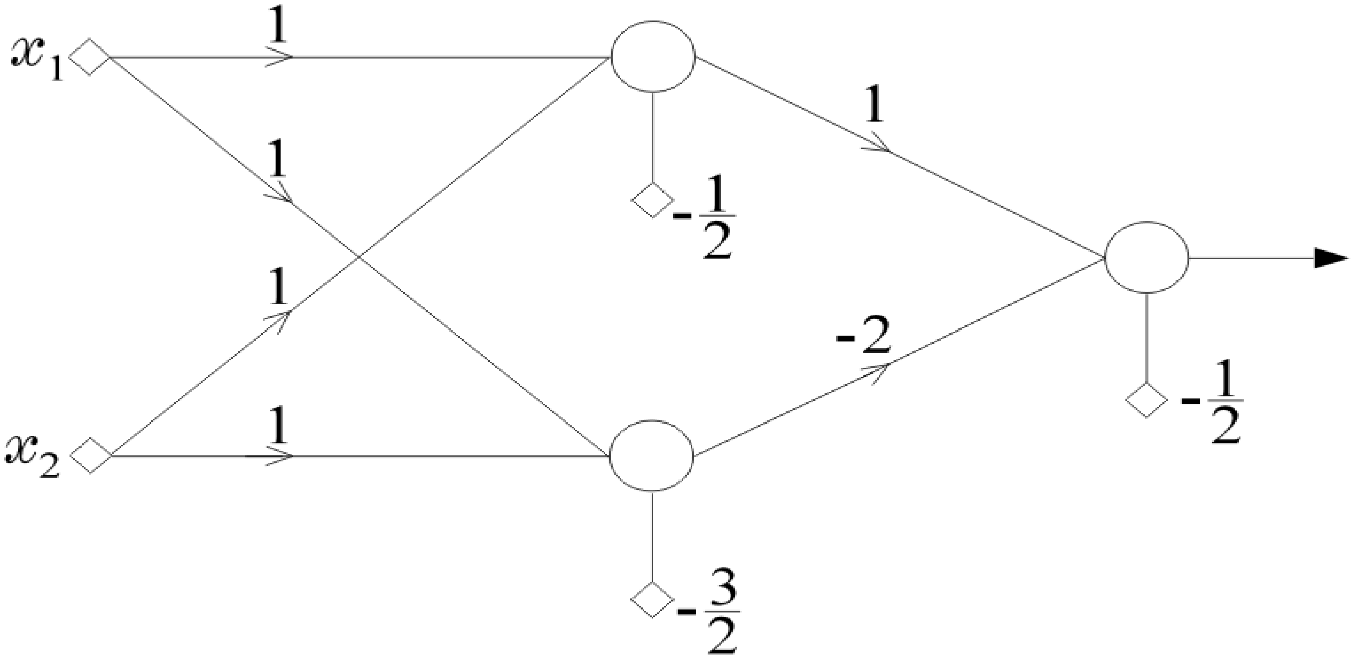


This can be performed via a second line, which can also be realized by a perceptron.

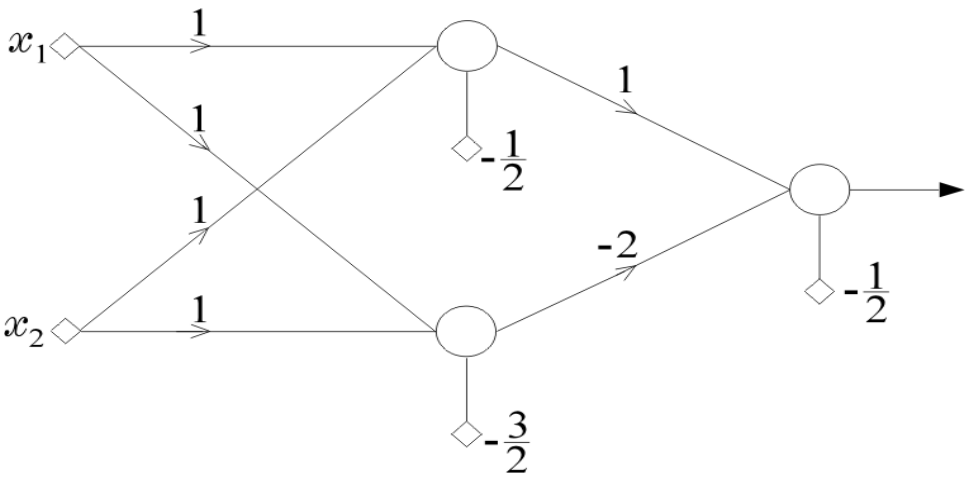
Two phases, Two Layers

- Computations of the first phase perform a mapping that transforms the nonlinearly separable problem to a linearly separable one.

- The architecture



Two Layer Perceptron



hidden layer

output layer

nodes realizes hyper planes:

$$g_1(\underline{x}) = x_1 + x_2 - \frac{1}{2} = 0$$

$$g_2(\underline{x}) = x_1 + x_2 - \frac{3}{2} = 0$$

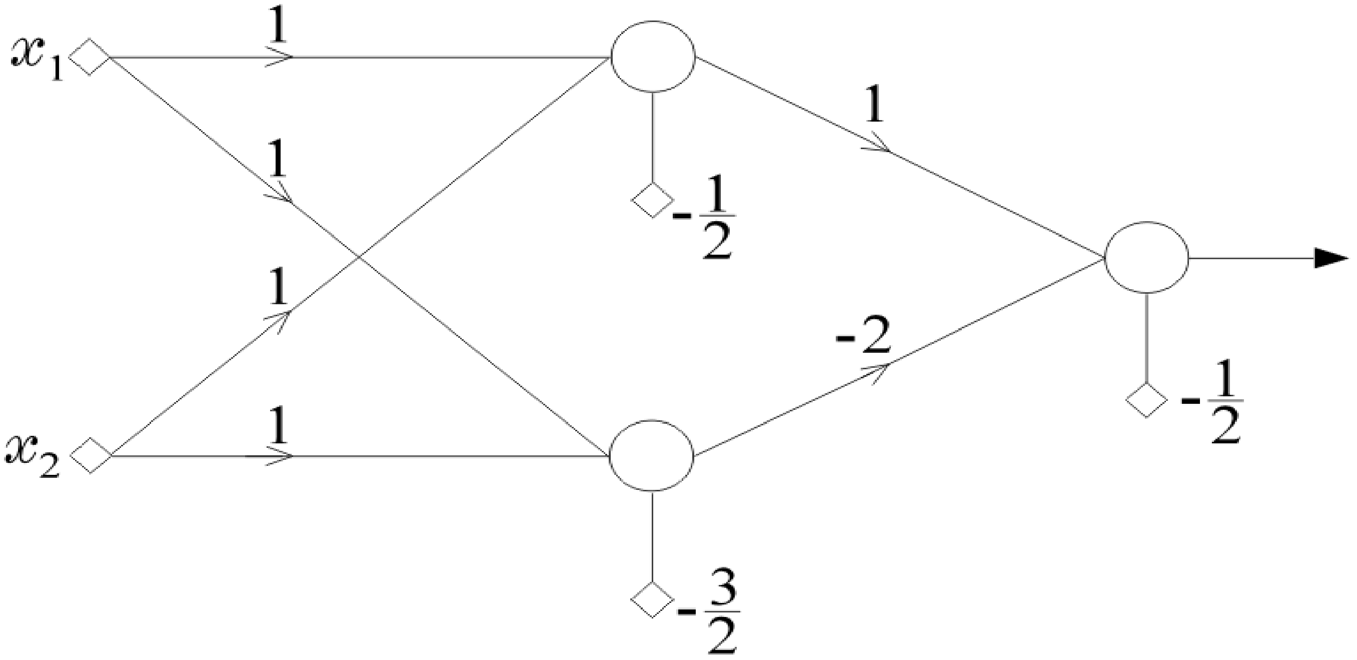
$$g(\underline{y}) = y_1 - 2y_2 - \frac{1}{2} = 0$$

Activation function:

$$f(.) = \begin{cases} 0 \\ 1 \end{cases}$$

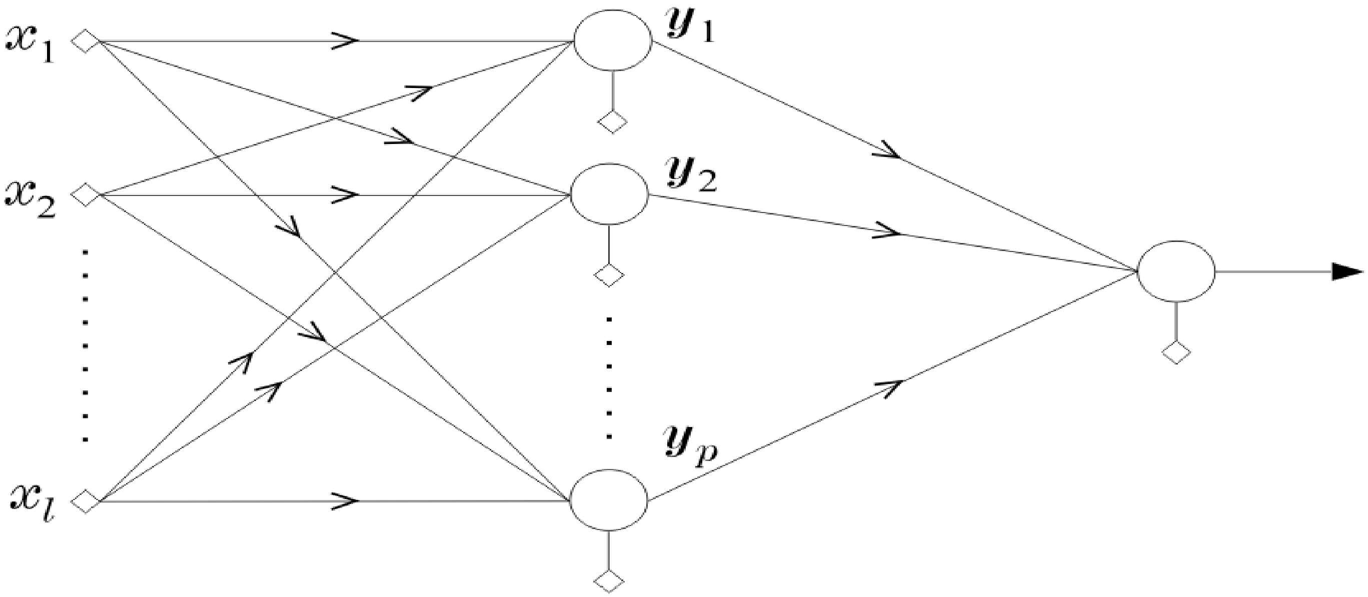
Classification Capabilities of Two Layer Perceptron

- The mapping performed by the first layer neurons is onto the vertices of the unit side square, e.g., (0, 0), (0, 1), (1, 0), (1, 1).



Classification Capabilities of Two Layer Perceptron

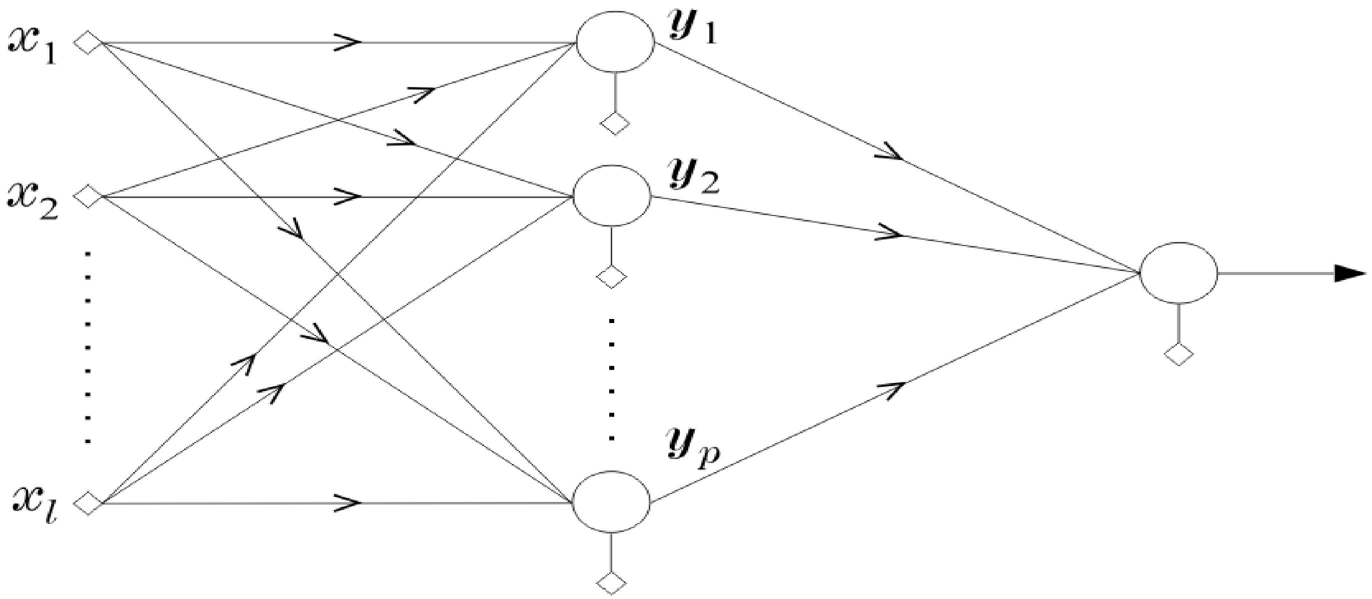
- Consider a more general case,



$$\underline{x} \in R^l$$

$$\underline{x} \rightarrow \underline{y} = [y_1,\ldots y_p]^T, \; y_i \in \{0,1\} \;\; i = 1, 2,\ldots p$$

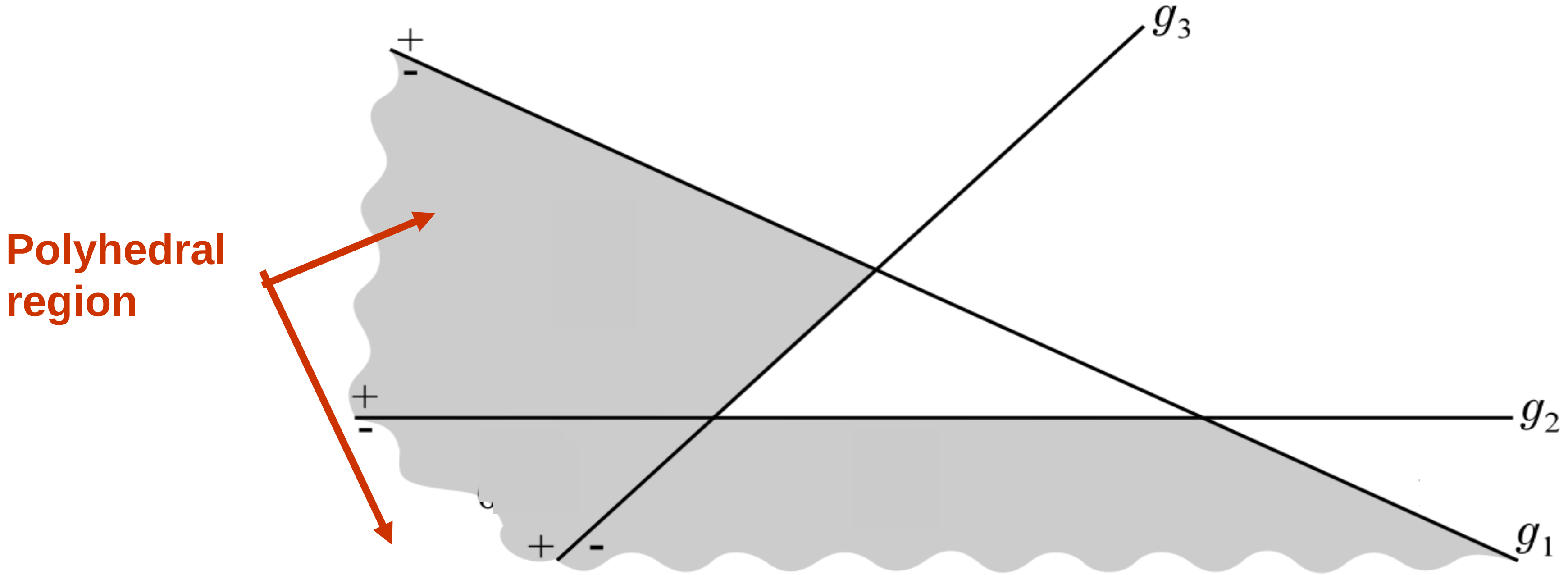
Classification Capabilities of Two Layer Perceptron



- maps a vector onto the vertices of the unit side hypercube, *Hp*
- mapping is through *p* neurons each realizing a hyper plane.
- The output of each of these neurons is 0 or 1

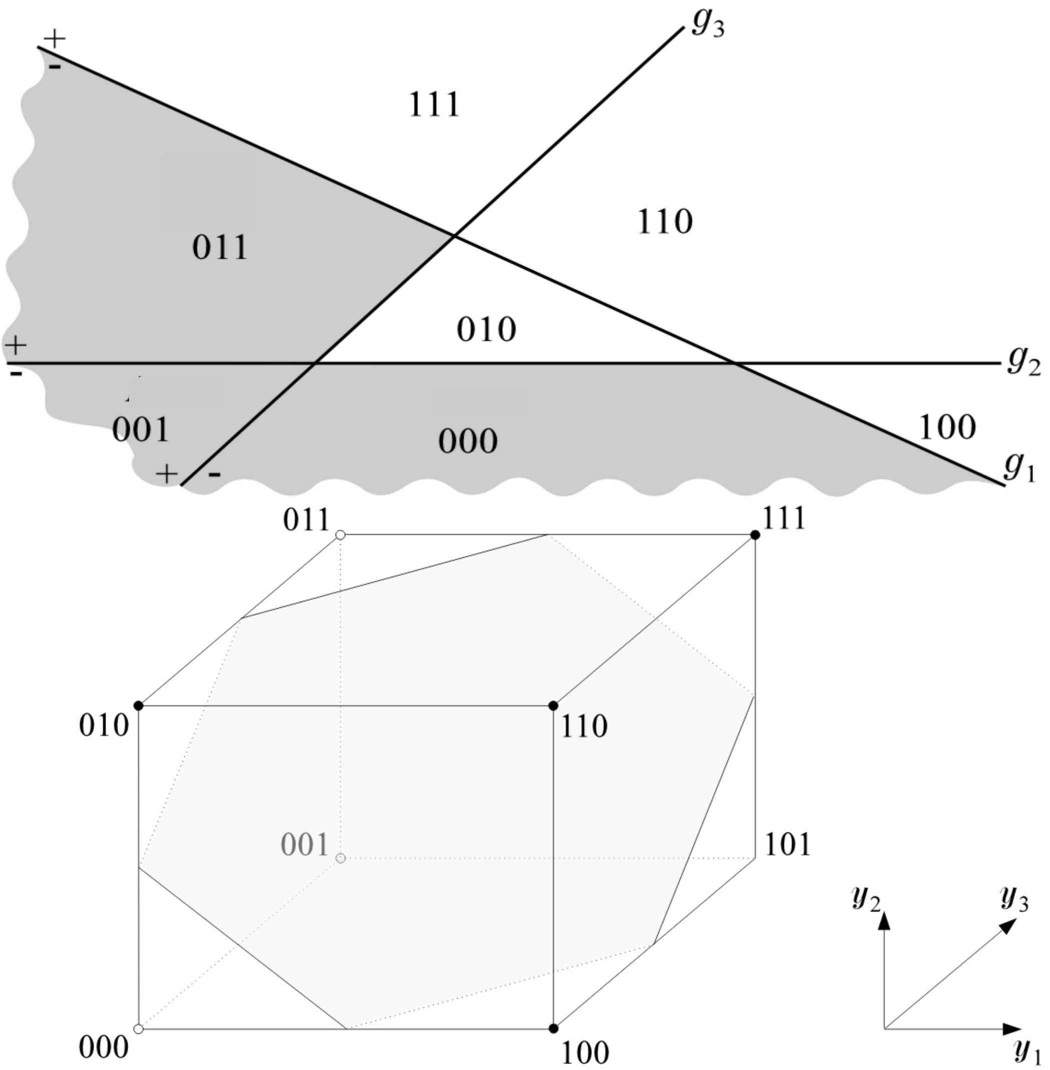
Classification Capabilities of Two Layer Perceptron

- Intersections of hyperplanes form regions.



Classification Capabilities of Two Layer Perceptron

- Intersections of hyperplanes form regions.
- Each region corresponds to a vertex of the H_p unit hypercube.



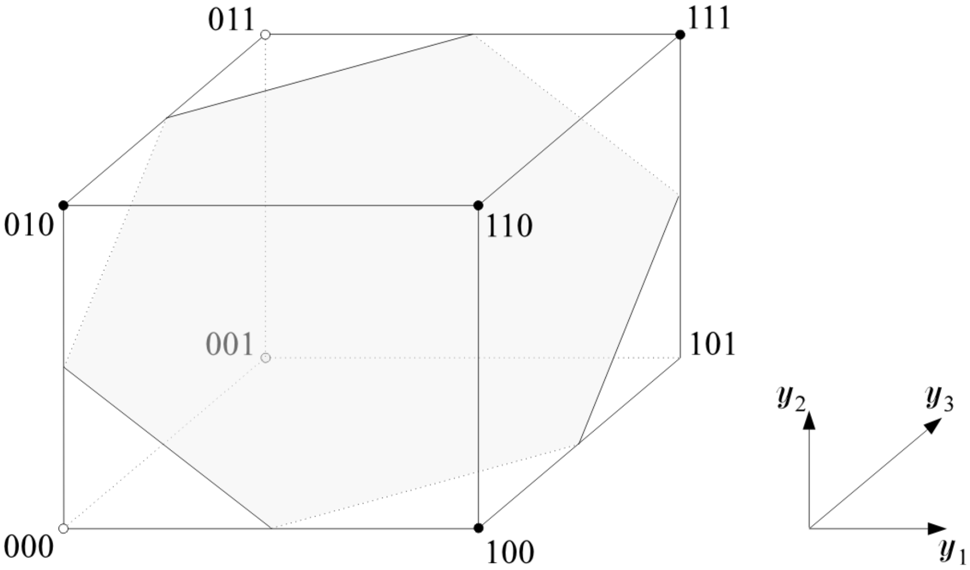
Classification Capabilities of Two Layer Perceptron

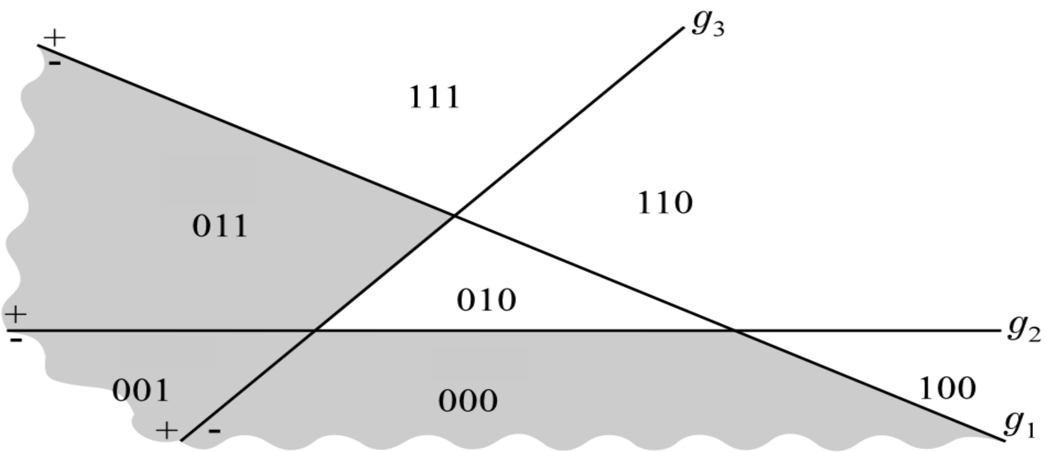
- For example, the 001 vertex corresponds to the region which is located

  to the (-) side of $g_1(\underline{x})=0$

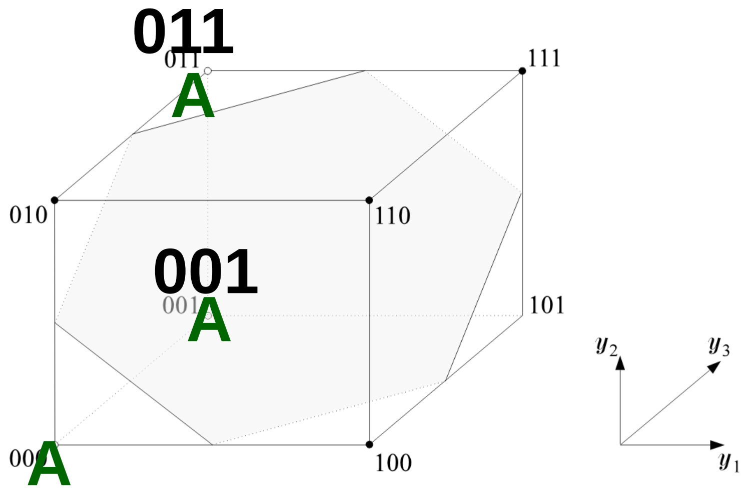
  to the (-) side of $g_2(\underline{x})=0$

  to the (+) side of $g_3(\underline{x})=0$

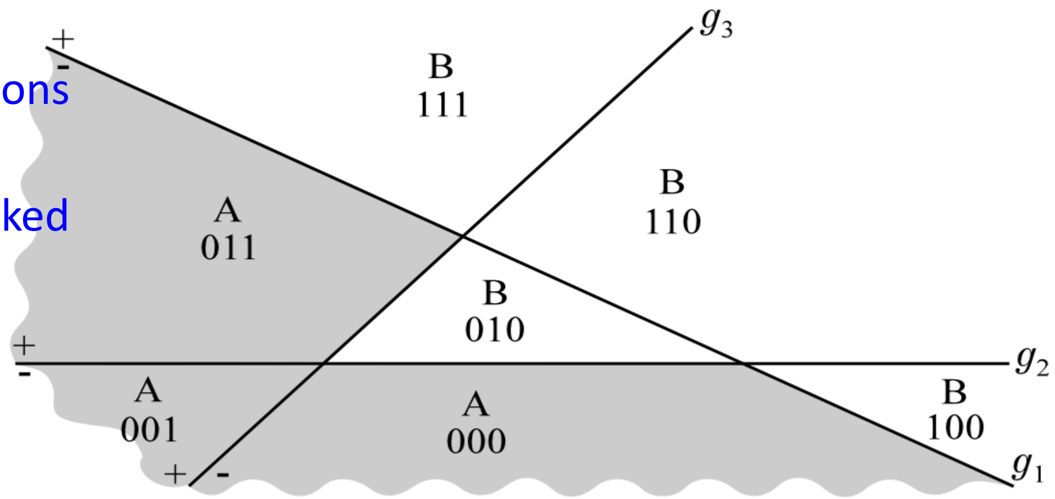




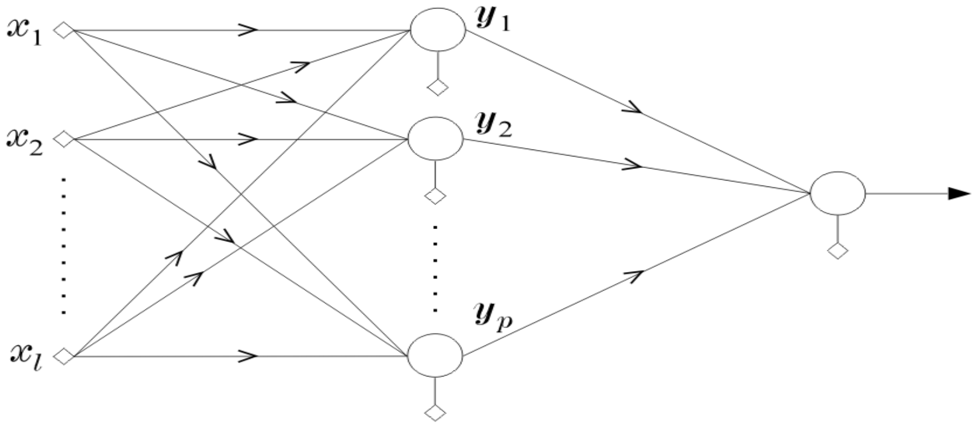
Classification Capabilities of Two Layer Perceptron

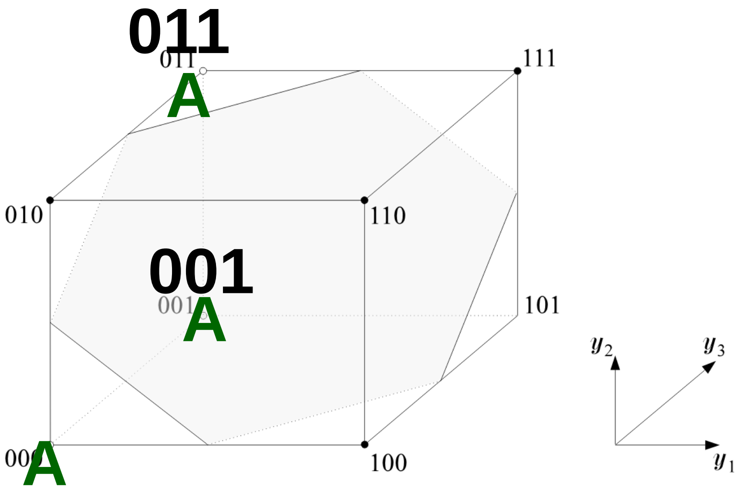


- A two-class problem
  - **Class A patterns** from regions marked as A
  - **Class B patterns** from regions marked as B

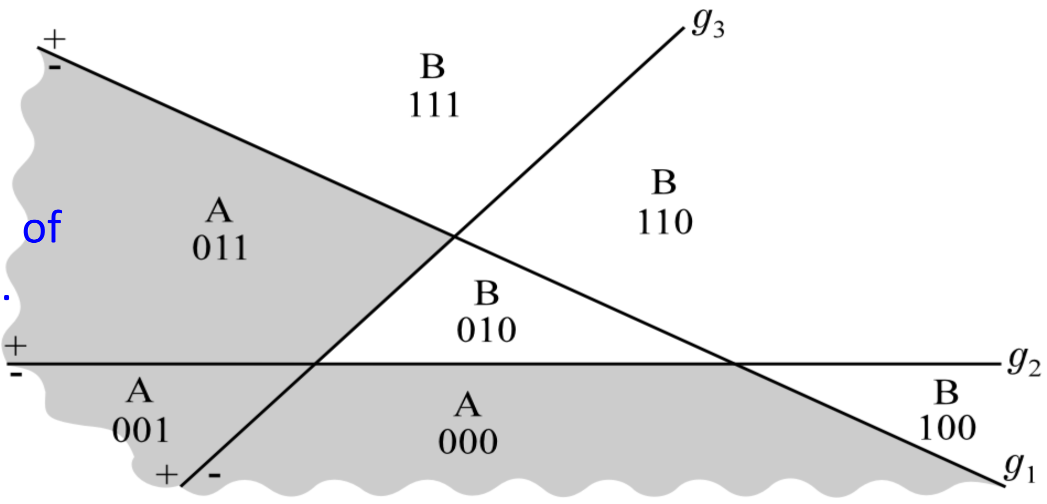


Classification Capabilities of Two Layer Perceptron

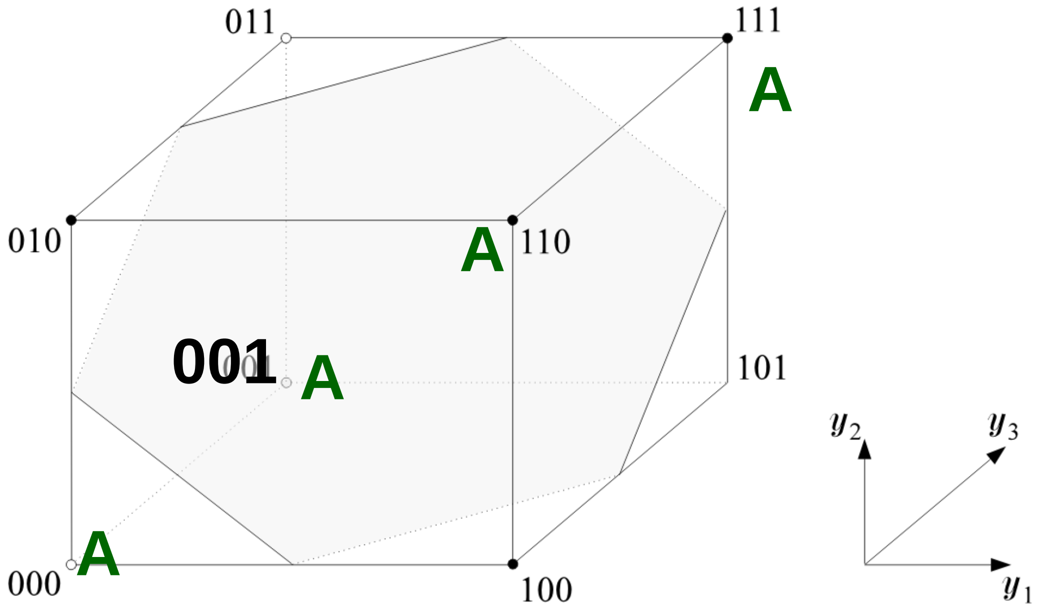




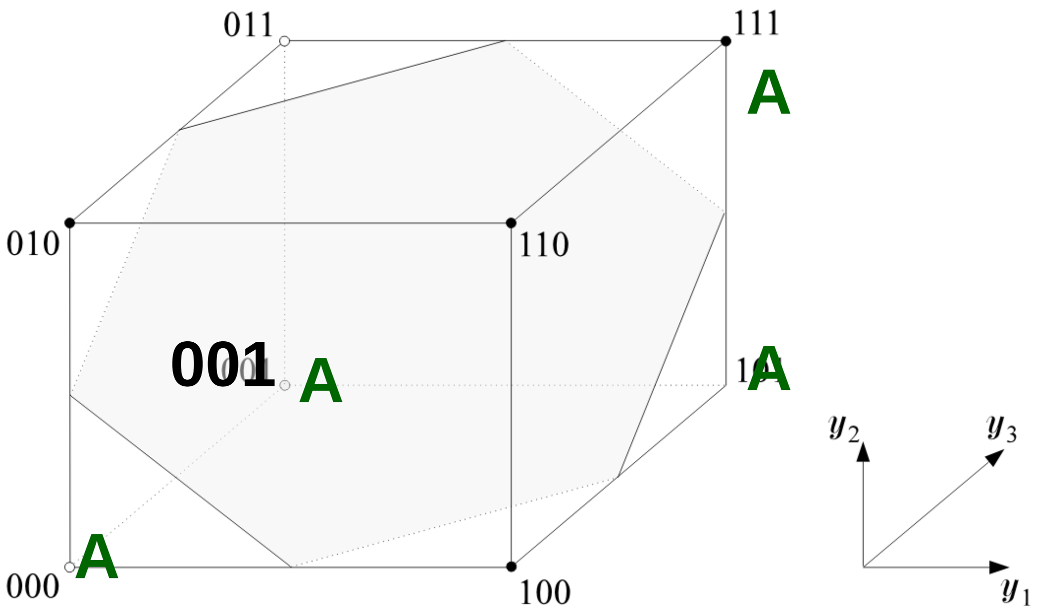
- The output neuron
  - realizes another hyperplane
  - separates the hypercube.
  - can classify vectors consisting of some unions of polyhedral regions.



Classification Capabilities of Two Layer Perceptron



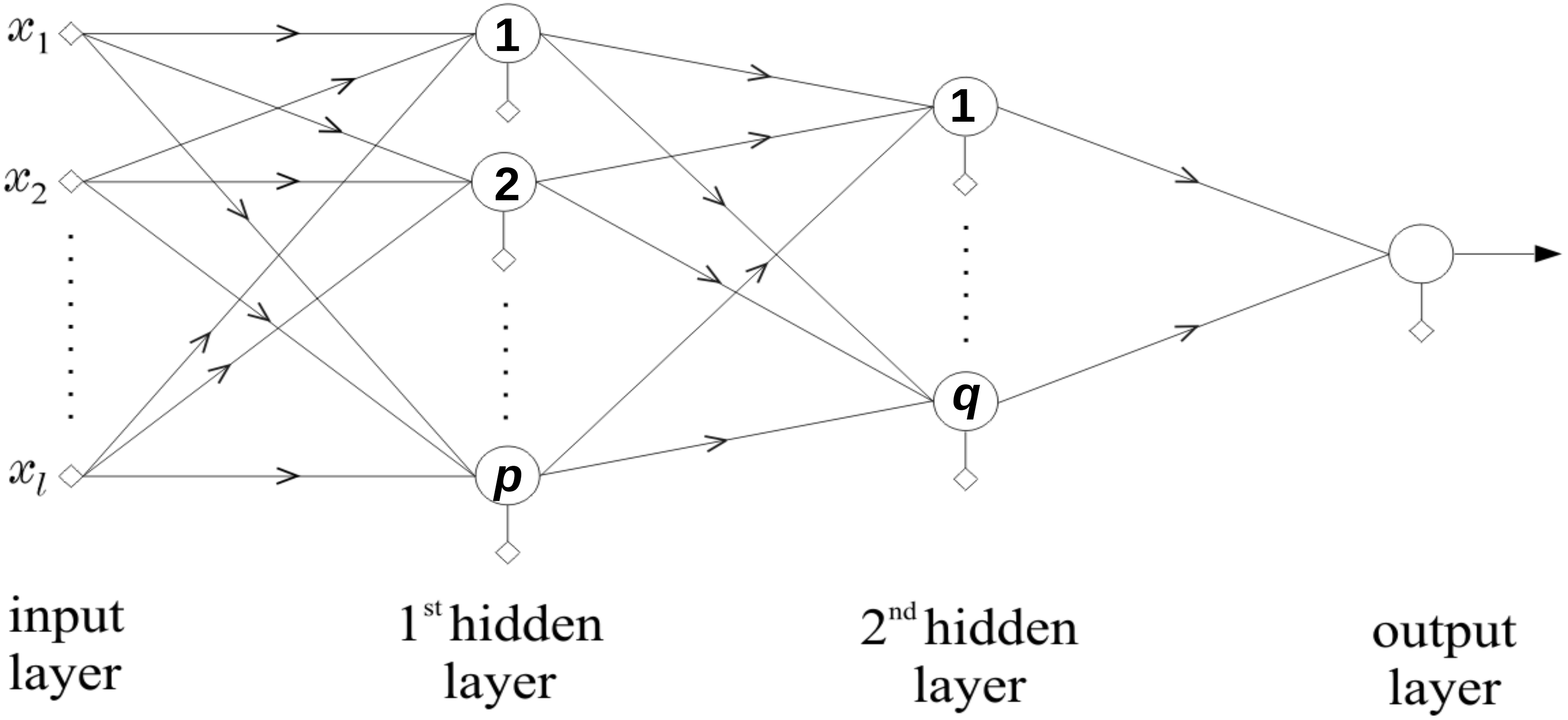
Classification Capabilities of Two Layer Perceptron



- The output neuron, *i.e.*, a 2 layer perceptron
  - – cannot classify vectors consisting of arbitrary unions of polyhedral regions.

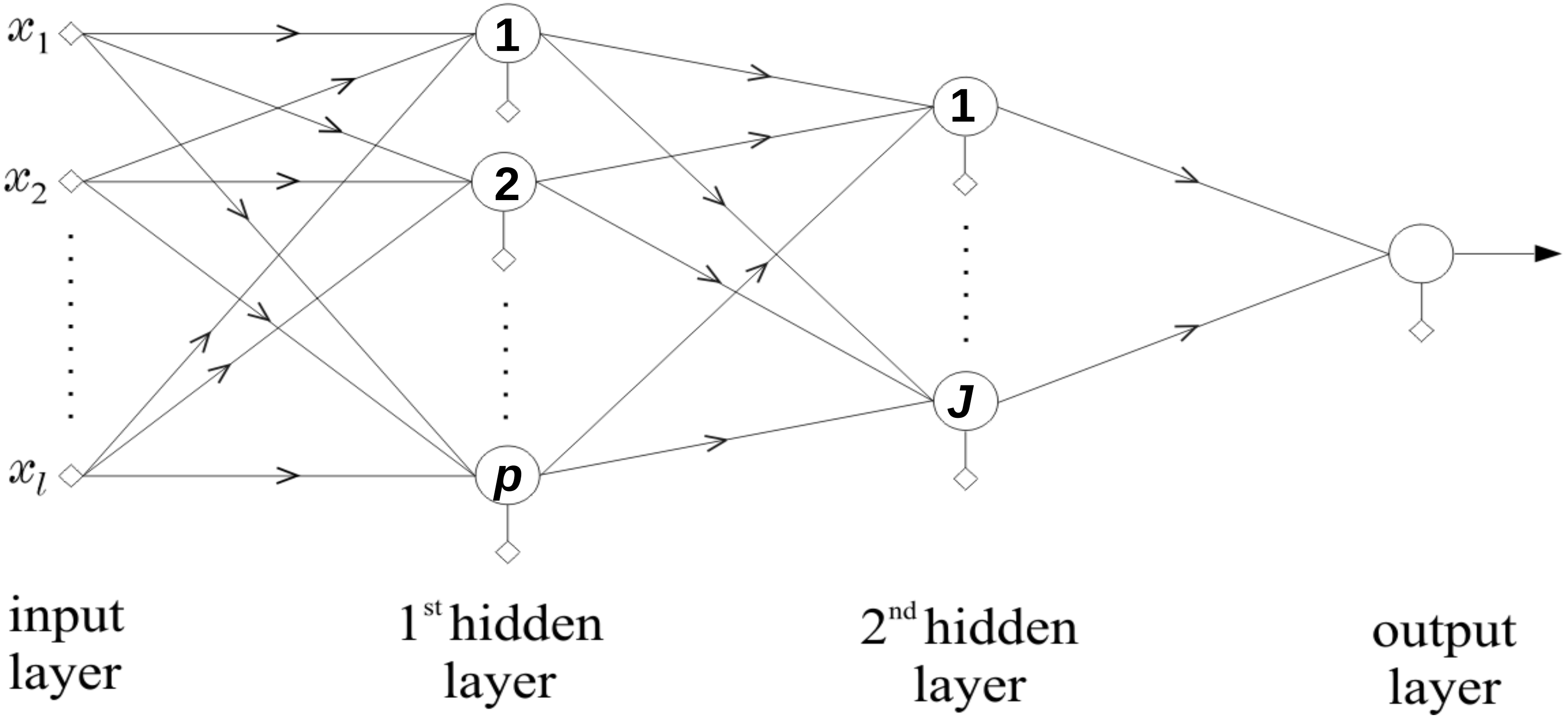
Solution: Three Layer Perceptron

- capable to classify vectors consisting of **ANY** union of polyhedral regions.
  - The idea is similar to the XOR problem.
  - Realizes more than one planes in the $\underline{y} \in R^p$ space.



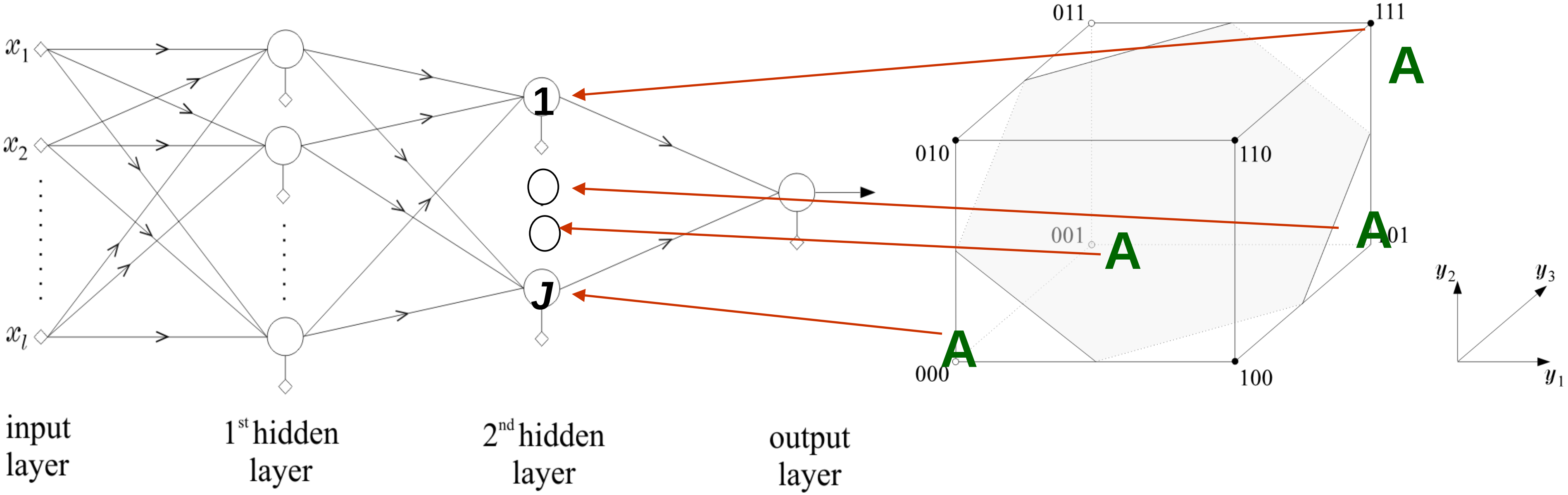
How does The Three Layer Perceptron Do It?

- Let, any *J* polyhedral regions constitutes vectors of class ***A.***
- Learn a neuron in the 2nd hidden layer for each of *J* regions



How does The Three Layer Perceptron Do It?

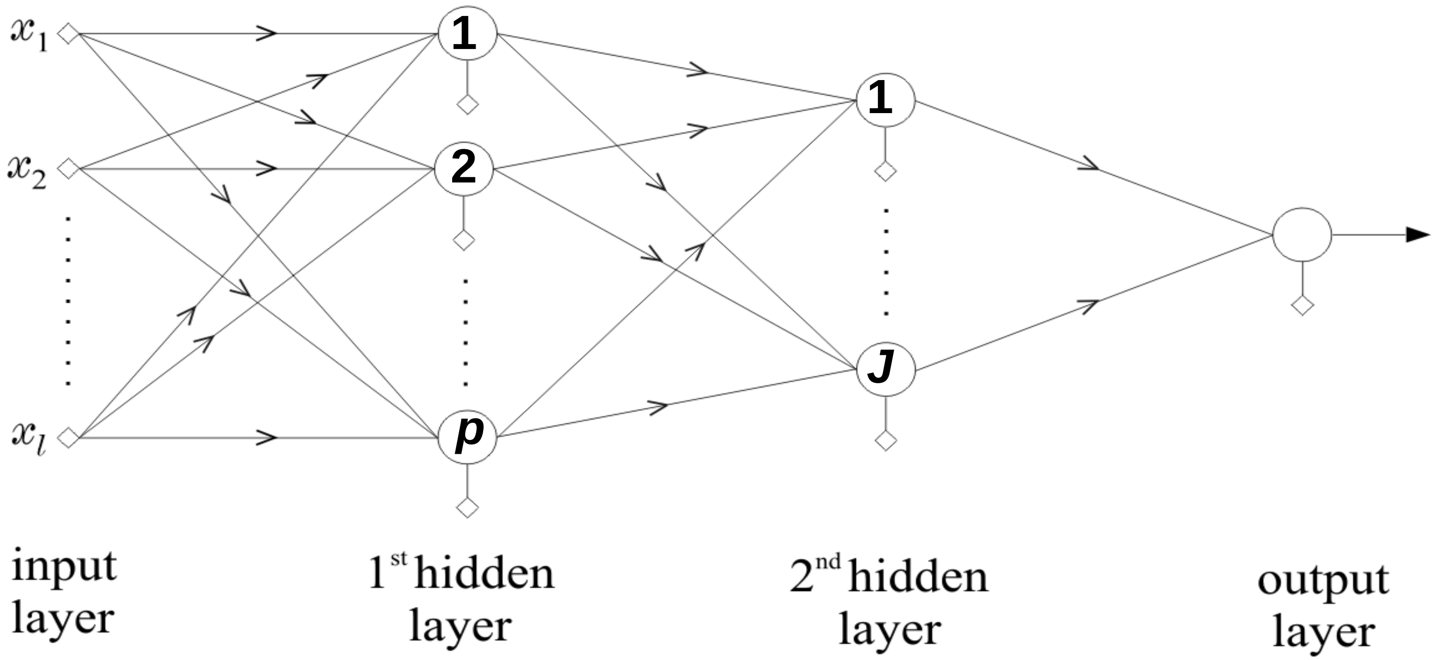
- Learn a neuron in the 2nd hidden layer for each of *J* regions



How does The Three Layer Perceptron Do It?

- For training vectors of a particular region of class ***A***, only one of the 2nd-layer neuron produces 1, the rest of neurons produce 0.

- Now realize the output neuron as an OR gate.

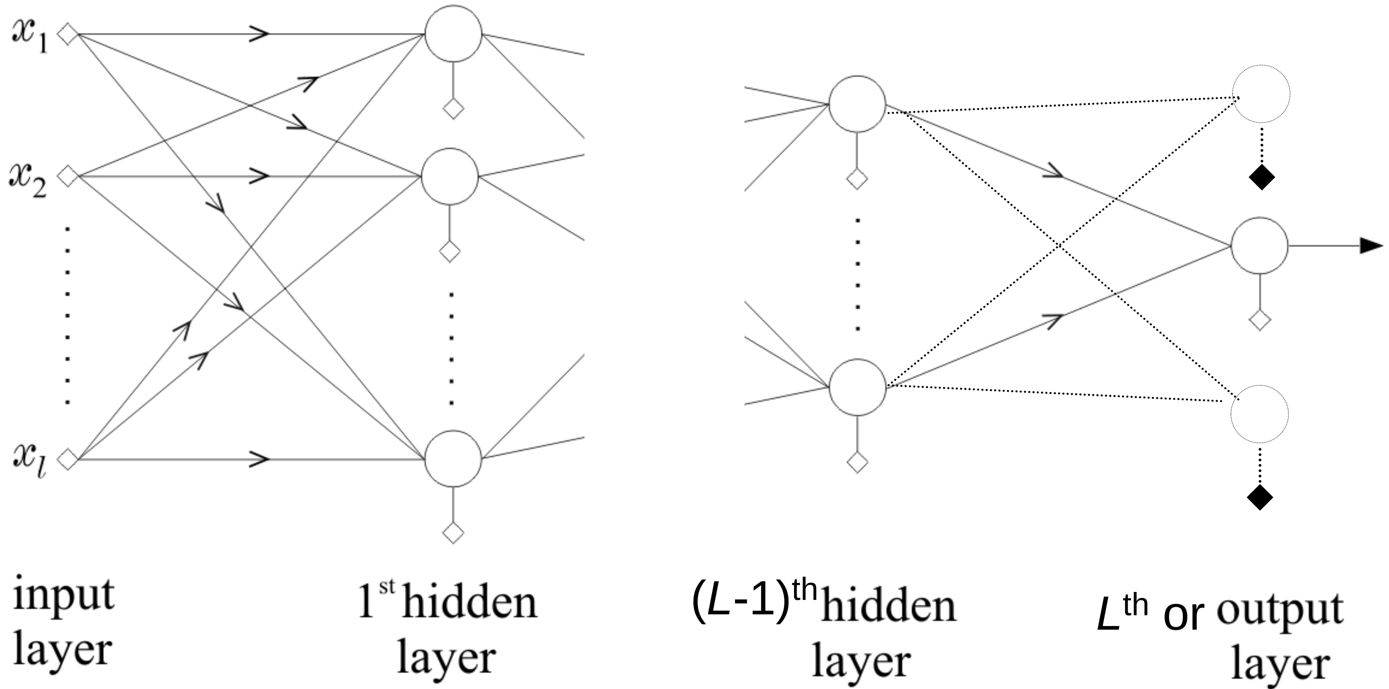


Training of a Multi Layer Perceptron (MLP)

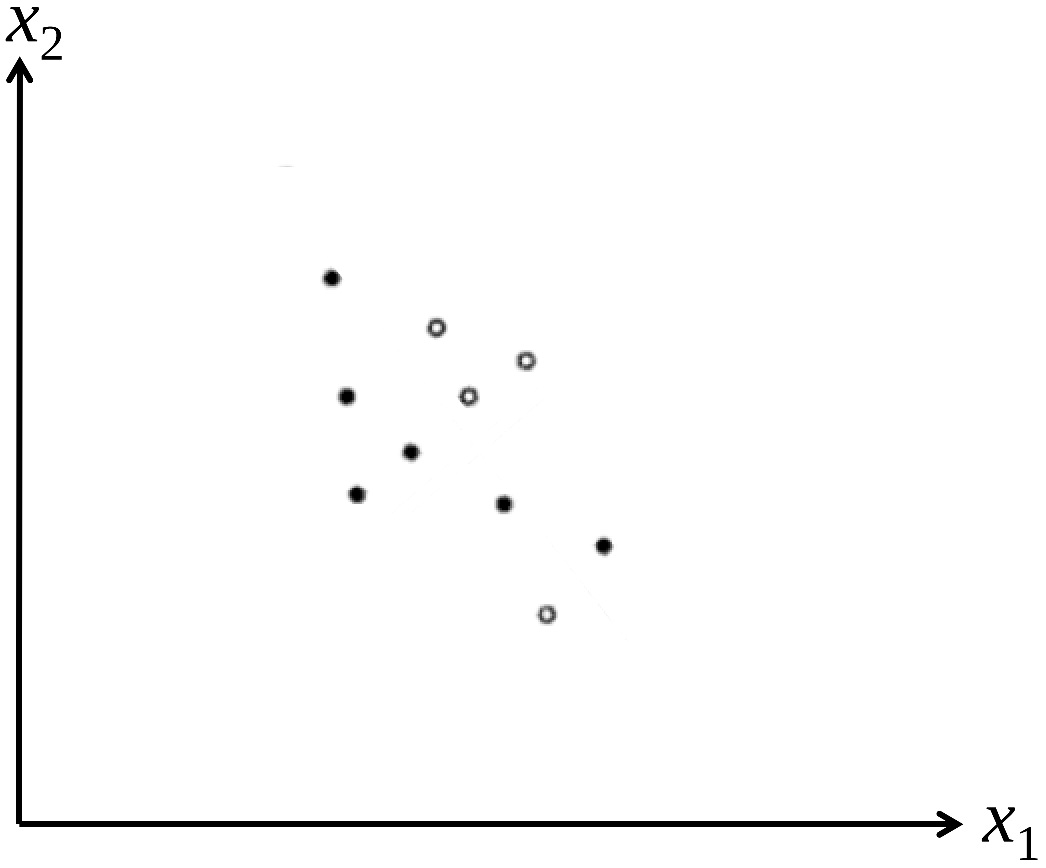
- use rationale and develop a structure that classifies **correctly all** the training patterns.

  OR

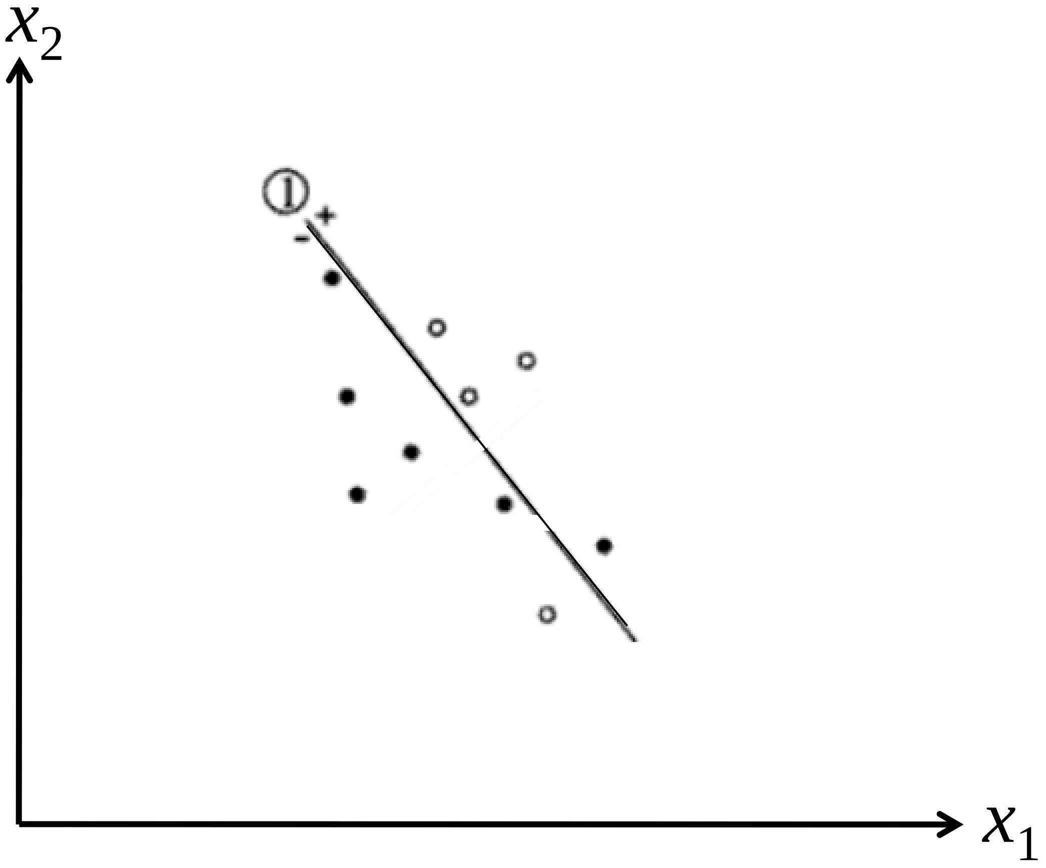
- choose a structure and compute the synaptic weights to optimize a cost function.

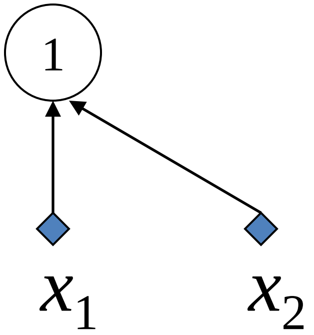


Algorithm Based on Exact Classification of Training Examples

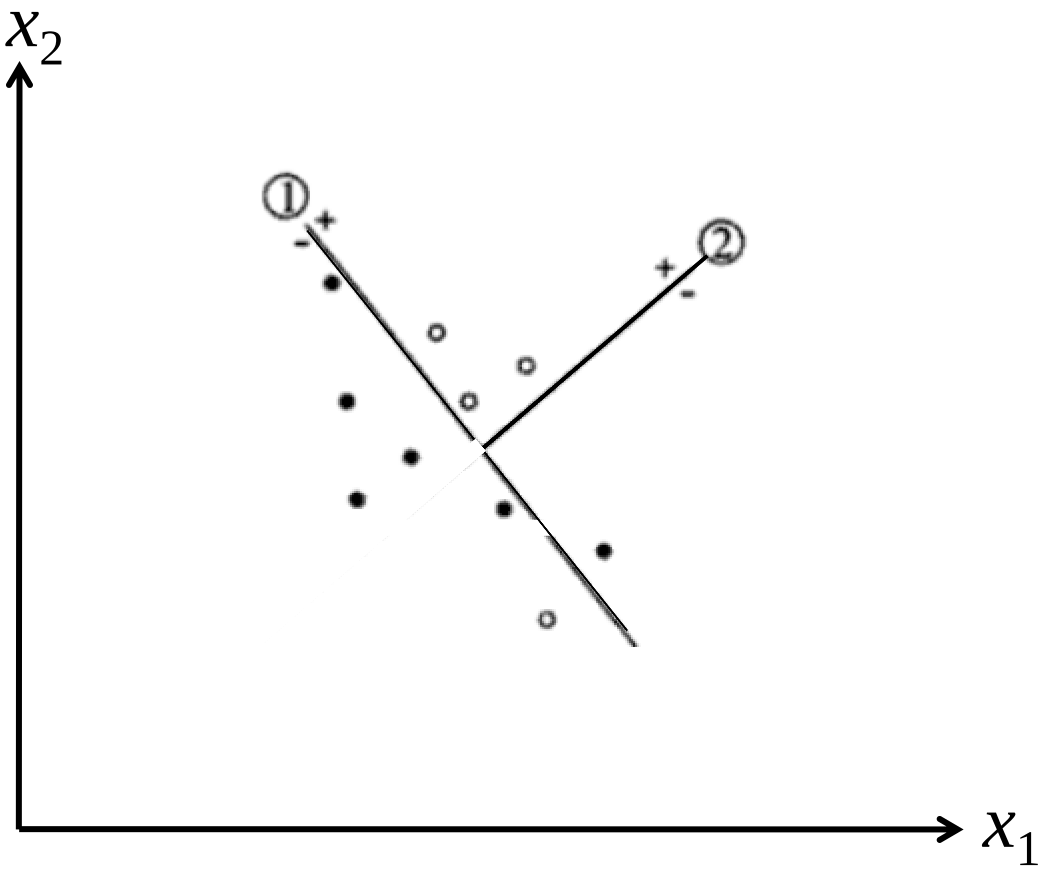


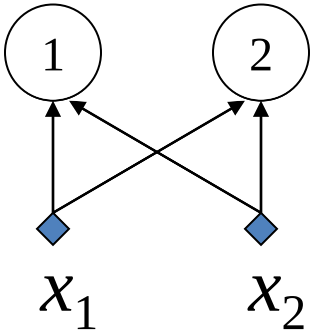
Algorithm Based on Exact Classification of Training Examples



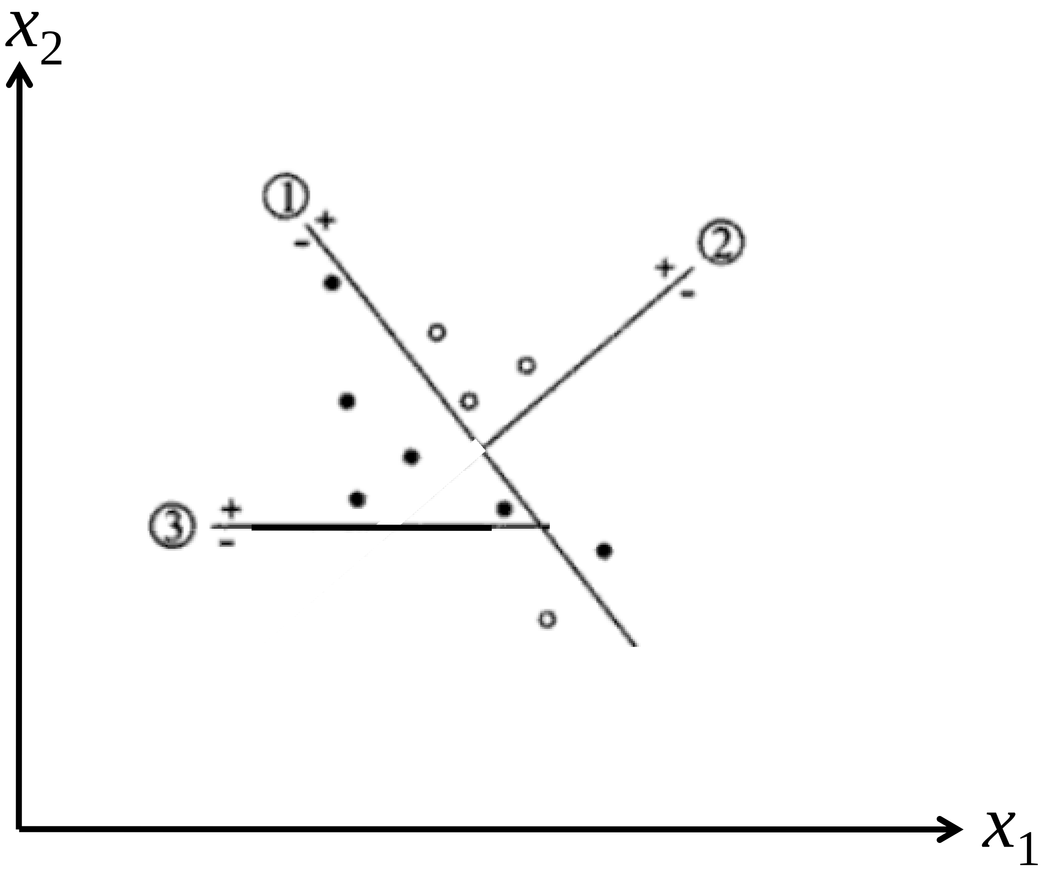


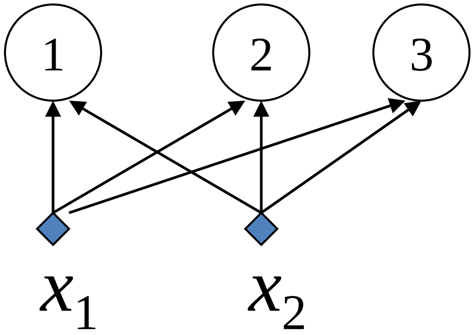
Algorithm Based on Exact Classification of Training Examples



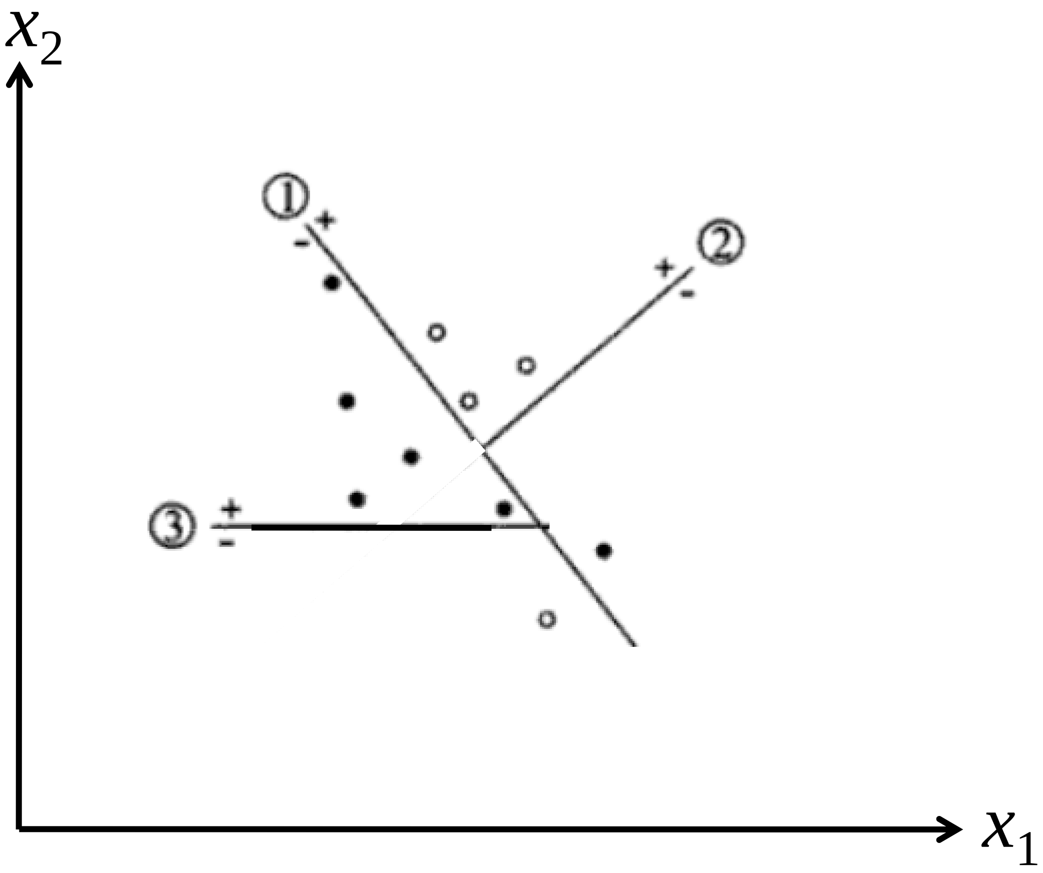


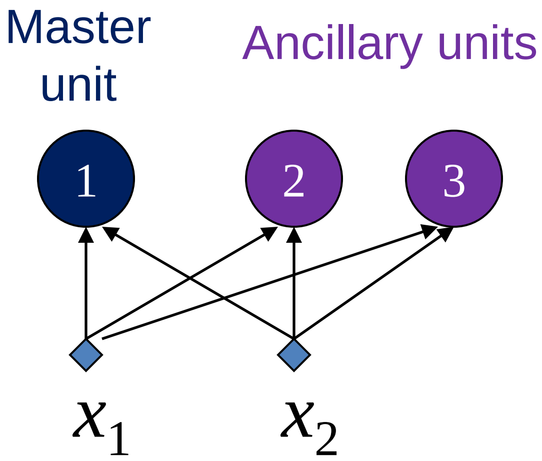
Algorithm Based on Exact Classification of Training Examples





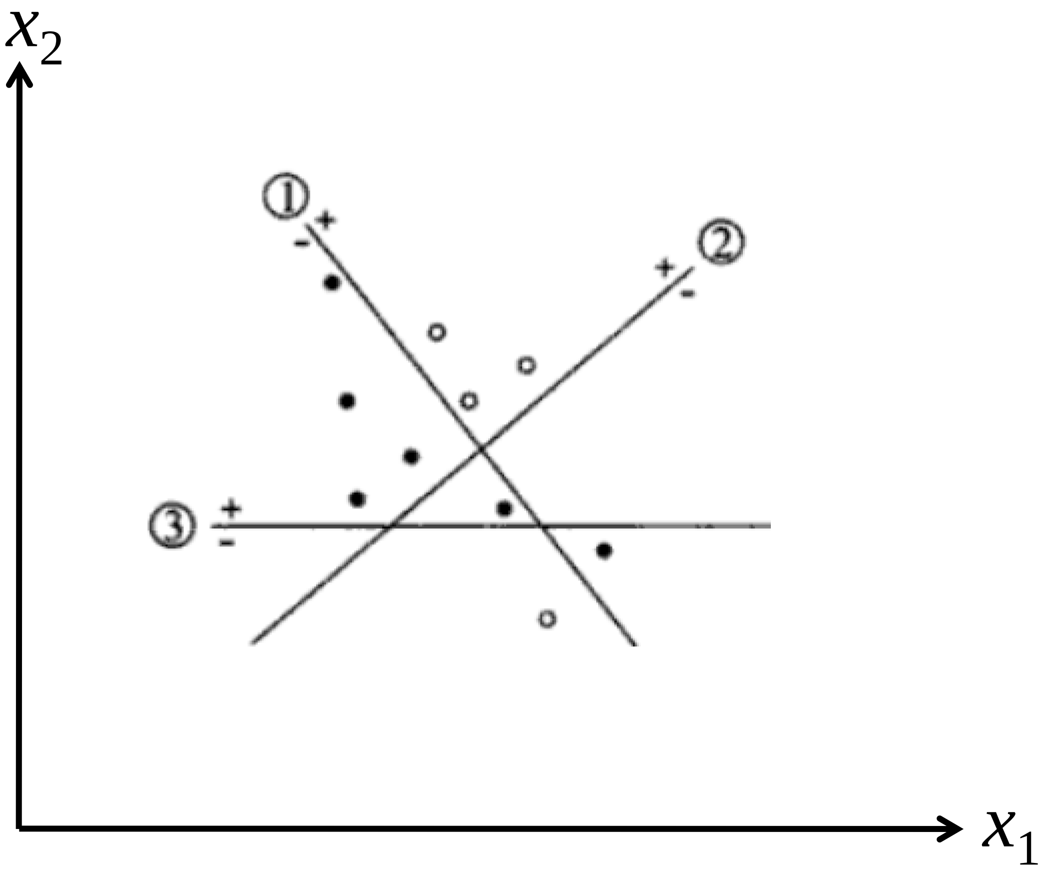
Algorithm Based on Exact Classification of Training Examples

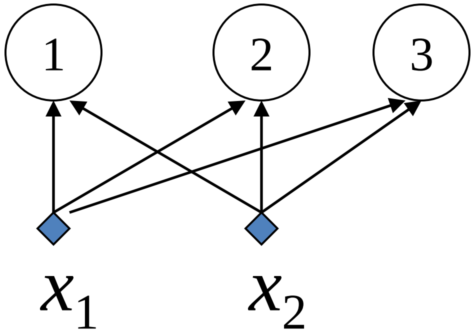



Master unit

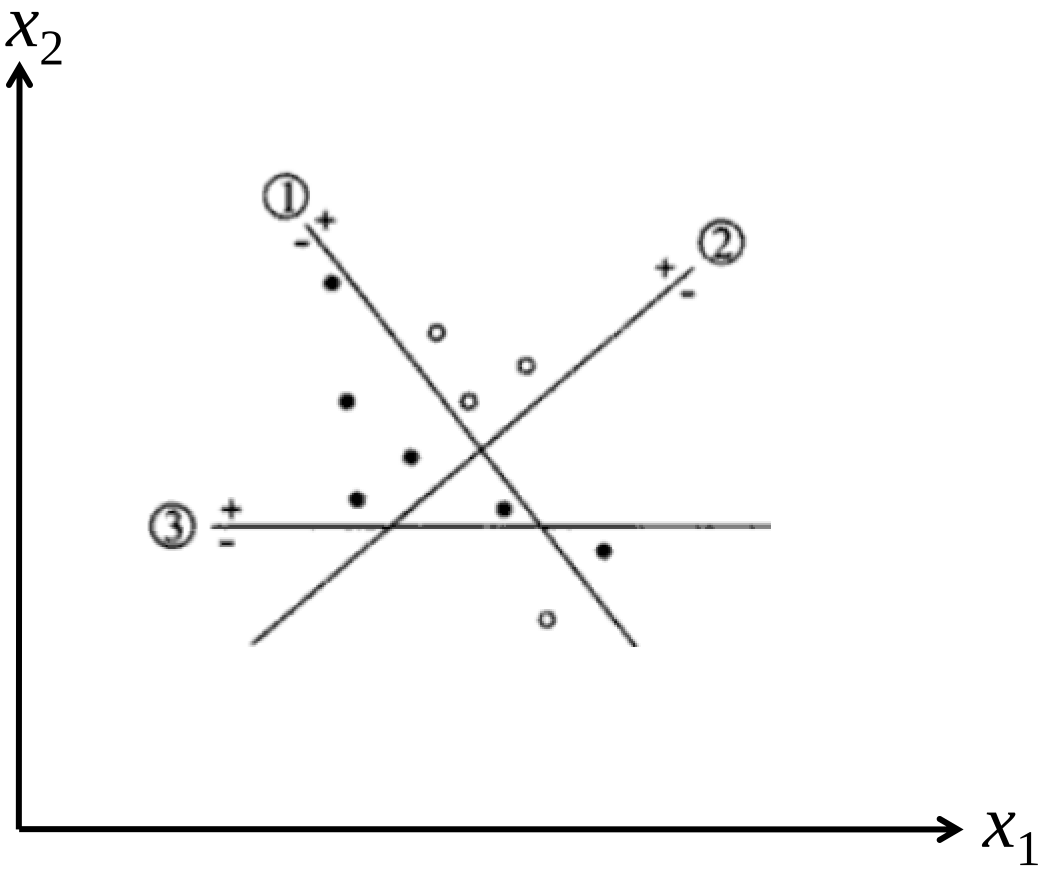
Ancillary units

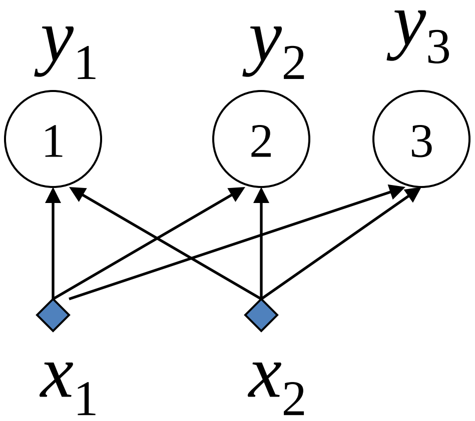
Algorithm Based on Exact Classification of Training Examples



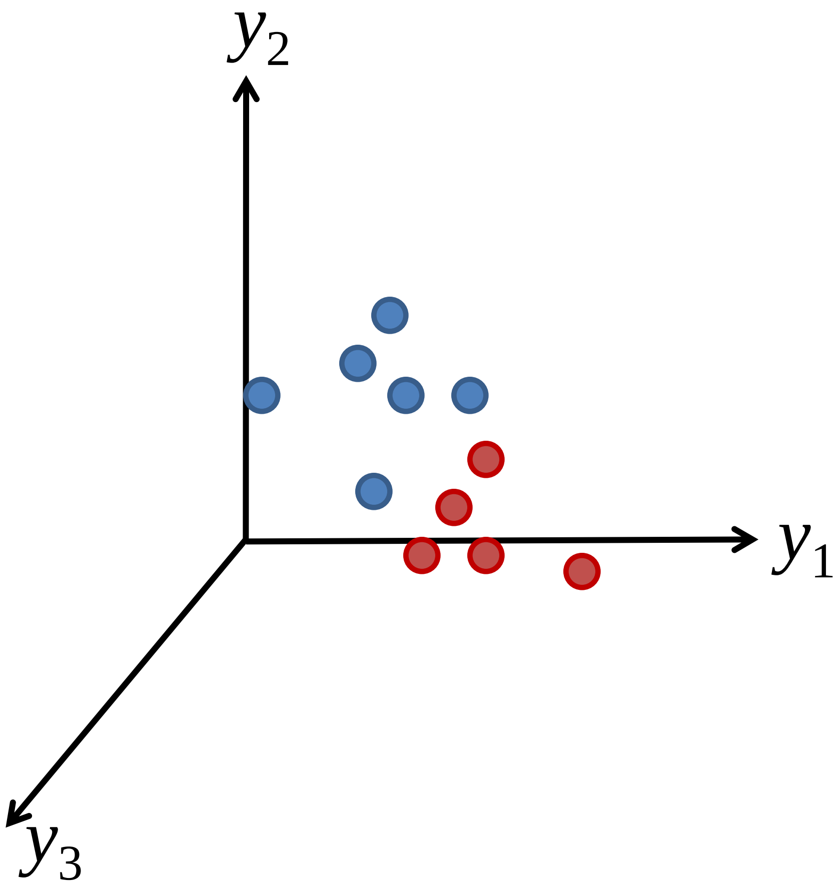


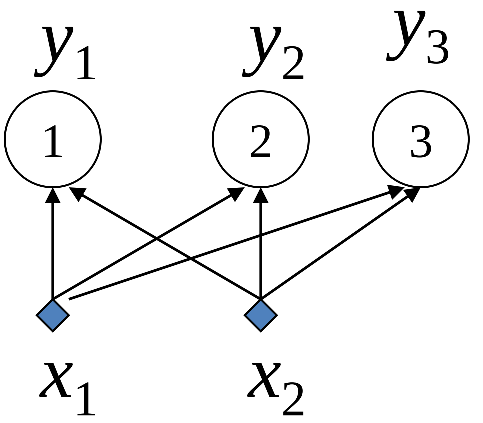
Algorithm Based on Exact Classification of Training Examples



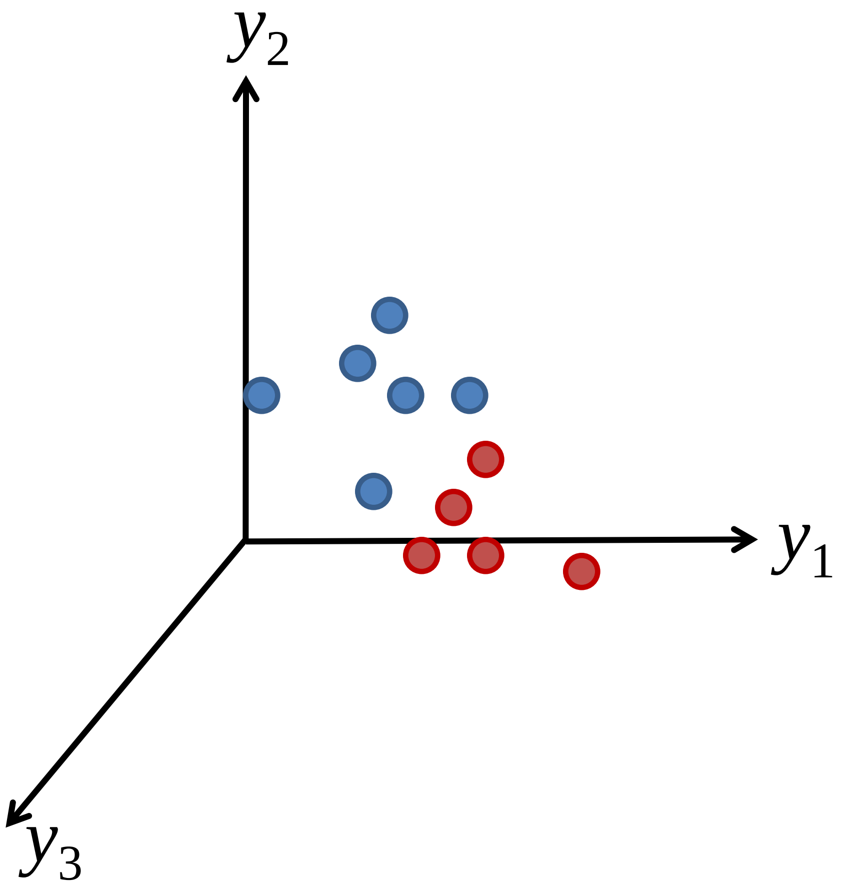


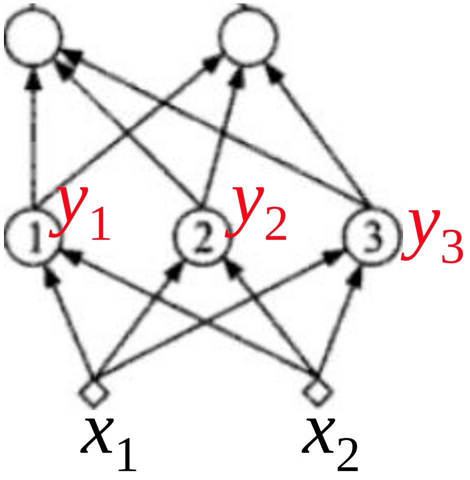
Algorithm Based on Exact Classification of Training Examples





Algorithm Based on Exact Classification of Training Examples





Algorithm Based on Exact Classification of Training Examples