Artificial Neural Network

Dr. Suyanto, S.T., M.Sc.
HP/WA: 0812 845 12345

Intelligence Computing Multimedia (ICM)
Informatics faculty – Telkom University

Bagaimana manusia belajar?

Menangis?

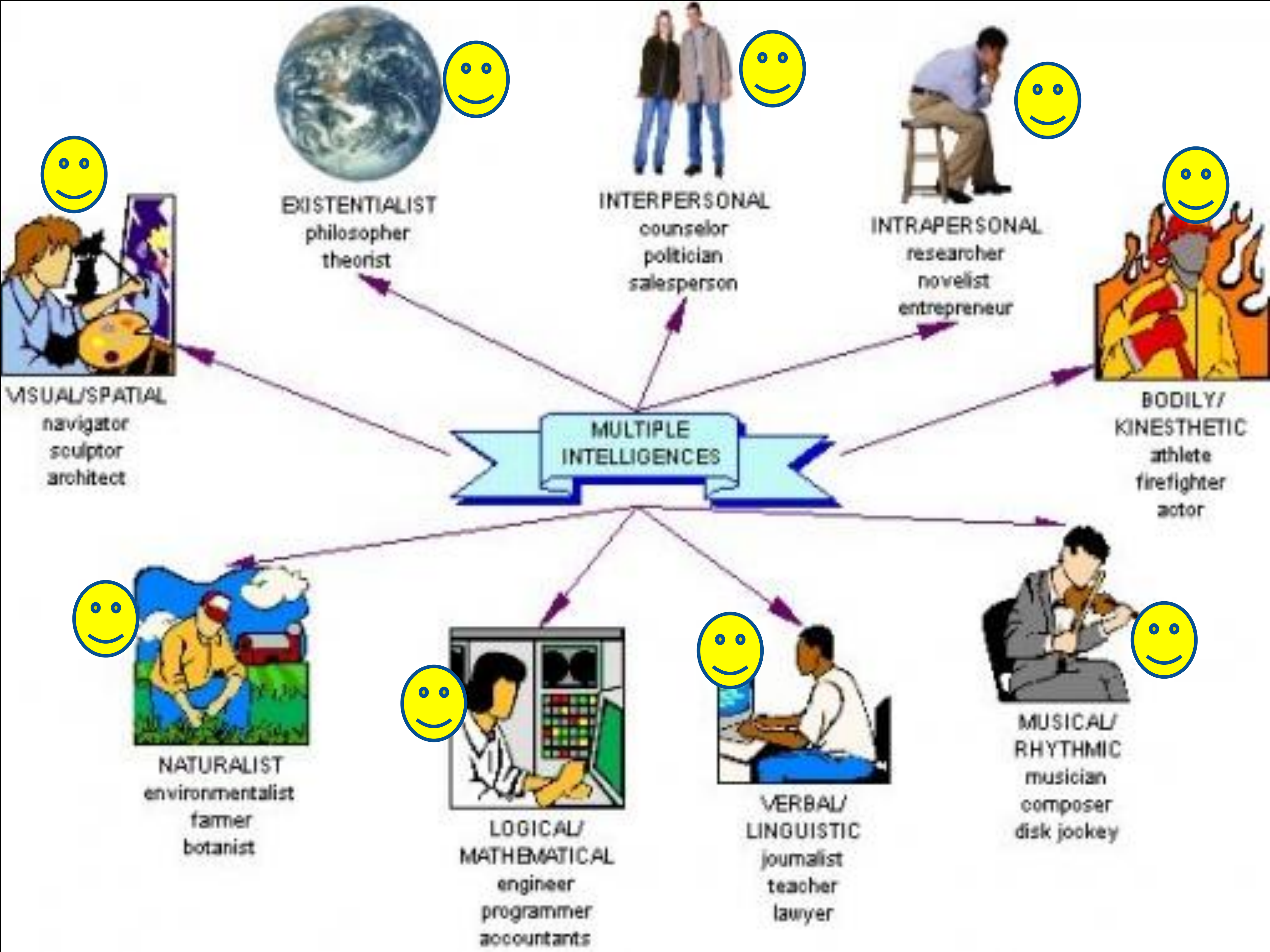
Merayap - Berjalan?

Arah?

Berbahasa?

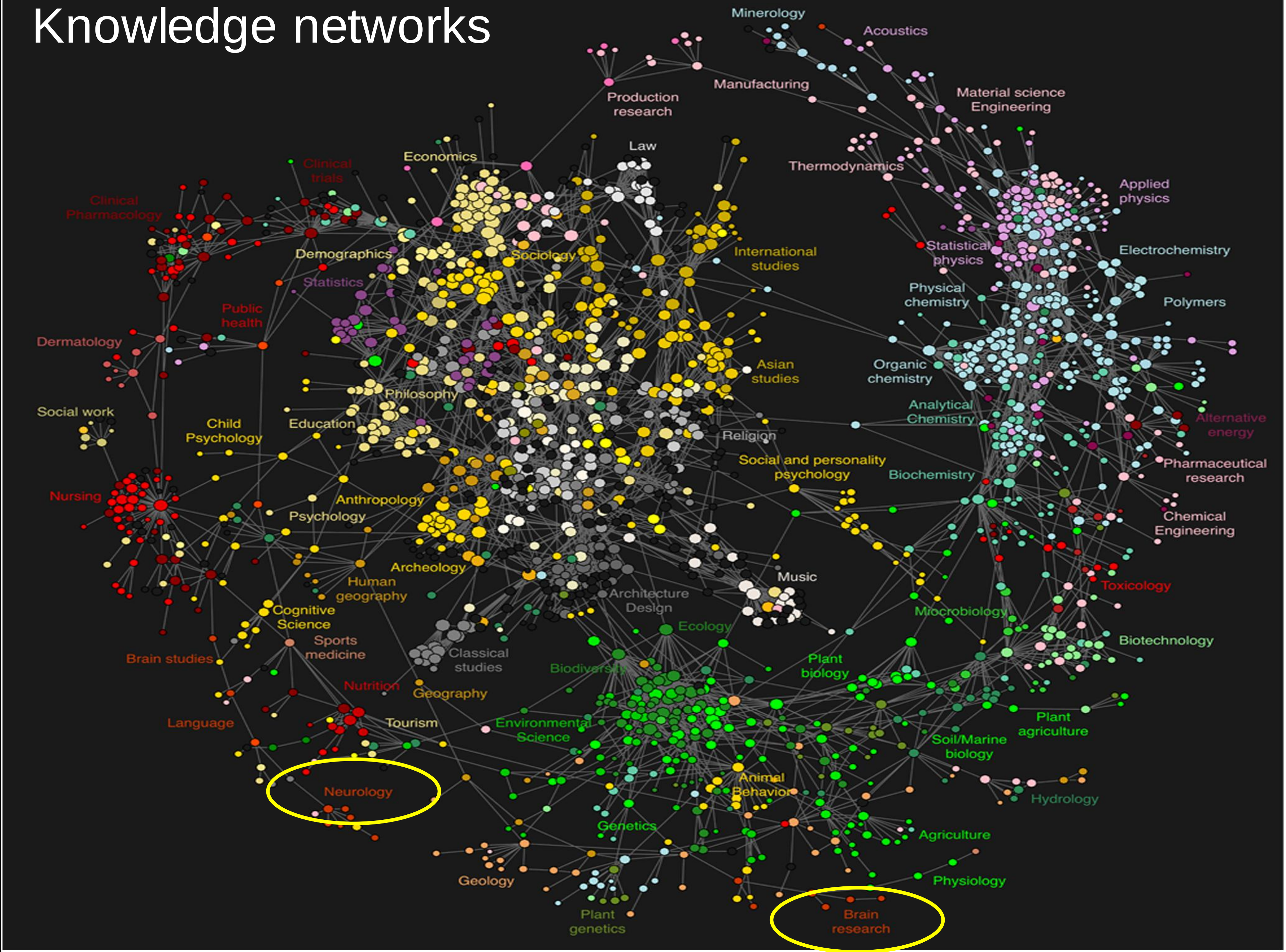
Logika?

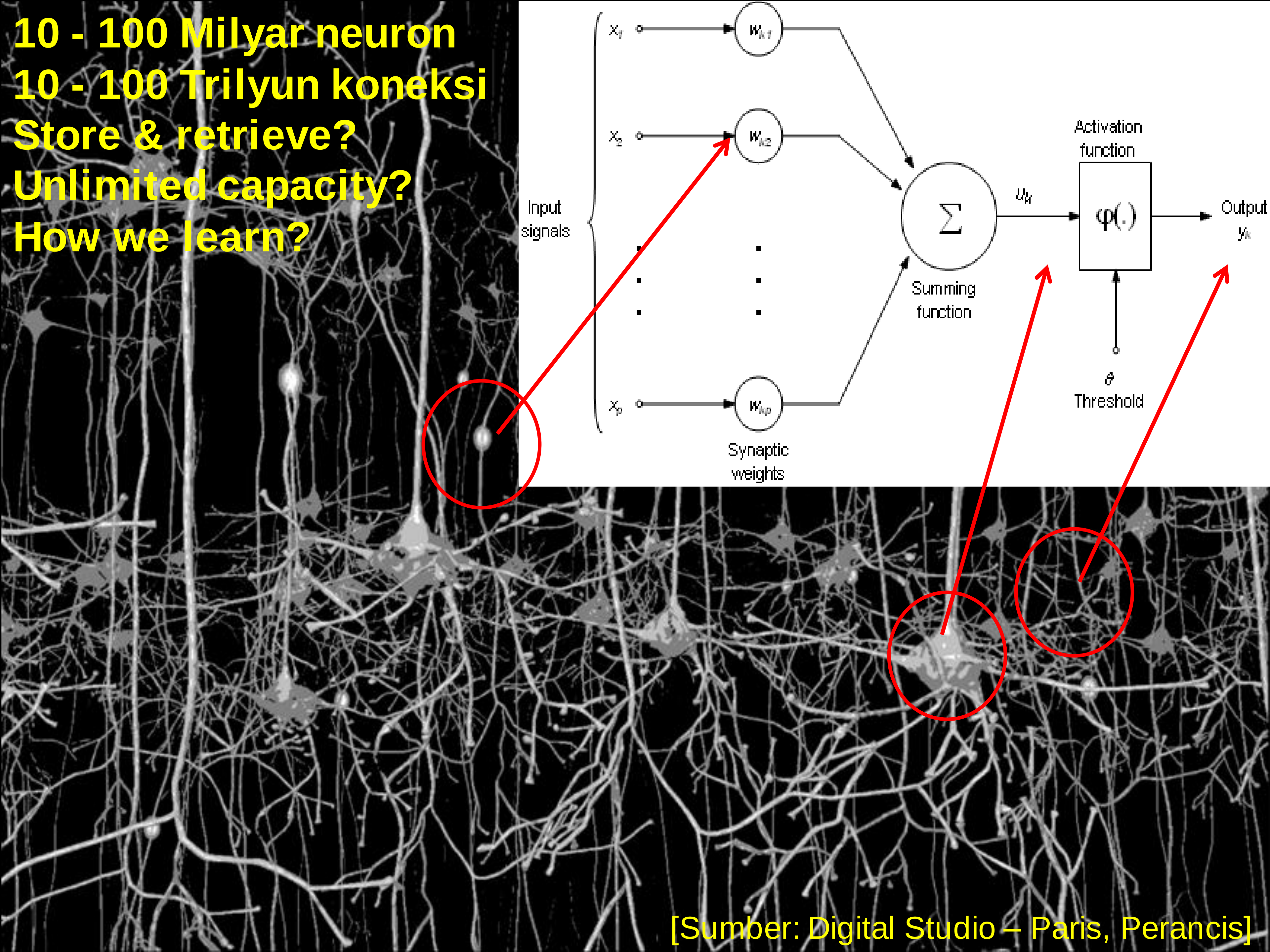
Multiple Intelligence !!!


EXISTENTIALIST
philosopher
theorist
INTERPERSONAL
counselor
politician
salesperson
INTRAPERSONAL
researcher
novelist
entrepreneur
VISUAL/SPATIAL
navigator
sculptor
architect
MULTIPLE INTELLIGENCES
BODILY/ KINESTHETIC
athlete
firefighter
actor
NATURALIST
environmentalist
farmer
botanist
LOGICAL/ MATHEMATICAL
engineer
programmer
accountants
VERBAL/ LINGUISTIC
journalist
teacher
lawyer
MUSICAL/ RHYTHMIC
musician
composer
disk jockey

http://newsfeed.time.com/2010/07/30/swedish-police-officer-fights-crime-with-sweet-dance-moves

Knowledge networks

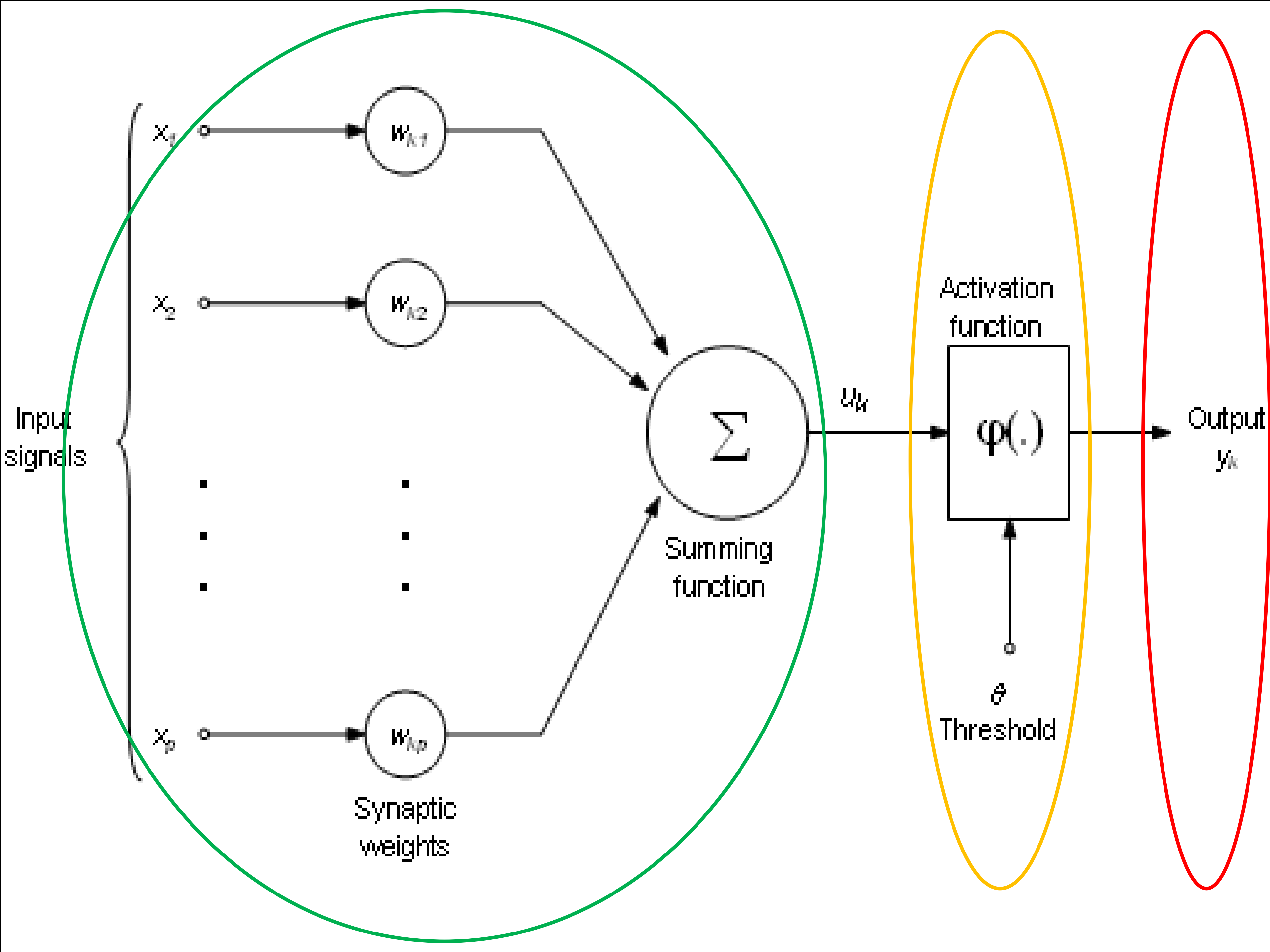




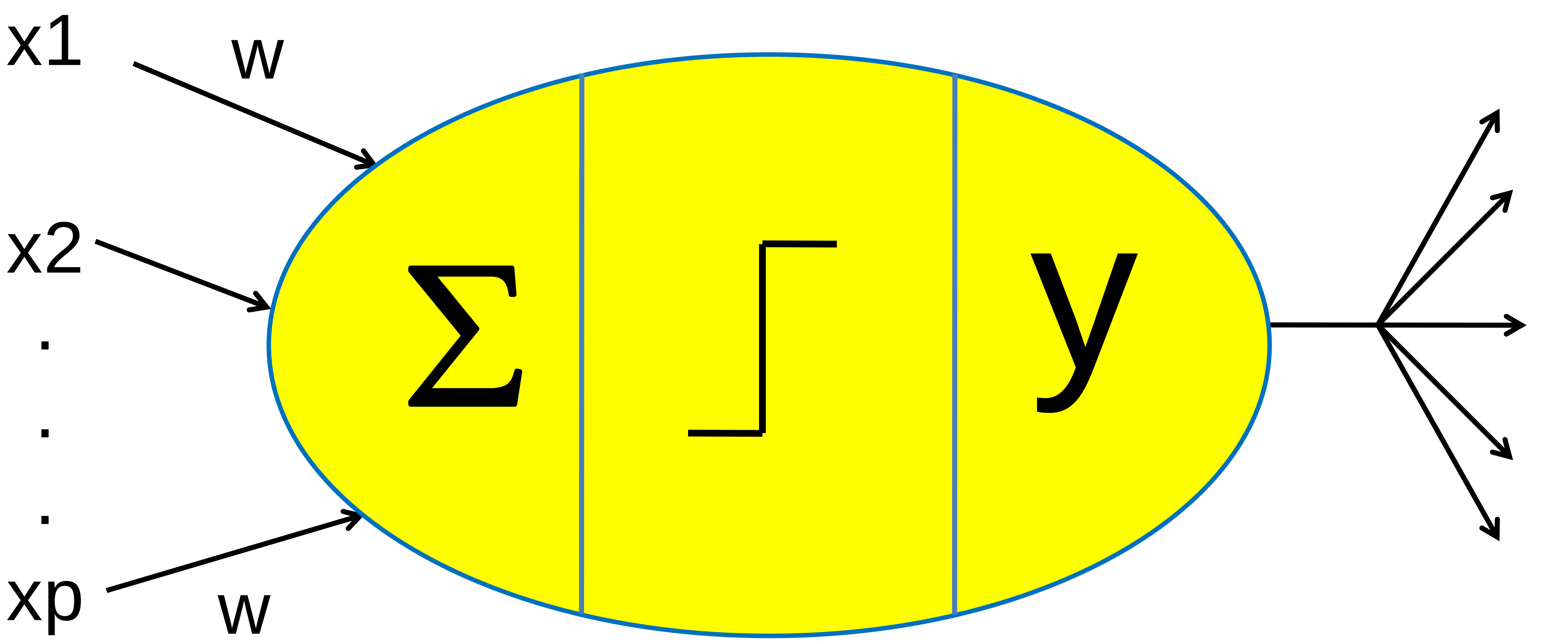
10 - 100 Milyar neuron
10 - 100 Trilyun koneksi
Store & retrieve?
Unlimited capacity?
How we learn?
Input signals
x_1
x_2
x_p
w_{k1}
w_{k2}
w_{kp}
Synaptic weights
Σ
Summing function
u_k
Activation function
$\varphi(.)$
θ
Threshold
Output
y_k
[Sumber: Digital Studio – Paris, Perancis]

Perceptron

- **Neuron**: Sel syaraf biologis
- **Perceptron**: Sel syaraf buatan
  - Input function
  - Activation function
  - Output


Input signals
x1
x2
xp
wk1
wk2
wkp
Synaptic weights
Σ
Summing function
uk
Activation function
φ(.)
θ
Threshold
Output
yk

Perceptron

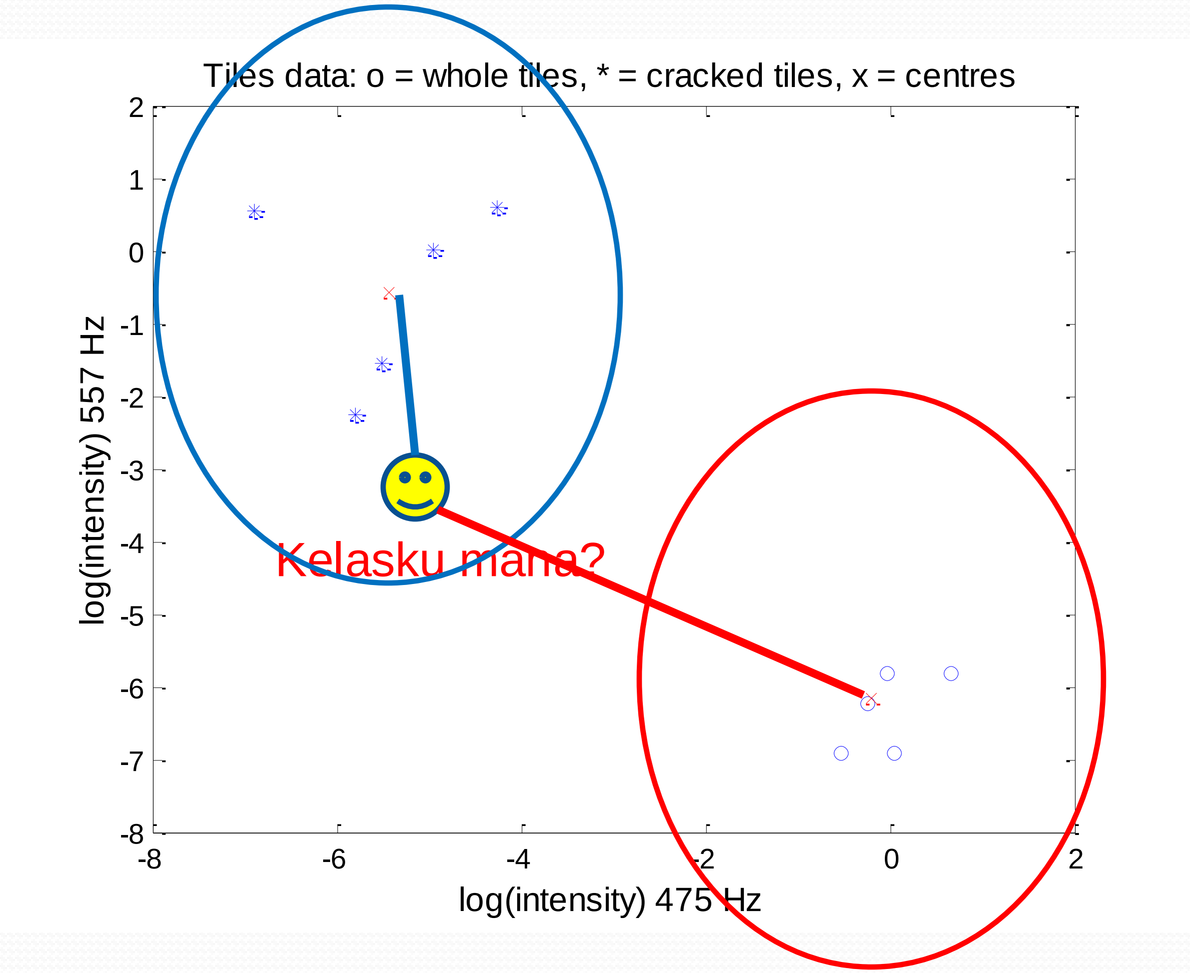


Perceptron

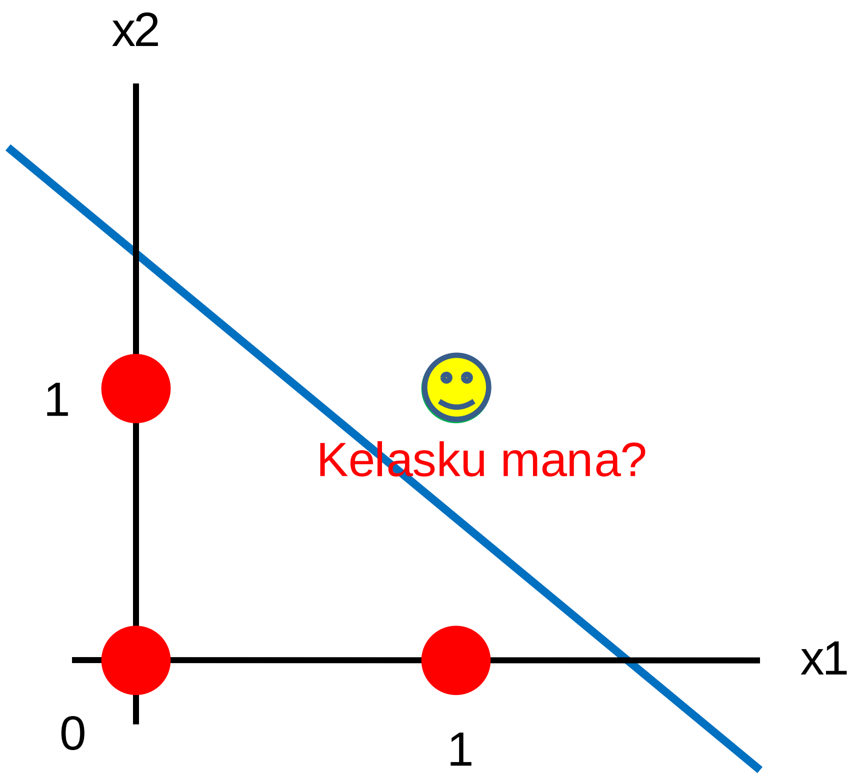
- Jika manusia punya 10 milyar neuron, apa yang bisa dilakukan?
- Sangat banyak hal bisa dilakukan
- Apalagi jika Multiple Intelligence

- Perceptron = MODEL SEDERHANA dari neuron
- Apa yang bisa dilakukan oleh satu perceptron?
- **Klasifikasi**
- **Prediksi**
- **Optimasi, …**

Fuzzy C-Means (FCM)



AND



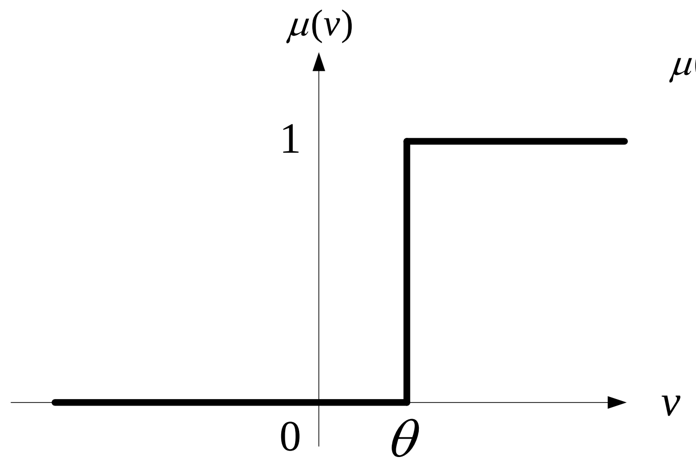
x1	x2	y
0	0	0
0	1	0
1	0	0
1	1	1

$$x1 + x2 - 1{,}5 = 0$$

$$w1.x1 + w2.x2 - 1{,}5 = 0$$

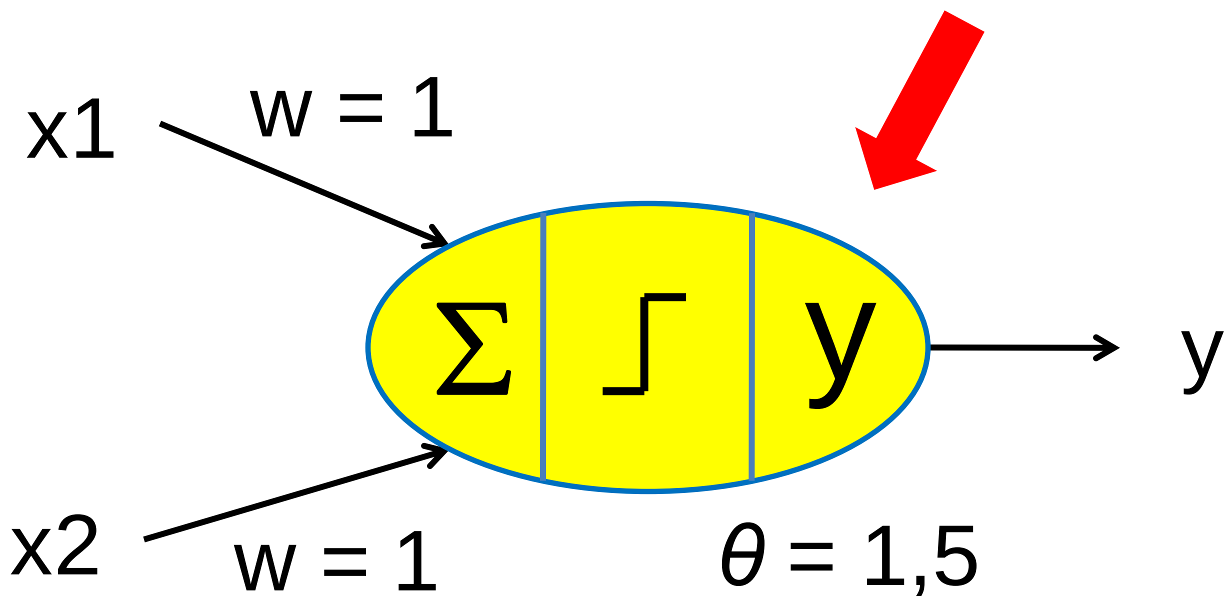
AND

x1	x2	y
0	0	0
0	1	0
1	0	0
1	1	1

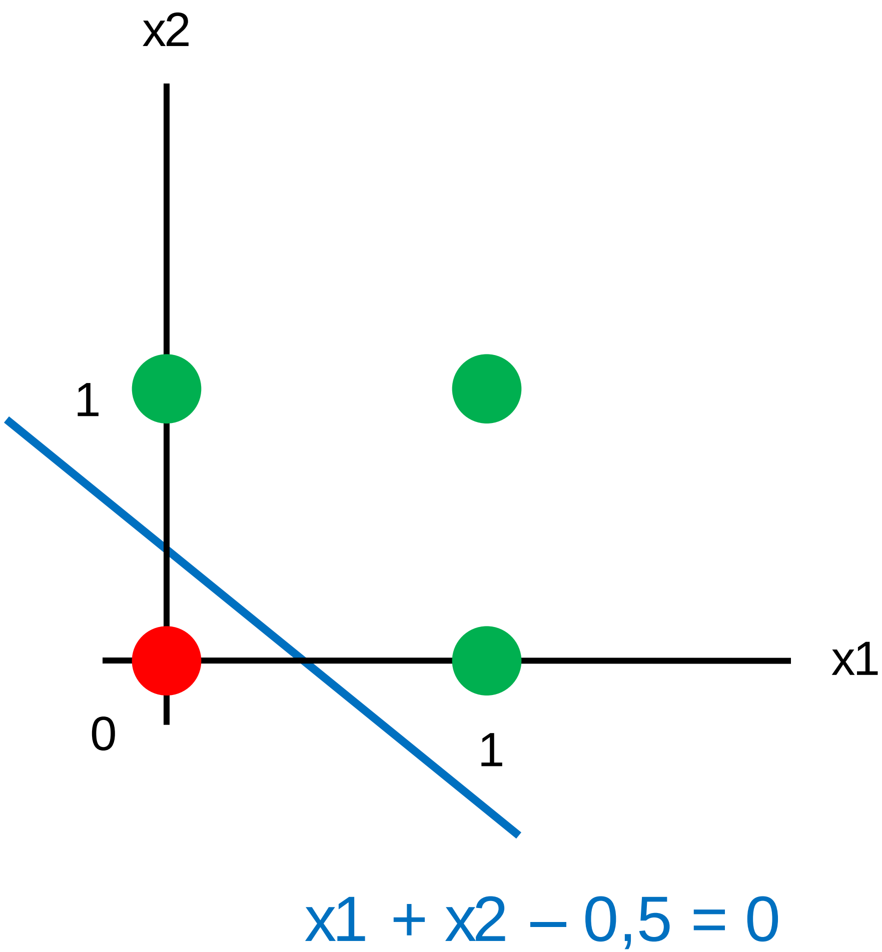


$$\mu(v) = \begin{cases} 1 & if\ v >= \theta \\ 0 & if\ v < \theta \end{cases}$$

$$v = \sum_{i=1}^{p} w_i x_i$$



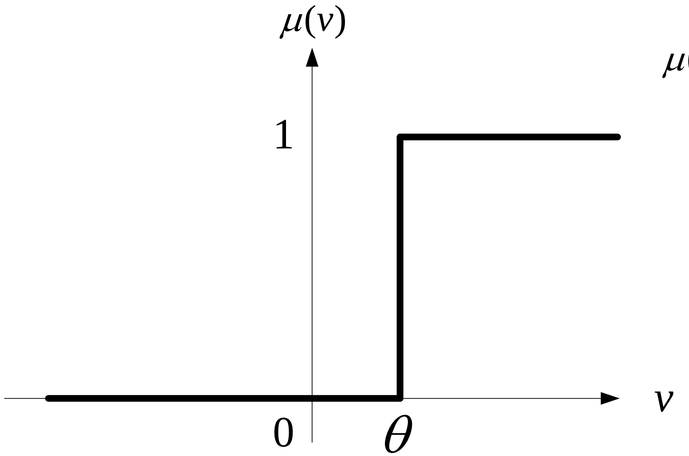
OR



x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

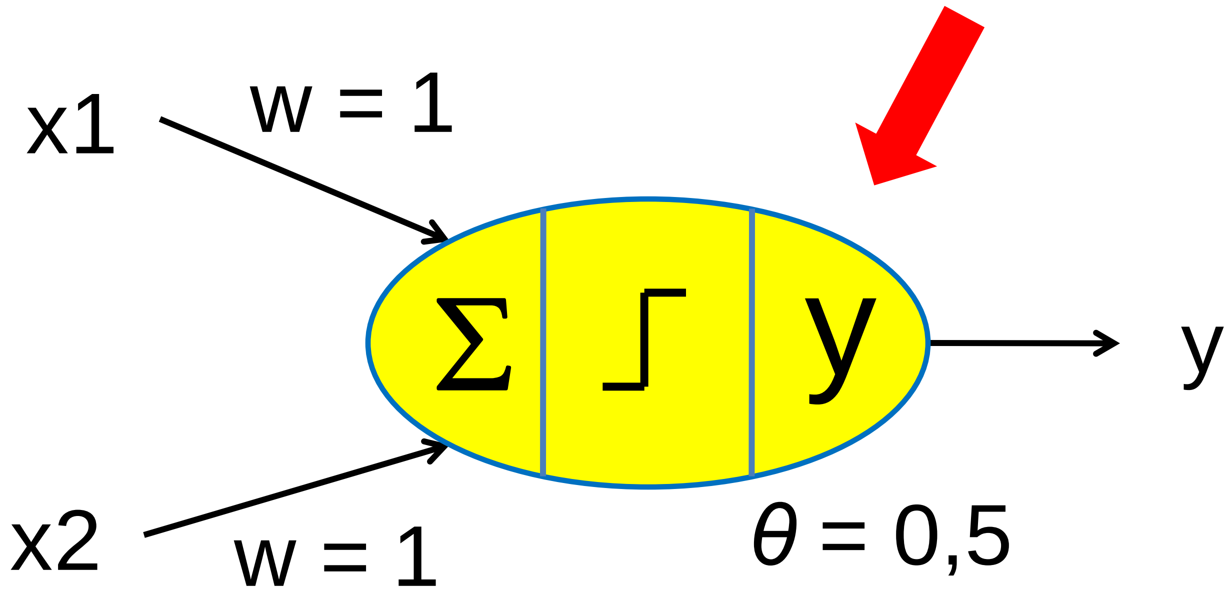
OR

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

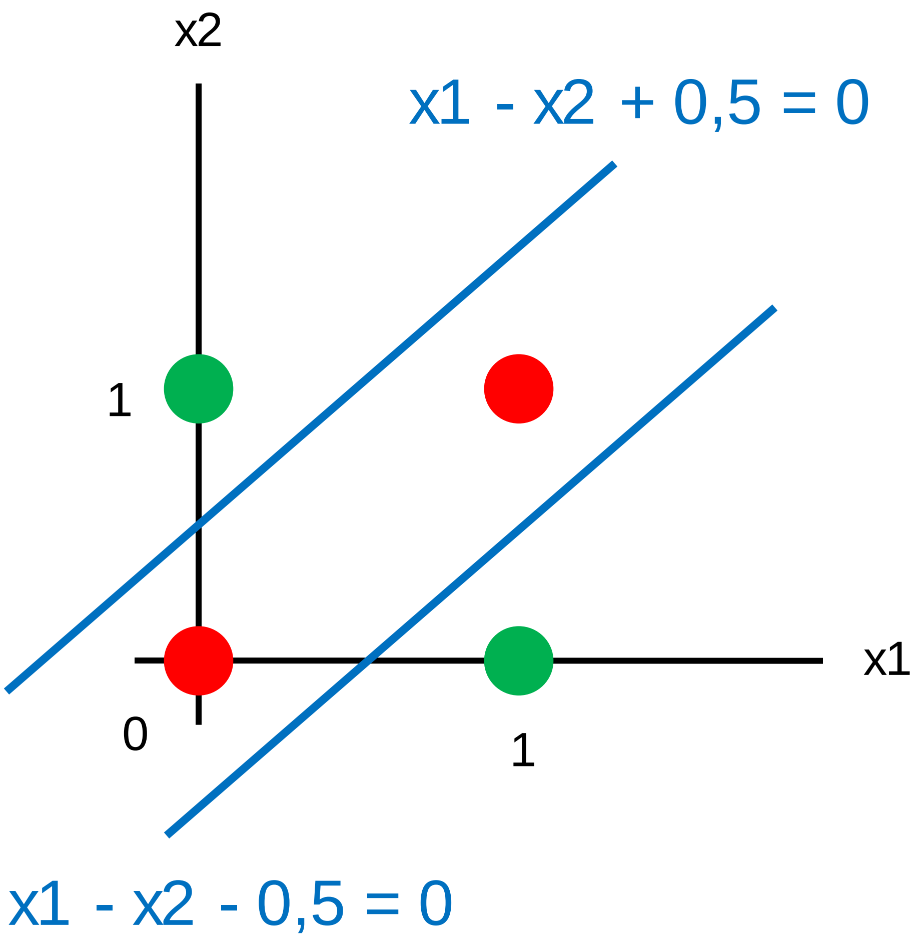


$$\mu(v) = \begin{cases} 1 & if\ v >= \theta \\ 0 & if\ v < \theta \end{cases}$$

$$v = \sum_{i=1}^{p} w_i x_i$$



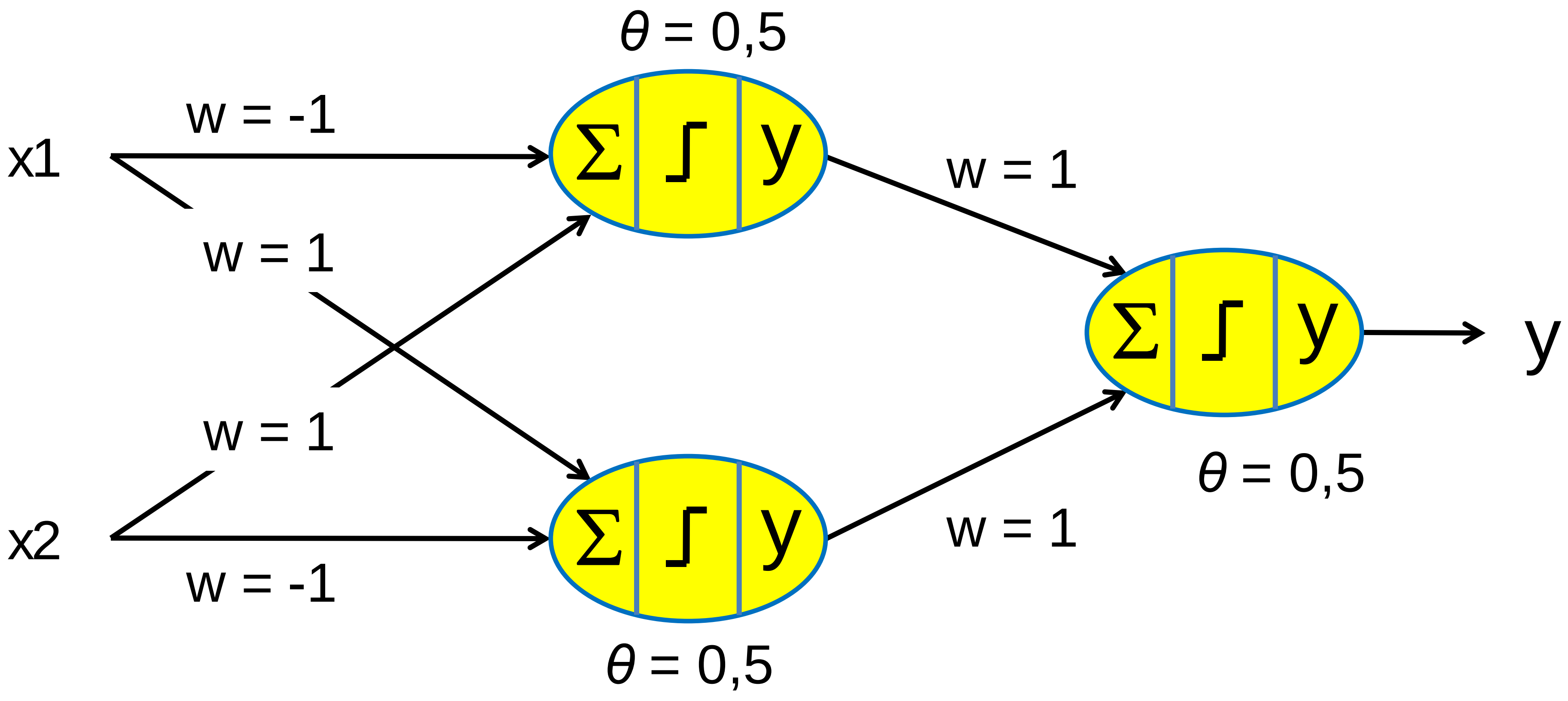
XOR



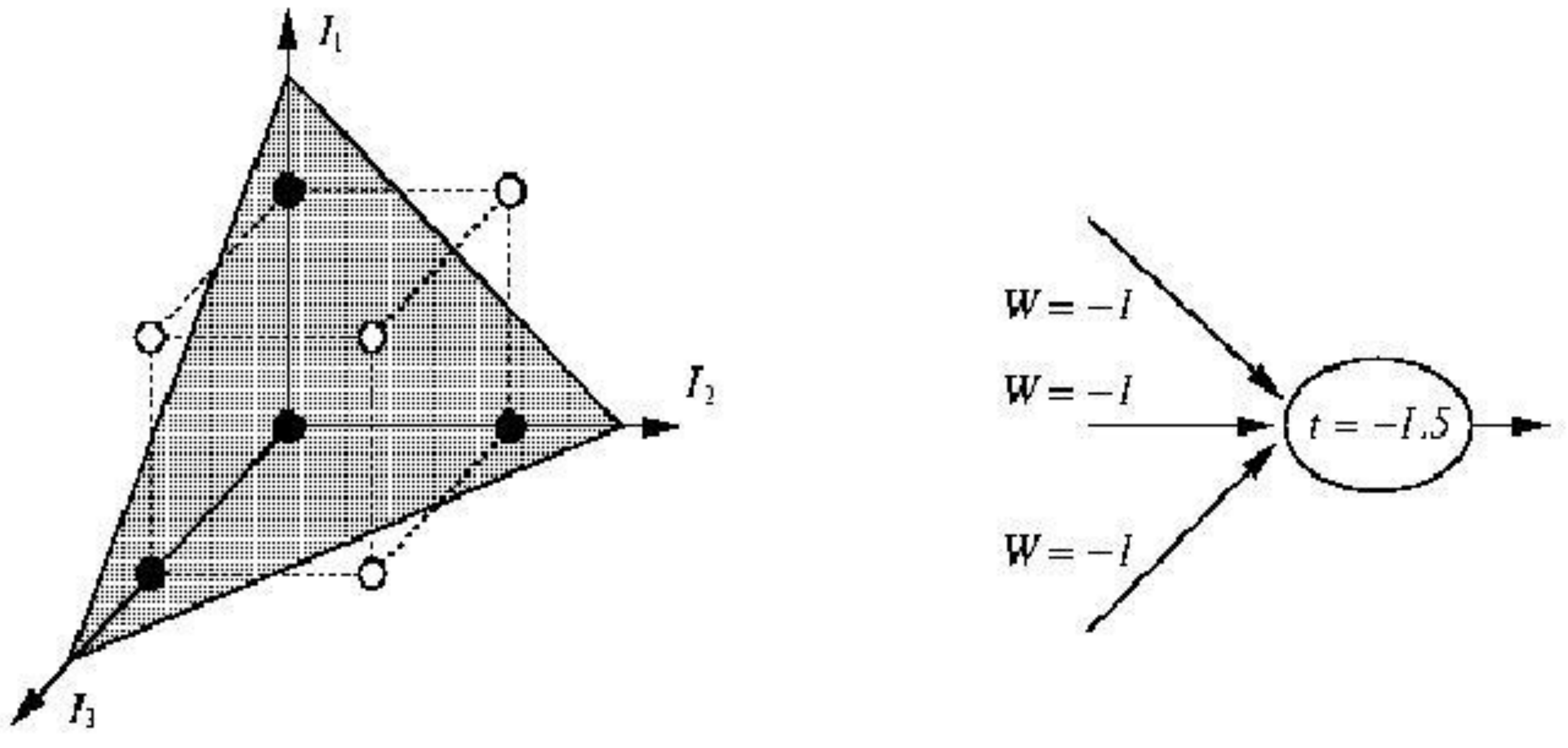
x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	0

XOR

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	0



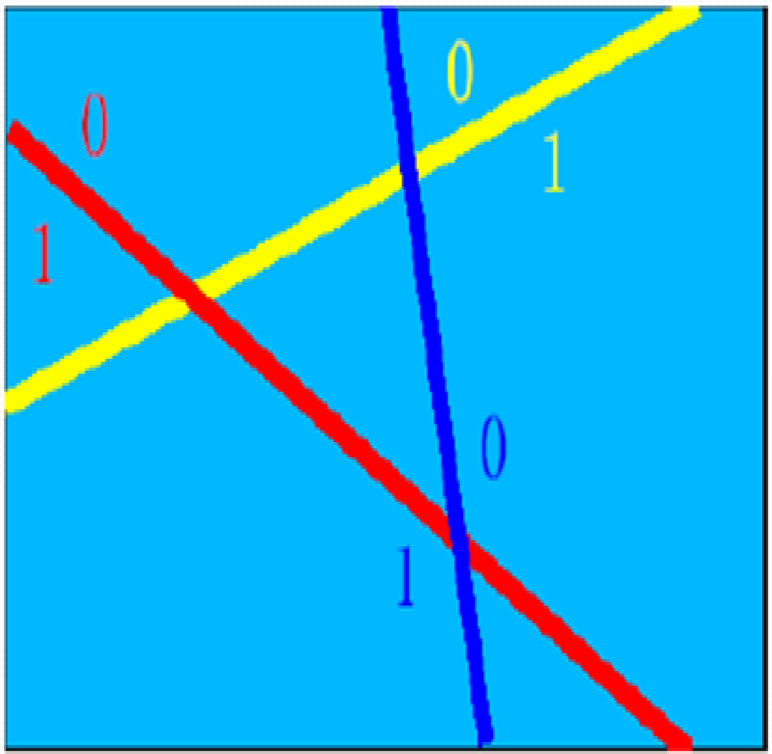
3 elemen input → 3 dimensi

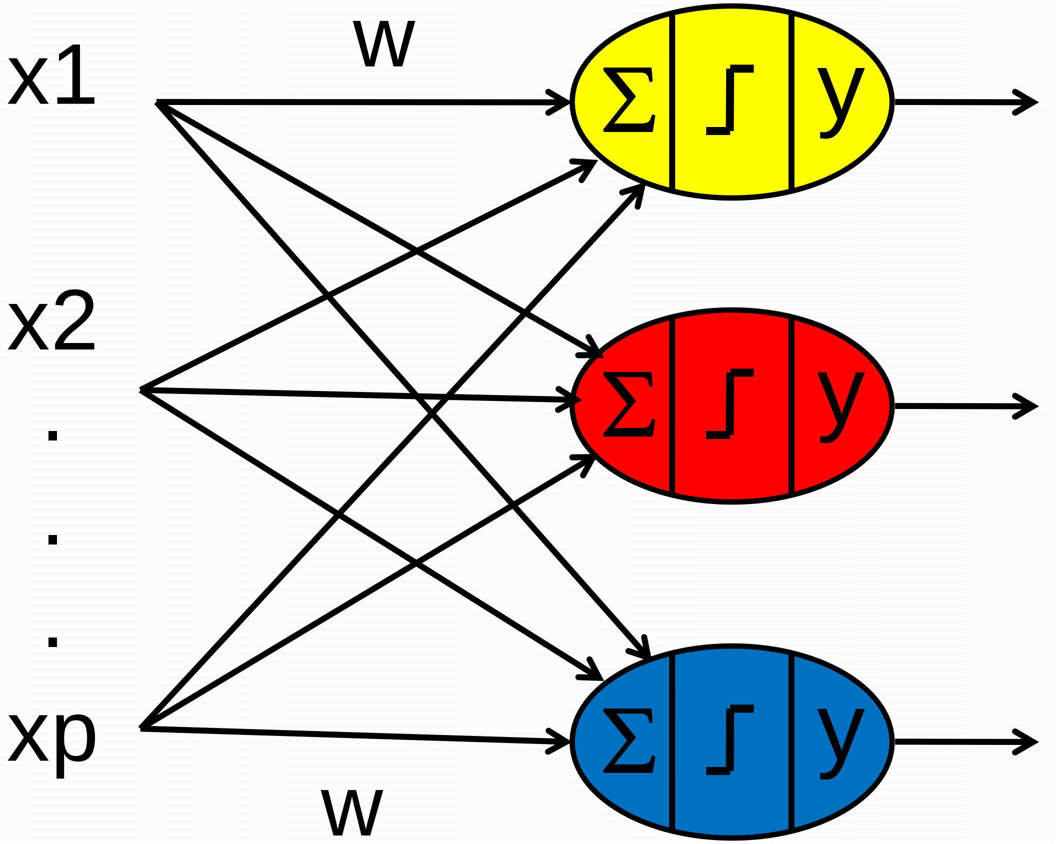


(a) Separating plane

(b) Weights and threshold

Perceptron Network

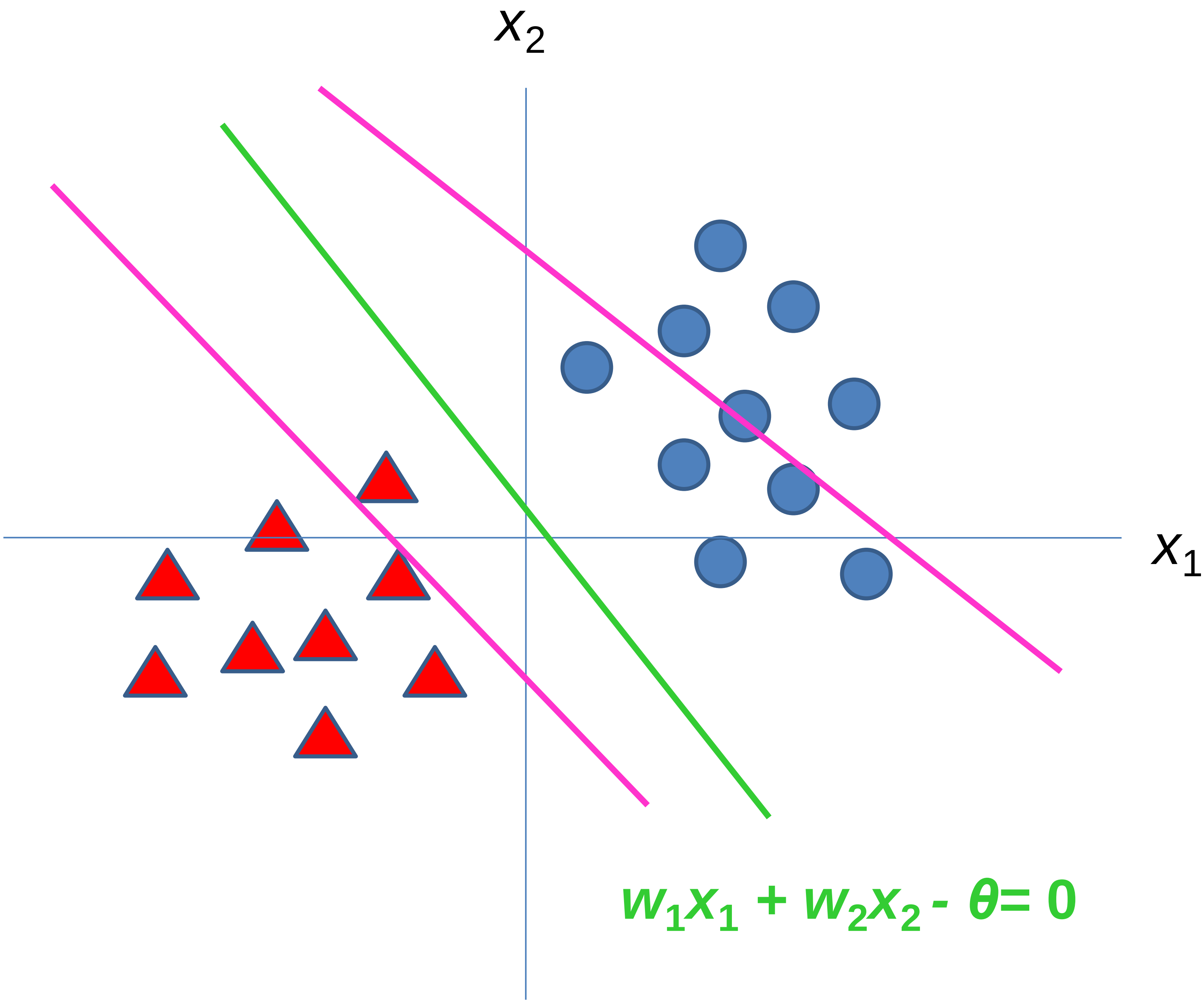


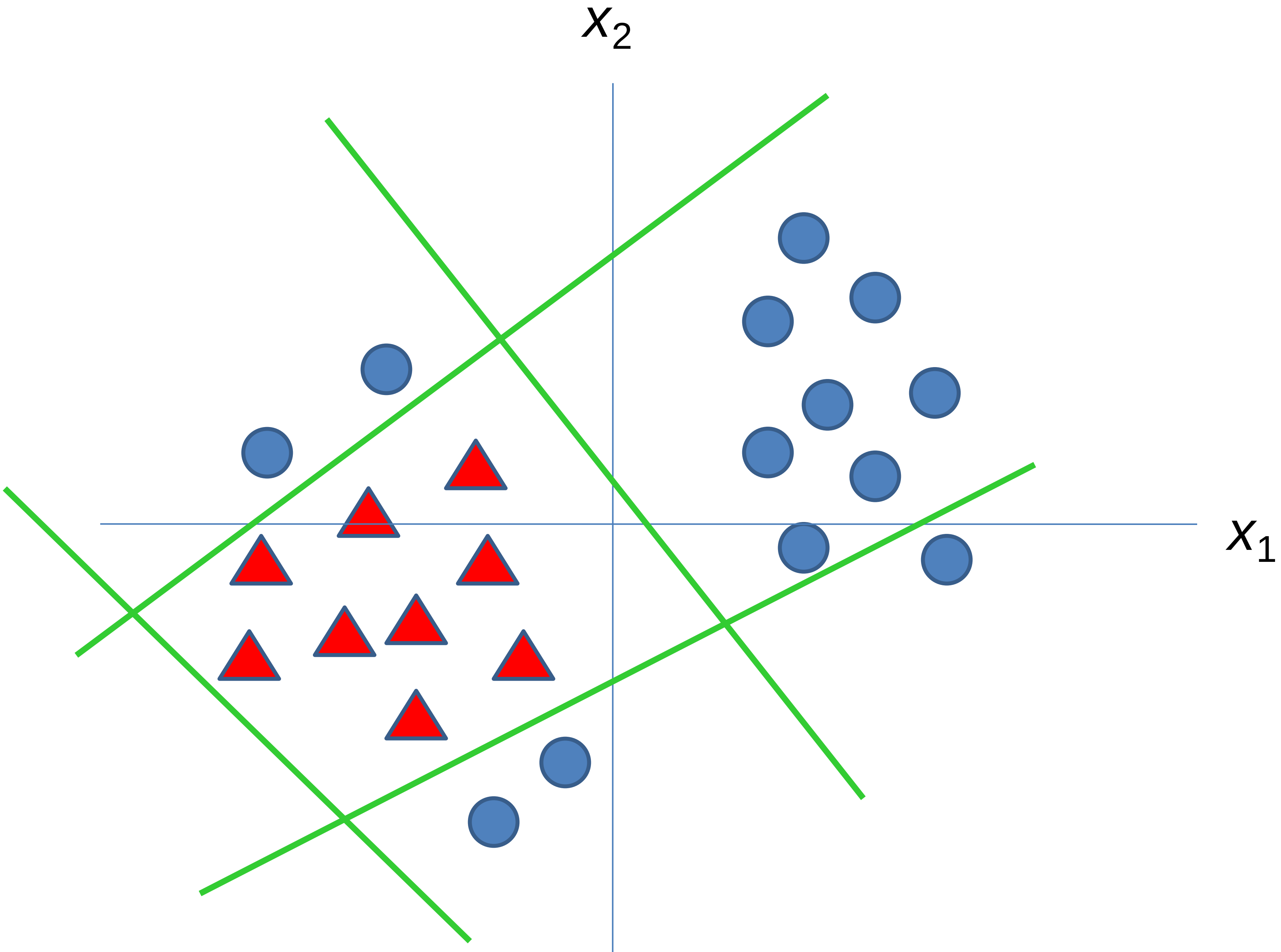


Learning

Bagaimana menemukan weights yang tepat?

Meminimumkan error

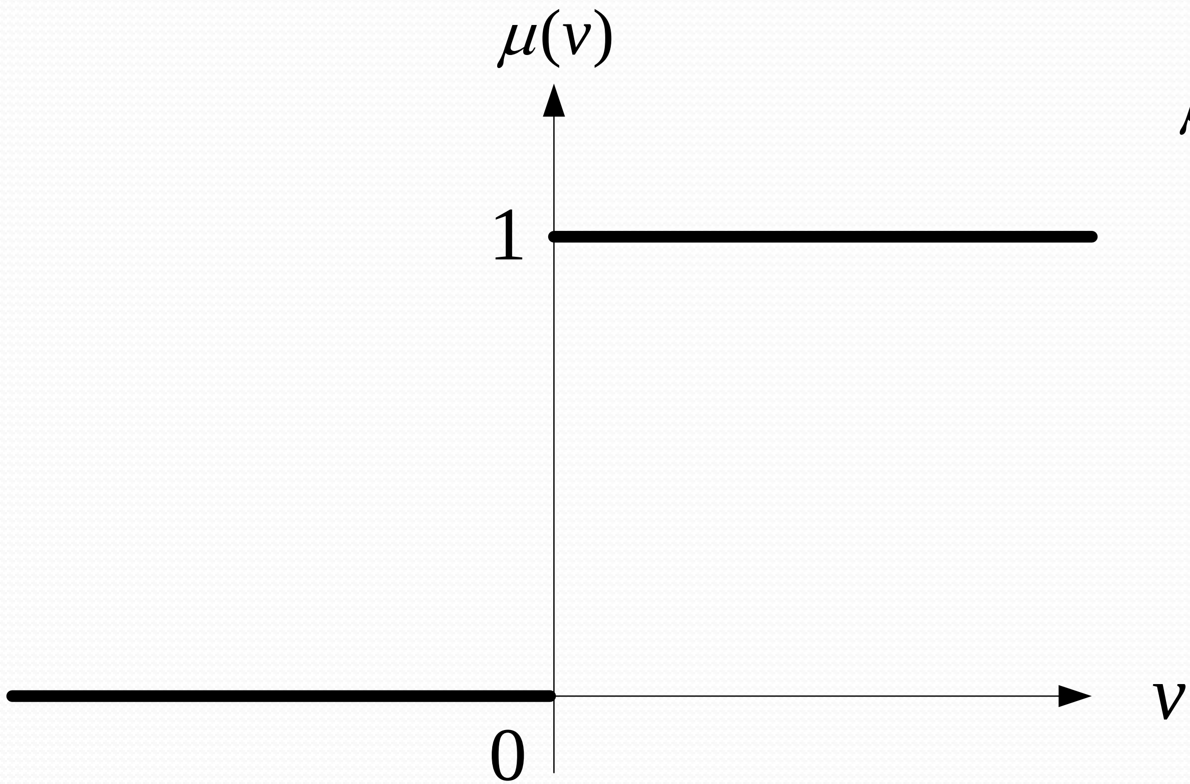

x_2
x_1
$w_1x_1 + w_2x_2 - \theta = 0$


x_2
x_1

Activation Functions

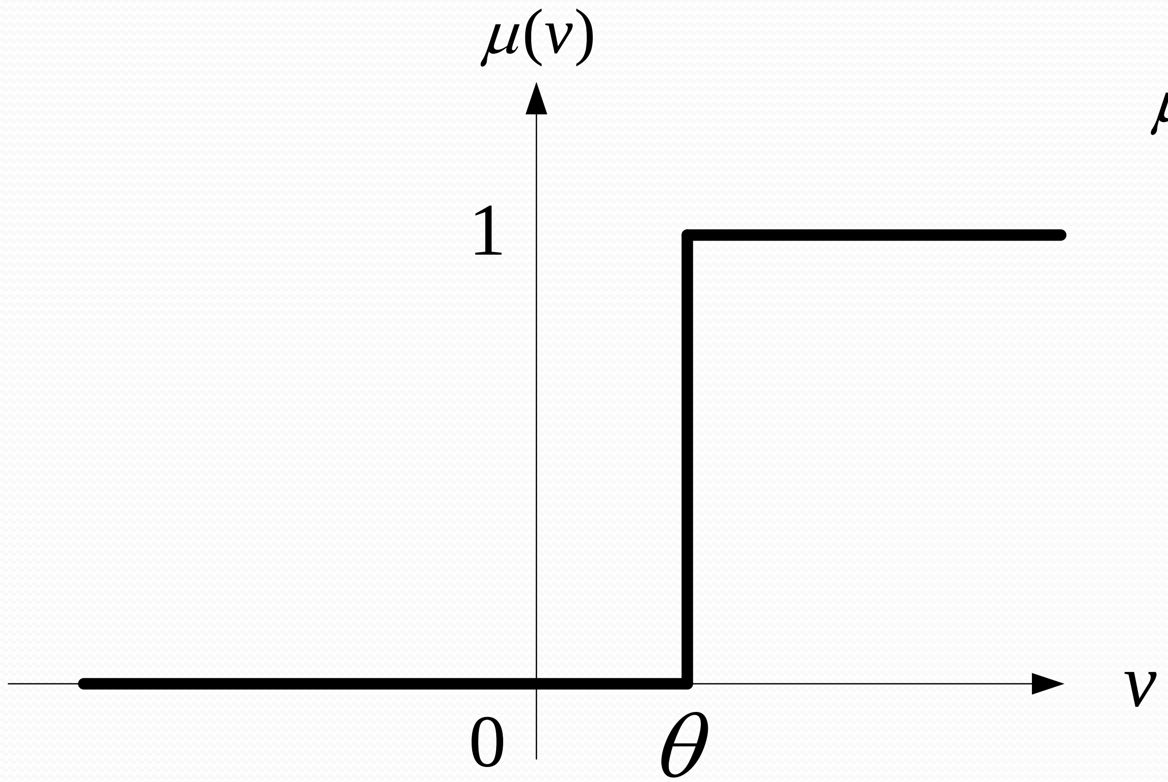
- *Hard Limit*
- *Threshold*
- *Linear (Identity)*
- *Sigmoid*
- *Radial Basis Function* (RBF)
- …

Hard Limit



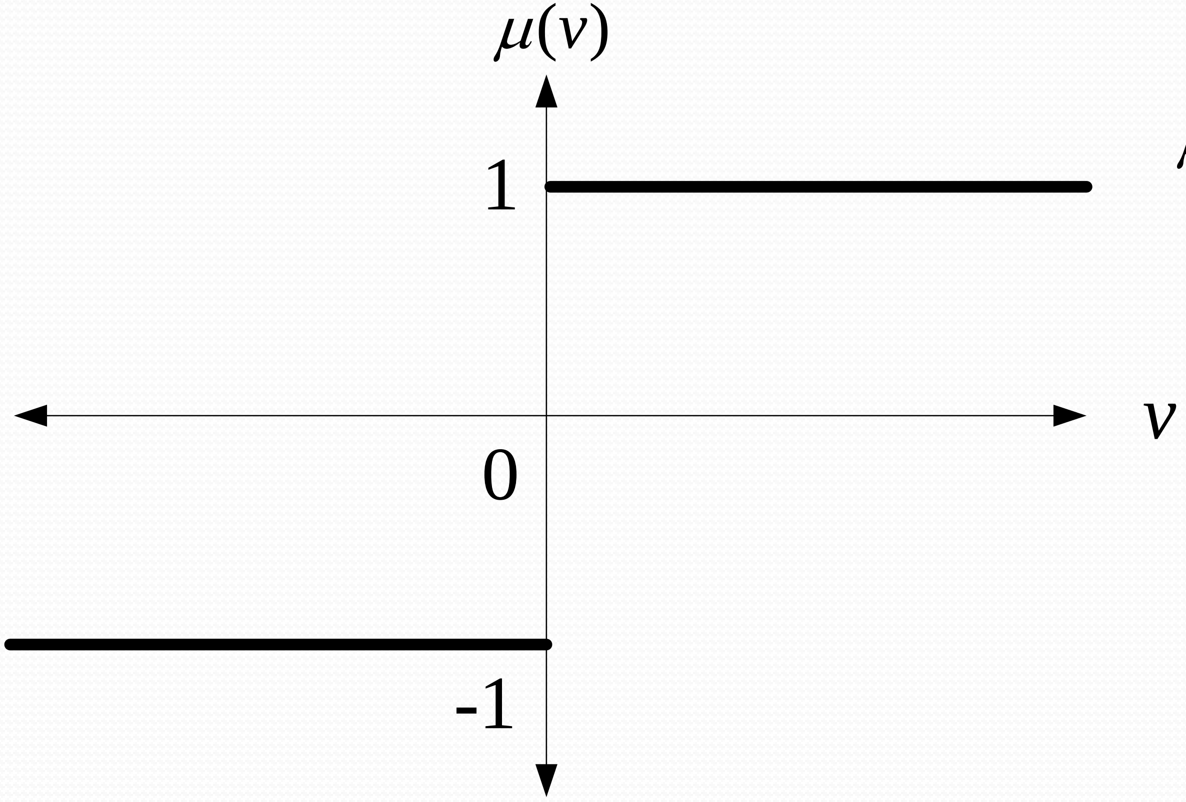
$$\mu(v) = \begin{cases} 1 & if\ v > 0 \\ 0 & if\ v \le 0 \end{cases}$$

Threshold



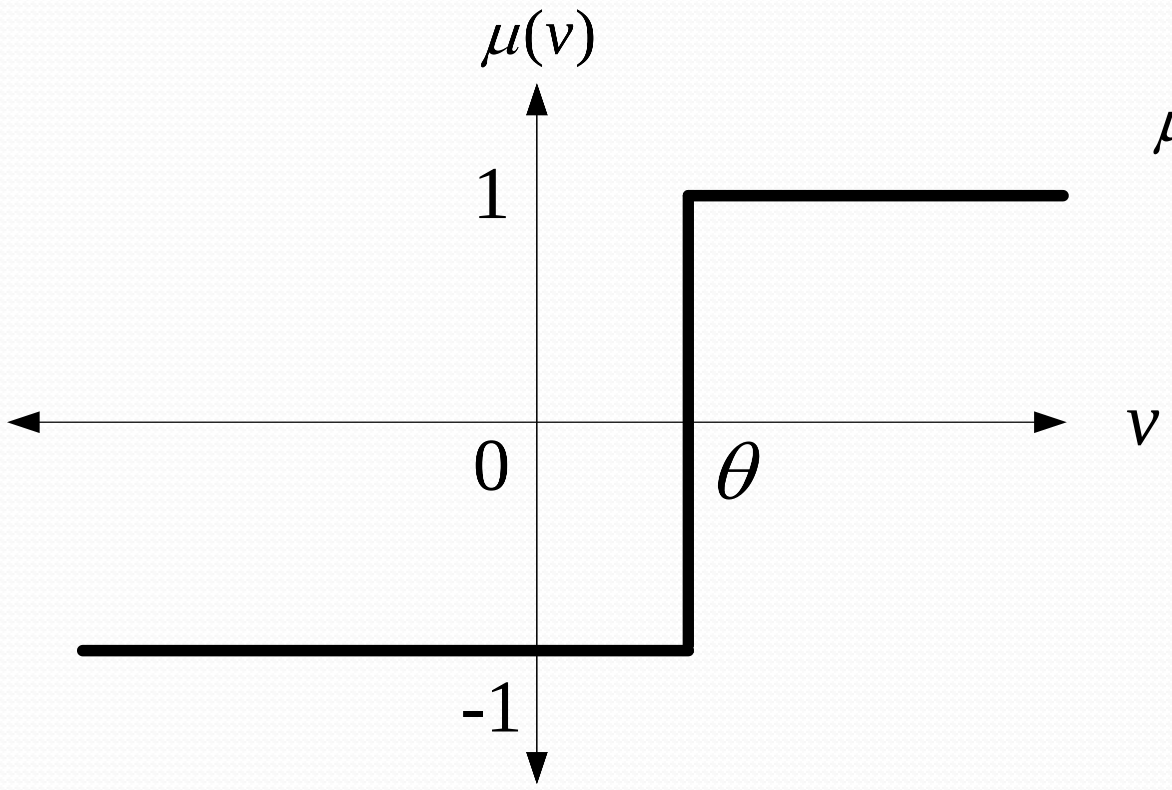
$$\mu(v) = \begin{cases} 1 & \text{if } v > \theta \\ 0 & \text{if } v \leq \theta \end{cases}$$

Symetric Hard Limit



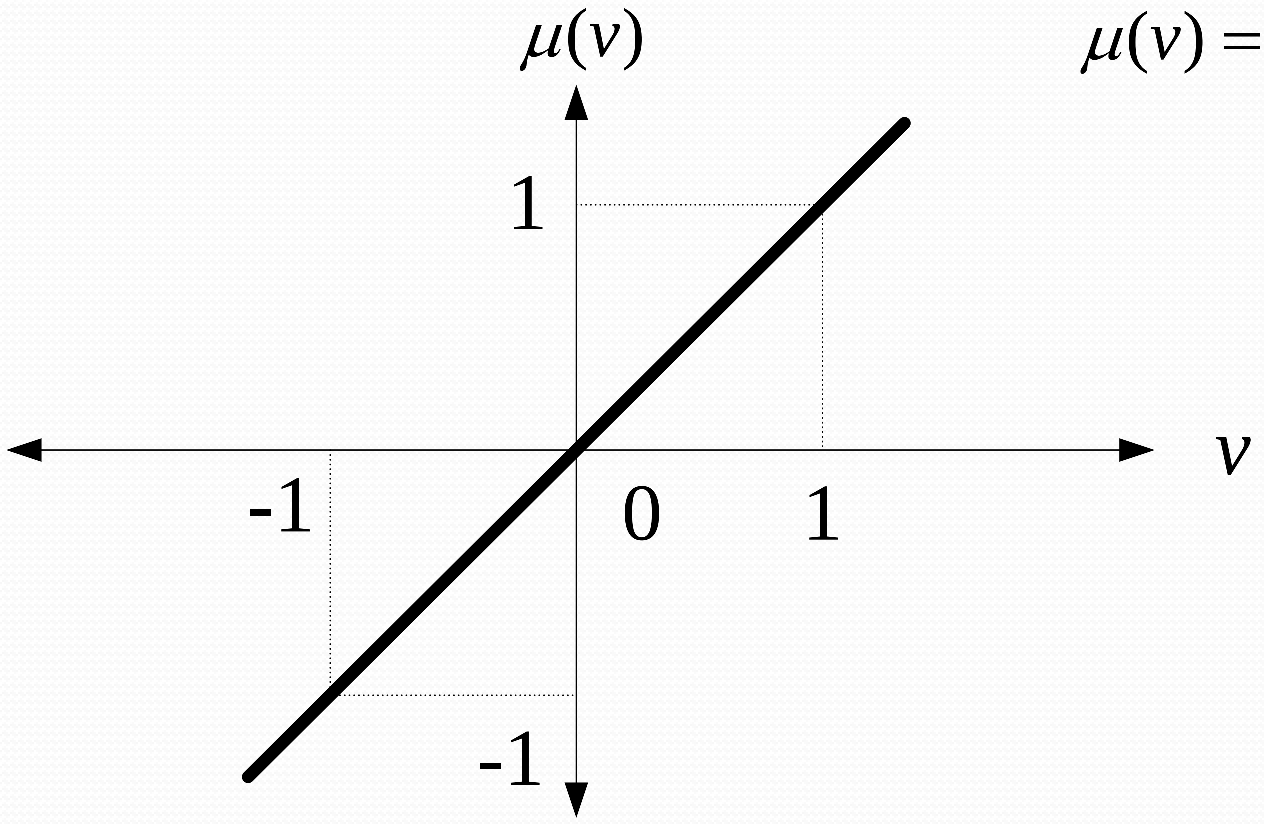
$$\mu(v) = \begin{cases} 1 & if\ v > 0 \\ 0 & if\ v = 0 \\ -1 & if\ v < 0 \end{cases}$$

Bipolar Threshold



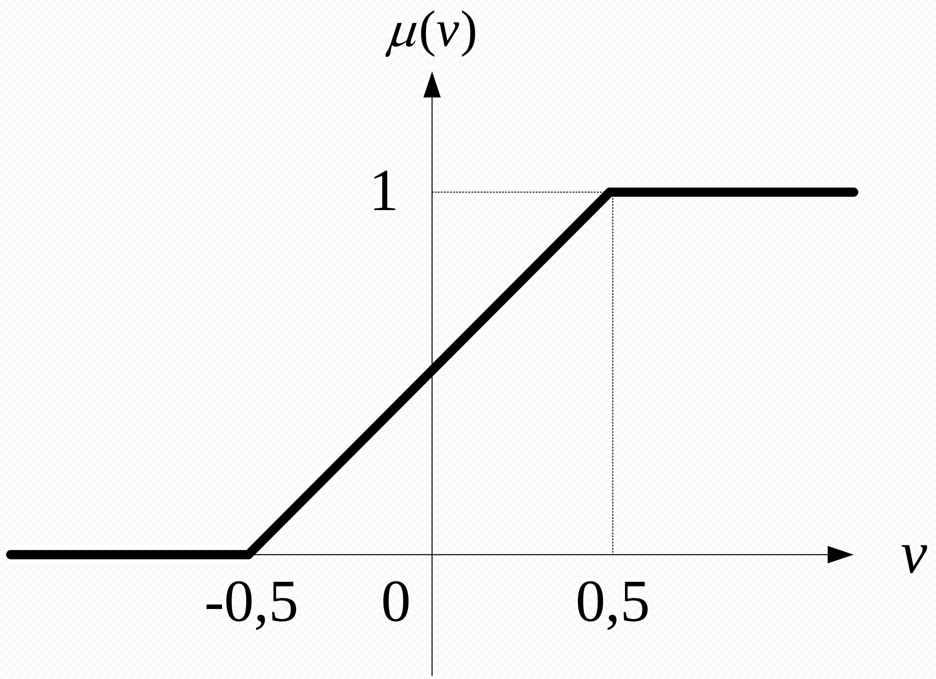
$$\mu(v) = \begin{cases} 1 & if\ v > \theta \\ 0 & if\ v = \theta \\ -1 & if\ v < \theta \end{cases}$$

Linear (Identity)



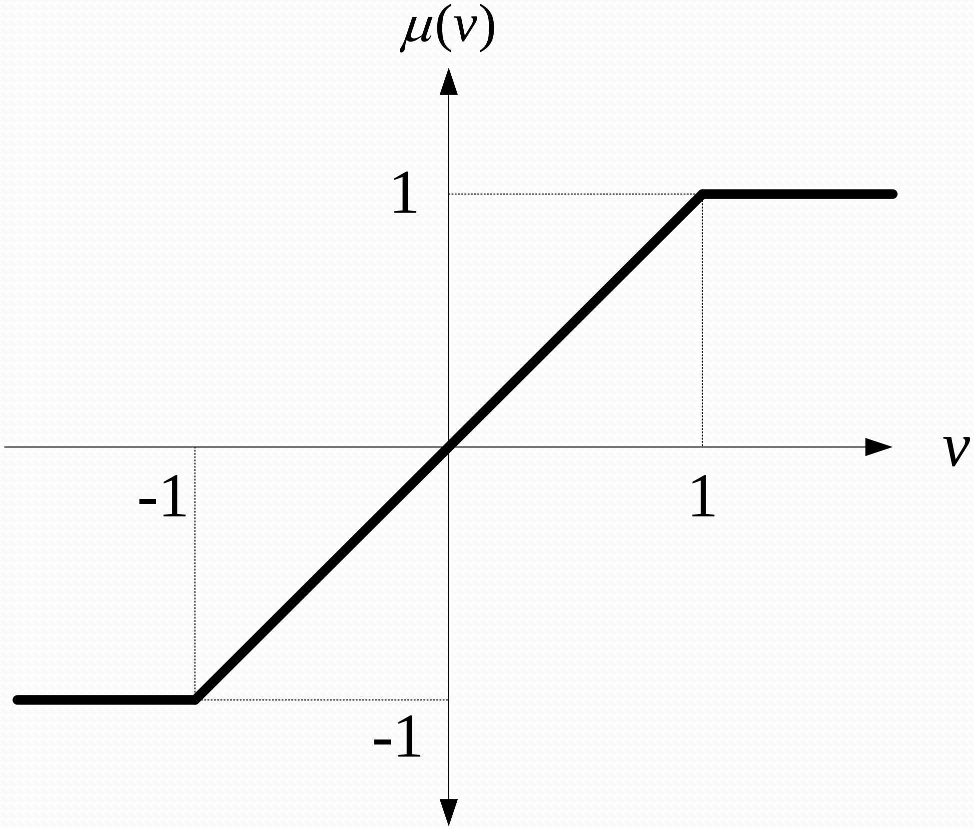
$$\mu(v) = v$$

Piecewise-linear



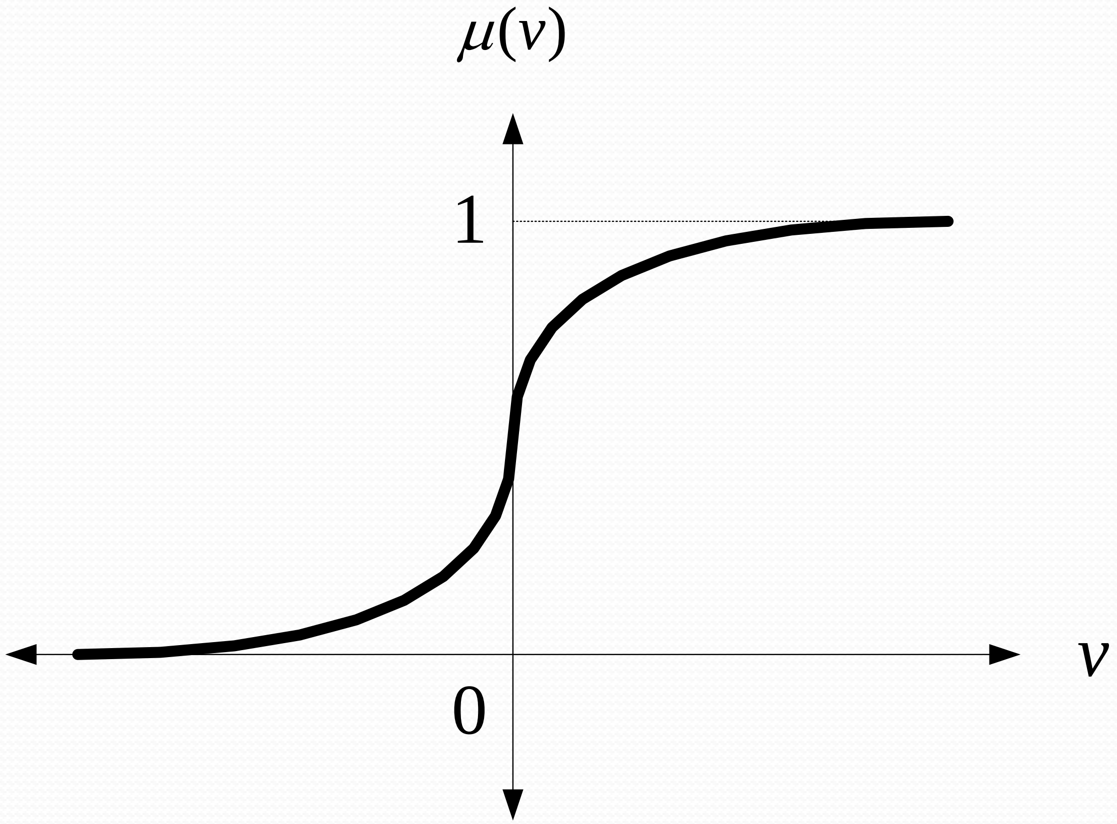
$$\mu(v) = \begin{cases} 1 & \text{jika } v \geq 0{,}5 \\ v + 0{,}5 & \text{jika } 0{,}5 > v > -0{,}5 \\ 0 & \text{jika } v \leq -0{,}5 \end{cases}$$

Symetric Piecewise-linear



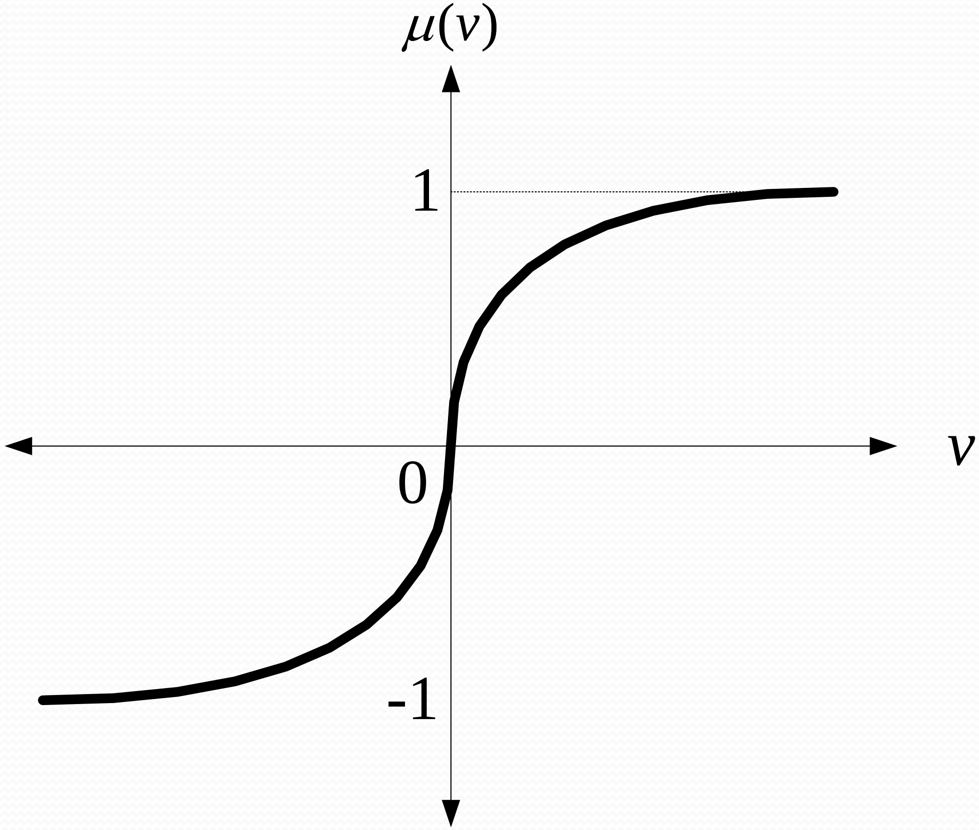
$$\mu(v) = \begin{cases} 1 & \text{jika } v \geq 1 \\ v & \text{jika } 1 > v > -1 \\ -1 & \text{jika } v \leq -1 \end{cases}$$

Sigmoid



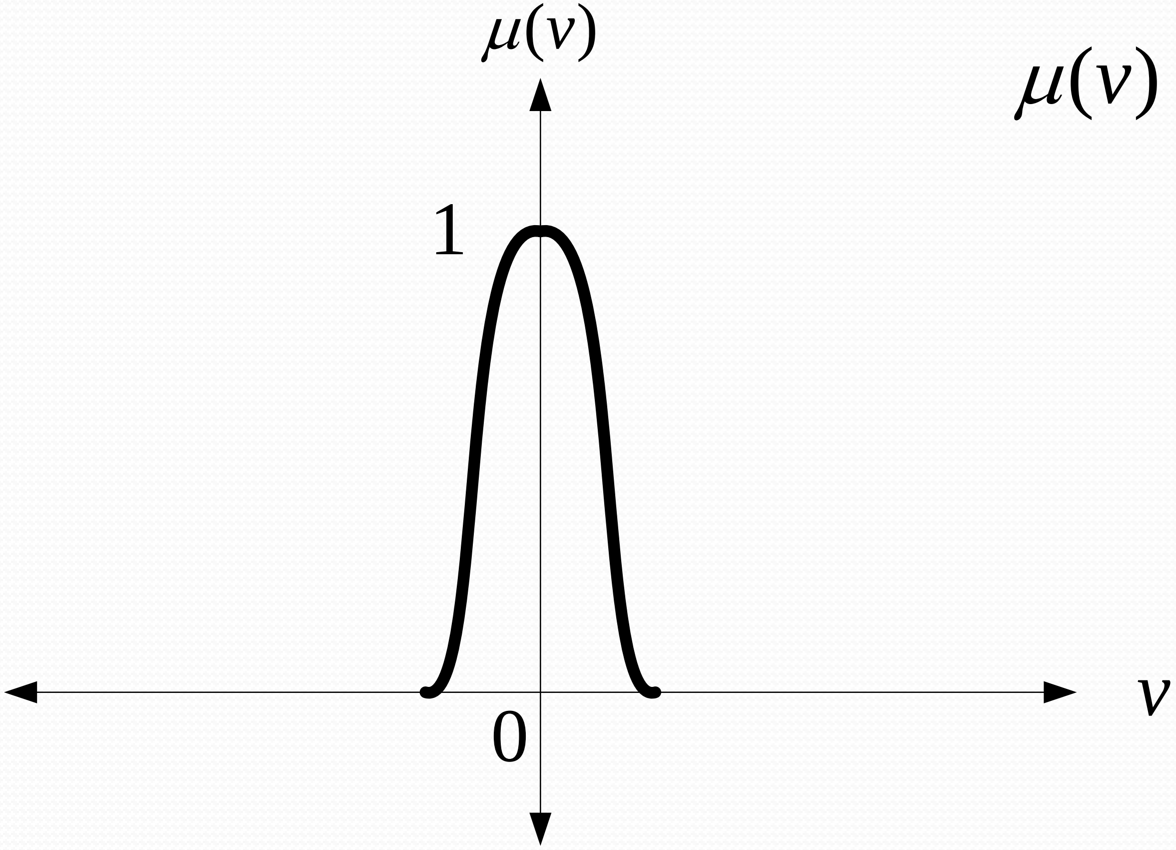
$$\mu(v) = \frac{1}{1+e^{-av}}$$

Symetric (Bipolar) Sigmoid



$$\mu(v) = \frac{1 - e^{-av}}{1 + e^{-av}}$$

Radial Basis Function (RBF)

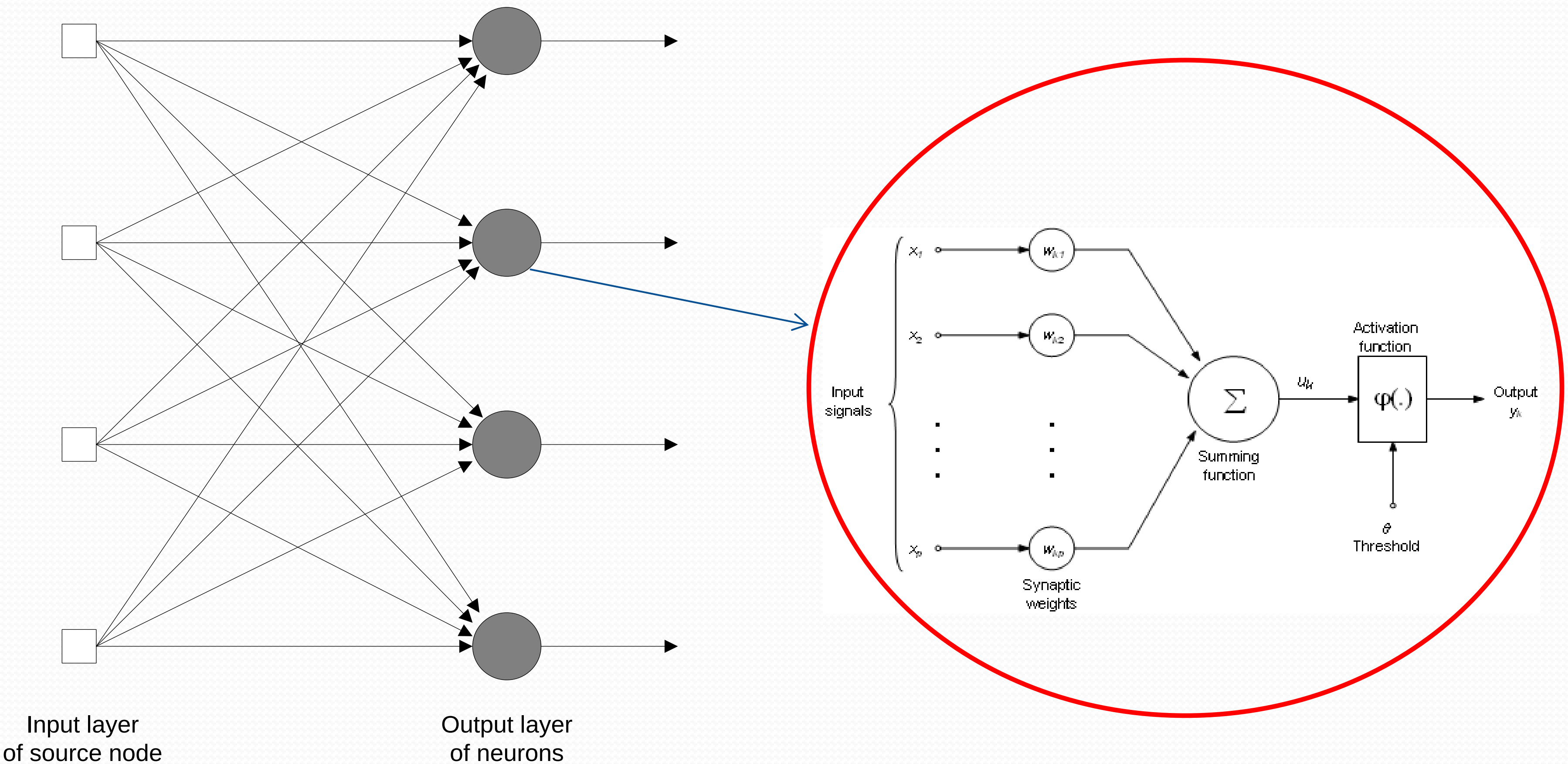


$$\mu(v) = e^{-av^2}$$

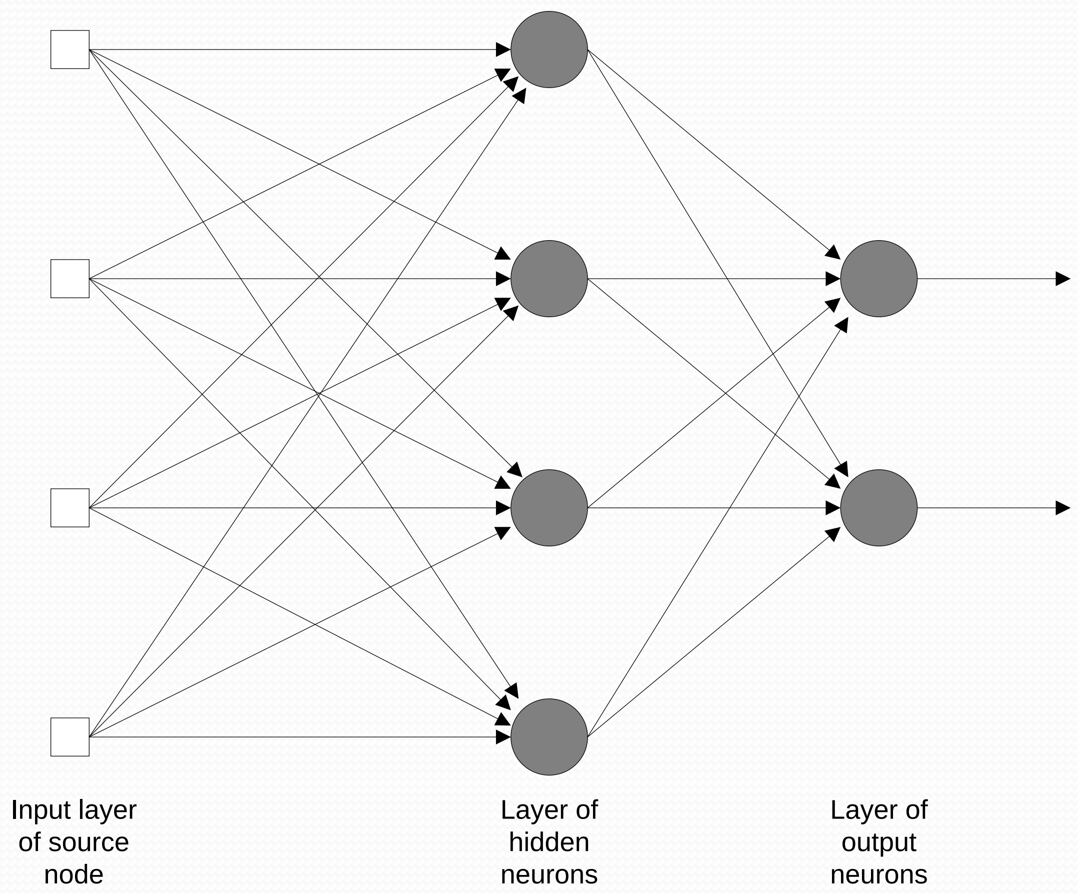
Arsitektur ANN

- Para ahli memodelkan sel syaraf otak manusia ke dalam berbagai arsitektur ANN (susunan *neuron*) yang berbeda-beda.
- Masing-masing arsitektur menggunakan algoritma belajar khusus.

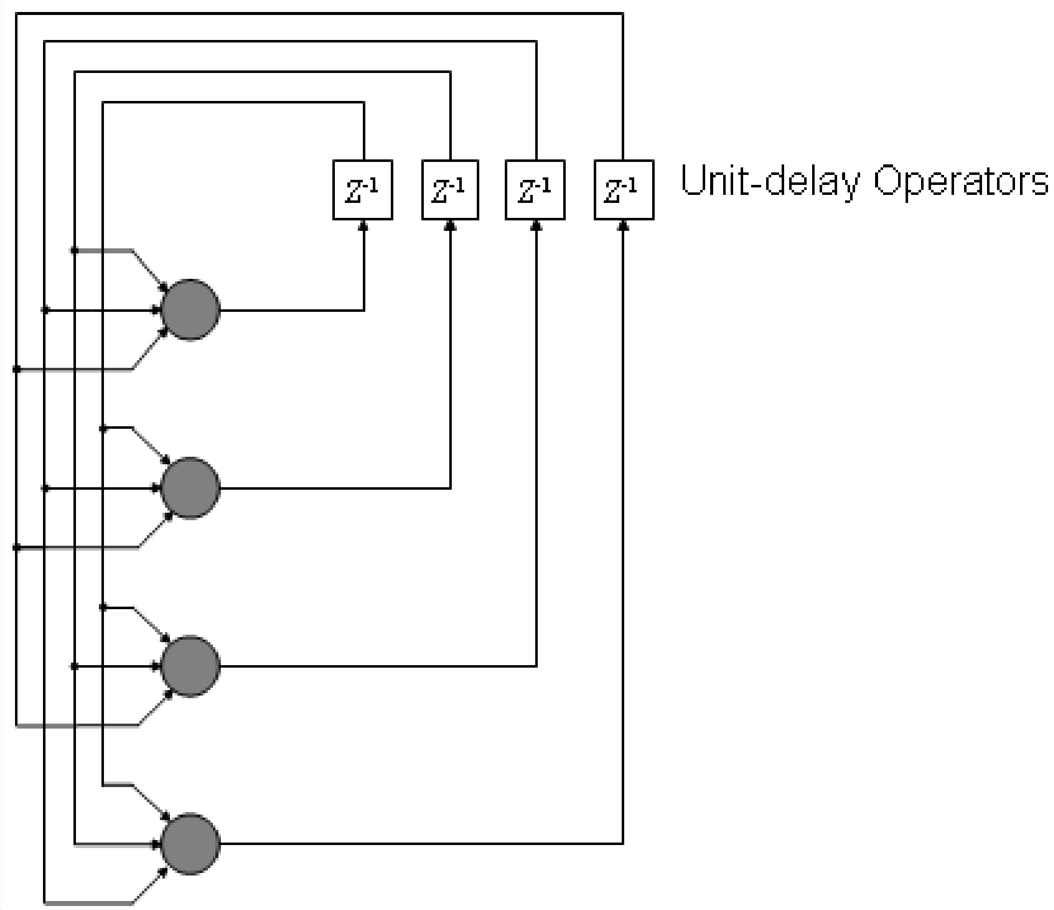
Single-Layer Feedforward Networks



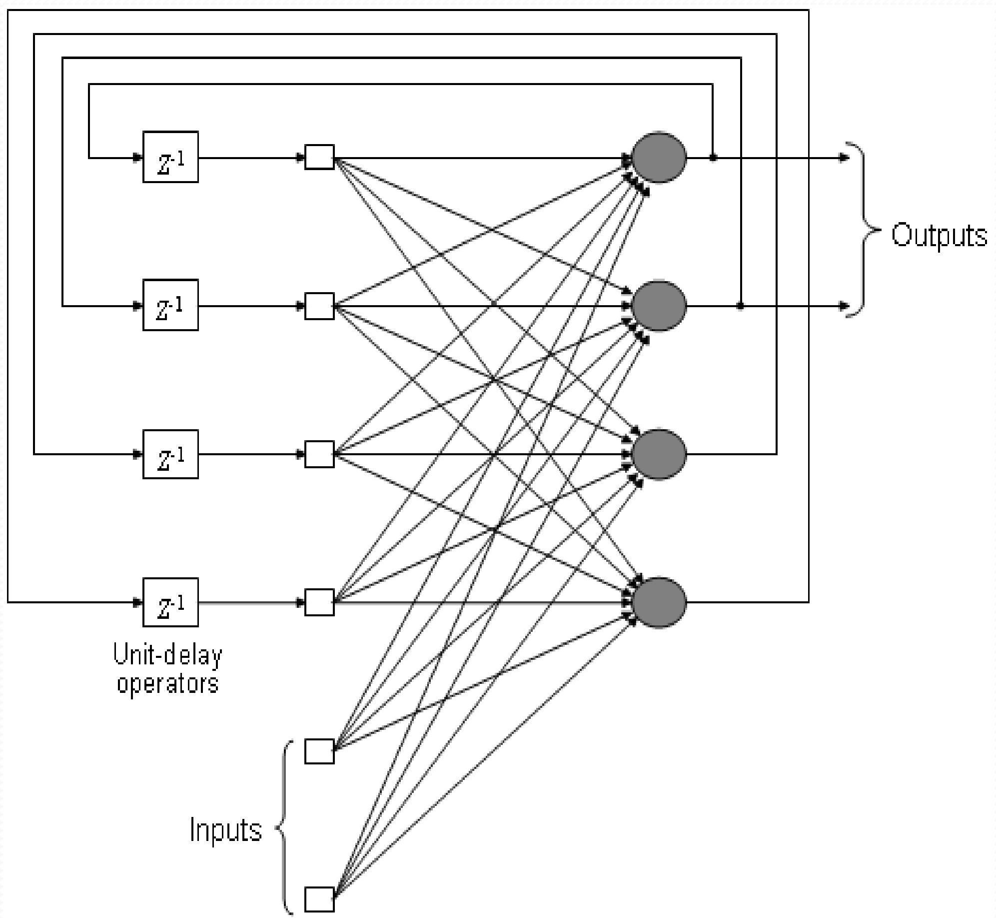
Multi-Layer Feedforward Networks



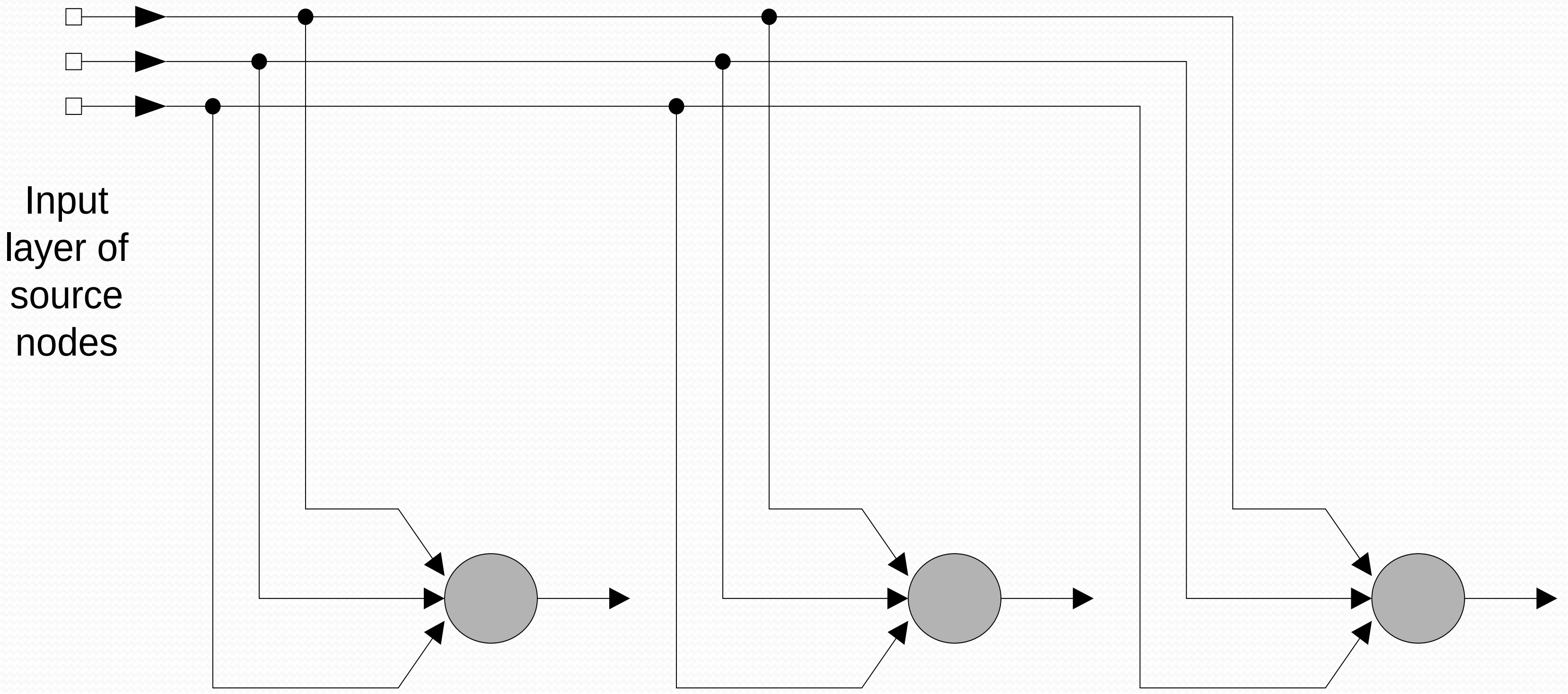
Recurrent Networks (tanpa *hidden neurons*)



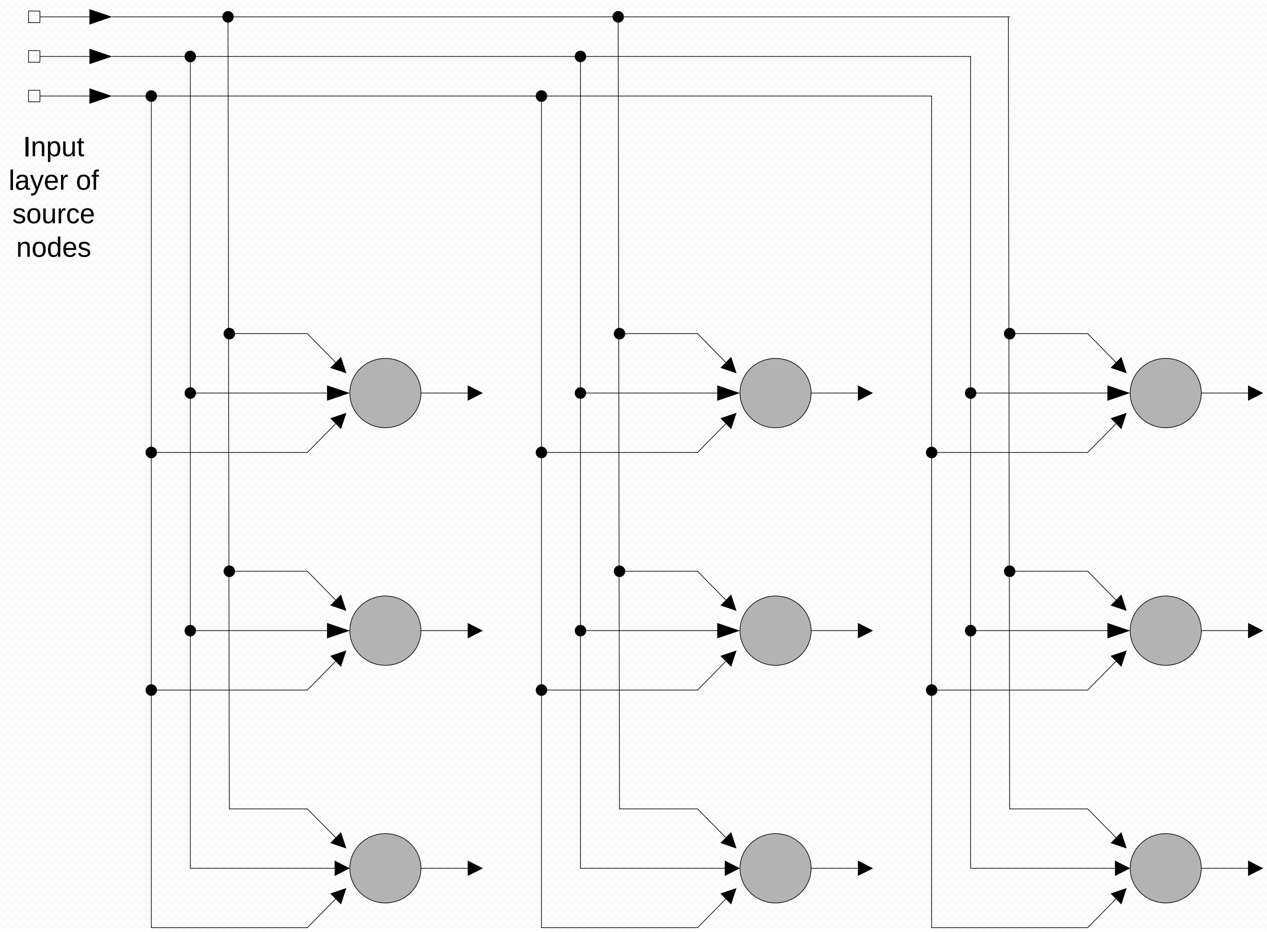
Recurrent Networks (dengan *hidden neurons*)



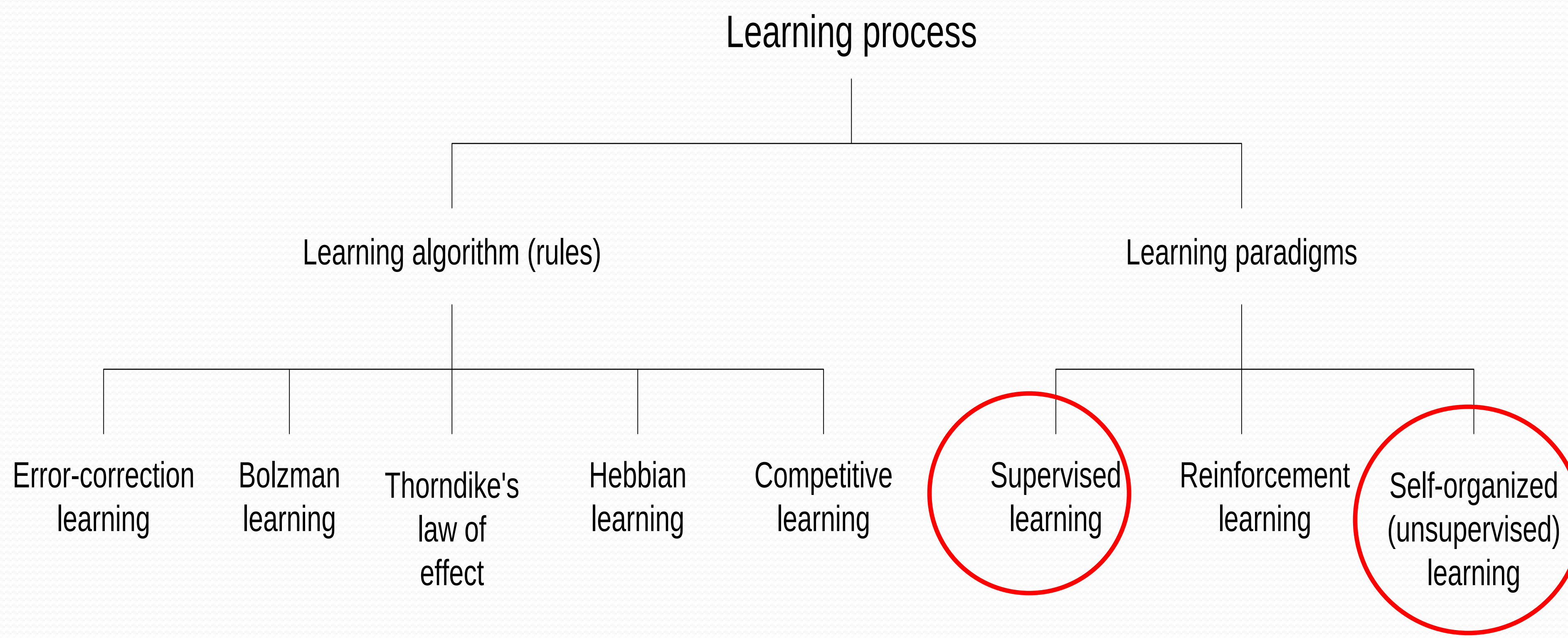
Lattice Structure (satu dimensi, 3 *neurons*)



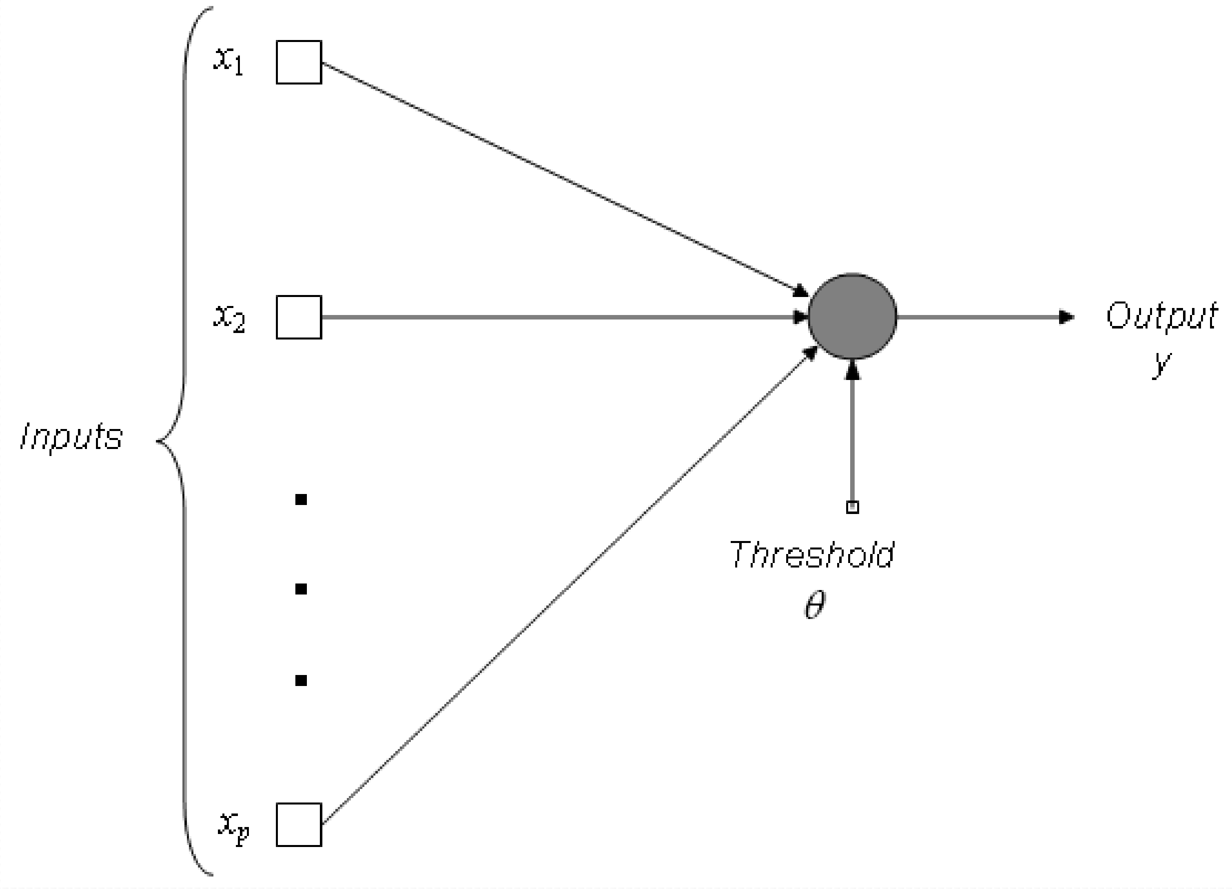
Lattice Structure (dua dimensi, 3x3 *neurons*)



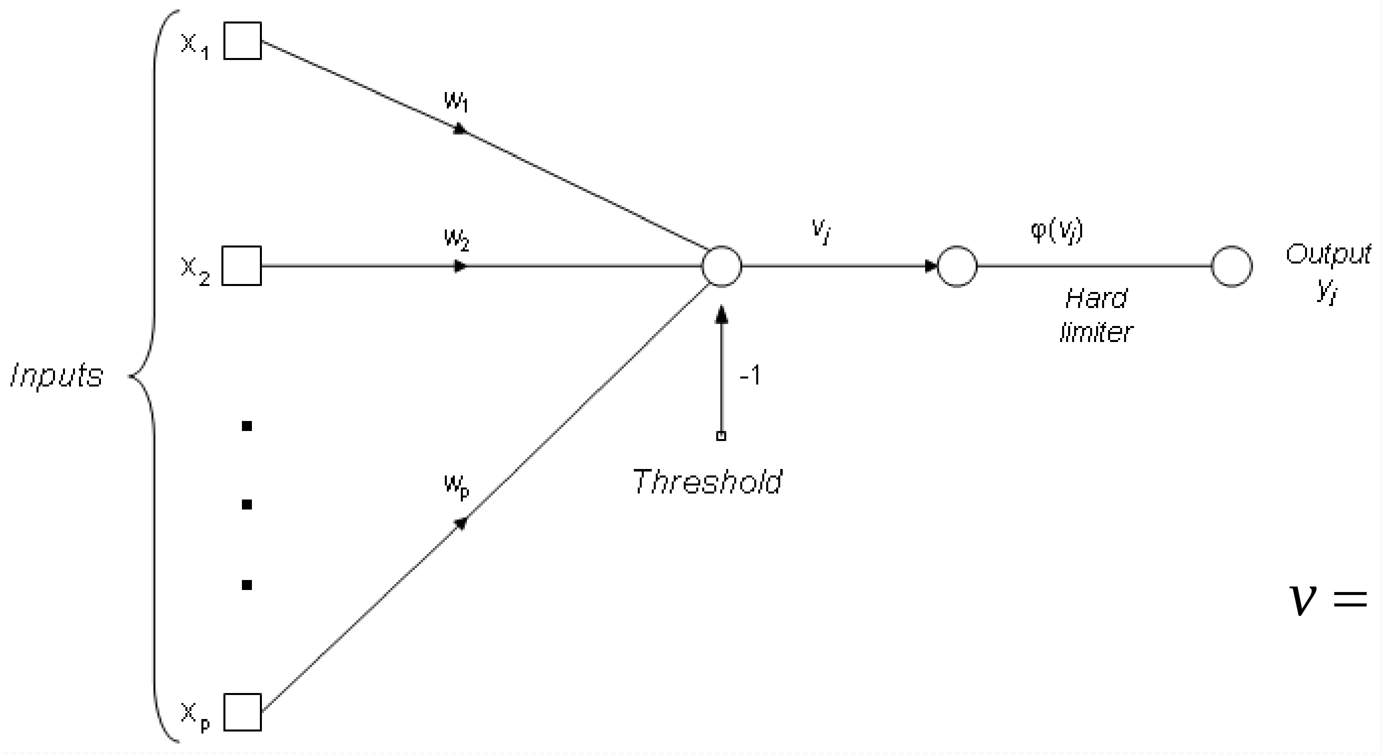
Proses Belajar (*Learning*)



Perceptron: Model

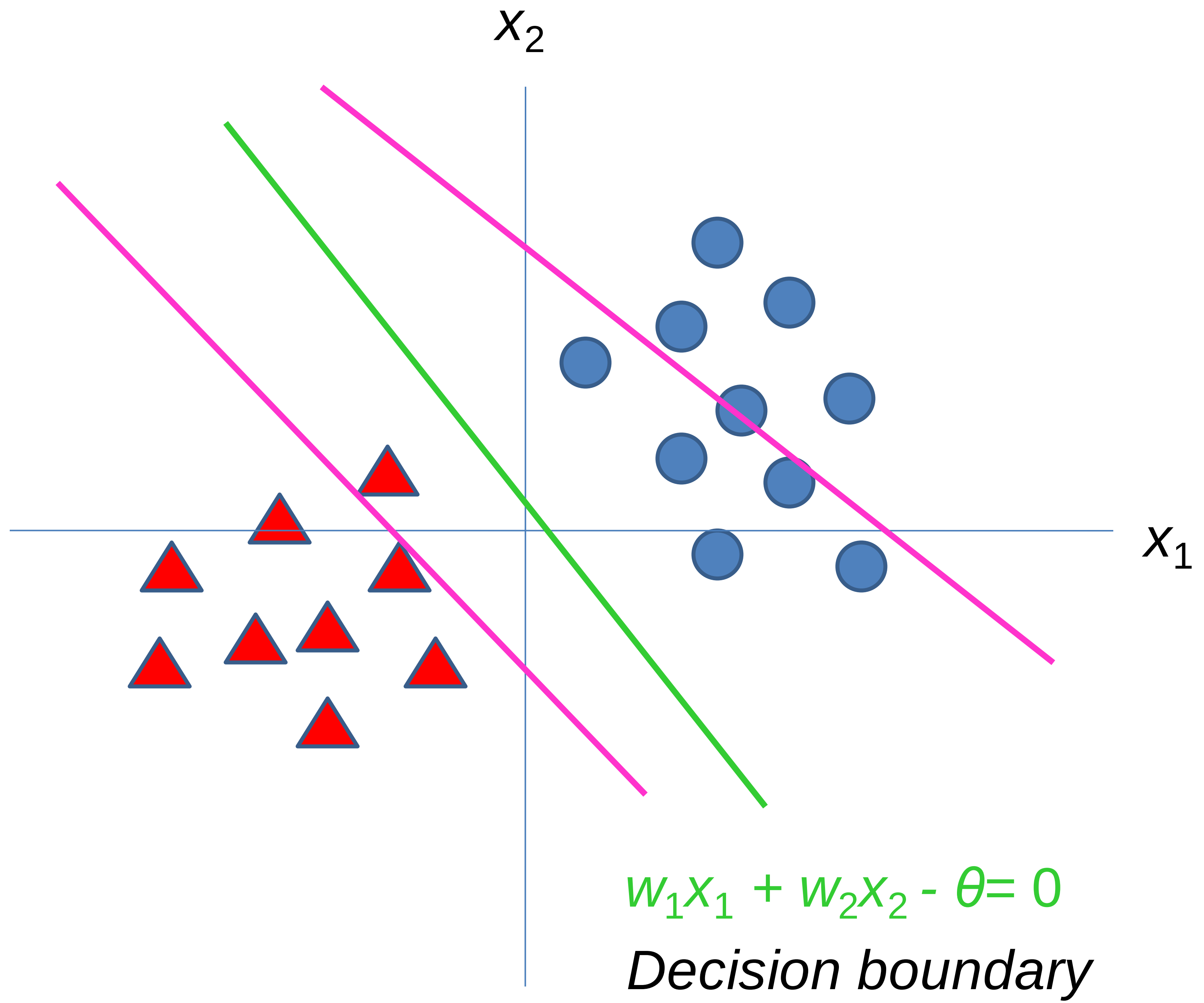


Perceptron: *Signal-Flow Graph*



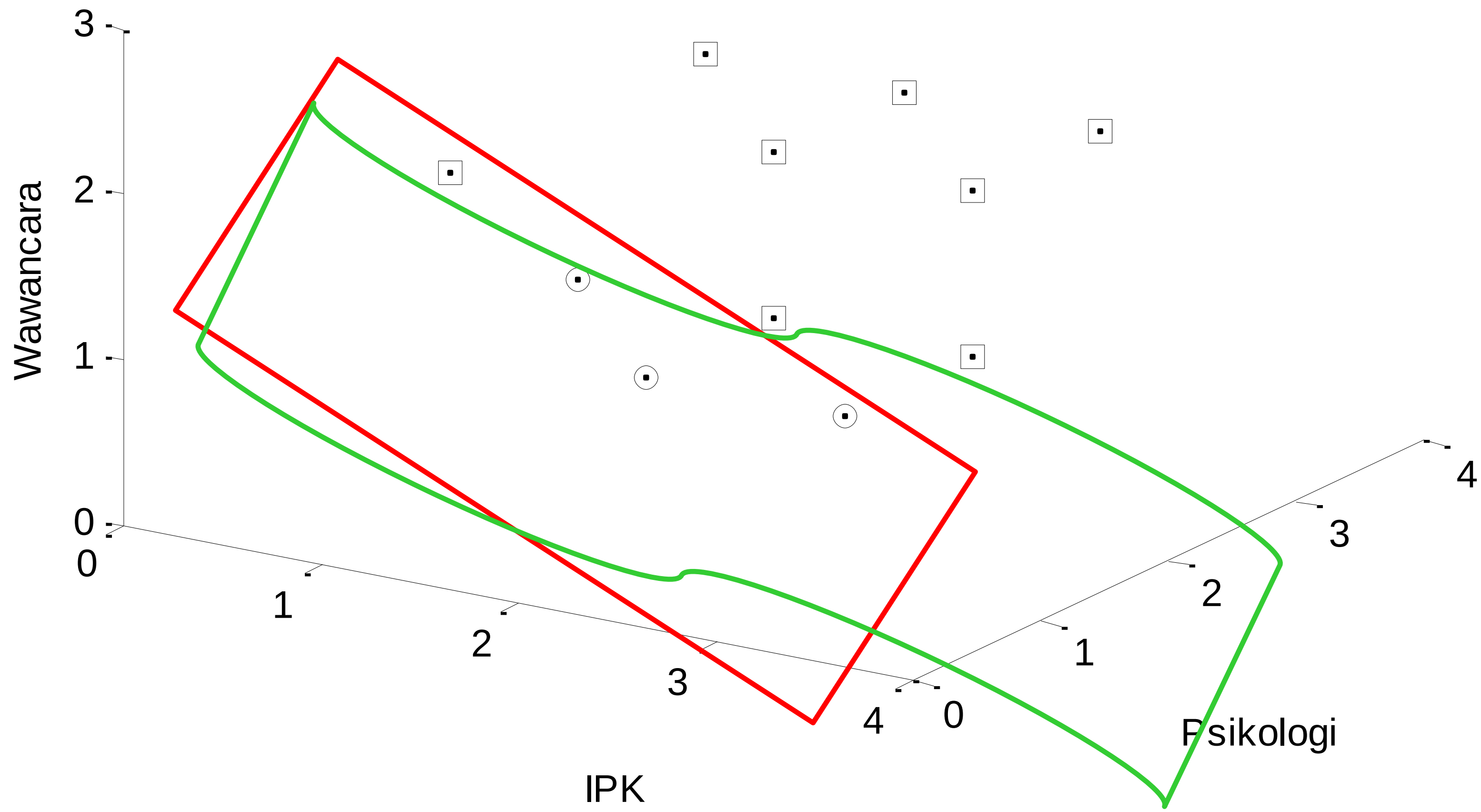
$$v = \sum_{i=1}^{p} w_i x_i - \theta$$

Decision boundary → $$\sum_{i=1}^{p} w_i x_i - \theta = 0$$

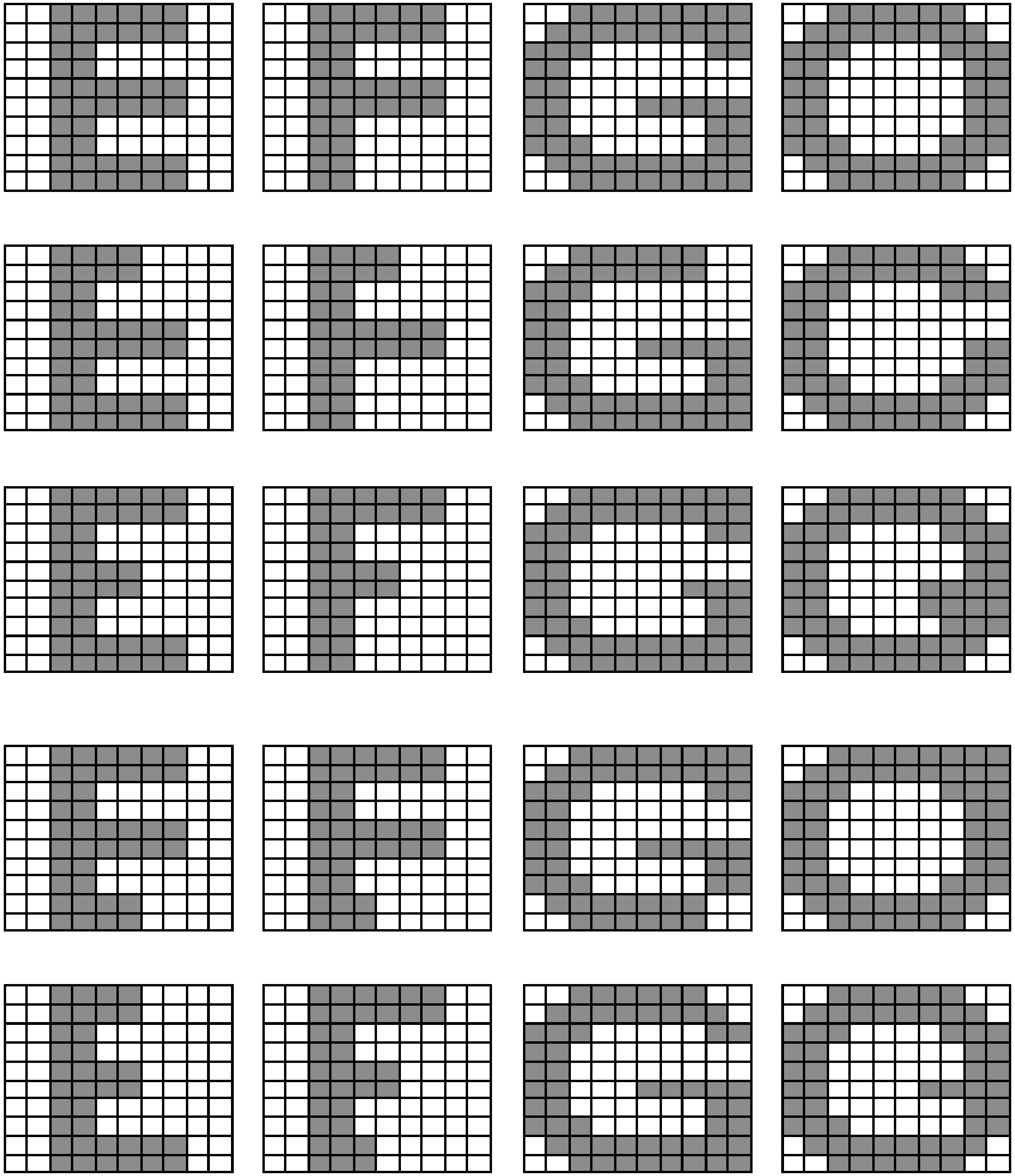


x_2
x_1
$w_1x_1 + w_2x_2 - \theta = 0$
Decision boundary

Pelamar	IPK	Psikologi	Wawancara	Diterima
P1	3	3	2	1
P2	3	2	2	1
P3	3	2	1	1
P4	3	1	1	0
P5	2	3	2	1
P6	2	2	2	1
P7	2	2	1	1
P8	2	1	1	0
P9	1	3	2	1
P10	1	2	1	0
P11	1	1	2	1



Wawancara
IPK
Psikologi
0
1
2
3
4

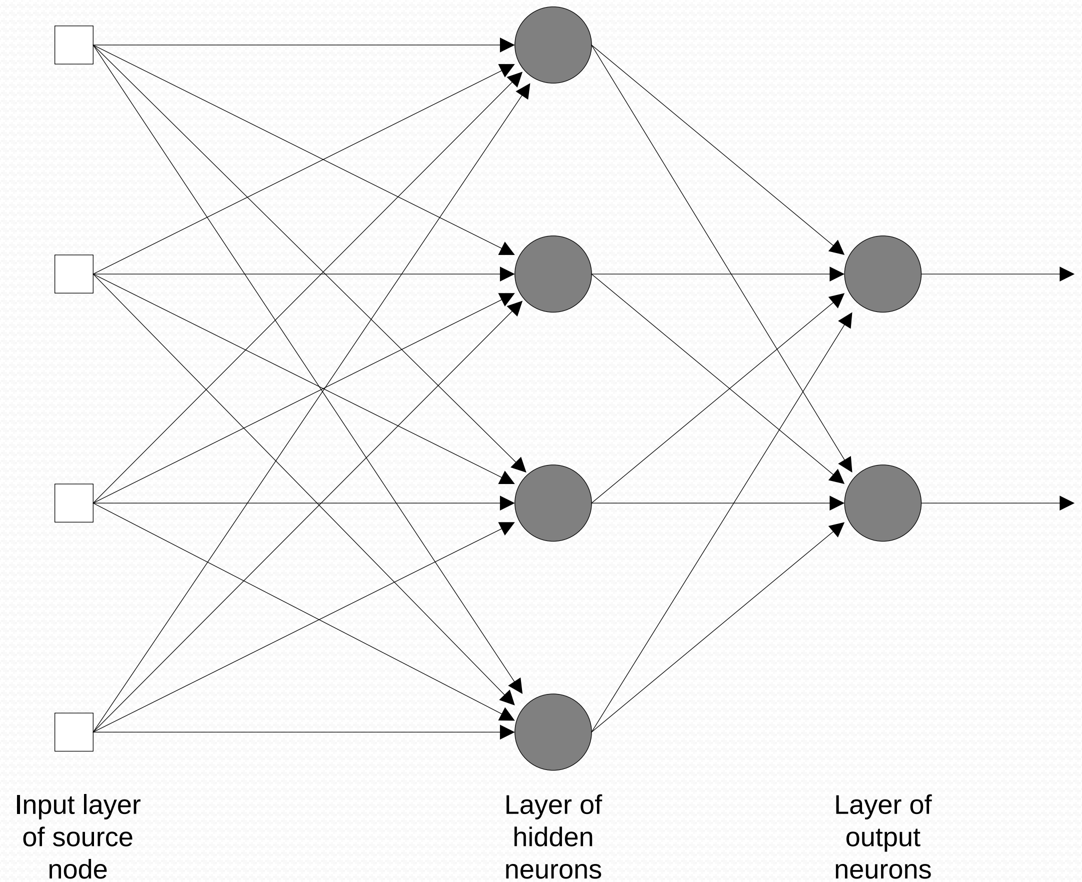


Visualisasi 100 dimensi?

Bisa dengan Perceptron?

Pola	Pix 1	Pix 2	Pix 3	Pix 4	Pix 5	…	Pix 100
E1	0	0	1	1	1	…	0
F1	0	0	1	1	1	…	0
G1	0	1	1	1	1	…	1
O1	0	1	1	1	1	…	1
..							
O5	0	1	1	1	1	…	1

Multi-Layer Perceptron (MLP)



Algoritma Belajar **Propagasi Balik**

- **Definisikan masalah**
  - Matriks pola masukan (P)
  - matriks target (T)
- **Inisialisasi parameter jaringan**
  - Arsitektur jaringan (misalkan I-H-O)
  - *Synaptic weights* → acak (atau dengan metode tertentu)
  - *Learning rate* (*lr*) → laju belajar
  - *Threshold* MSE → untuk menghentikan *learning*

Algoritma Belajar **Propagasi Balik**

- **Pelatihan Jaringan**
  - Perhitungan Maju

$$A1 = \frac{1}{1 + e^{(W1 * P + B1)}}$$

$$A2 = W2 * A1 + B2$$

$$E = T - A2$$

$$MSE = \frac{\sum E^2}{N}$$

Algoritma Belajar **Propagasi Balik**

- **Pelatihan Jaringan**
  - Perhitungan Mundur

$$D2 = (1 - A2^2) * E$$

$$D1 = (1 - A1^2) * (W2 * D2)$$

$$dW1 = dW1 + (lr * D1 * P)$$

$$dB1 = dB1 + (lr * D1)$$

$$dW2 = dW2 + (lr * D2 * P)$$

$$dB2 = dB2 + (lr * D2)$$

Algoritma Belajar **Propagasi Balik**

- **Pelatihan Jaringan**
  - Perhitungan Mundur

$$W1 = W1 + dW1$$

$$B1 = B1 + dB1$$

$$W2 = W2 + dW2$$

$$B2 = B2 + dB2$$

Algoritma Belajar **Propagasi Balik**

- Langkah-langkah di atas adalah untuk satu kali siklus pelatihan (satu *epoch*).
- Biasanya, pelatihan harus diulang-ulang lagi hingga jumlah siklus tertentu atau telah tercapai MSE yang diinginkan.
- Hasil akhir dari pelatihan jaringan adalah bobot-bobot ***W*1**, ***W*2**, ***B*1** dan ***B*2**.

Pengenalan Karakter E, F, G, O

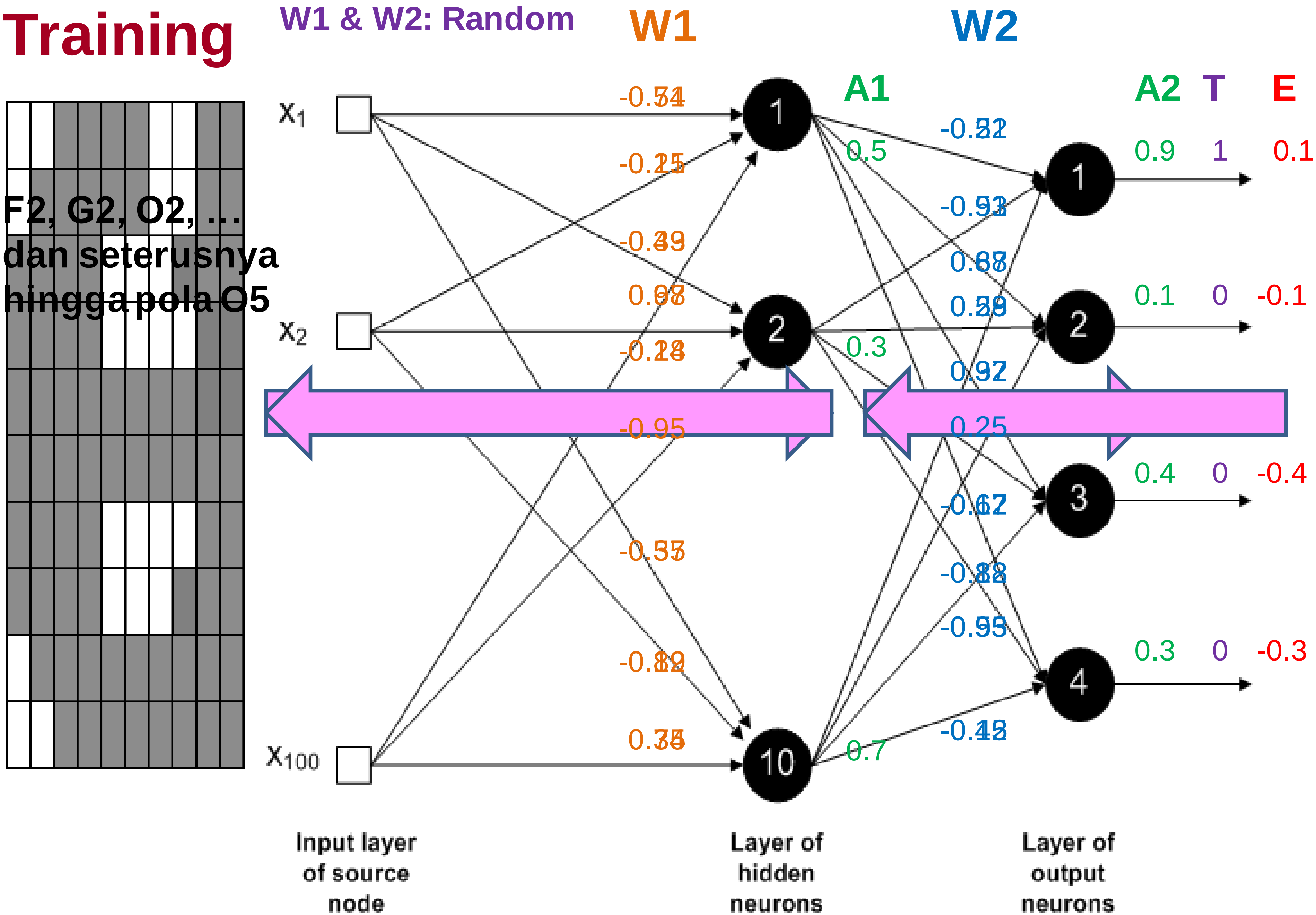
Matriks P

Pola	Pix 1	Pix 2	Pix 3	Pix 4	Pix 5	...	Pix 100
E1	0	0	1	1	1	...	0
F1	0	0	1	1	1	...	0
G1	0	1	1	1	1	...	1
O1	0	1	1	1	1	...	1
E2	0	0	1	1	1	...	0
...	...	...	...	...	...	...	...
O5	0	1	1	1	1	...	1

Matriks T

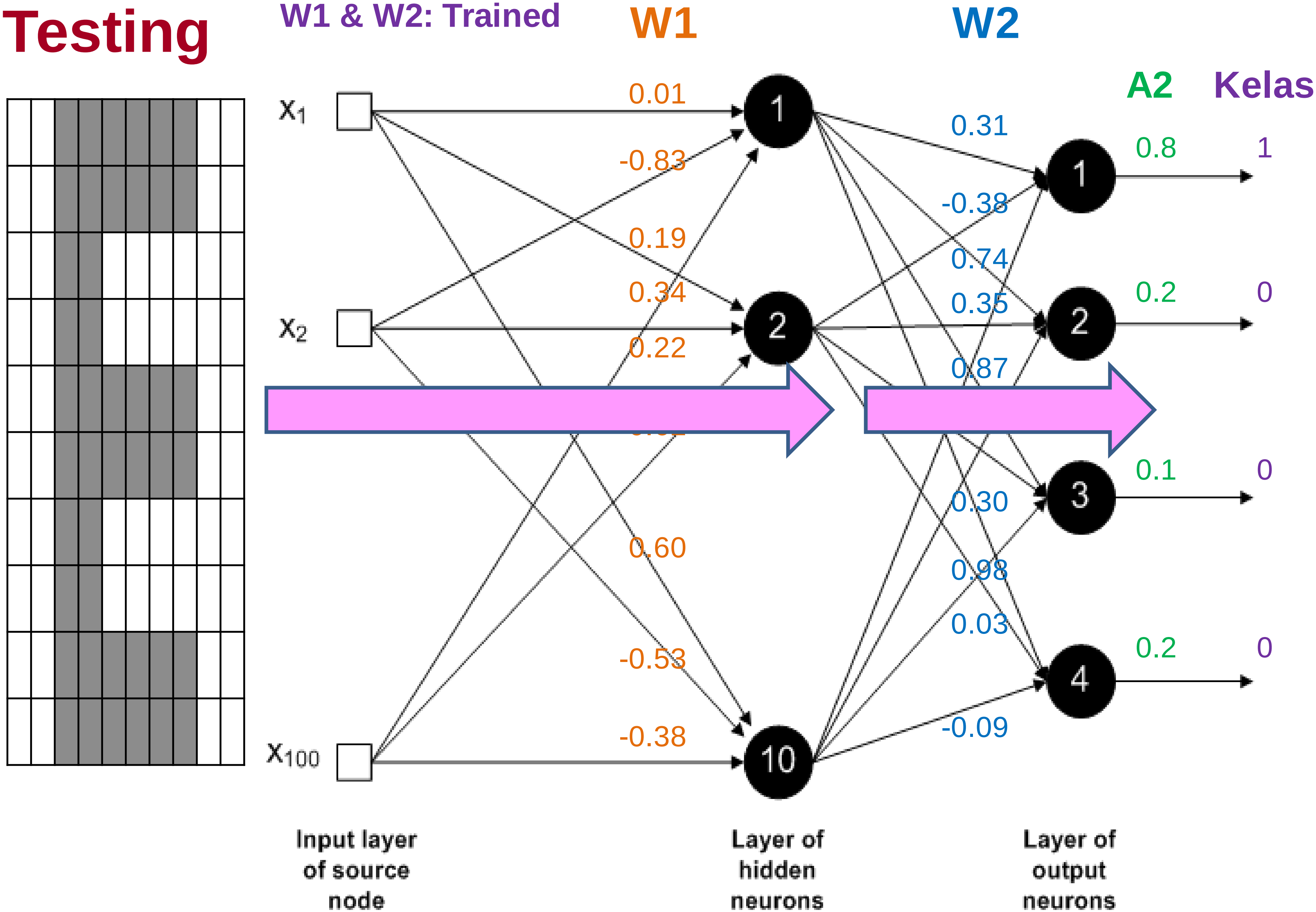
N1	N2	N3	N4	Kelas
1	0	0	0	E
0	1	0	0	F
0	0	1	0	G
0	0	0	1	O
1	0	0	0	E
...	...	...	...	...
0	0	0	1	O

Training


W1 & W2: Random
W1
W2
F2, G2, O2, ...
dan seterusnya
hingga pola O5
x1
x2
x100
1
2
10
A1
0.5
0.3
0.7
-0.95
0.25
A2
T
E
0.9
1
0.1
0.1
0
-0.1
0.4
0
-0.4
0.3
0
-0.3
Input layer
of source
node
Layer of
hidden
neurons
Layer of
output
neurons

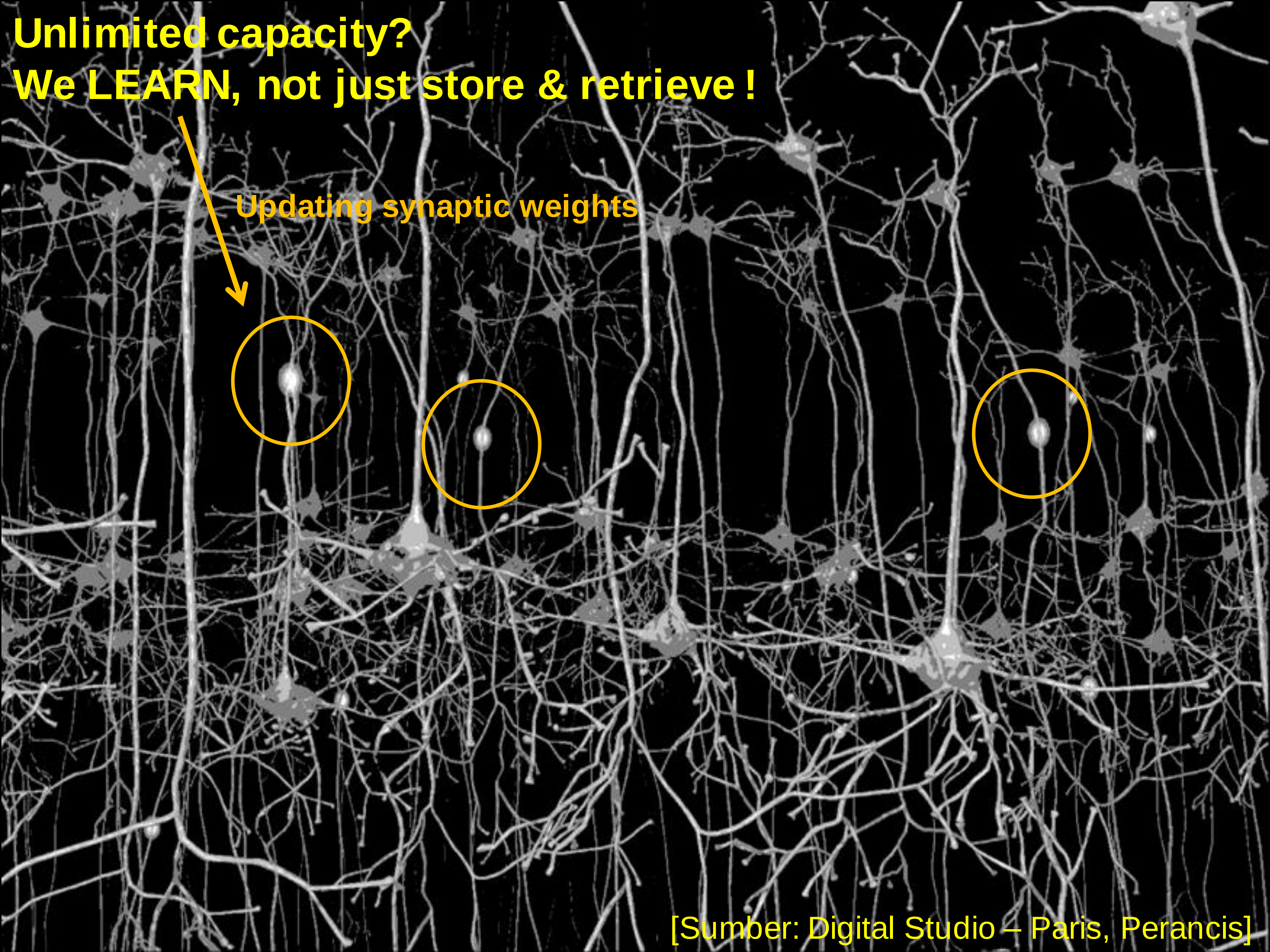
Testing

W1 & W2: Trained



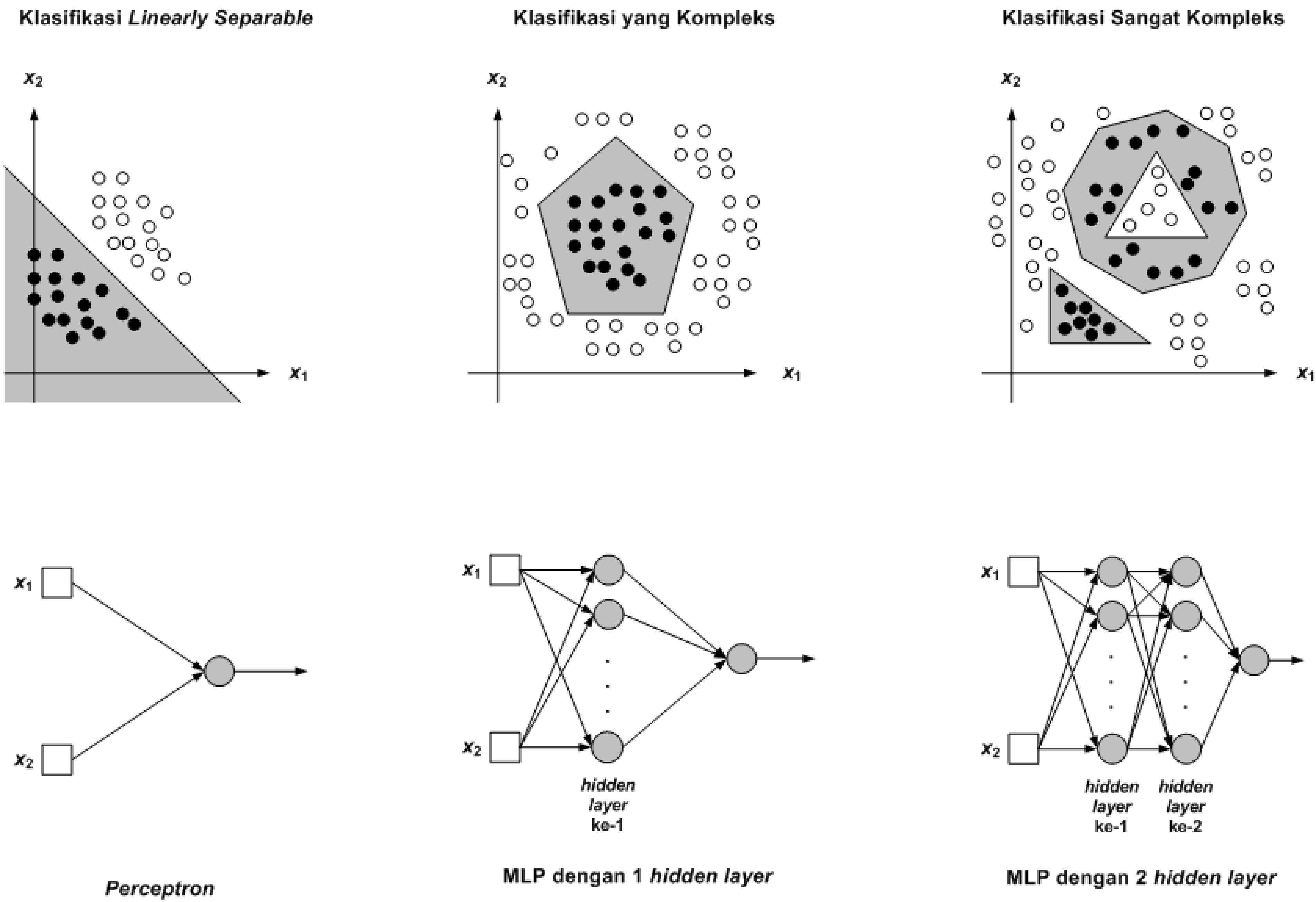
Demo Program

- Pengenalan Karakter
- 4 kelas huruf: E, F, G, O
- Data Latih
  - 4 kelas, setiap kelas berisi 5 pola → 20 pola huruf
  - MLP = 100-10-4
  - Algoritma Learning Propagasi Balik


Unlimited capacity?
We LEARN, not just store & retrieve !
Updating synaptic weights
[Sumber: Digital Studio – Paris, Perancis]

Permasalahan pada MLP

- Bagaimana struktur ANN yang optimal?
  - Jumlah *hidden layer*
  - Jumlah neuron pada hidden layer
  - Jumlah neuron pada output layer
  - Fungsi aktivasi yang optimal
- *Learning Rate*
- Kapan Menghentikan *Learning*

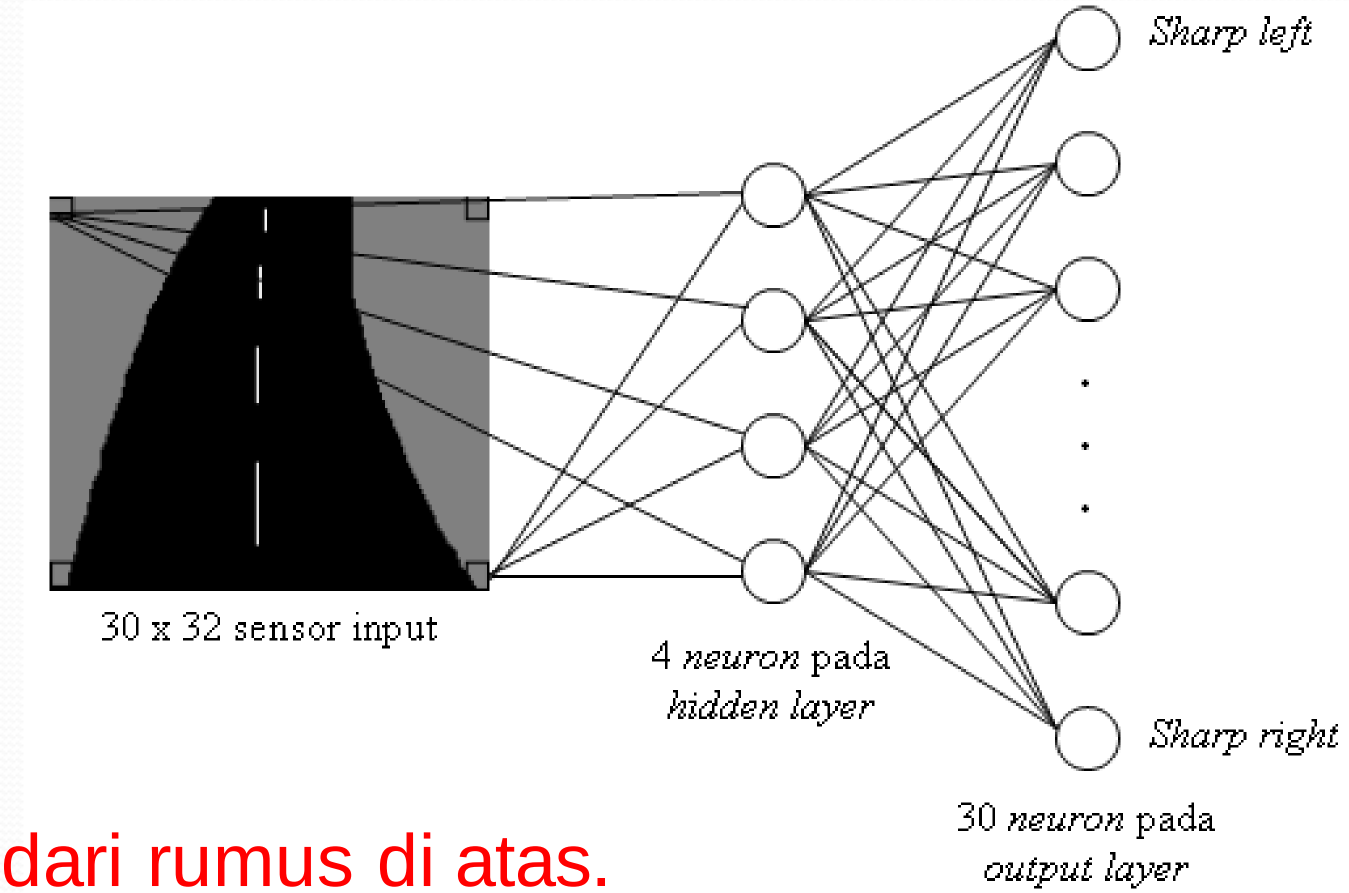

Klasifikasi Linearly Separable
Klasifikasi yang Kompleks
Klasifikasi Sangat Kompleks
x2
x1
x1
x2
Perceptron
hidden layer ke-1
MLP dengan 1 hidden layer
hidden layer ke-1
hidden layer ke-2
MLP dengan 2 hidden layer

Jumlah neuron pada hidden layer?

$$N_H = \sqrt{N_I N_O}$$

Perhatian: Rumus ini hanya perkiraan (tidak pasti).

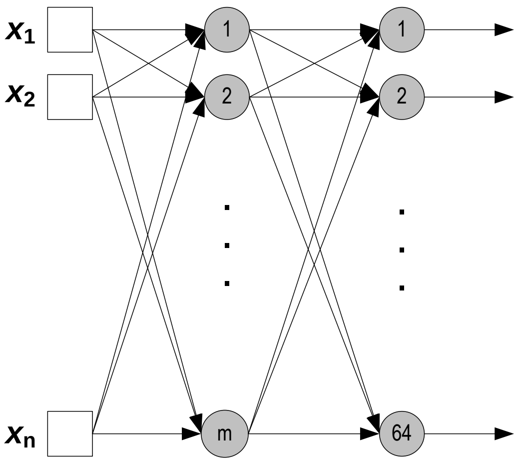
ALVINN: MLP 960-4-30

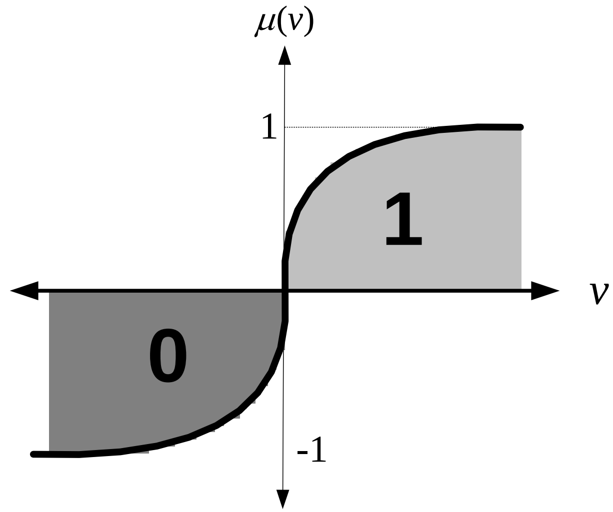


Jauh dari rumus di atas.

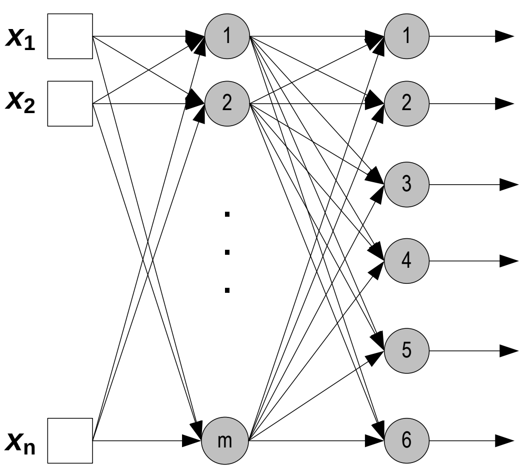
Jumlah neuron pada output layer

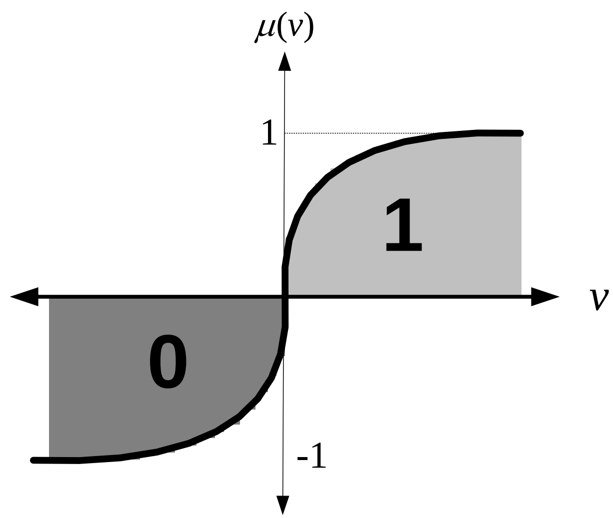
- **Deterministik**: mudah dihitung berdasarkan permasalahan yang dihadapi.
- Untuk pengenalan karakter dengan **64 kelas**: ('a', 'b', ..., 'z', 'A', 'B', ..., 'Z', '0', '1', ... '9', '-', '+'), perlu berapa output neuron?



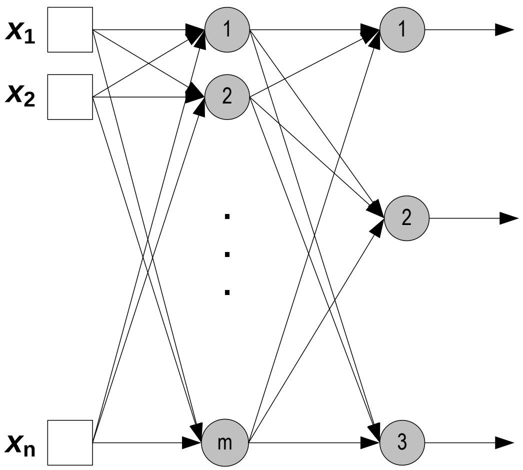


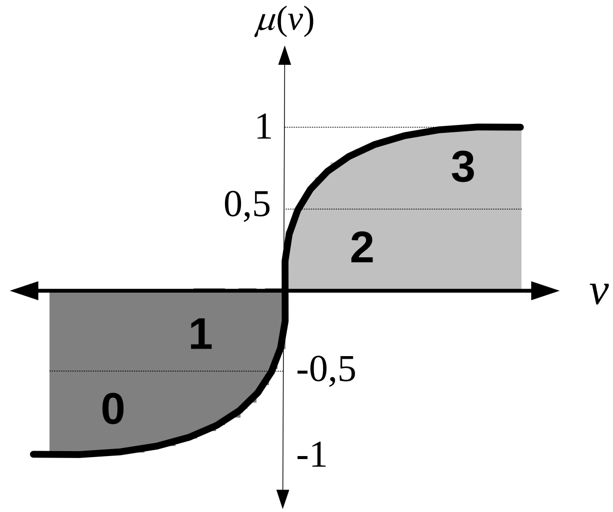
Output dari *neuron* ke-							Kelas
1	2	3	4	5	...	64	
1	0	0	0	0	...	0	a
0	1	0	0	0	...	0	b
0	0	1	0	0	...	0	c
0	0	0	1	0	...	0	d
0	0	0	0	1	...	0	e
...	...	...	...	...	...	...	...
0	0	0	0	0	...	1	+



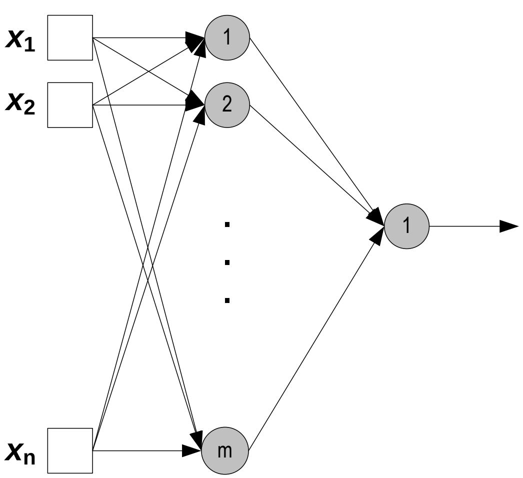


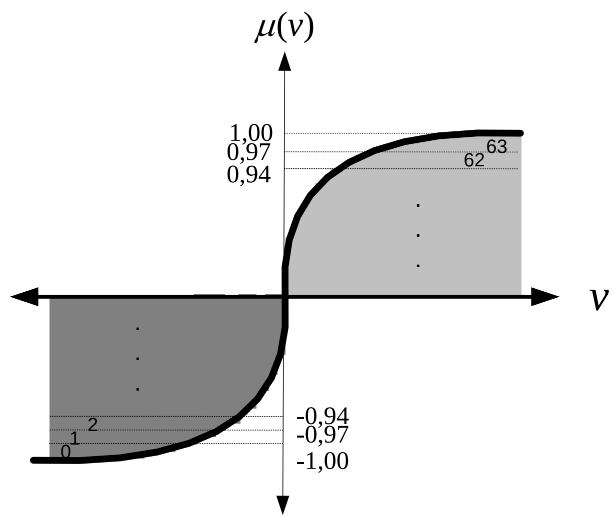
Output neuron ke-						Kelas
1	2	3	4	5	6	
0	0	0	0	0	0	a
0	0	0	0	0	1	b
0	0	0	0	1	0	c
0	0	0	0	1	1	d
0	0	0	1	0	0	e
...	...	...	...	...	...	...
1	1	1	1	1	1	+



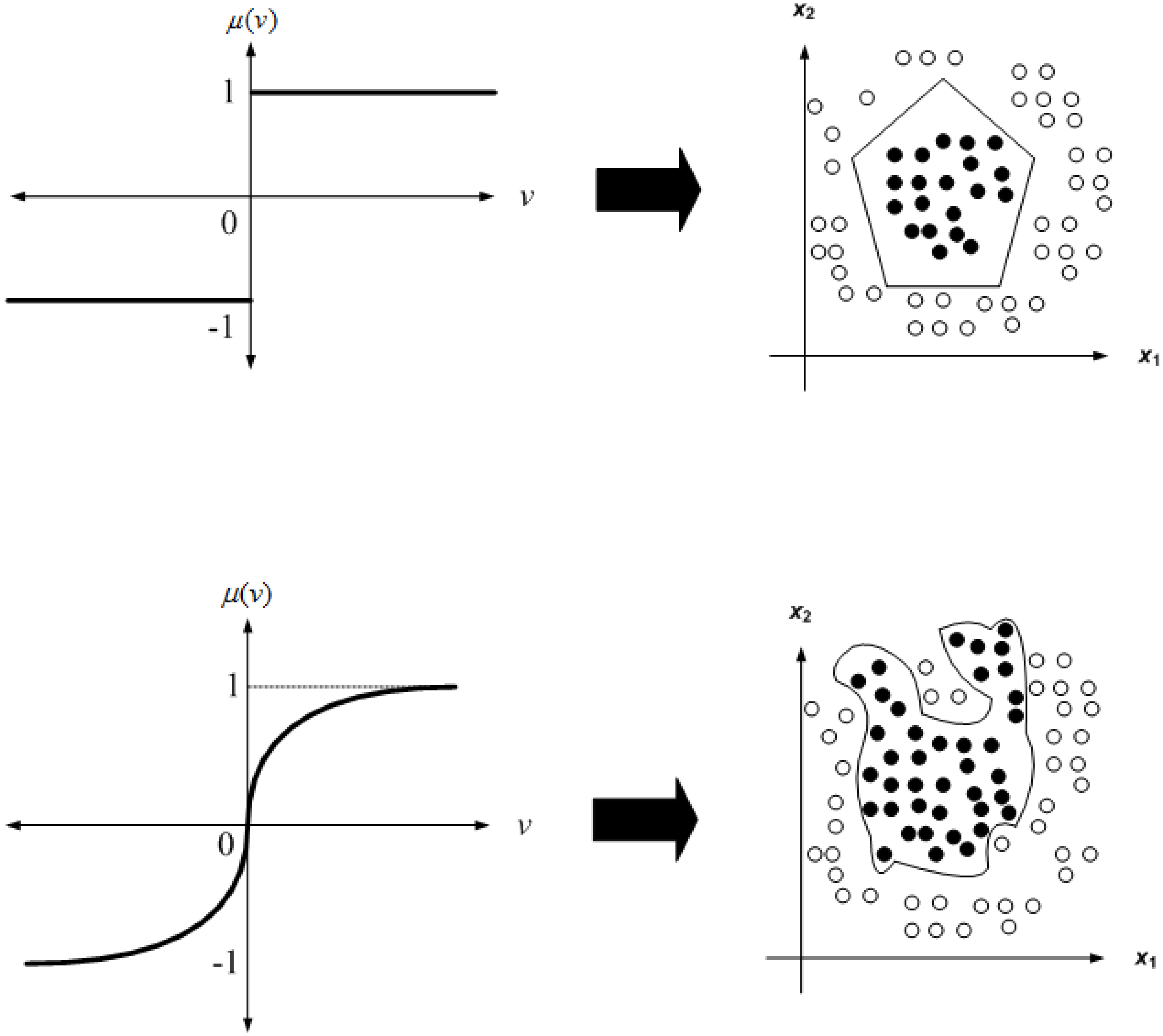


Output neuron ke-			Kelas
1	2	3	
0	0	0	a
0	0	1	b
0	0	2	c
0	0	3	d
0	1	0	e
...	...	...	...
3	3	3	+



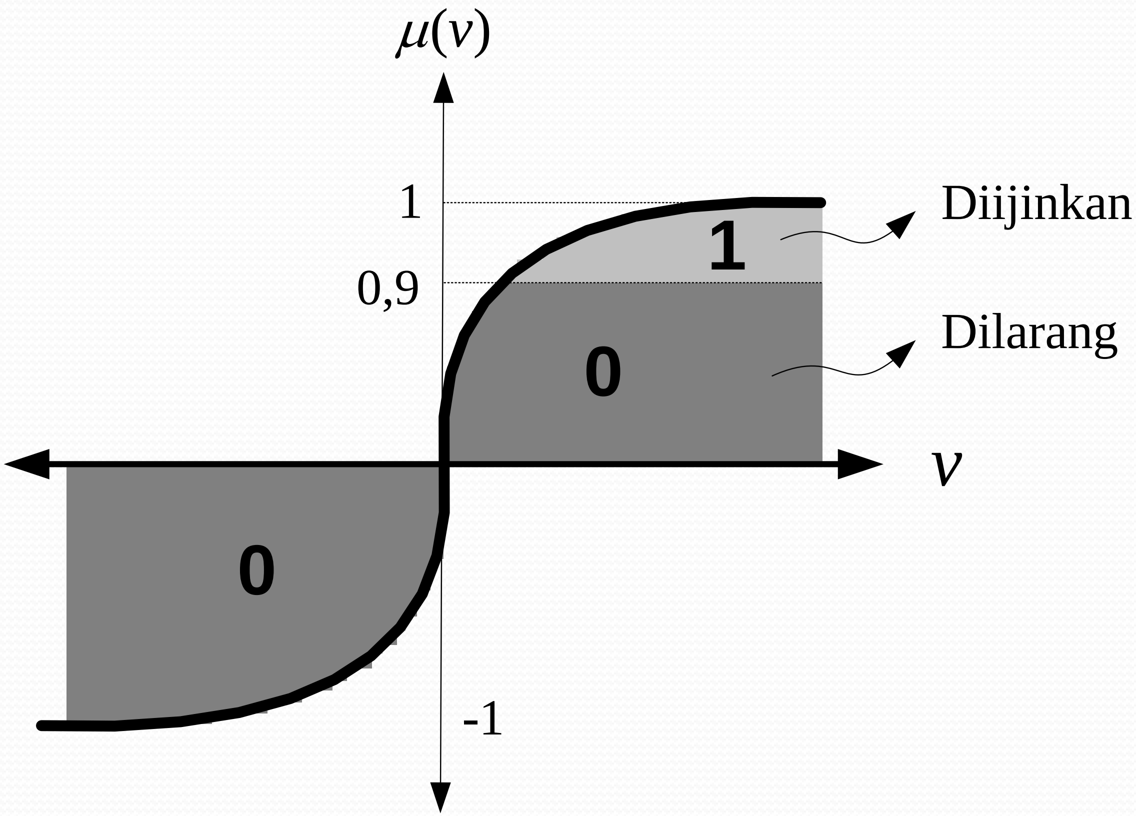


Output neuron ke-	Kelas
1	
0	a
1	b
2	c
3	d
4	e
...	...
63	+

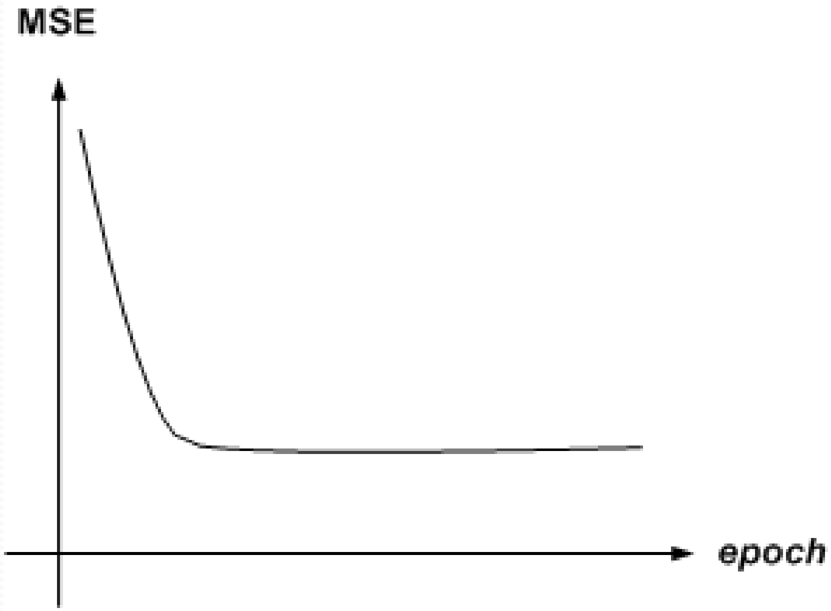


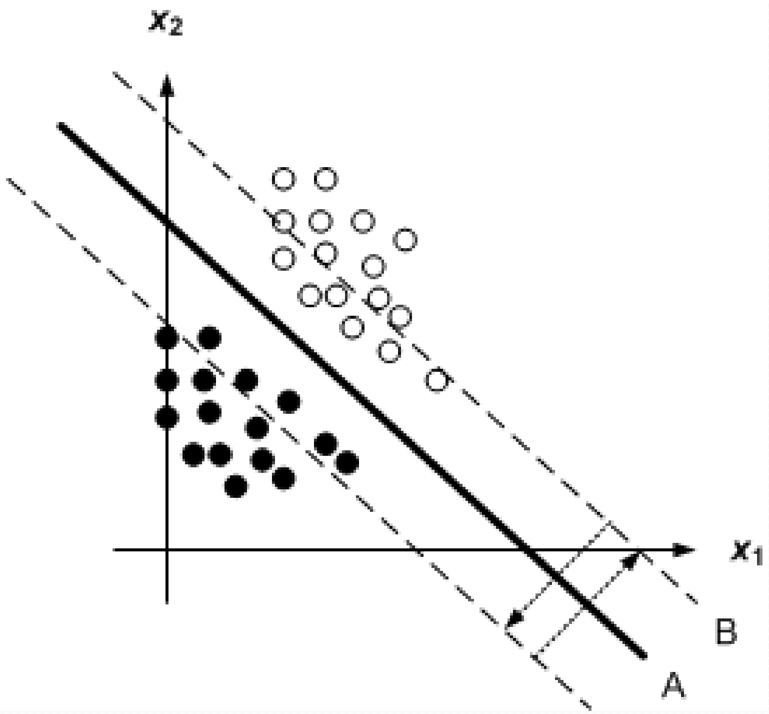
$\mu(v)$
1
0
-1
v
x_2
x_1
$\mu(v)$
1
0
-1
v
x_2
x_1

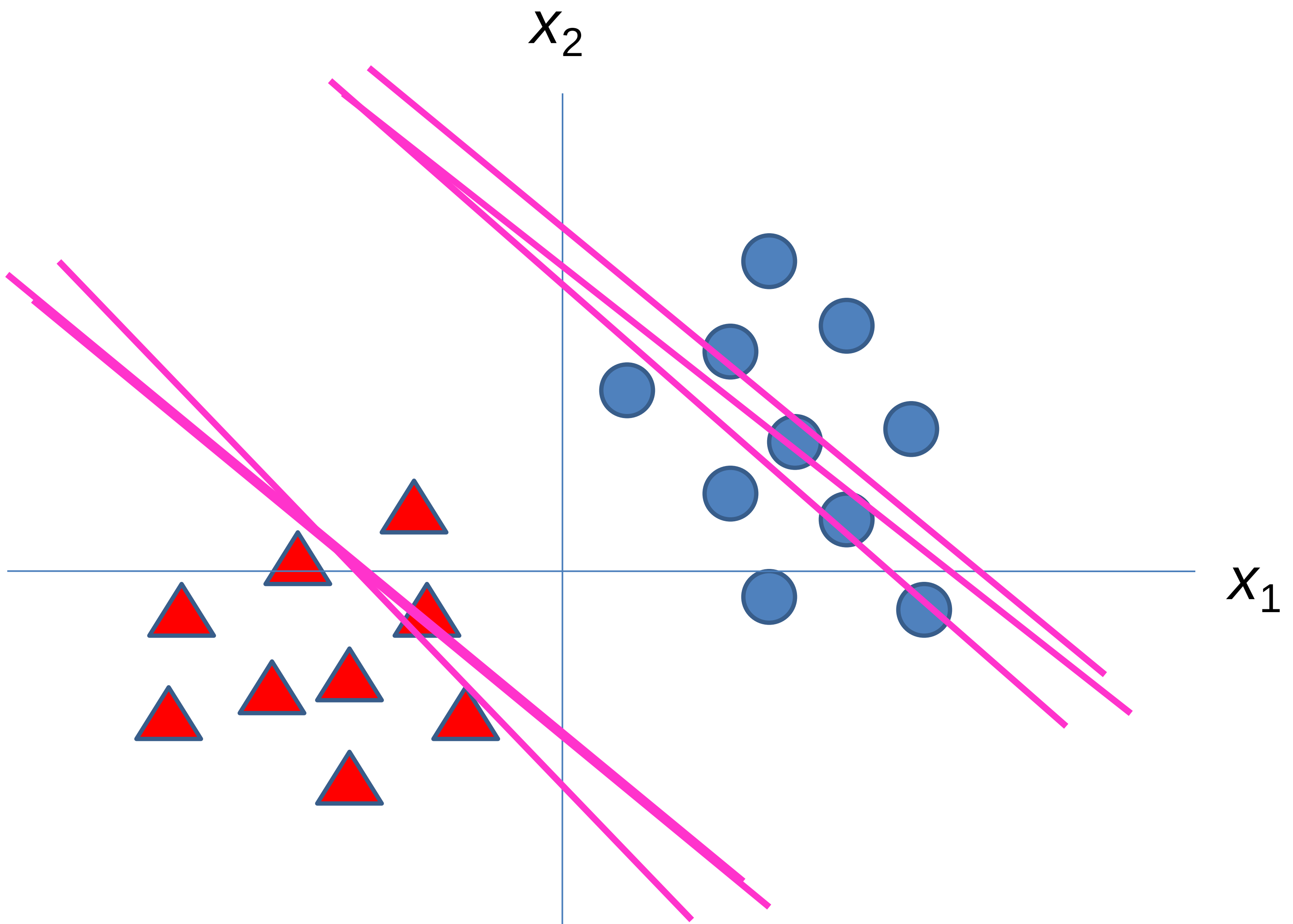
Security Systems



Learning Rate: **Besar**







$$dW1 = dW1 + (lr * D1 * P)$$

Algoritma Belajar **Propagasi Balik**

- **Pelatihan Jaringan**
  - Perhitungan Mundur

$$D2 = (1 - A2^2) * E$$

$$D1 = (1 - A1^2) * (W2 * D2)$$

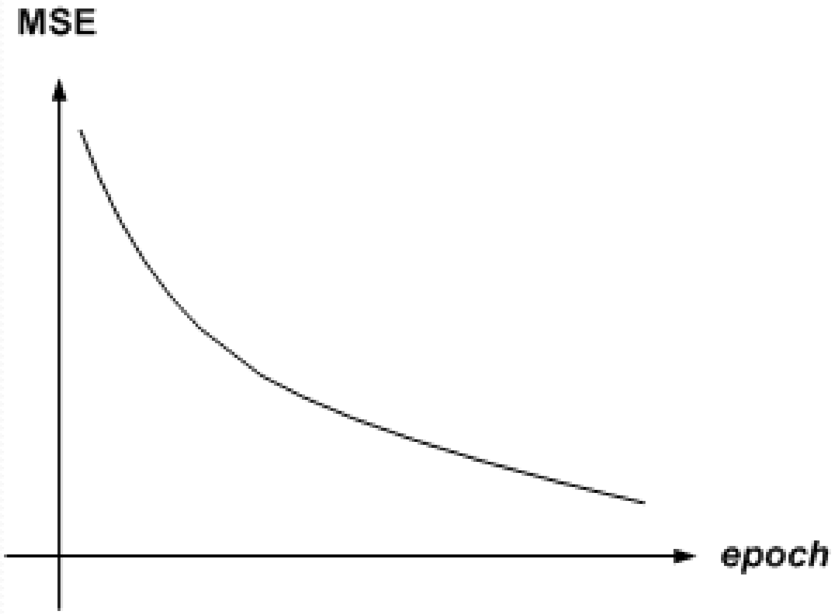
$$dW1 = dW1 + (lr * D1 * P)$$

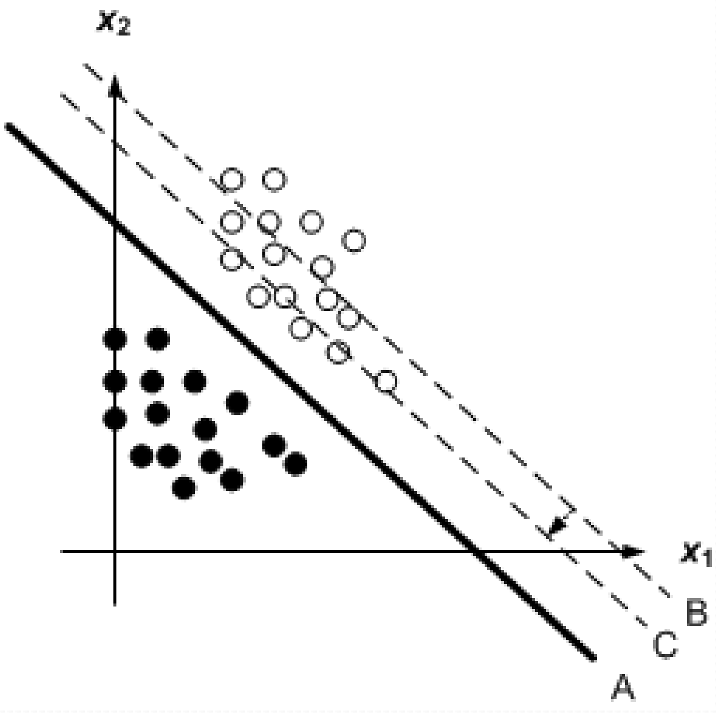
$$dB1 = dB1 + (lr * D1)$$

$$dW2 = dW2 + (lr * D2 * P)$$

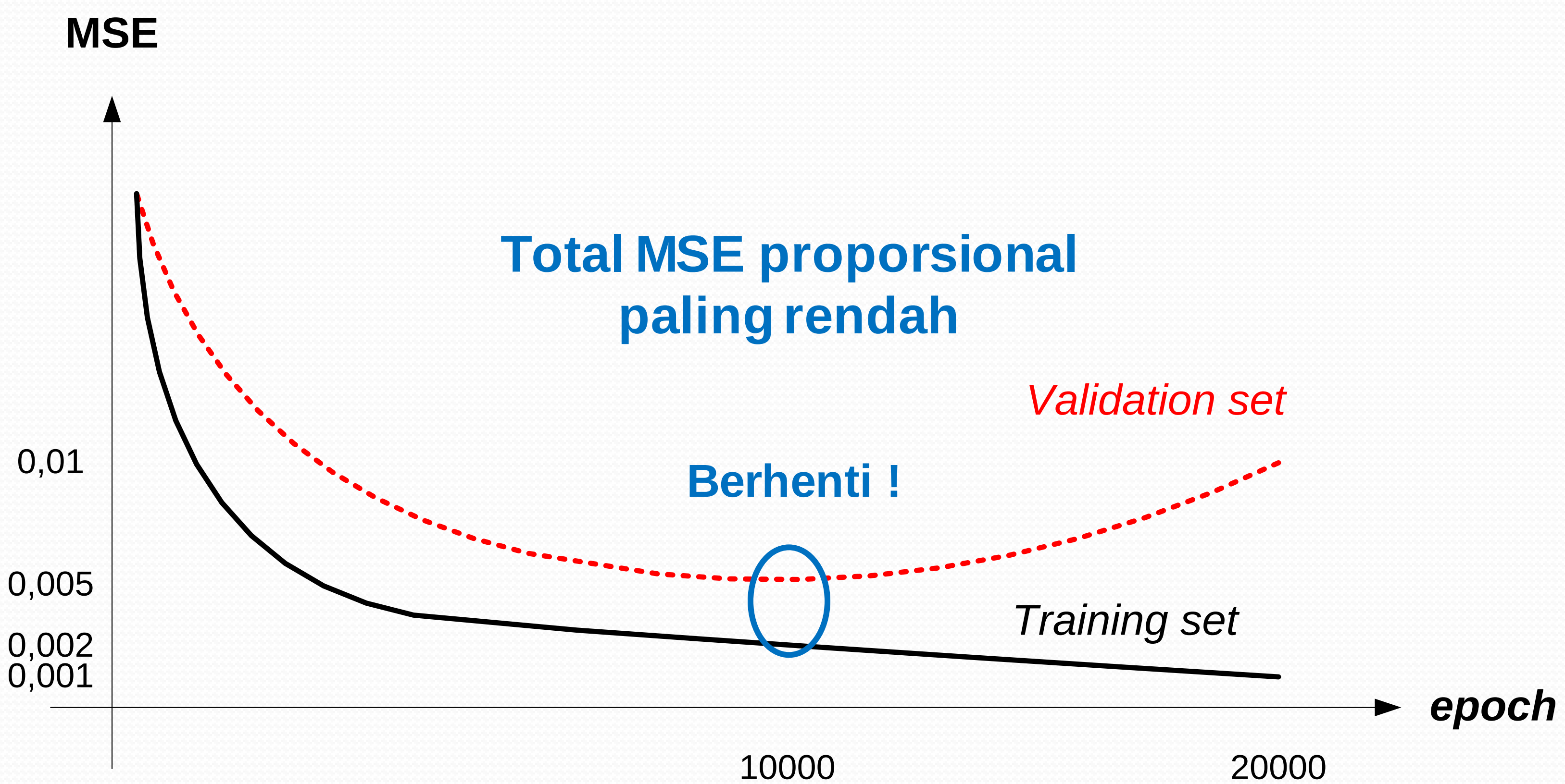
$$dB2 = dB2 + (lr * D2)$$

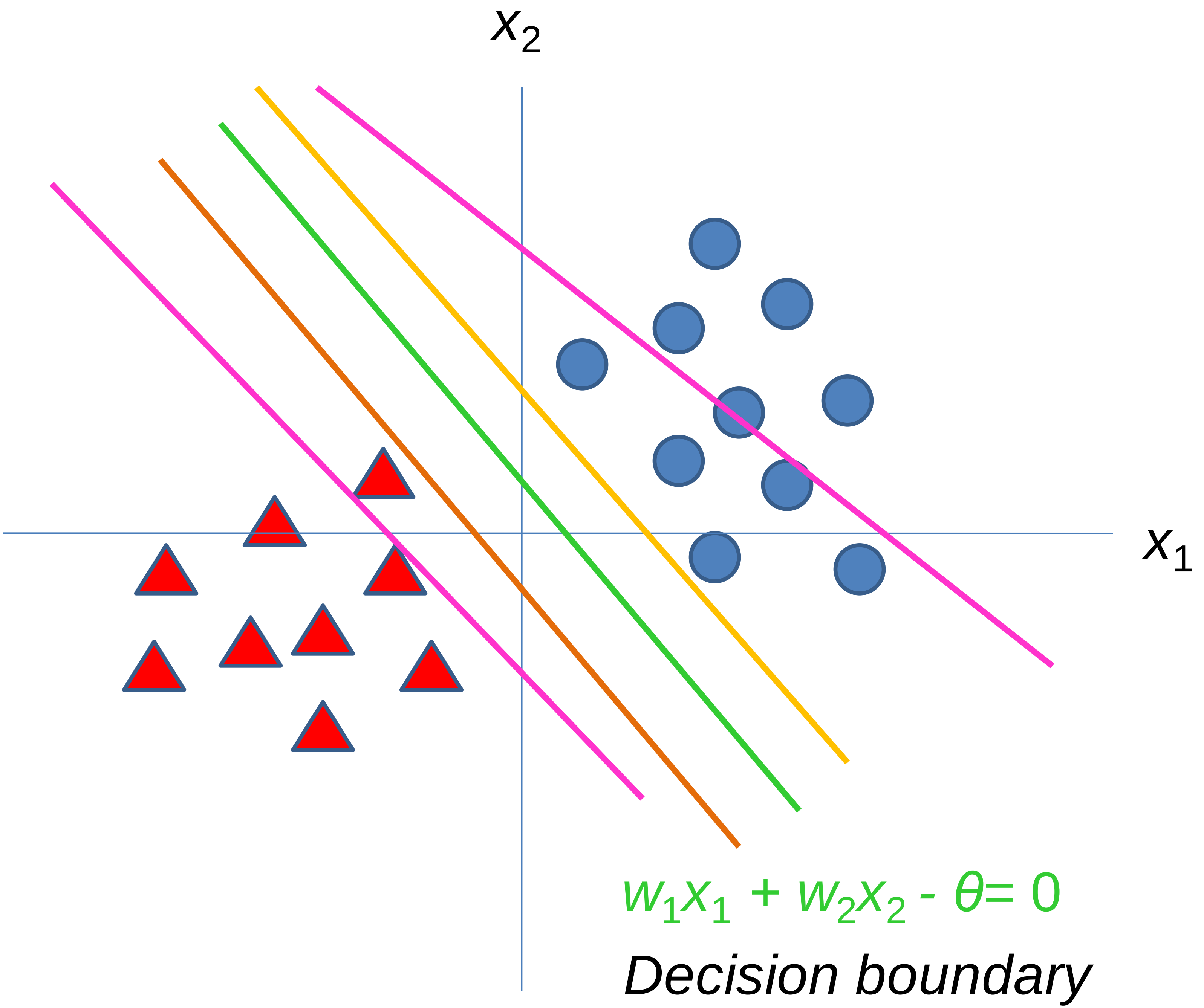
Learning Rate: **Kecil**





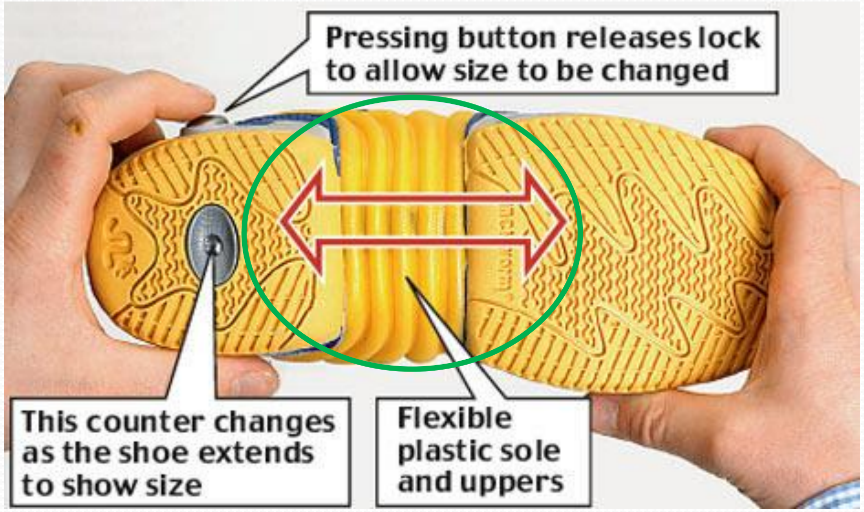
Kapan Menghentikan *Learning*?





x_2
x_1
$w_1x_1 + w_2x_2 - \theta = 0$
Decision boundary

Overfit, Oversize, Flexible



Strategi penggunaan ANN

- **Cara memandang masalah**:
  - Klasifikasi
  - Sekuriti
  - Prediksi
  - Optimasi
- **Teknik learning**: Supervised/Unsupervised
- **Desain Arsitektur**
  - Jumlah layer
  - Jumlah neuron
  - Pemetaan output
- **Strategi *learning***
  - Penyiapan data: filterisasi data, pembagian data (training, validasi, test)
  - Parameter: inisialisasi (acak atau memakai algoritma), laju belajar, dsb.
  - Penghentian learning

Studi Kasus

- Klasifikasi
- Verifikasi
- Sekuriti
- Prediksi
- Deteksi
- Optimasi

Kasus 1 Verifikasi tandatangan

- OFFLINE atau ONLINE?
- Citra: 100 x 100 pixel grayscale
- Satu juta tandatangan?

- Input & Output?
- P dan T?
- Struktur dan parameter MLP?
- Perlu *preprocessing*?

Kasus 2: Sistem keamanan

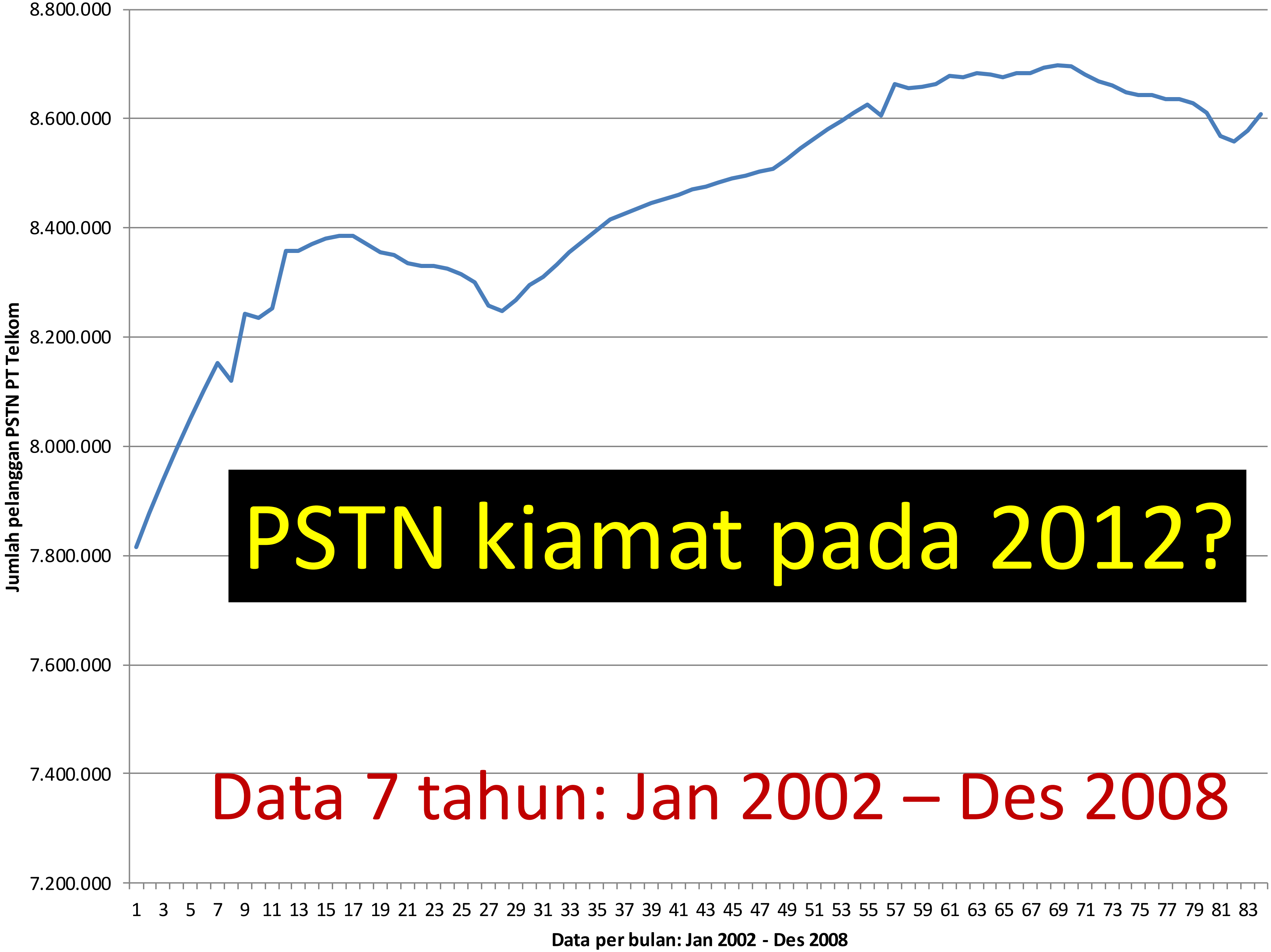
- Satu ruangan hanya 10 orang yang boleh masuk
- Setiap orang yang akan masuk ruangan harus menempelkan ibu jari untuk diverifikasi sidik jarinya
- Citra: 300 x 300 pixels

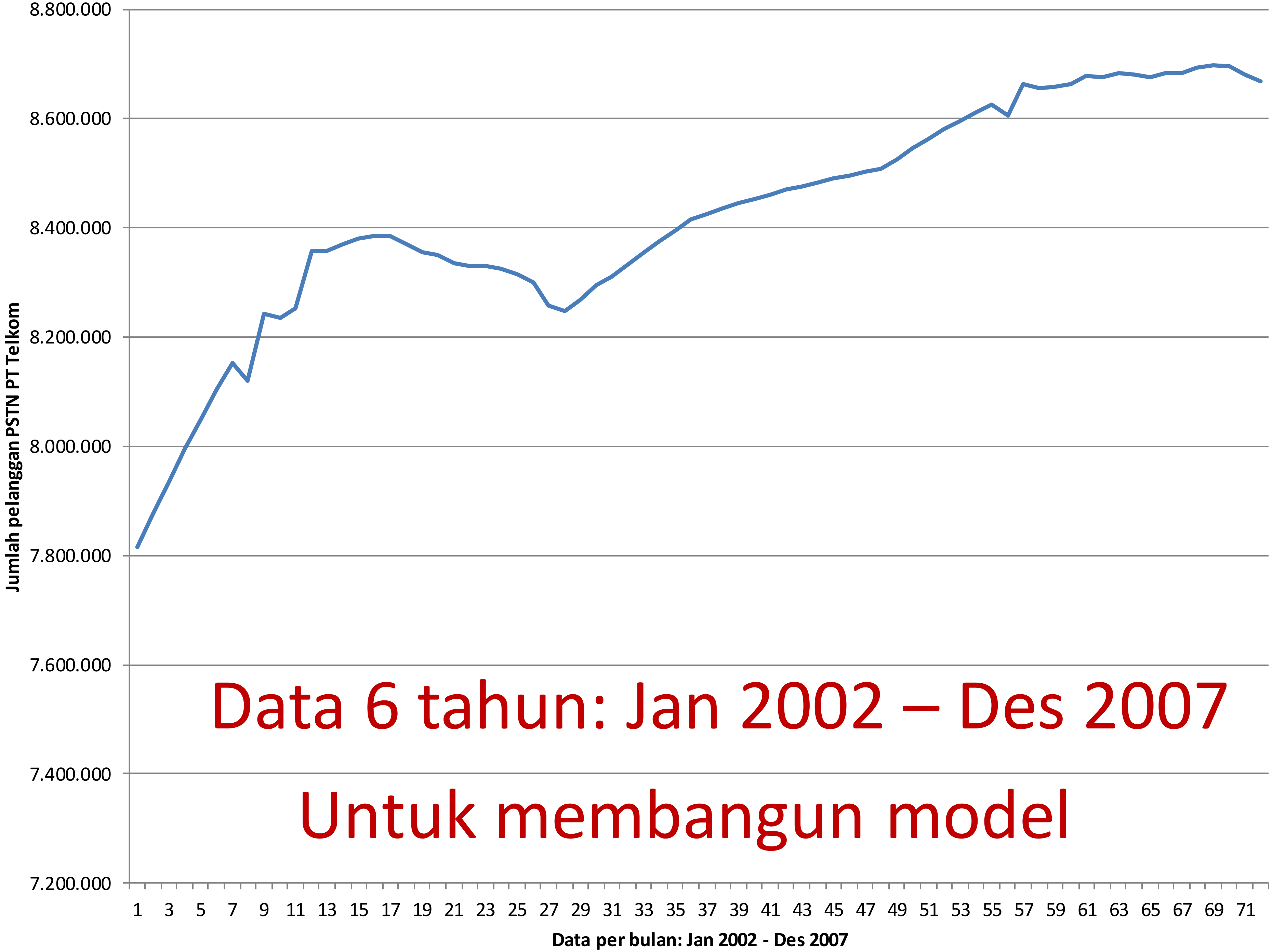
- Input & Output?
- P dan T?
- Struktur dan parameter MLP?
- Perlu *preprocessing*?

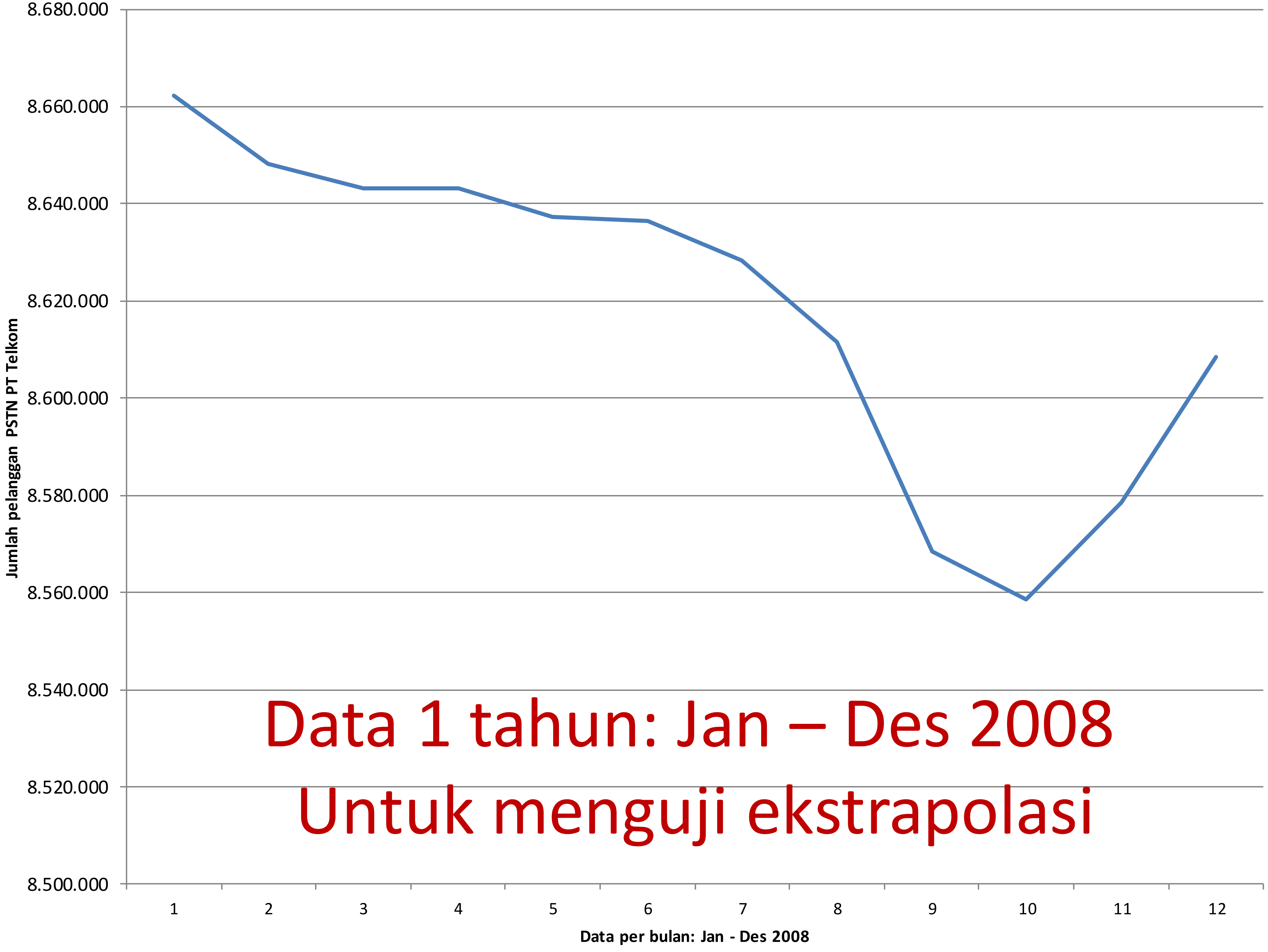
Kasus 3: Prediksi pelanggan PSTN

- Data riil dari PT Telkom
- Jumlah pelanggan bulanan selama 7 tahun
- Error harus < 1 %

- Input & Output?
- P dan T?
- Struktur dan parameter MLP?
- Perlu *preprocessing*?


PSTN kiamat pada 2012?
Data 7 tahun: Jan 2002 – Des 2008
Jumlah pelanggan PSTN PT Telkom
Data per bulan: Jan 2002 - Des 2008
8.800.000
8.600.000
8.400.000
8.200.000
8.000.000
7.800.000
7.600.000
7.400.000
7.200.000
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83


8.800.000
8.600.000
8.400.000
8.200.000
8.000.000
7.800.000
7.600.000
7.400.000
7.200.000
Jumlah pelanggan PSTN PT Telkom
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71
Data per bulan: Jan 2002 - Des 2007
Data 6 tahun: Jan 2002 – Des 2007
Untuk membangun model


8.680.000
8.660.000
8.640.000
8.620.000
8.600.000
8.580.000
8.560.000
8.540.000
8.520.000
8.500.000
Jumlah pelanggan PSTN PT Telkom
1
2
3
4
5
6
7
8
9
10
11
12
Data per bulan: Jan - Des 2008
Data 1 tahun: Jan – Des 2008
Untuk menguji ekstrapolasi

Pembangunan Model

- **Skenario 1**
  - Data *training* : 24 bulan (2002 - 2003)
  - Data *validation* : 24 bulan (2004 - 2005)
  - Data *testing* : 24 bulan (2006 - 2007)
- **Skenario 2**
  - Data *training* : 12 bulan (2002)
  - Data *validation* : 12 bulan (2003)
  - Data *testing* : 48 bulan (2004 - 2007)
- **Skenario 3**
  - Data *training* : 48 bulan (2002 - 2005)
  - Data *validation* : 12 bulan (2006)
  - Data *testing* : 12 bulan (2007)

Formulasi Masalah

- Input & Output?
- P dan T?
- Struktur dan parameter MLP?
- Perlu *preprocessing*?

Normalisasi

$$Xn_i = \left(\frac{(X_i - \min(X))}{(\max(X) - \min(X))} \; x \; 0{,}8 \right) + 0{,}1$$

- Xn_i = data aktual normalisasi ke-i
- X_i = data aktual dengan range data asli ke-i
- X = data aktual dengan range data asli

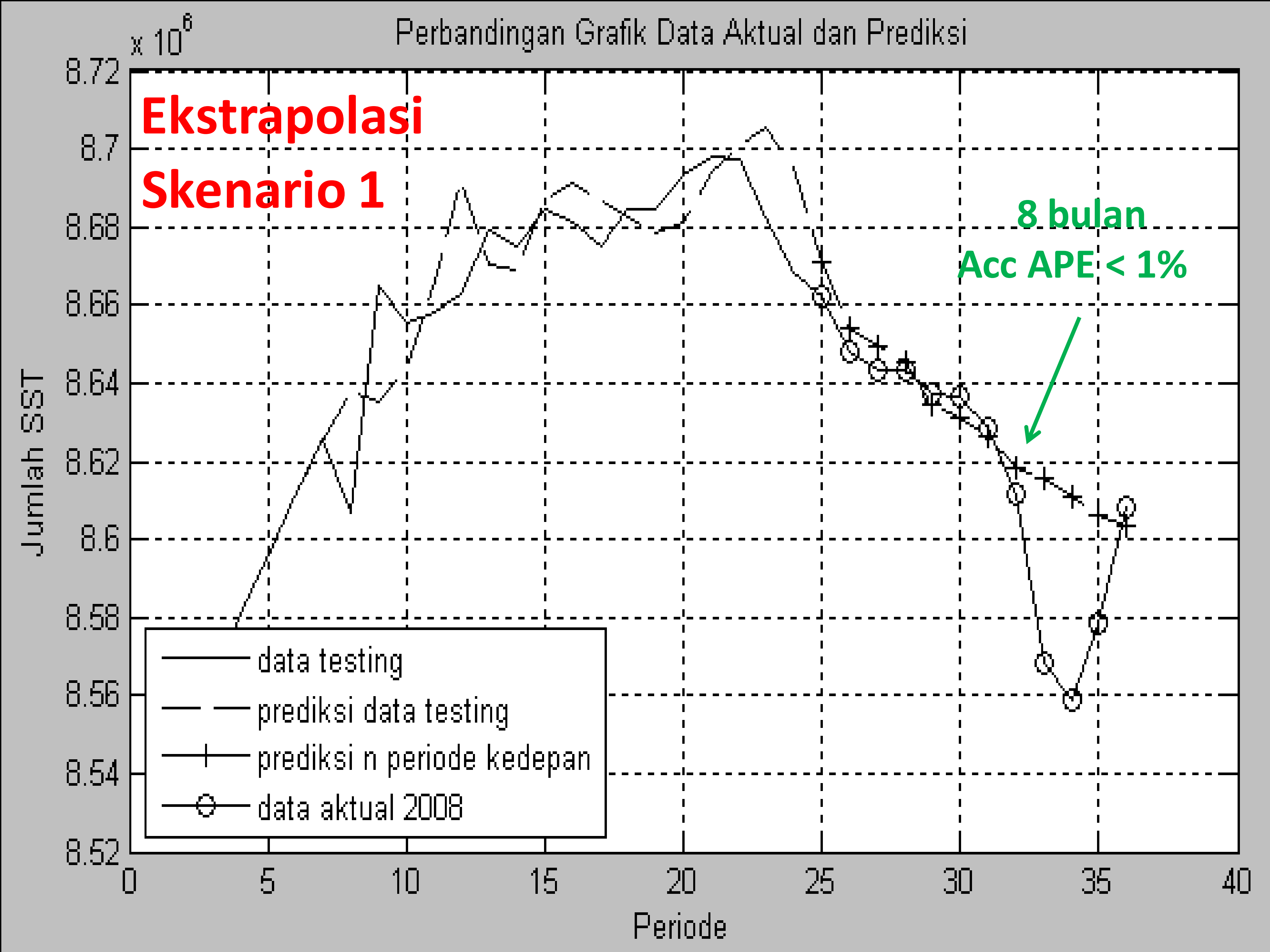
Denormalisasi

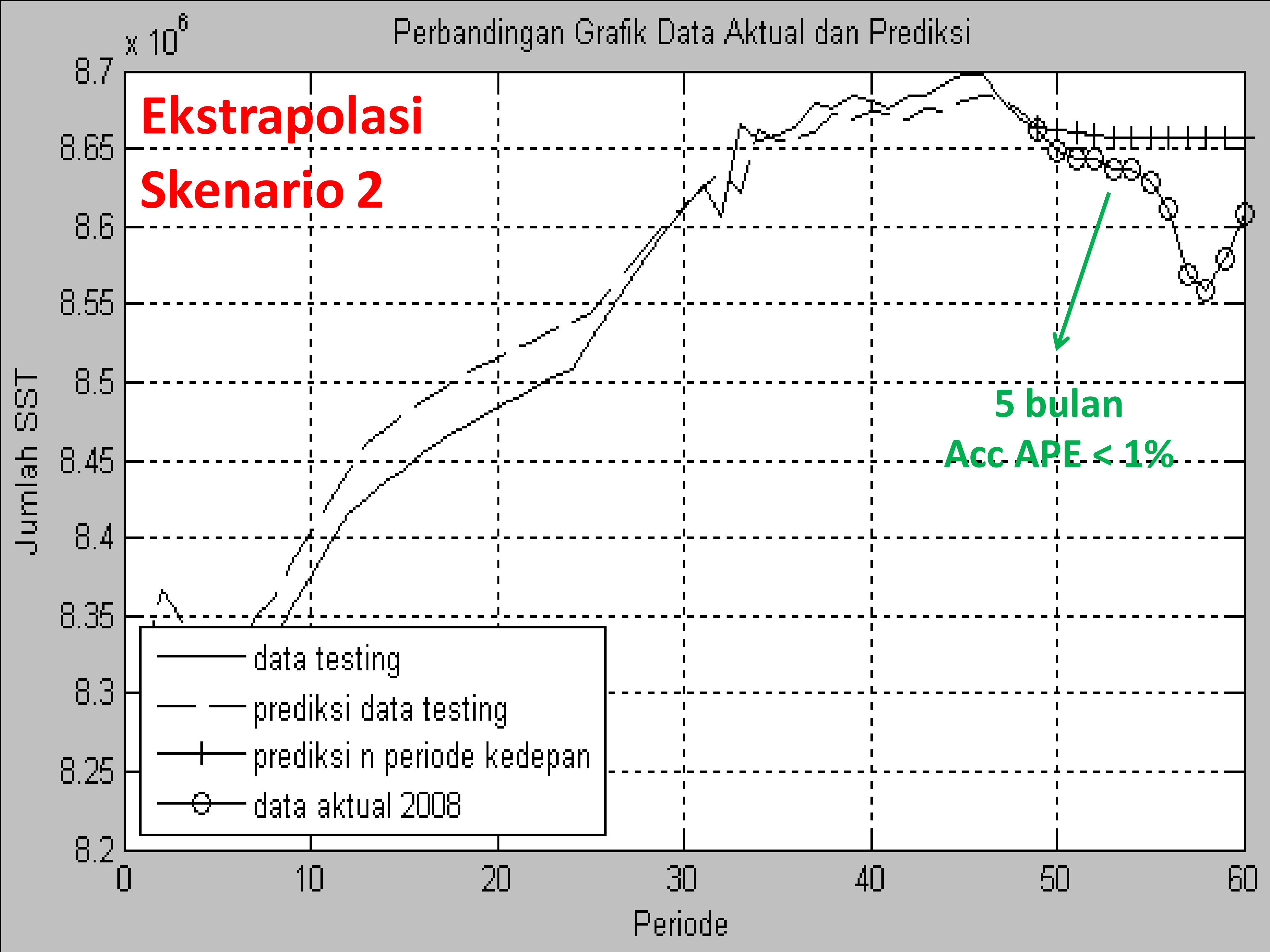
$$F_i = \left[\frac{((F'_i) - 0{,}1)}{0{,}8}\right] \times (\max(X) - \min(X)) + \min(X)$$

- F_i = nilai prediksi dengan range nilai asli
- F'_i = nilai prediksi dari hasil data yang dinormalisasi
- X = data aktual

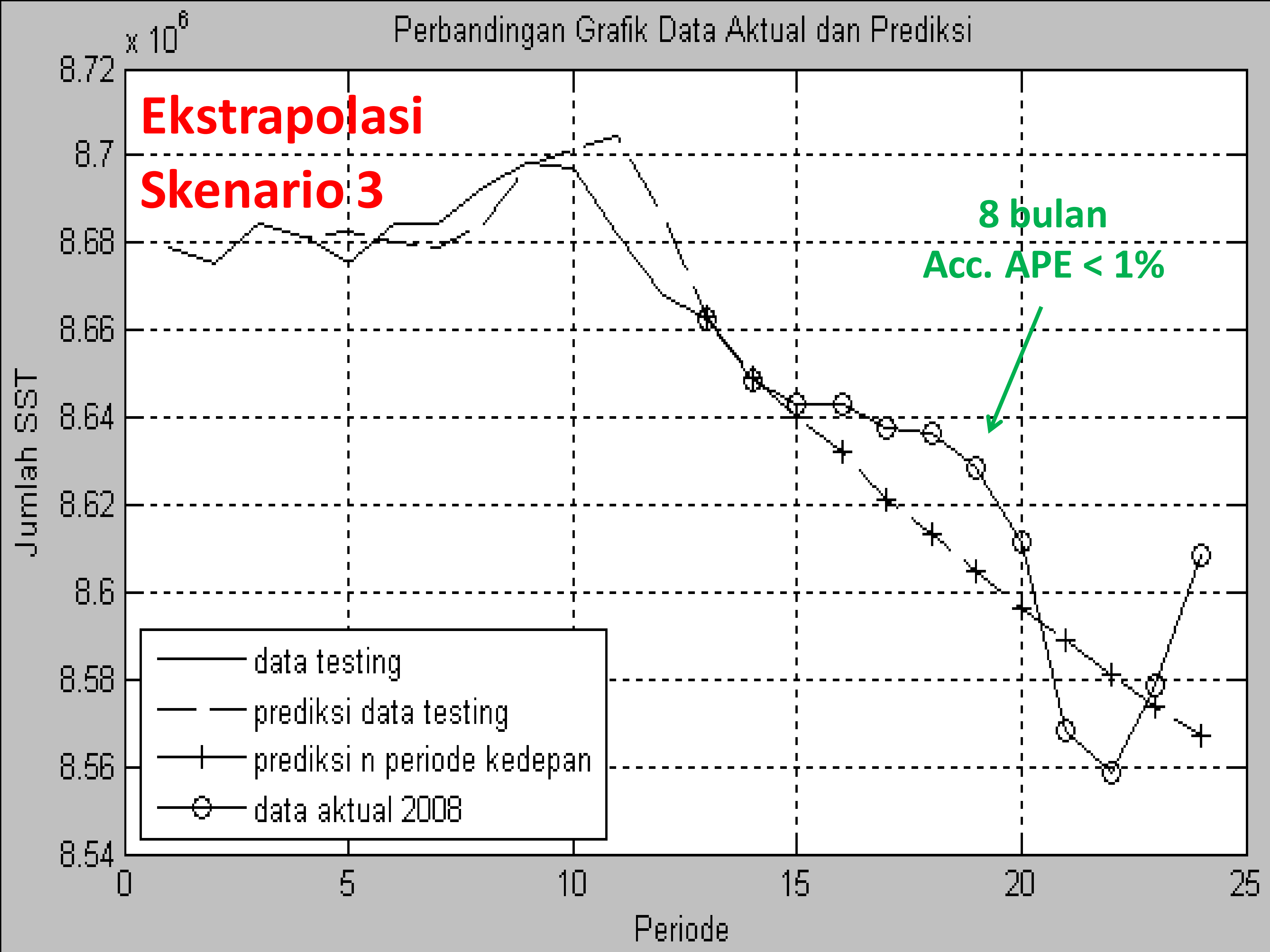
Pembangunan Model

- **Skenario 1**
  - Data *training* : 24 bulan (2002 - 2003)
  - Data *validation* : 24 bulan (2004 - 2005)
  - Data *testing* : 24 bulan (2006 - 2007)
- **Skenario 2**
  - Data *training* : 12 bulan (2002)
  - Data *validation* : 12 bulan (2003)
  - Data *testing* : 48 bulan (2004 - 2007)
- **Skenario 3**
  - Data *training* : 48 bulan (2002 - 2005)
  - Data *validation* : 12 bulan (2006)
  - Data *testing* : 12 bulan (2007)


Perbandingan Grafik Data Aktual dan Prediksi
x 10^6
Ekstrapolasi
Skenario 1
8 bulan
Acc APE < 1%
Jumlah SST
Periode
8.72
8.7
8.68
8.66
8.64
8.62
8.6
8.58
8.56
8.54
8.52
0
5
10
15
20
25
30
35
40
data testing
prediksi data testing
prediksi n periode kedepan
data aktual 2008



Perbandingan Grafik Data Aktual dan Prediksi
x 10^6
Ekstrapolasi
Skenario 2
5 bulan
Acc APE < 1%
Jumlah SST
Periode
8.7
8.65
8.6
8.55
8.5
8.45
8.4
8.35
8.3
8.25
8.2
0
10
20
30
40
50
60
data testing
prediksi data testing
prediksi n periode kedepan
data aktual 2008



Perbandingan Grafik Data Aktual dan Prediksi
x 10^6
Ekstrapolasi
Skenario 3
8 bulan
Acc. APE < 1%
Jumlah SST
Periode
8.72
8.7
8.68
8.66
8.64
8.62
8.6
8.58
8.56
8.54
0
5
10
15
20
25
data testing
prediksi data testing
prediksi n periode kedepan
data aktual 2008

Kasus 4: Deteksi Kecurangan

- Jumlah pelanggan: 10 juta
- Data yg tersedia: tagihan bulanan selama 5 tahun
- Kecurangan:
  - Jika tagihan pada bulan ini jauh lebih sedikit atau lebih besar dibandingkan bulan-bulan sebelumnya
  - Jika tunggakan lebih dari 3 bulan dengan total tagihan jauh lebih besar dibandingkan bulan-bulan sebelumnya

Formulasi Masalah

- Input & Output?
- P dan T?
- Struktur dan parameter MLP?
- Perlu *preprocessing*?

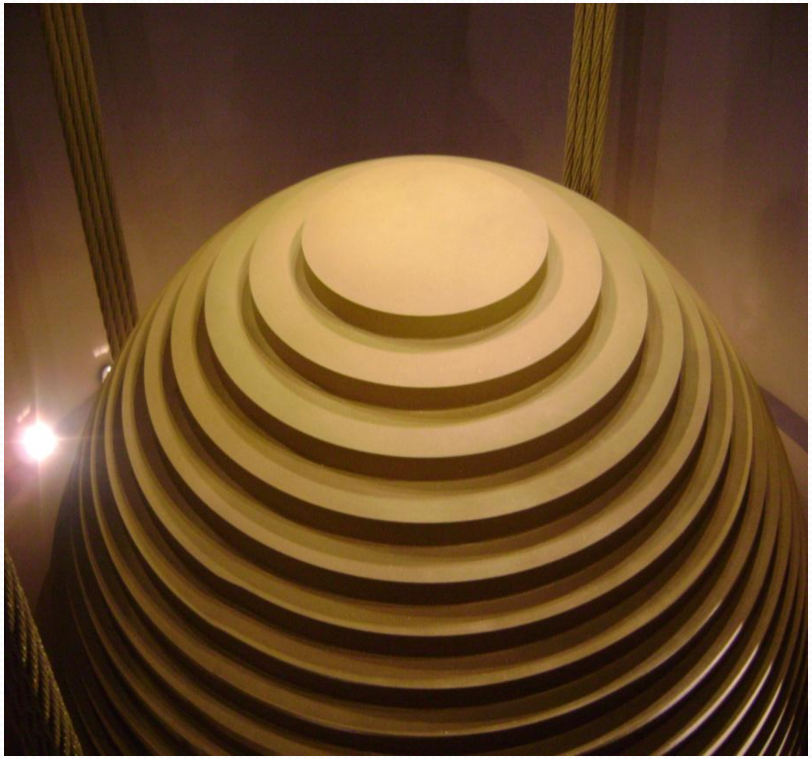
Kasus 5: Deteksi Churn

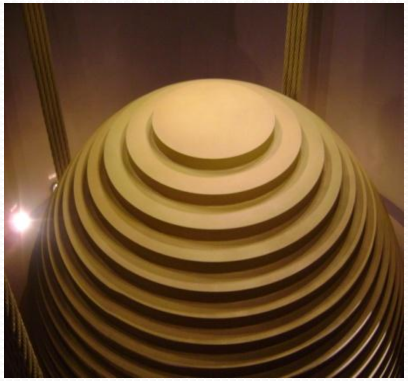
Kota	Tipe	Tipe Pembayaran	Tagihan Bulanan	Jumlah Panggilan	Panggilan TidakNormal	Churn
Jakarta	Pemerintah	Cash	Besar	10	Sedikit	Tidak
Jakarta	Corporate	Kartu Kredit	Sedang	8	Sedang	Tidak
Jakarta	Corporate	Kartu Kredit	Kecil	5	Banyak	Tidak
Surabaya	Corporate	Cash	Kecil	3	Banyak	Ya
Surabaya	Corporate	Cash	Kecil	2	Banyak	Ya
Surabaya	Corporate	Kartu Kredit	Besar	1	Sedang	Tidak
Jakarta	Corporate	Kartu Kredit	Sedang	9	Sedang	Tidak
Jakarta	Corporate	Kartu Kredit	Sedang	7	Sedang	Tidak
Jakarta	Corporate	Kartu Kredit	Sedang	6	Sedang	Tidak
Jakarta	Corporate	Kartu Kredit	Sedang	4	Sedang	Tidak
Jakarta	Corporate	Kartu Kredit	Sedang	9	Sedang	Tidak

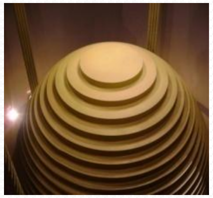
Formulasi Masalah

- Input & Output?
- P dan T?
- Struktur dan parameter MLP?
- Perlu *preprocessing*?

Kasus 6: Kompresi Citra



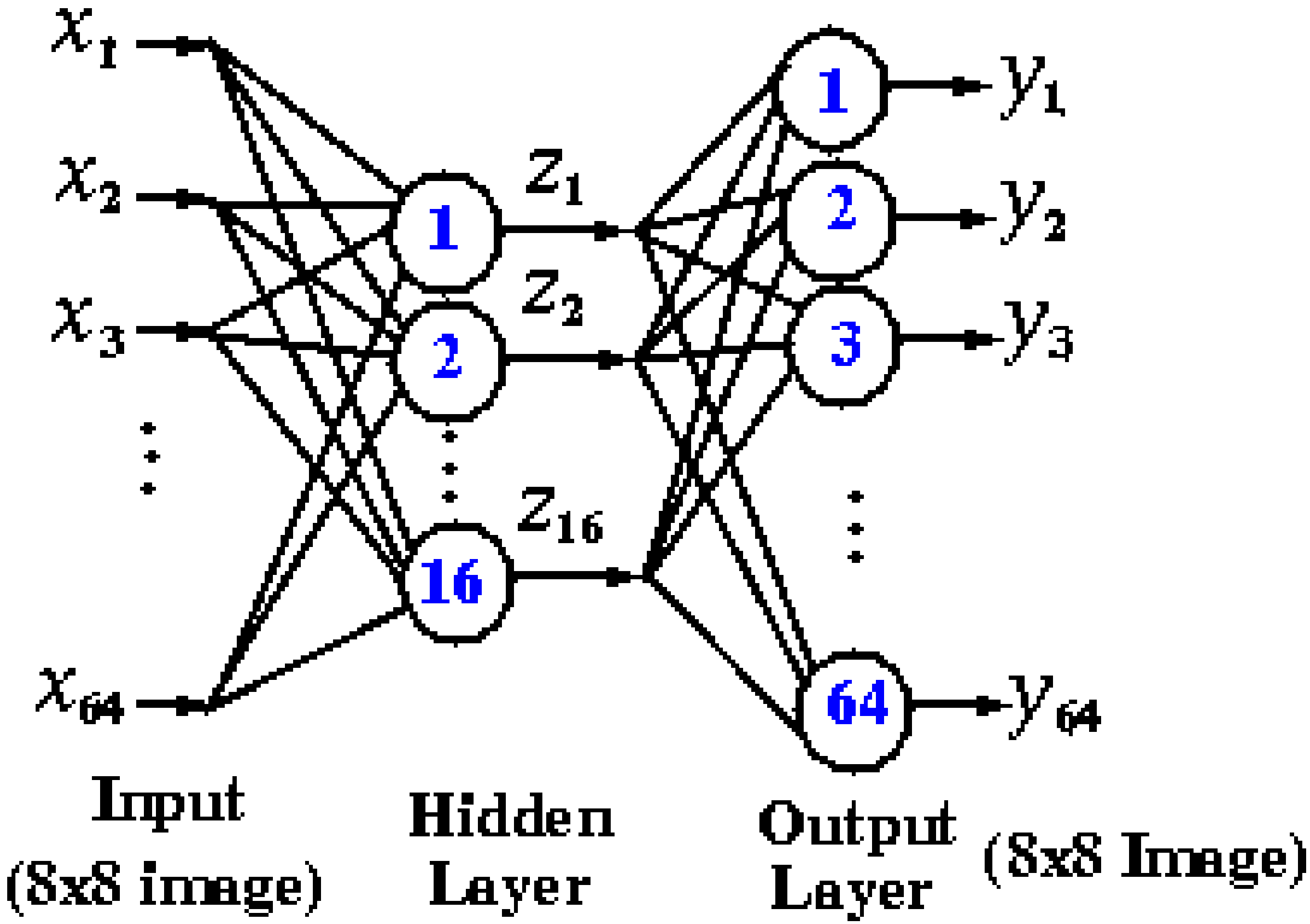




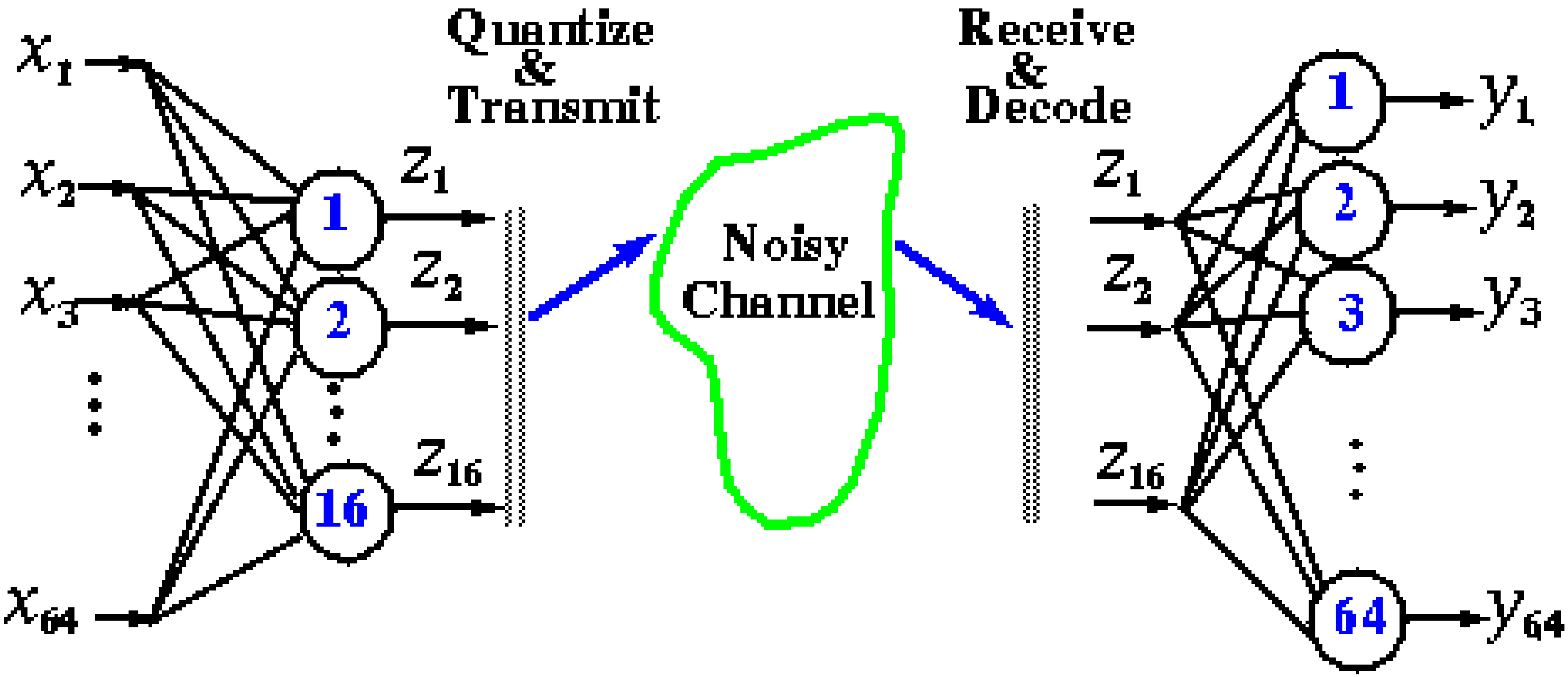
1024x1024 pixel, size: **3 MB**

Formulasi Masalah

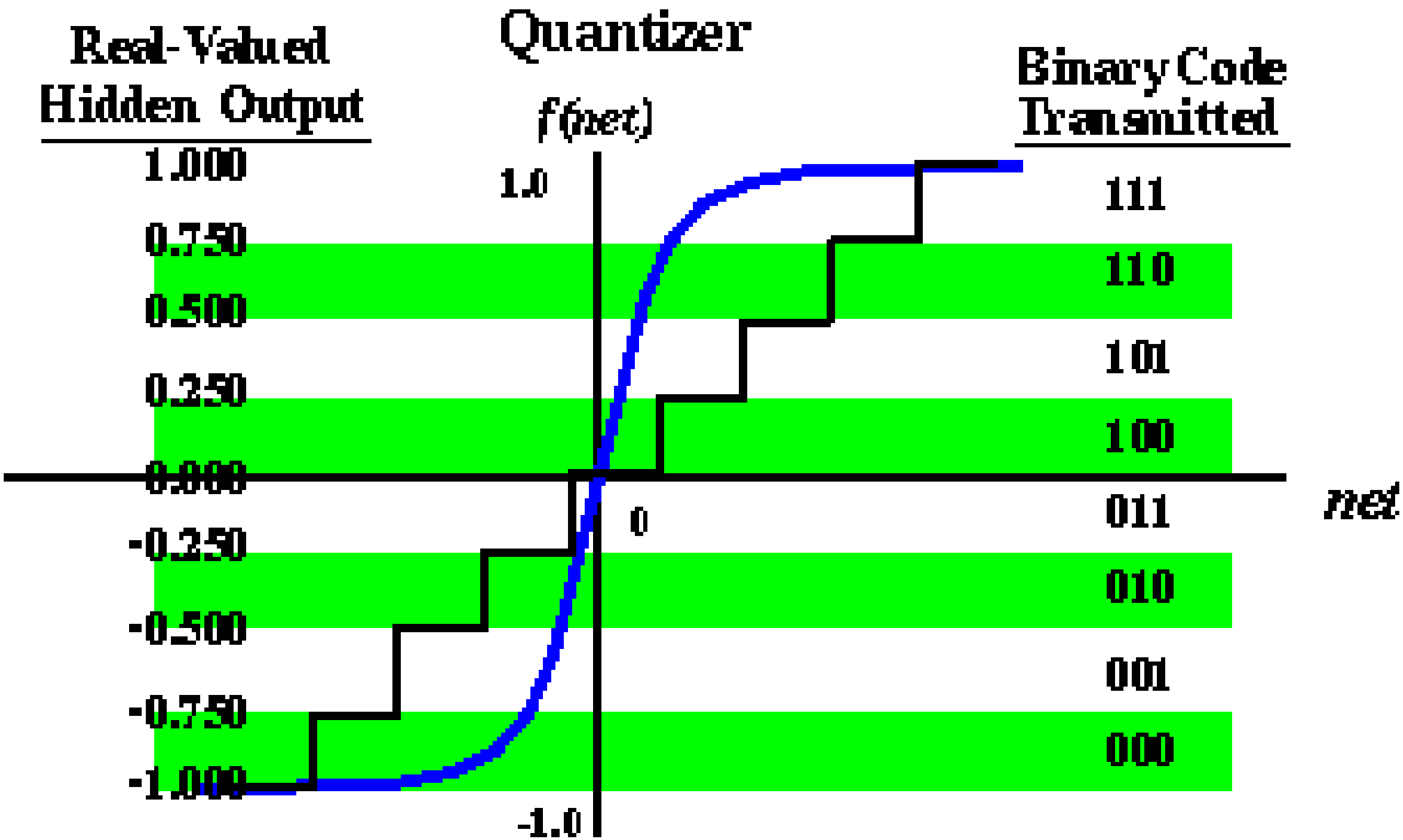
- Input & Output?
- P dan T?
- Struktur dan parameter MLP?
- Perlu *preprocessing*?



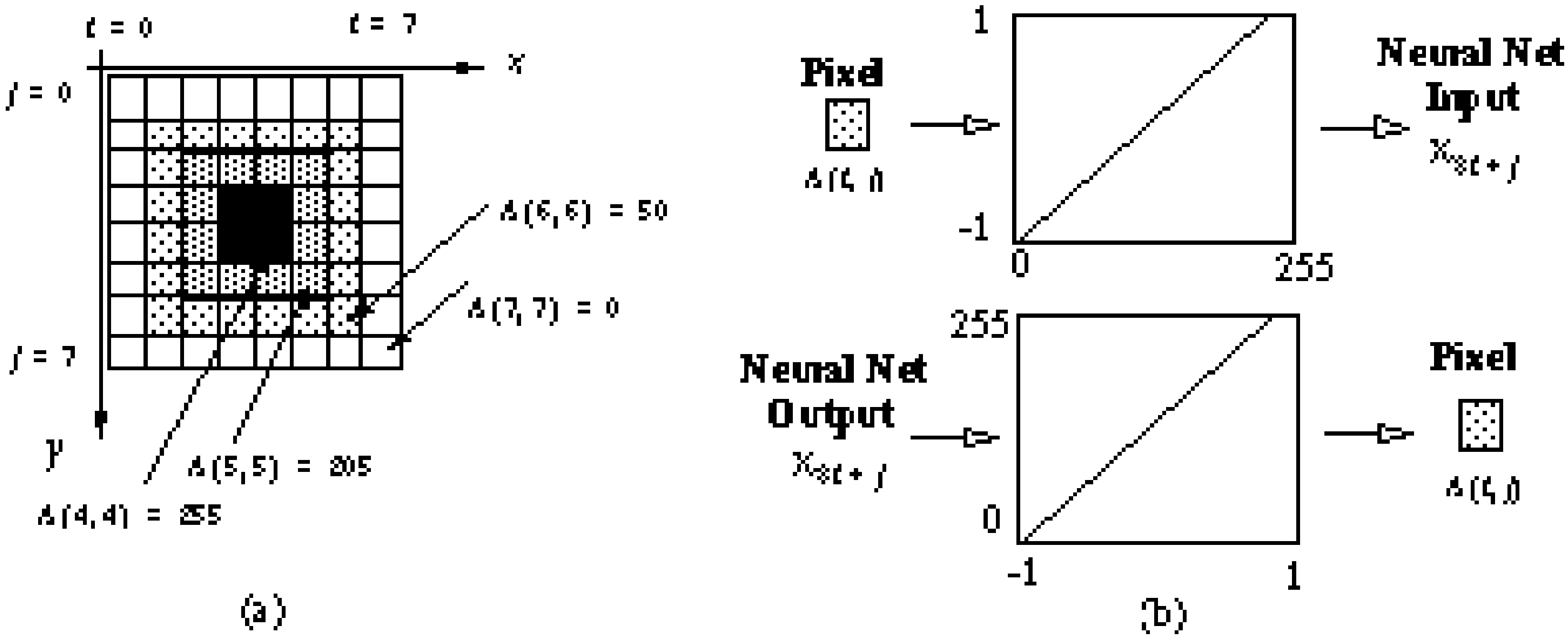
[Paul Watta, et. al. 1996]



[Paul Watta, et. al. 1996]



[Paul Watta, et. al. 1996]



[Paul Watta, et. al. 1996]

MLP

- Fungsi Aktivasi: umumnya (0, 1) atau (-1, 1)
  - Preprocessing
  - Normalisasi Data Input
  - Denormalisasi Data Output
- Output dari suatu neuron bisa biner (0 atau 1) maupun real dalam interval (0, 1) atau (-1, 1)
  - Klasifikasi → setiap kelas dikodekan dalam biner
  - Prediksi → output didenormalisasi
  - Optimasi → preprocessing yang agak rumit

Supervised vs Unsupervised

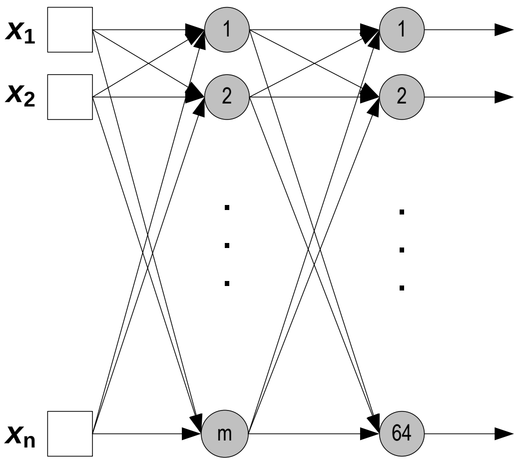
Supervised	Unsupervised
Klasifikasi	Clustering
Kelas harus diketahui	Kelas tidak harus diketahui
Waktu training lambat	Waktu training cepat

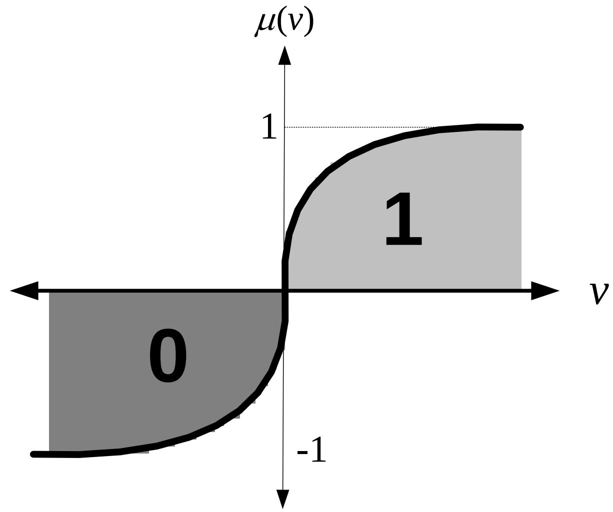
ANN

- Klasifikasi atau Clustering
- Mudah implementasi
- Akurasi tinggi
- Tahan *noise*
- Implementasi hardware (CHIP)
- Harus tersedia data latih dengan kelas yang jelas
- Waktu training lama
- Training ulang
- Penalarannya tidak bisa dijelaskan (*Weights*)

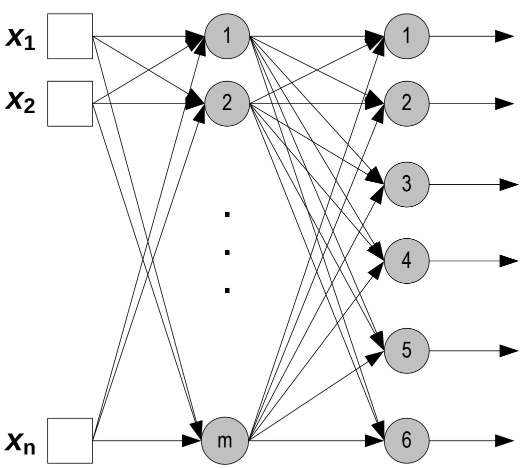
Jumlah neuron pada output layer

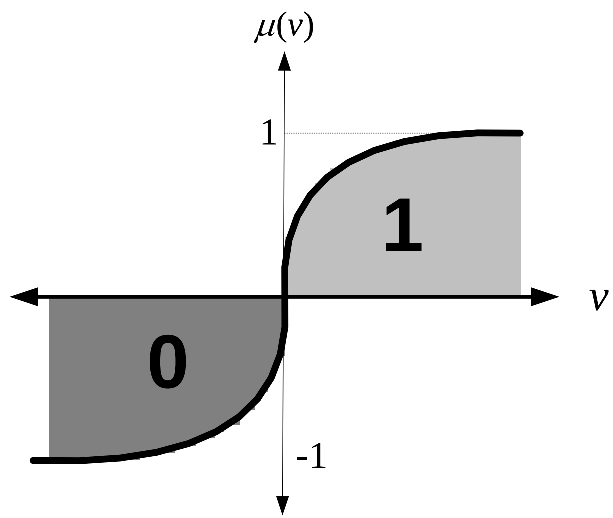
- **Deterministik**: mudah dihitung berdasarkan permasalahan yang dihadapi.
- Untuk pengenalan karakter dengan **64 kelas**: ('a', 'b', ..., 'z', 'A', 'B', ..., 'Z', '0', '1', ... '9', '-', '+'), perlu berapa output neuron?



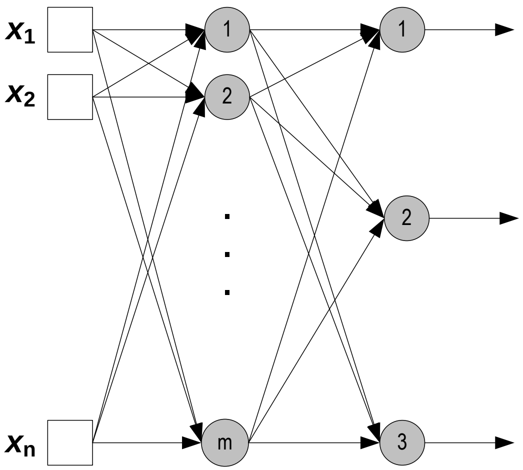


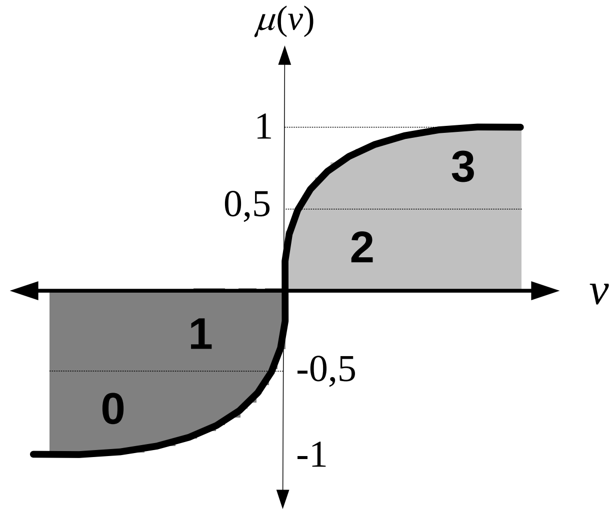
Output dari *neuron* ke-							Kelas
1	2	3	4	5	...	64	
1	0	0	0	0	...	0	a
0	1	0	0	0	...	0	b
0	0	1	0	0	...	0	c
0	0	0	1	0	...	0	d
0	0	0	0	1	...	0	e
...	...	...	...	...	...	...	...
0	0	0	0	0	...	1	+



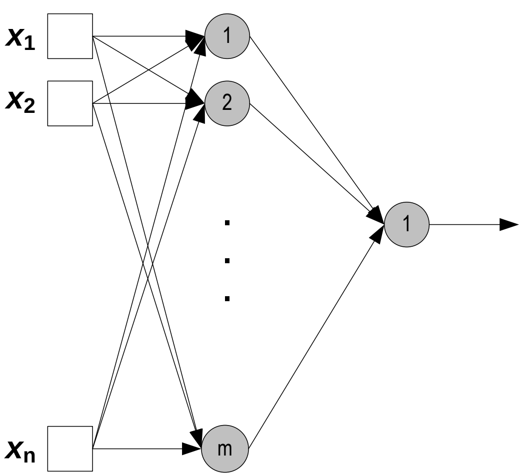


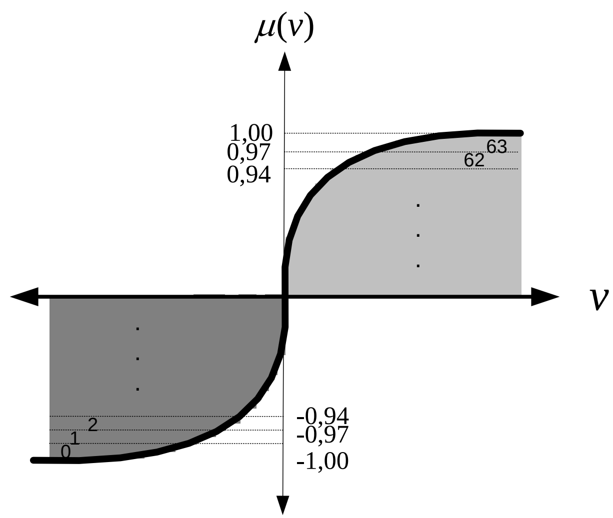
Output neuron ke-						Kelas
1	2	3	4	5	6	
0	0	0	0	0	0	a
0	0	0	0	0	1	b
0	0	0	0	1	0	c
0	0	0	0	1	1	d
0	0	0	1	0	0	e
...	...	...	...	...	...	...
1	1	1	1	1	1	+



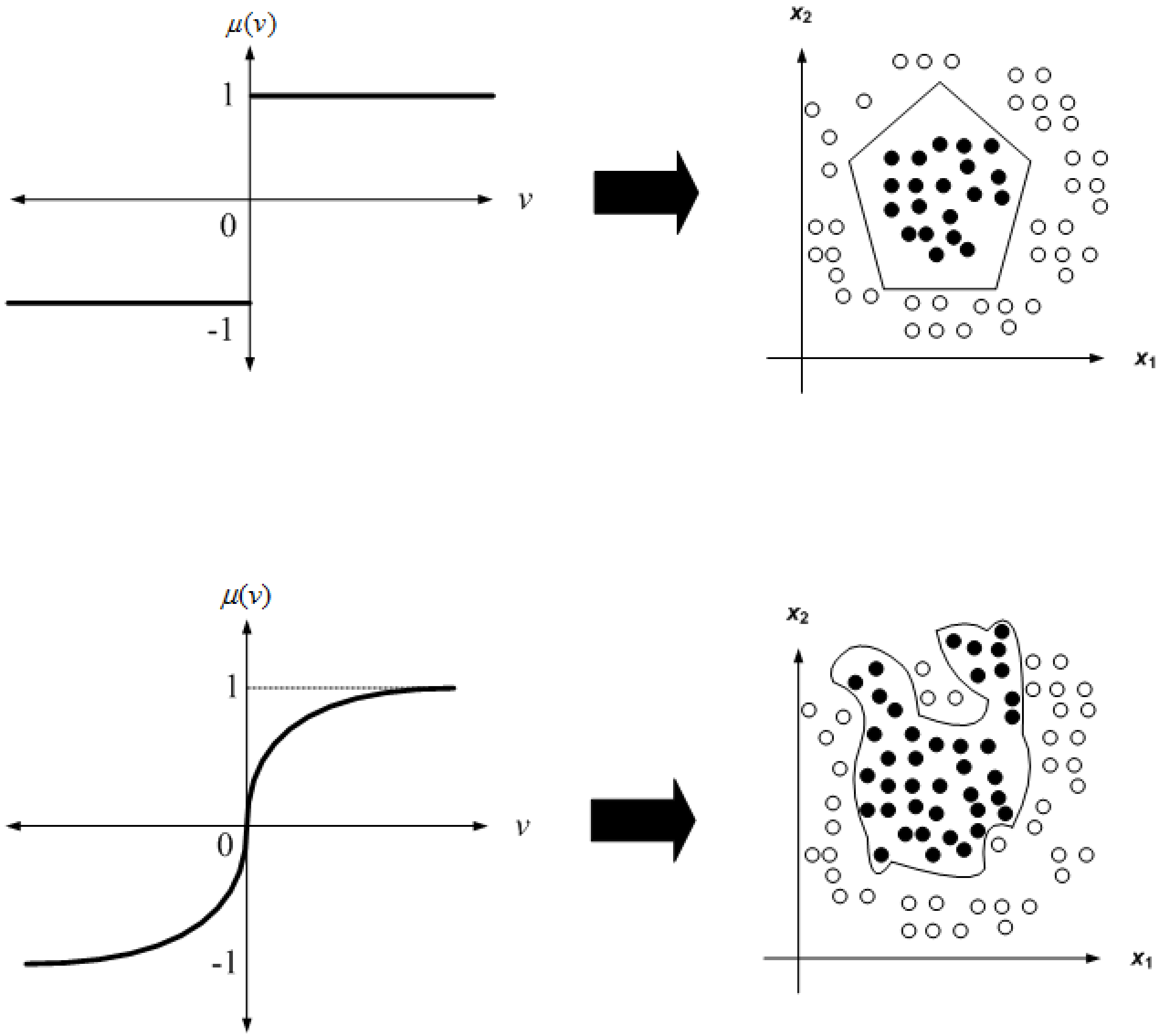


Output neuron ke-			Kelas
1	2	3	
0	0	0	a
0	0	1	b
0	0	2	c
0	0	3	d
0	1	0	e
...	...	...	...
3	3	3	+

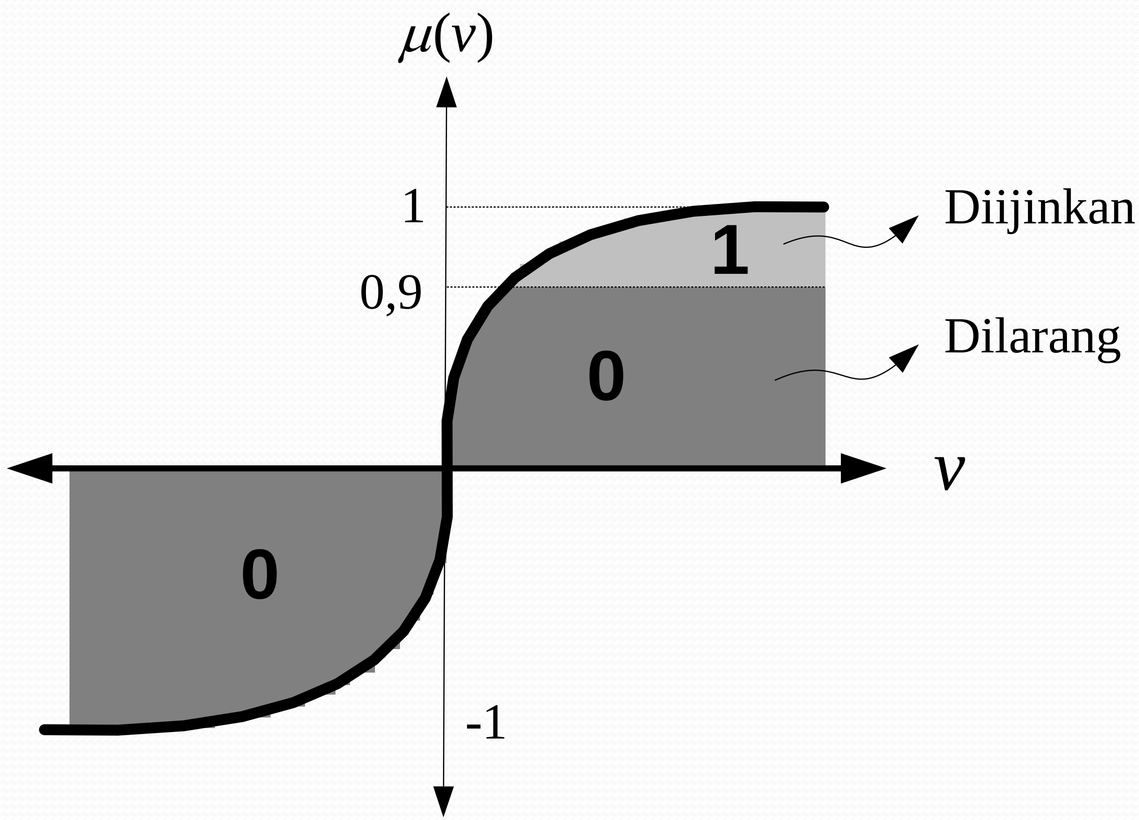




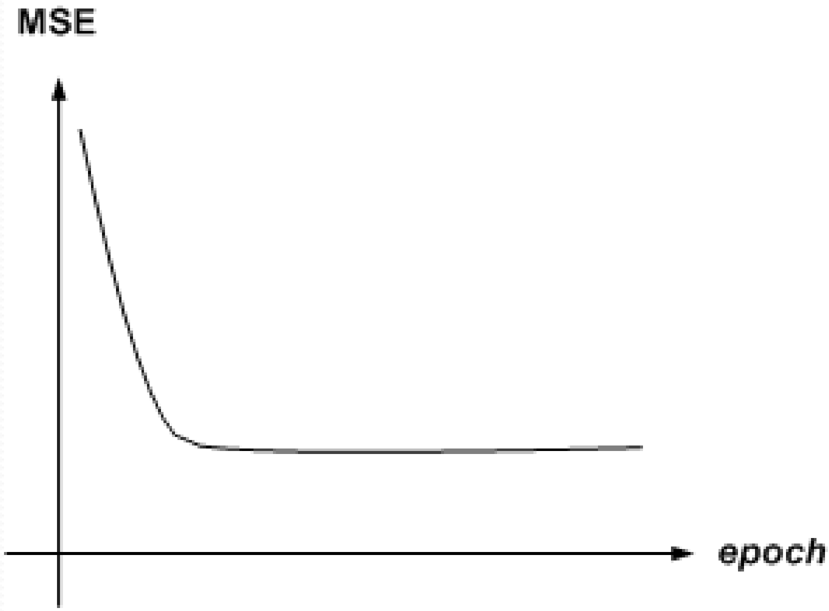
Output neuron ke-	Kelas
1	
0	a
1	b
2	c
3	d
4	e
...	...
63	+

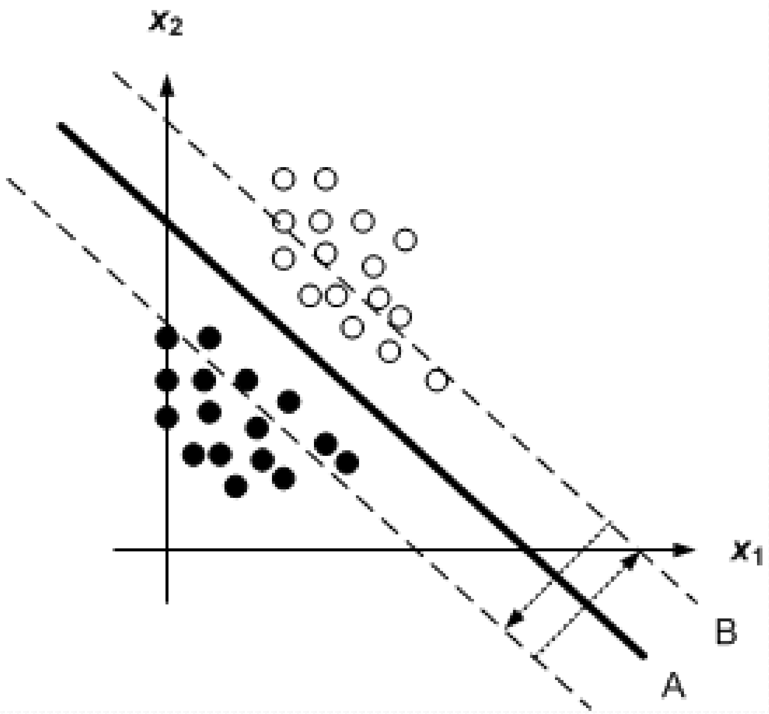

$\mu(v)$
1
0
-1
v
x_2
x_1
$\mu(v)$
1
0
-1
v
x_2
x_1

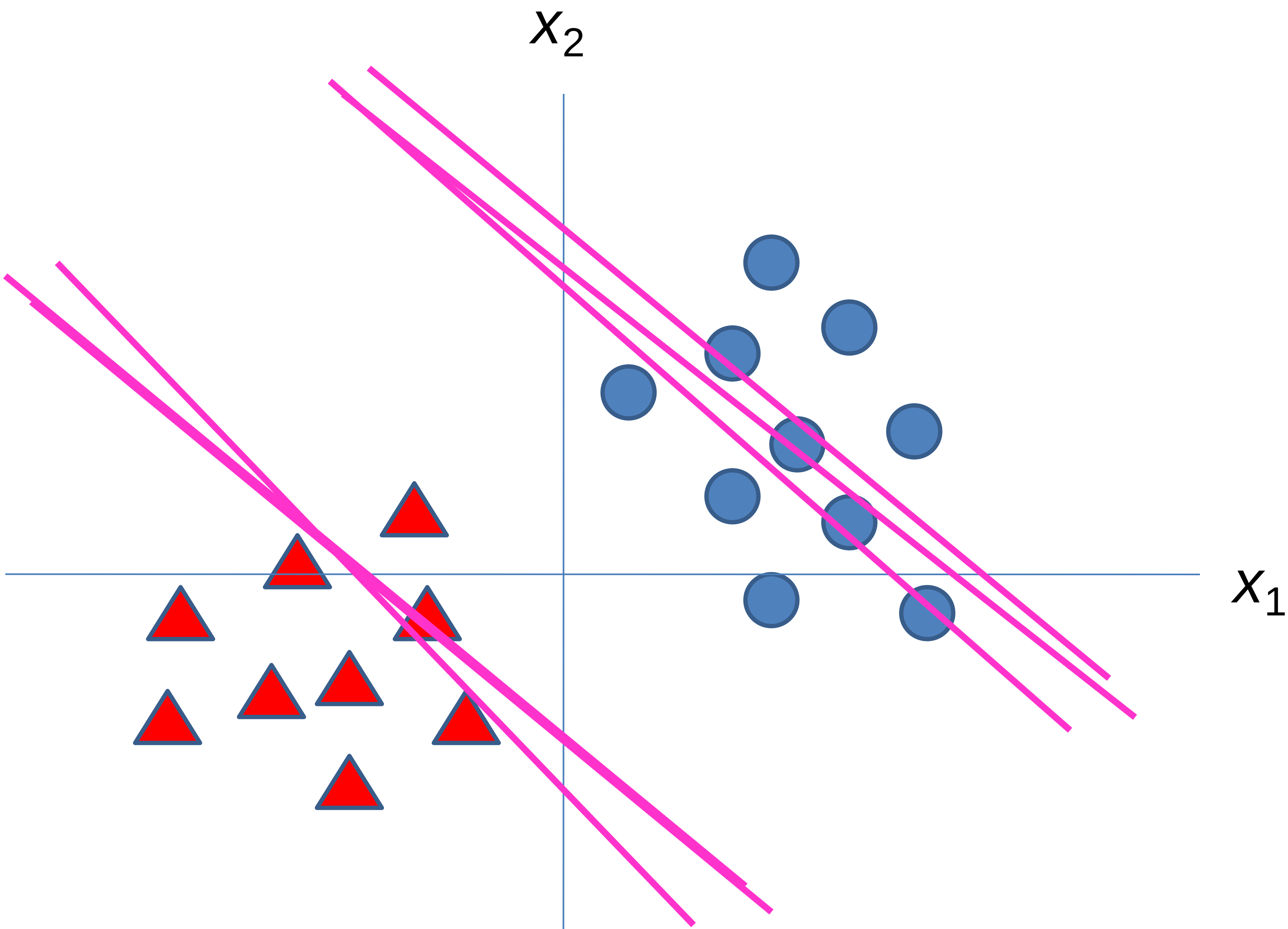
Security Systems



Learning Rate: **Besar**







$$dW1 = dW1 + (lr * D1 * P)$$

Algoritma Belajar **Propagasi Balik**

- **Pelatihan Jaringan**
  - Perhitungan Mundur

$$D2 = (1 - A2^2) * E$$

$$D1 = (1 - A1^2) * (W2 * D2)$$

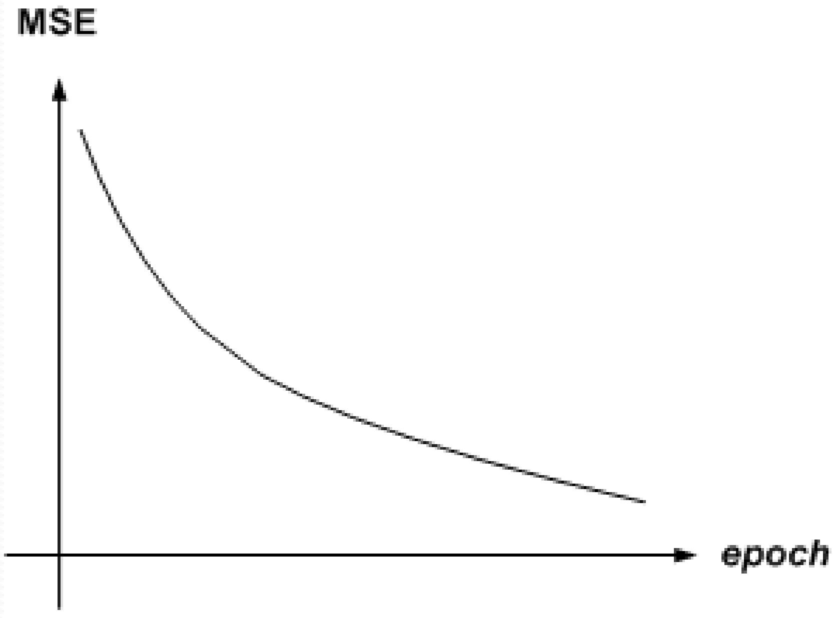
$$dW1 = dW1 + (lr * D1 * P)$$

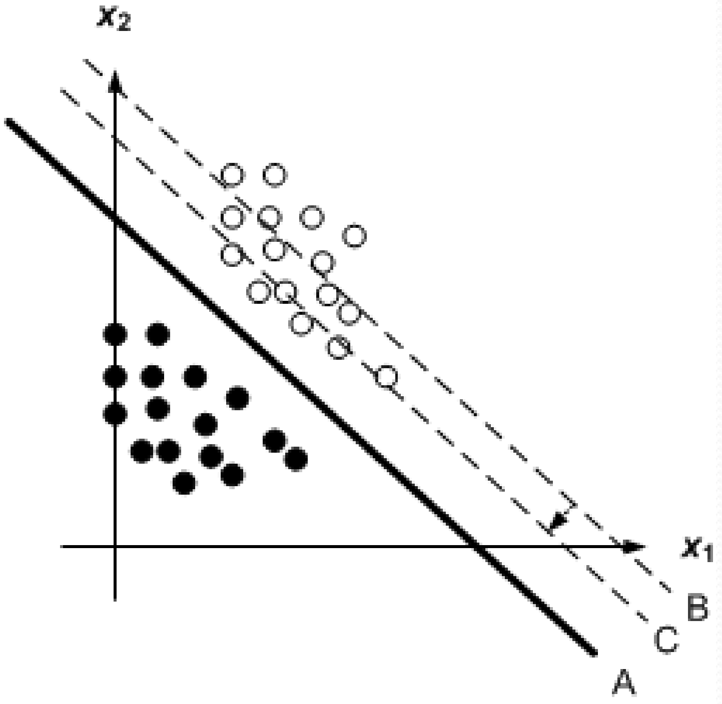
$$dB1 = dB1 + (lr * D1)$$

$$dW2 = dW2 + (lr * D2 * P)$$

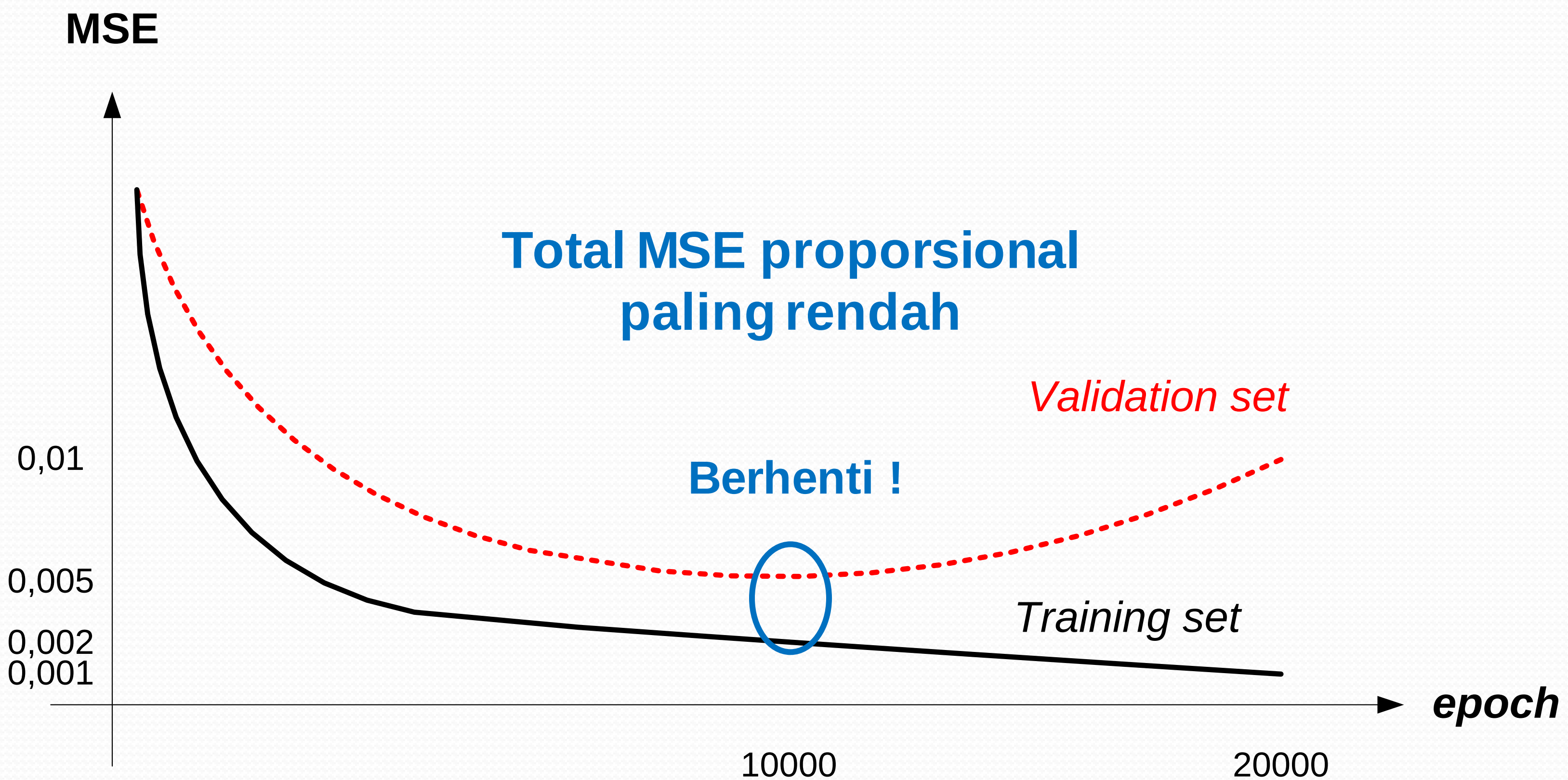
$$dB2 = dB2 + (lr * D2)$$

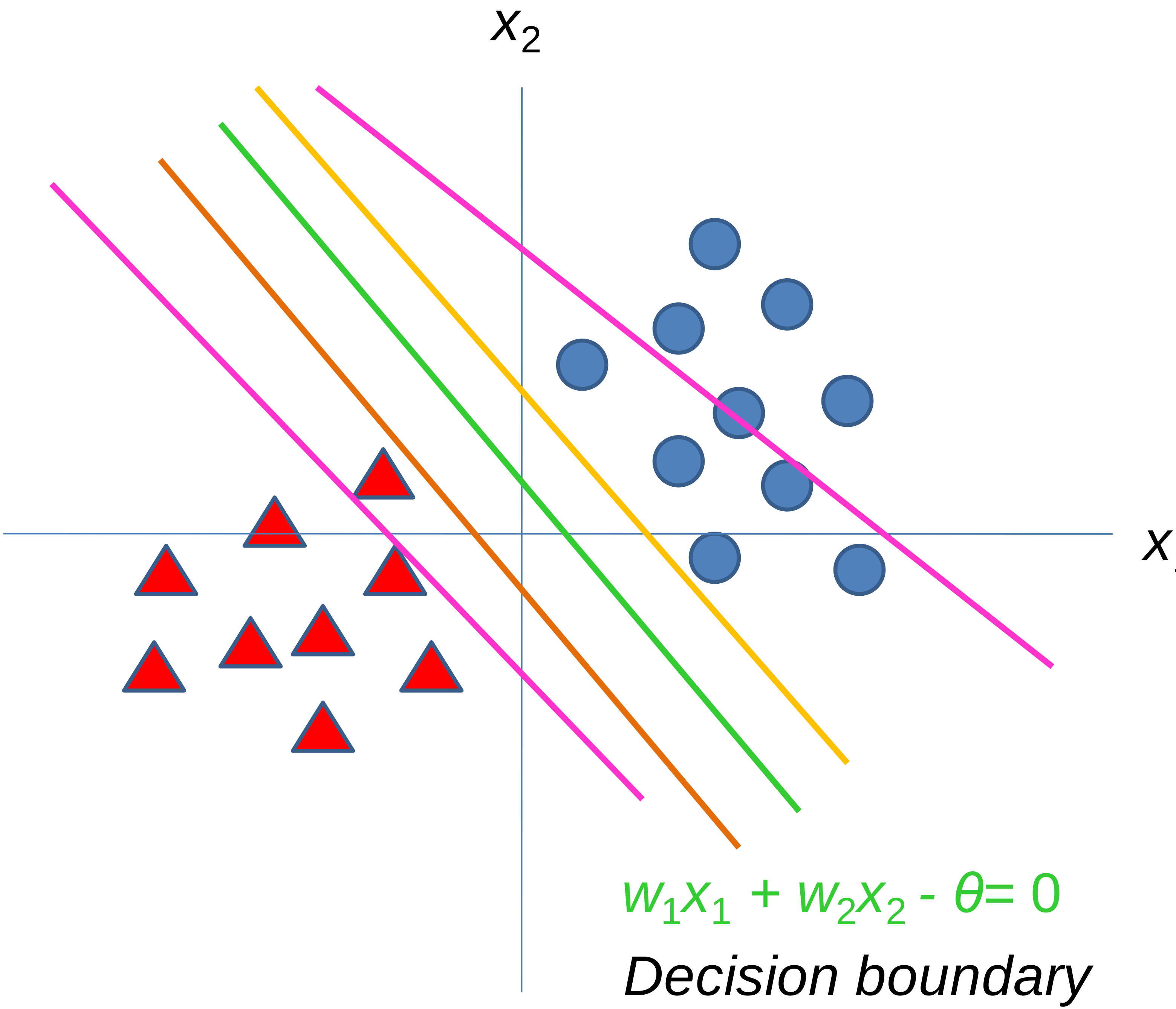
Learning Rate: **Kecil**



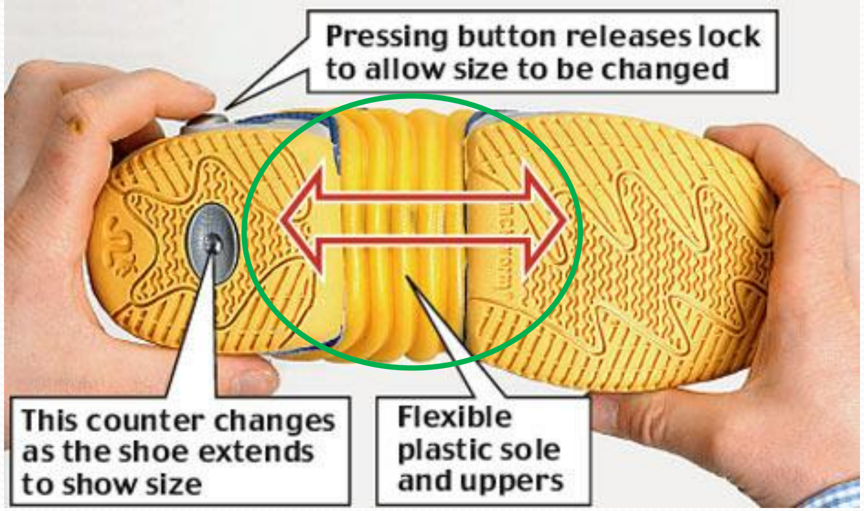


Kapan Menghentikan *Learning*?




x_2
x_1
$w_1x_1 + w_2x_2 - \theta = 0$
Decision boundary

Overfit, Oversize, Flexible



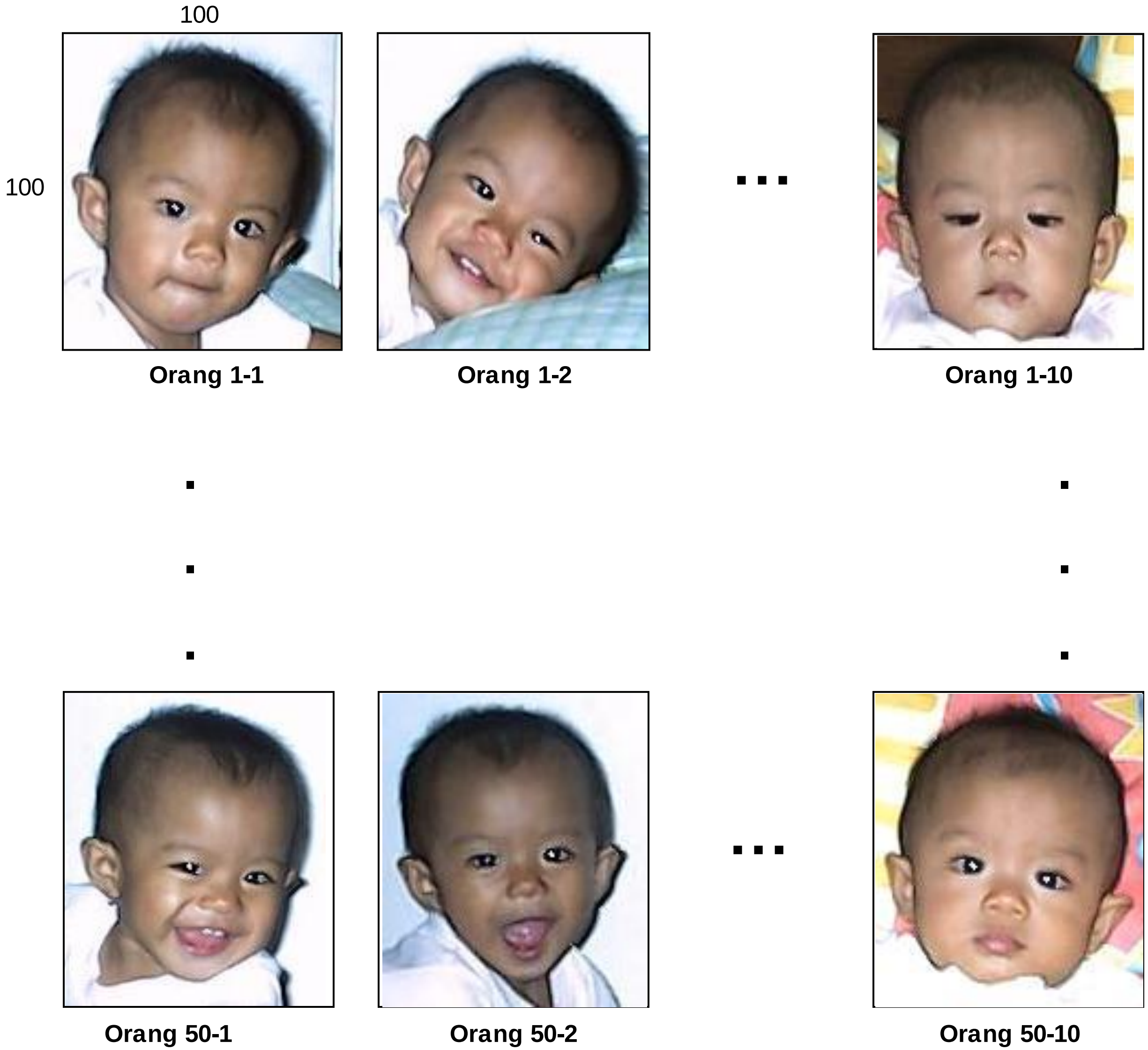
Strategi penggunaan ANN

- **Cara memandang masalah**:
  - Klasifikasi
  - Sekuriti
  - Prediksi
  - Optimasi
- **Teknik learning**: Supervised/Unsupervised
- **Desain Arsitektur**
  - Jumlah layer
  - Jumlah neuron
  - Pemetaan output
- **Strategi *learning***
  - Penyiapan data: filterisasi data, pembagian data (training, validasi, test)
  - Parameter: inisialisasi (acak atau memakai algoritma), laju belajar, dsb.
  - Penghentian learning


Menemukan persamaan dari objek berbeda?
Menemukan perbedaan dari objek mirip?
Menemukan fitur / ciri penting?
Tidak pernah diajari !!!
UNSUPERVISED Learning

Perbedaan?

- Wajah
- Suara
- Bau
- Gesture
- Sidik Jari?
- Iris Mata?
- DNA?



Face Recognition

- 50 orang
- @ 10 variation
- 100x100 pixels
- Matris 500x10000

- Efisien?
- Bisa direduksi?

PCA

PCA

- Principle Component Analysis
- Reduksi dimensi data
- Analisa data
- Ekstraksi ciri
- Visualisasi
- Mengurangi noise

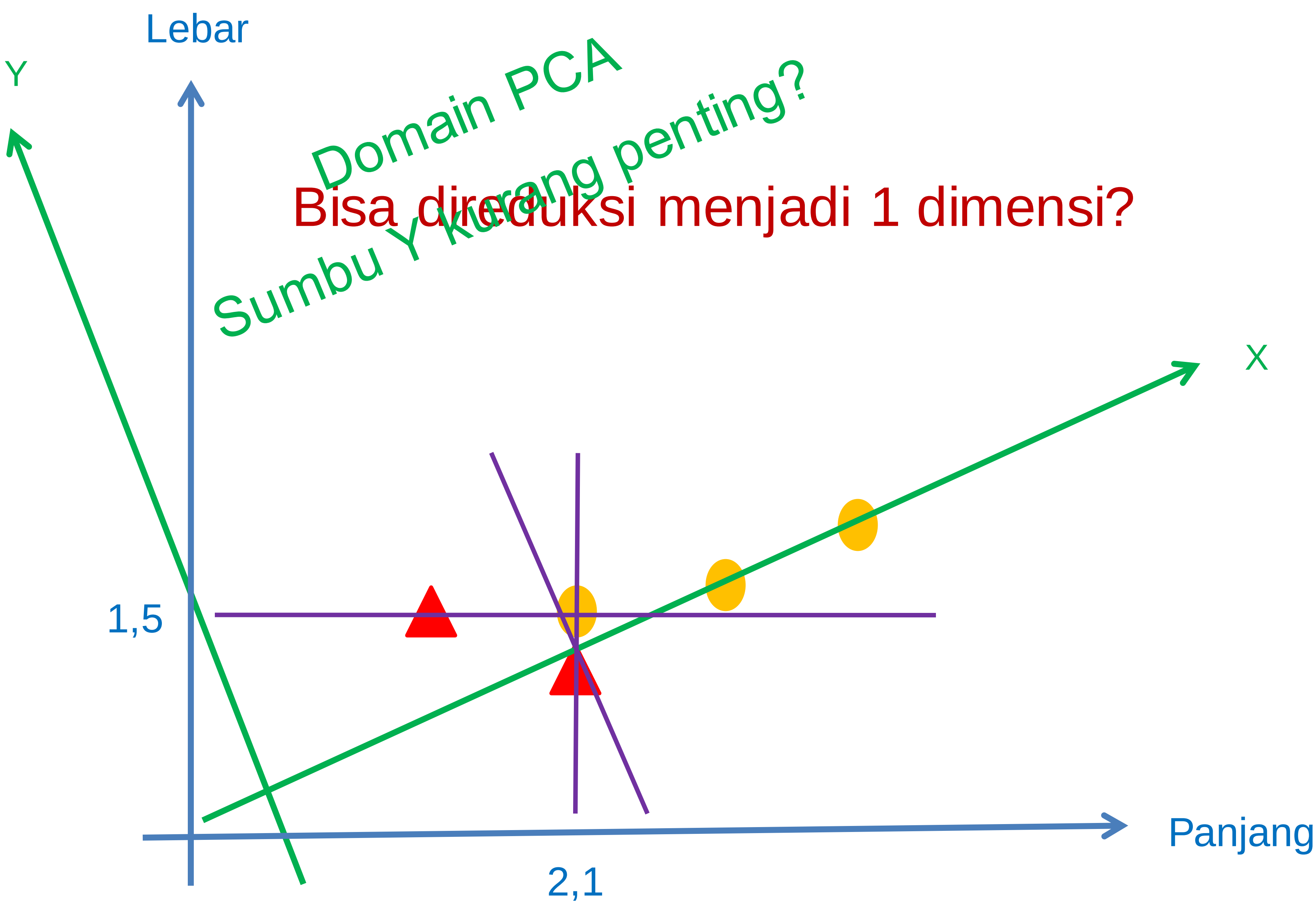
Tidak penting

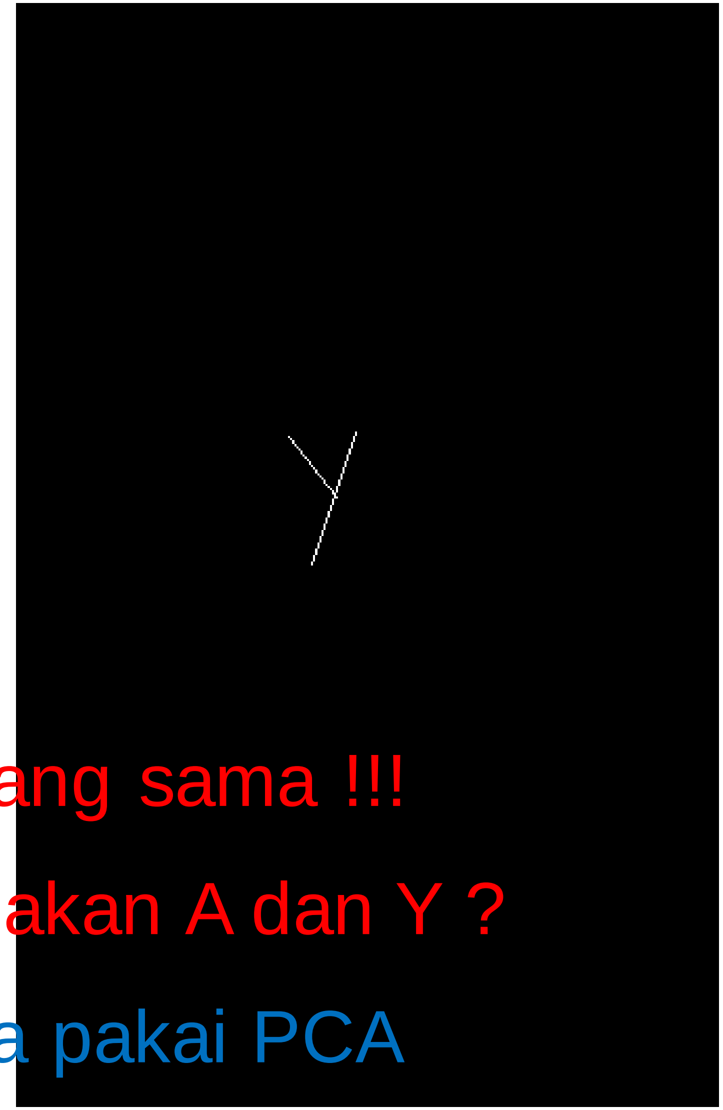
Objek	Panjang	Lebar	Tinggi	Kelas
Objek 1	2,1	1,5	0,8	Meja
Objek 2	2,3	1,7	0,8	Meja
Objek 3	2,1	1,3	0,8	Kursi
Objek 4	1,6	1,5	0,8	Kursi
Objek 5	2,5	1,9	0,8	Meja

Bisa direduksi menjadi 2 dimensi? Bisa.

Objek	Panjang	Lebar	Kelas
Objek 1	2,1	1,5	Meja
Objek 2	2,3	1,7	Meja
Objek 3	2,1	1,3	Kursi
Objek 4	1,6	1,5	Kursi
Objek 5	2,5	1,9	Meja

Bisa direduksi menjadi 1 dimensi? Tidak bisa. Ada data overlap.


Lebar
Y
Domain PCA
Sumbu Y Kurang penting?
Bisa direduksi menjadi 1 dimensi?
X
1,5
Panjang
2,1



Lebih banyak pixel yang sama !!!

Bagaimana membedakan A dan Y ?

Reduksi dimensi data pakai PCA

60000 dimensi → 40 dimensi

$$B = \begin{array}{c c} \begin{bmatrix} 3 & 5 & 4 & 5 & 2 \\ 2 & 7 & 8 & 9 & 7 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} & \begin{array}{c} x_1 \\ x_2 \\ x_3 \end{array} \end{array}$$

Rata-rata pada setiap dimensi

$$Cov(B) = \frac{(B - \overline{B}) * (B - \overline{B})^T}{N - 1}$$

Transpose

Jumlah data (observasi)

$$C_B = Cov(B) = \begin{bmatrix} 1,7 & 1,6 & 0 \\ 1,6 & 7,3 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Vektor Covarian

- Dari matriks covarian tersebut, dapat dihitung suatu basis orthogonal dengan mencari *EIGEN VALUE* dan *EIGEN VECTOR*-nya.
- *Eigenvectors* $\boldsymbol{v_i}$ dan *eigenvalues* $\boldsymbol{L_i}$ adalah solusi dari persamaan:

$$C_B v_i = L_i v_i \quad , i = 1, 2, 3, \ldots, n$$

dimensi

Jika L_i sudah ditemukan, maka *eigen vector* dapat ditemukan dari persamaan:

$$\left| C_B - L_i I \right|$$

Matriks identitas yang berukuran sama dengan $\mathbf{C}_B$

Determinant dari matriks di dalamnya

$$B = \begin{array}{c} \begin{bmatrix} 3 & 5 & 4 & 5 & 2 \\ 2 & 7 & 8 & 9 & 7 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \begin{array}{c} x_1 \\ x_2 \\ x_3 \end{array}$$

$$\Big\downarrow \quad [\mathrm{v}\ \mathrm{L}] = \mathrm{eig}(C_B)$$

$$v = \begin{array}{c} \begin{array}{ccc} v_3 & v_2 & v_1 \end{array} \\ \begin{bmatrix} 0{,}0000 & -0.9648 & 0{,}2631 \\ 0{,}0000 & 0{,}2631 & 0{,}9648 \\ 1{,}0000 & 0{,}0000 & 0{,}0000 \end{bmatrix} \end{array} \qquad L = \begin{array}{c} \begin{array}{ccc} L_3 & L_2 & L_1 \end{array} \\ \begin{bmatrix} 0{,}0000 & 0{,}0000 & 0{,}0000 \\ 0{,}0000 & 1{,}2500 & 0{,}0000 \\ 0{,}0000 & 0{,}0000 & 7{,}7500 \end{bmatrix} \end{array}$$

Jika v_{row} didapat dari v^T yang diurutkan berdasarkan *eigenvalues* terbesar

$$v_{row} = \begin{bmatrix} 0,2631 & 0,9648 & 0,000 \\ -0.9648 & 0,2631 & 0,0000 \\ 0,0000 & 0,0000 & 1,0000 \end{bmatrix}$$

maka matriks B dapat ditransformasikan ke dalam representasi PCA:

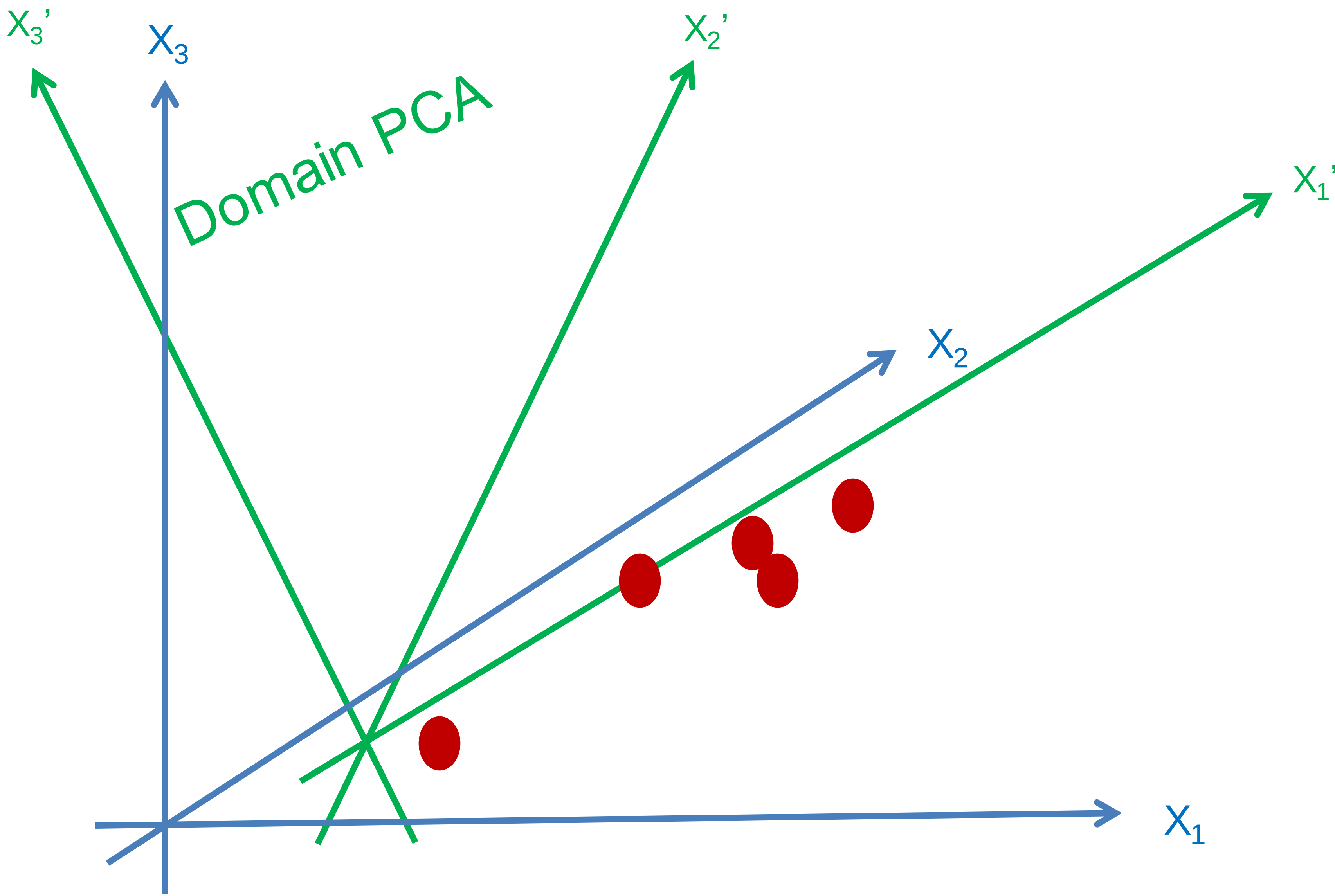
$$Y = v_{row} * B$$

$$v_{row} = \begin{bmatrix} 0,2631 & 0,9648 & 0,000 \\ -0.9648 & 0,2631 & 0,0000 \\ 0,0000 & 0,0000 & 1,0000 \end{bmatrix} * B = \begin{array}{cc} \begin{bmatrix} 3 & 5 & 4 & 5 & 2 \\ 2 & 7 & 8 & 9 & 7 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} & \begin{array}{c} x_1 \\ x_2 \\ x_3 \end{array} \end{array}$$

PCA

$$Y = \begin{bmatrix} 2,7189 & 8,0689 & 8,7706 & 9,9985 & 7,2796 \\ -2,3681 & -2,9820 & -1,7541 & -2,4558 & -0,0877 \\ 0,0000 & 0,0000 & 0,0000 & 0,0000 & 0,0000 \end{bmatrix}$$

$$B = \begin{array}{c} \begin{bmatrix} 3 & 5 & 4 & 5 & 2 \\ 2 & 7 & 8 & 9 & 7 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \begin{array}{c} x_1 \\ x_2 \\ x_3 \end{array} \qquad Y = \begin{bmatrix} 2{,}7189 & 8{,}0689 & 8{,}7706 & 9{,}9985 & 7{,}2796 \\ -2{,}3681 & -2{,}9820 & -1{,}7541 & -2{,}4558 & -0{,}0877 \\ 0{,}0000 & 0{,}0000 & 0{,}0000 & 0{,}0000 & 0{,}0000 \end{bmatrix}$$



Jika $\boldsymbol{v}_{\mathbf{row}}$ dibangun dari ketiga *eigenvector* yang ada, maka matriks B dapat dibangun kembali dengan tepat (**tanpa kesalahan**) menggunakan Y dan v_{row}

$$B_{rev} = v_{row}^{T} * Y$$

$$v_{row}^{T} = \begin{bmatrix} 0{,}0000 & -0.9648 & 0{,}2631 \\ 0{,}0000 & 0{,}2631 & 0{,}9648 \\ 1{,}0000 & 0{,}0000 & 0{,}0000 \end{bmatrix} * \quad Y = \begin{bmatrix} 2{,}7189 & 8{,}0689 & 8{,}7706 & 9{,}9985 & 7{,}2796 \\ -2{,}3681 & -2{,}9820 & -1{,}7541 & -2{,}4558 & -0{,}0877 \\ 0{,}0000 & 0{,}0000 & 0{,}0000 & 0{,}0000 & 0{,}0000 \end{bmatrix}$$

$$B_{rev} = \begin{bmatrix} 3 & 5 & 4 & 5 & 2 \\ 2 & 7 & 8 & 9 & 7 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 3 & 5 & 4 & 5 & 2 \\ 2 & 7 & 8 & 9 & 7 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Jika $\mathbf{v}_{\mathbf{row}}$ dibangun dari **satu *eigenvector*** yang *eigenvalue*-nya **terbesar** (v_1), maka matriks B dapat dibangun kembali dengan kesalahan (galat) sebesar norm(B – B_{rev}).

$$v_{row}^T = \begin{bmatrix} 0{,}2631 \\ 0{,}9648 \\ 0{,}0000 \end{bmatrix} \quad * \quad \boxed{Y = \begin{bmatrix} 2{,}7189 & 8{,}0689 & 8{,}7706 & 9{,}9985 & 7{,}2796 \end{bmatrix}}$$

$$\Downarrow$$

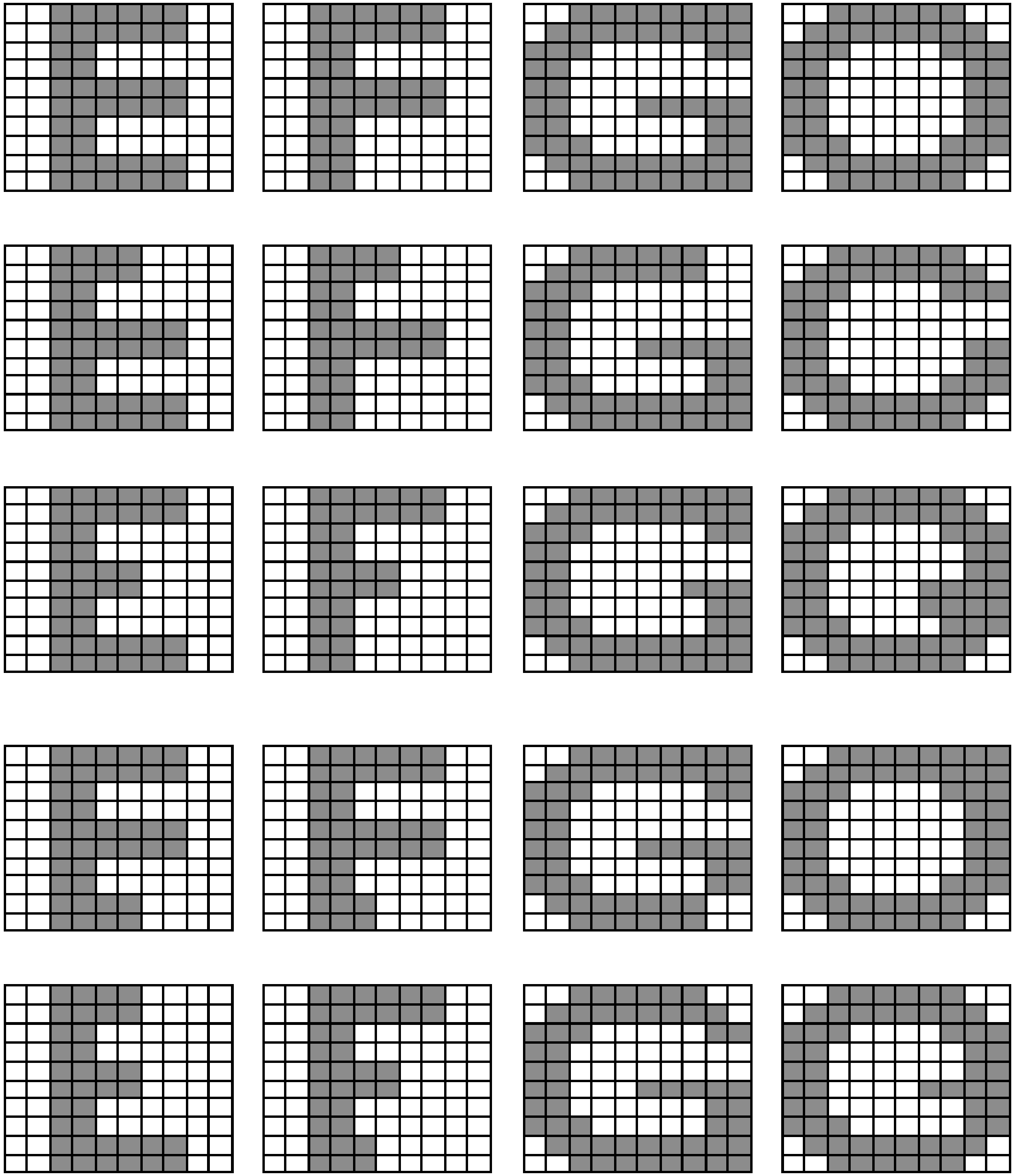
$$B_{rev} = \begin{bmatrix} 0{,}7154 & 2{,}1231 & 2{,}3077 & 2{,}6308 & 1{,}9154 \\ 2{,}6231 & 7{,}7846 & 8{,}4615 & 9{,}6462 & 7{,}0231 \\ 0{,}0000 & 0{,}0000 & 0{,}0000 & 0{,}0000 & 0{,}0000 \end{bmatrix} \qquad B = \begin{bmatrix} 3 & 5 & 4 & 5 & 2 \\ 2 & 7 & 8 & 9 & 7 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Jika $\boldsymbol{v}_{\textbf{row}}$ dibangun dari dua *eigenvector* yang *eigenvalue*-nya terbesar, (v_1 dan v_2), maka matriks *B* dapat dibangun kembali dengan kesalahan (galat) yang lebih kecil dibandingkan dengan jika $\boldsymbol{v}_{\textbf{row}}$ dibangun dari satu *eigenvector*.

$$v_{row}^T = \begin{bmatrix} 0,2631 & -0,9648 \\ 0,9648 & 0,2631 \\ 0,0000 & 0,000 \end{bmatrix} * \boxed{Y = \begin{bmatrix} 2,7189 & 8,0689 & 8,7706 & 9,9985 & 7,2796 \\ -2,3681 & -2,9820 & -1,7541 & -2,4558 & -0,0877 \end{bmatrix}}$$

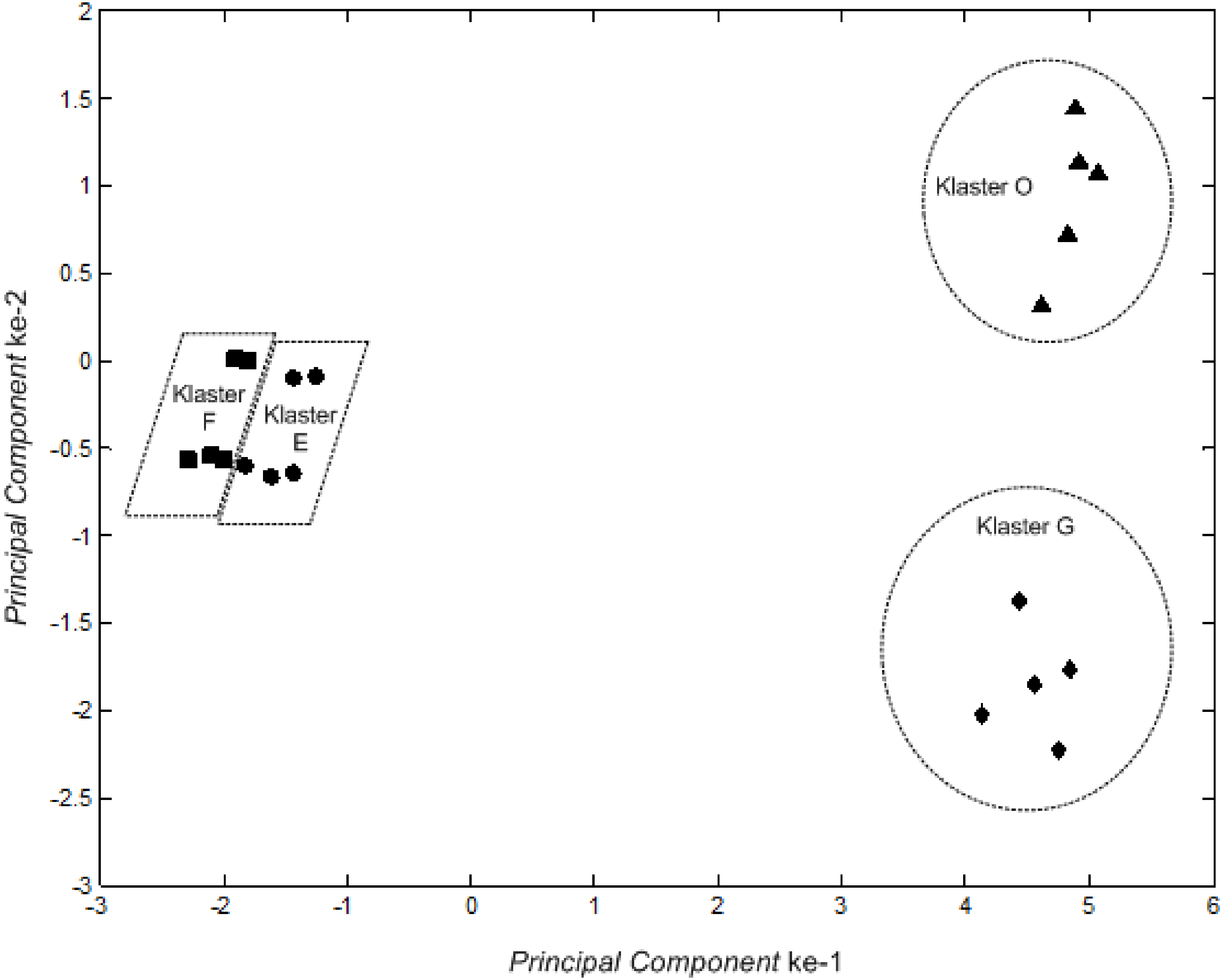
$$\Downarrow$$

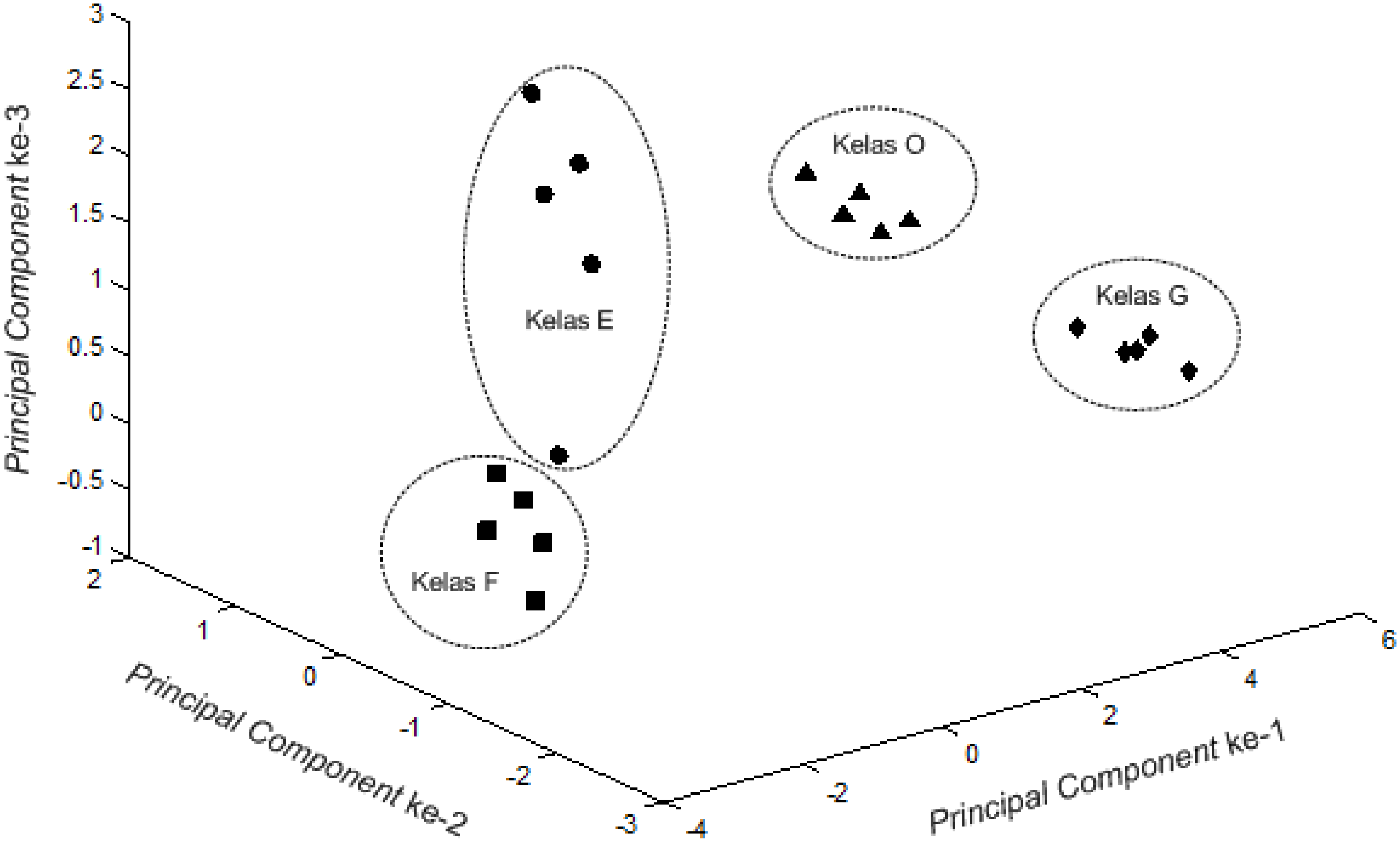
$$B_{rev} = \begin{bmatrix} 3,0001 & 5,0000 & 3,9999 & 5,0000 & 1,9999 \\ 2,0001 & 7,0003 & 8,0004 & 9,0004 & 7,0003 \\ 0,0000 & 0,0000 & 0,0000 & 0,0000 & 0,0000 \end{bmatrix} \qquad B = \begin{bmatrix} 3 & 5 & 4 & 5 & 2 \\ 2 & 7 & 8 & 9 & 7 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$



- Visualisasi 100 dimensi?
- Tidak bisa dilakukan
- Reduksi 100 → 2 dimensi
- Gunakan PCA dari dua *eigenvector* yang *eigenvalue*-nya terbesar.

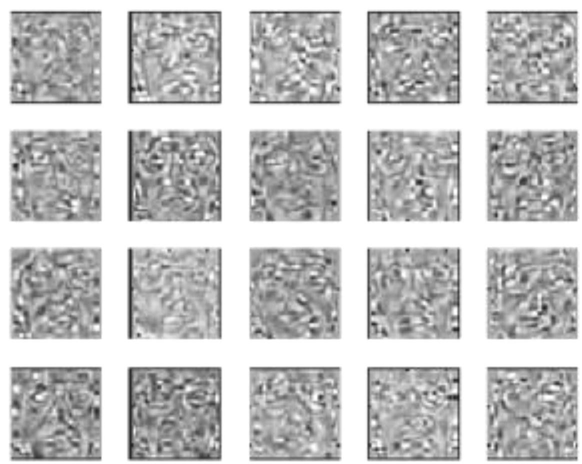
Pola	Pix 1	Pix 2	Pix 3	Pix 4	Pix 5	…	Pix 100
E1	0	0	1	1	1	…	0
F1	0	0	1	1	1	…	0
G1	0	1	1	1	1	…	1
O1	0	1	1	1	1	…	1
..							
O5	0	1	1	1	1	…	1


Klaster O
Klaster G
Klaster F
Klaster E
Principal Component ke-2
Principal Component ke-1
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
-3
-2
-1
0
1
2
3
4
5
6

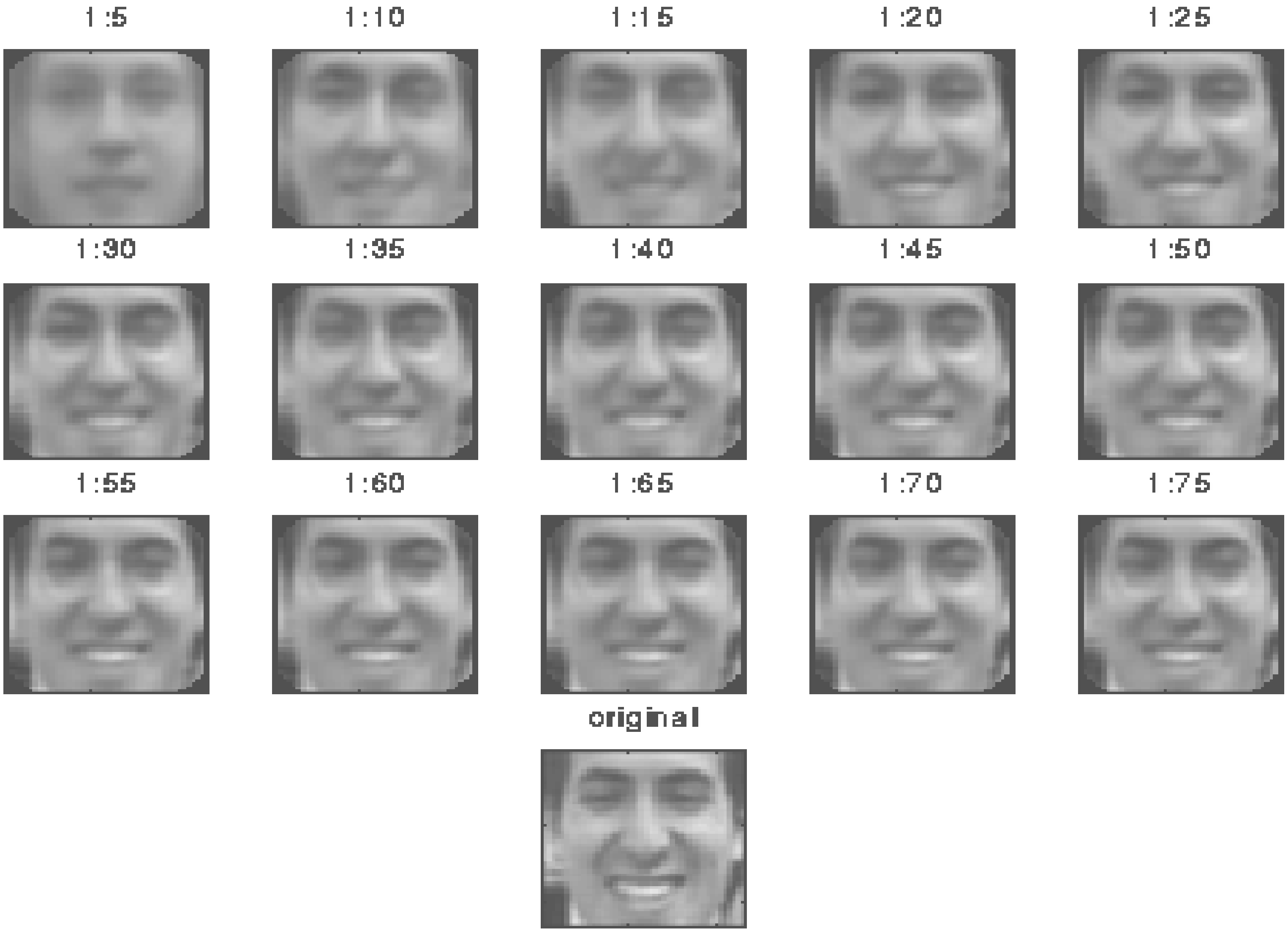


Principal Component ke-3
3
2.5
2
1.5
1
0.5
0
-0.5
-1
Kelas E
Kelas O
Kelas G
Kelas F
Principal Component ke-2
2
1
0
-1
-2
-3
Principal Component ke-1
-4
-2
0
2
4
6

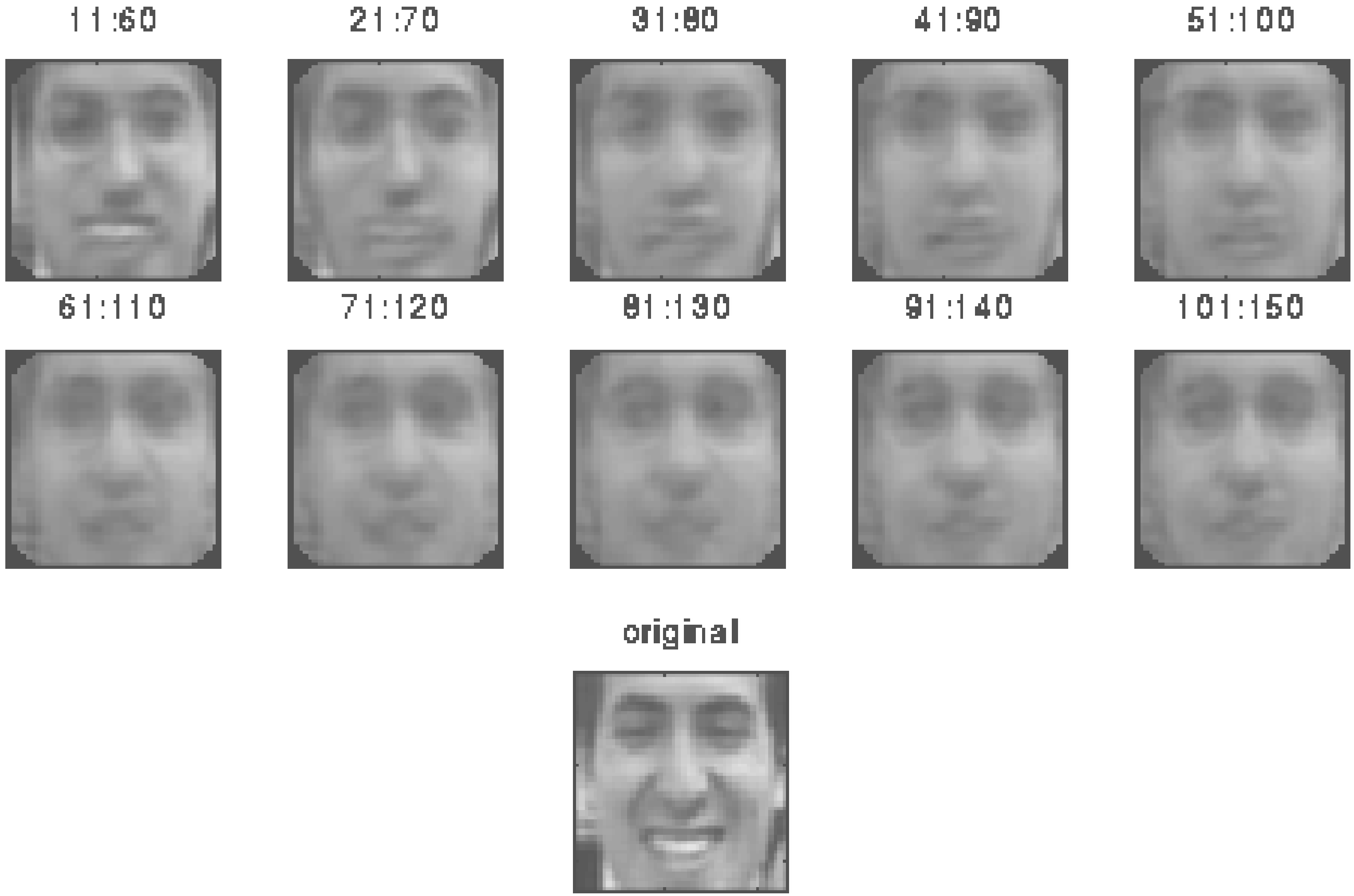
The first 20 eigen faces with the highest eigen values



Eigenfaces with eigenvalues ranked from 141 to 160



Faces reconstructed using eigenfaces with high eigenvalues. The label above each face is the range of eigenfaces used.



Faces reconstructed using eigenfaces with low eigenvalues. The label above each face is the range of eigenfaces used.

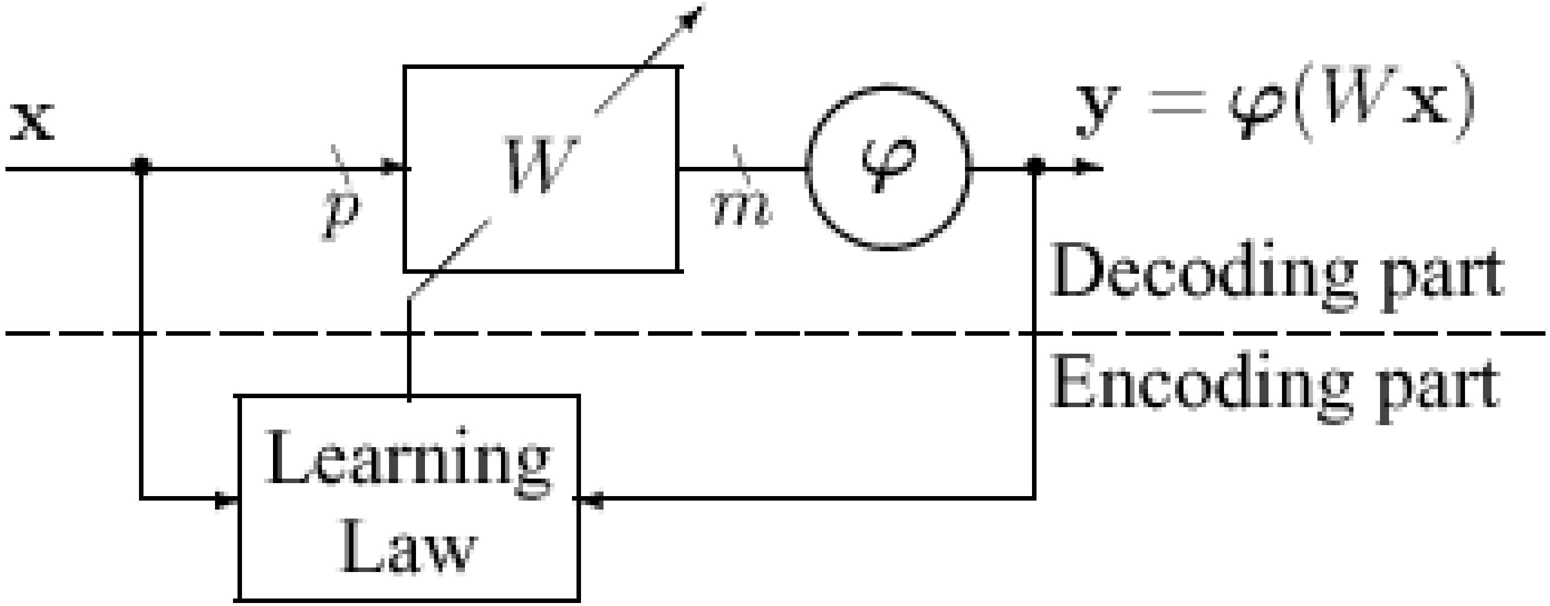
Perhitungan PCA

$$C_B = Cov(B) = \frac{(B-\overline{B})*(B-\overline{B})^T}{N-1}$$

$$C_B v_i = L_i v_i \quad ,i=1,2,3,\ldots,n$$

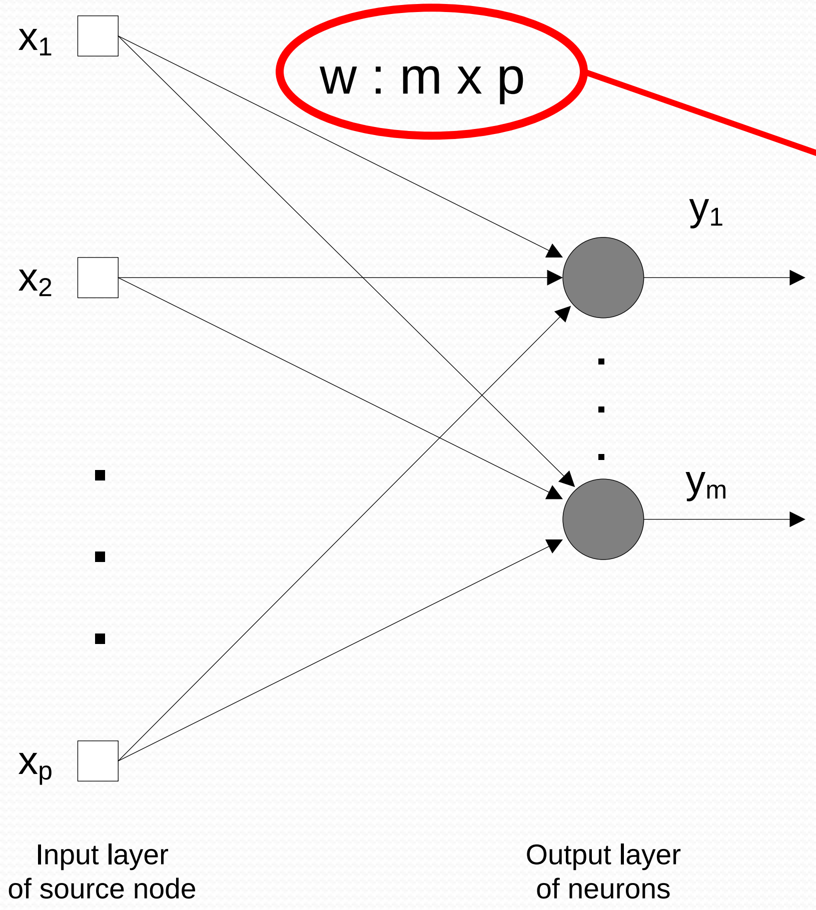
$$|C_B - L_i I|$$

- Sejuta citra @ 800 x 600 pix, kompleksitas & memory?
- Kompleksitas tinggi → prosesnya lama
- Memory besar
- Solusinya?
- ANN: Hebbian Learning



- Decoding: matrix *W* berukuran *m* × *p*
- Satu baris *weight vector* = satu neuron

Arsitektur Hebbian Network



Nilai pendekatan untuk V_{row} (yang dihitung secara matematis)

Hebbian-Based PCA

- Generalized Hebbian Algorithm (GHA)
- Adaptive Principal Components Extraction (APEX)

Generalized Hebbian Algorithm (GHA)

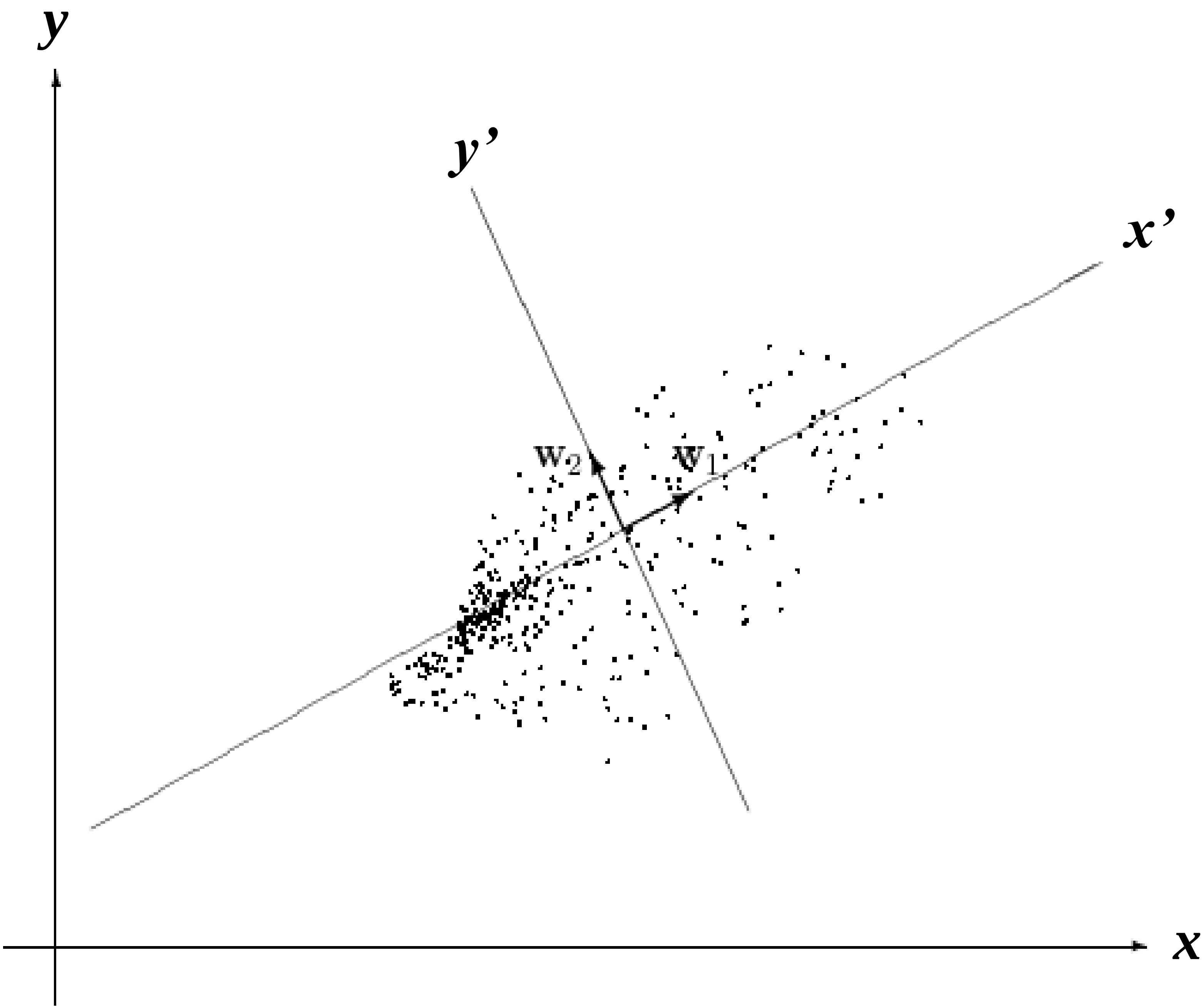
1. Initialize the weights to a small random values at n = 1. Assign a small positive value to the learning-rate.
2. Compute

$$y_j(n) = \sum_{i=1}^{m} w_{ji}(n) x_i(n),$$

$$\Delta w_{ji}(n) = \eta \left[y_j(n) x_i(n) - y_j(n) \sum_{k=1}^{j} w_{ki}(n) y_k(n) \right],$$

for $j = 1,2,\ldots,l$, and $i = 1,2,\ldots,m$..

3. Increment n by 1, and go to step 2. Repeat until all synaptic weights reach steady-state values (||w|| → 1).

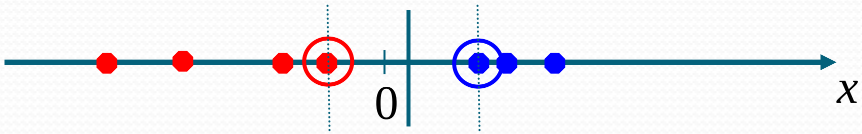

y
y'
x'
w2
w1
x

Real Problems

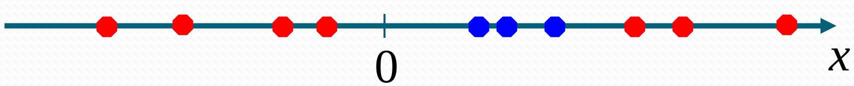
- Reduksi dimensi
  - Ekstraksi fitur: PCA, ICA, Fourier, dsb.
  - Klasifikasi menjadi lebih mudah
- Penambahan dimensi
  - Support Vector Machine (SVM)
  - Fungsi Kernel
  - Klasifikasi menjadi lebih mudah
- Reduksi dan Penambahan

SVM

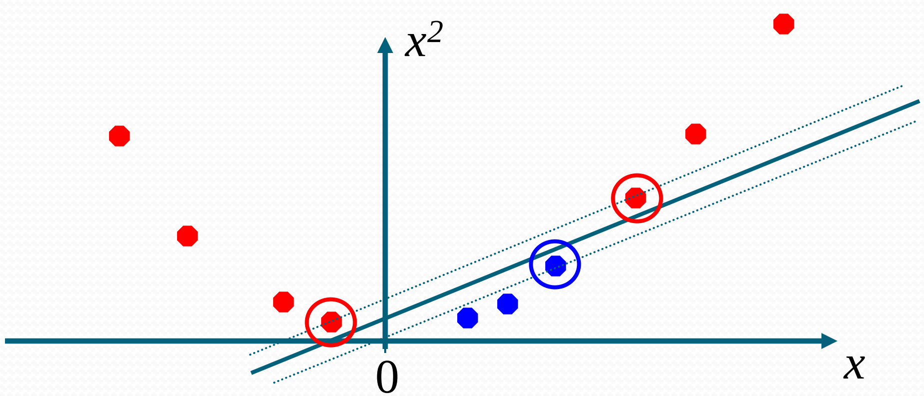
- Datasets yang *linearly separable* masih bisa diselesaikan:



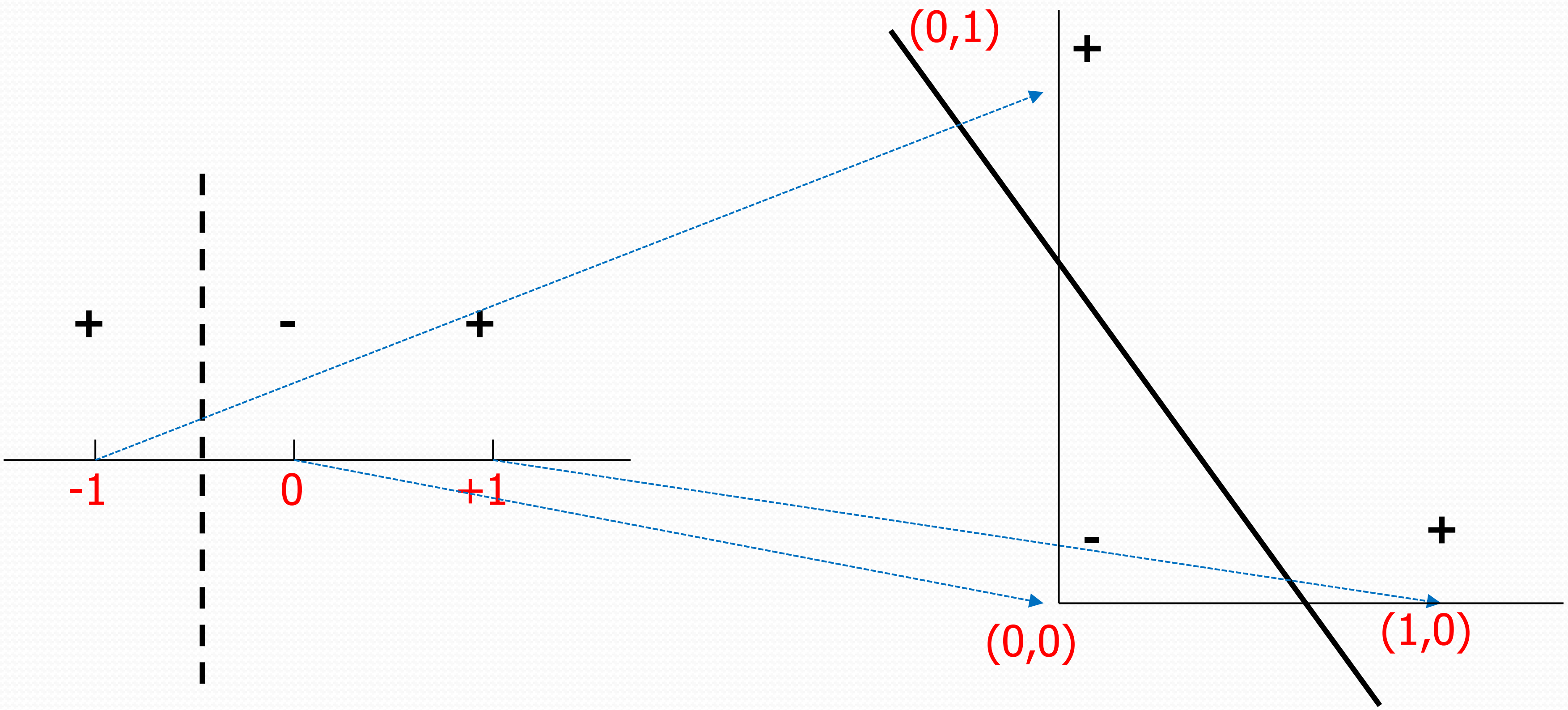
- Tetapi, apa yang bisa dilakukan jika *dataset*-nya tidak *linearly separable*?



- Petakan dataset ke ruang berdimensi lebih tinggi:



SVM



Strategi penggunaan ANN

- **Cara memandang masalah**:
  - Klasifikasi
  - Sekuriti
  - Prediksi
  - Optimasi
- **Teknik learning**: Supervised/Unsupervised
- **Desain Arsitektur**
  - Jumlah layer
  - Jumlah neuron
  - Pemetaan output
- **Strategi *learning***
  - Penyiapan data: filterisasi data, pembagian data (training, validasi, test)
  - Setting parameter: inisialisasi, laju belajar, dsb.
  - Penghentian learning

Case 1: Spam Filtering

- KDD: Knowledge Discovery from Data
- International competition about Personalised Spam Filtering held in Humboldt-Universität zu Berlin, Germany
- Find set of rules or strategy to distinguish an email as spam or not

From: DOUGLAS M ROBIN douglas_markrobin@yahoo.co.uk
Subject: COMPLETE THE FORM WITH YOUR PASSPORT PHOTO ATTACHED
The National Lottery
PAYMENT/PROCESSING OFFICE, LONDON, UK.
3240 RIDGE-WAY,LONDON NW71RN. 00447040112422

Batch/074/05/ZY3
Ticket/ 5647600545188
Serial No /5073-11

Dear (),

I acknowledge the receipt of your mail, as regard your request the reason is that over the years we have had cases of double claim of winnings and with the help of the verification form its earlier to detect the winners in various category.

Your information is need to process other vital document before your cash prize can be release to you for claim.

I need those vital information alongside passport photo to proceed with the processing of your winnings.I need urgent response in 24 hrs because you have less 2 weeks .

Regards,
Douglas Robin.
(FIDUCIARY OFFICER)

Data

- 4000 emails as training set for learning. 2000 spam and 2000 non-spam.
- 4000 emails as tune set (validation set). 2000 spam and 2000 non-spam.
- 2500 emails as validation set. This data is from a user inbox containing 1250 spam and 1250 non-spam.
- 7500 emails as test set, without label of spam nor non-spam. The data are from 3 inboxes. Each box contains 2500 email, 1250 spam and 1250 non-spam.

Rules of the game

- All 18.000 emails are encoded as numerical data
- Firstly, build a wordlist atau dictionary from all emails and produce a dictionary of 200.000 words
- Secondly, each email is parsed and each word is replaced by the suitable index of word in the dictionary and calculate the word frequency
- Email spam is labeled by **1**, email non-spam **-1**, and unknown email **0**.

Example: Wordlist (Dictionary)

No	Word
1	attached
2	document
3	form
4	lottery
5	national
6	open
7	passport
8	ticket
…	…
200.000	urgent

Example: Spam Email

- **1** 35:1 73:1 77:1 206:1 16176:1
- **1** 2058:1 27162:1 49588:1
- **1** 9:3 94:1 109:1 163:1 405:1 406:1 415:2 416:1 435:3 436:3 437:4 440:4 450:3 456:1 457:1 461:1 466:1 467:1 477:1 478:1 751:1 1015:1 1034:14 1041:1 1216:1 1226:1 1231:1 1666:1 2344:1 2345:1 2505:1 2528:1 3498:1 4339:1 4463:1 7480:1 8143:1 15050:1 17176:1 19051:1 20895:1 22963:1 35908:1 48253:1 49469:1 60004:1 78684:1 84924:1 85550:1 93429:1 95839:1 106782:1 106783:4 106784:2 106785:2 106786:1 106788:1 106802:1 106803:1

Example: Non-Spam Email

- **-1** 9:1 82:1 92:1 104:1 231:1 308:1 338:1 351:1 390:1 440:2 693:1 933:1 975:1 984:1 1631:1 2404:2 2560:2 2589:2 3361:1 3630:1 4042:1 4059:1 6515:1 7851:1 8762:1 10427:1 16178:1 37517:1 44973:1 53347:1 109089:2 109090:1 110944:1 111668:1 133323:1 140060:1 155590:1

Example: Unlabeled Email

- **0** 94:1 204:1 257:1 582:1 4898:1 6371:1

Classification technique?

- ID3
- Bayesian Learning
- Genetic Algorithm
- Support Vector Machine
- ANN: MLP
- Or other techniques?

Feature Extraction?

- Information Gain
- PCA
- ICA
- Atau teknik lainnya

Training Set

Email ke-	K1	K2	K3	K4	K5	K6	...	K200000	Kelas
1	3	9	0	0	0	2	...	1	1
2	7	3	0	2	0	1	...	0	-1
3	2	0	17	1	0	8	...	0	1
4	9	2	4	16	5	7	...	8	1
5	1	0	2	6	0	5	...	16	-1
6	7	0	0	0	0	0	...	0	1
7	0	3	0	0	0	8	...	0	-1
8	5	12	3	1	0	0	...	0	1
9	6	8	0	18	0	0	...	5	1
...	...	...	...	...	...	...	...	...	...
4000	8	2	0	23	0	1	...	9	-1

Information Gain (IG)

- Untuk memilih kata-kata yang paling efektif dalam mengklasifikasikan data (*spam* atau *non-spam*).
- IG digunakan untuk mengukur efektivitas suatu atribut dalam mengklasifikasikan data dengan cara:
  1. Hitung IG dari semua kata yang ada (sekitar 200.000)
  2. Urutkan kata-kata tersebut berdasarkan IG-nya, dari yang terbesar sampai yang terkecil
  3. Ambil sejumlah kata (yang IG-nya paling besar)

Email spam:

1 35:11 73:5 77:9 206:12 16176:6

Look up

Daftar 60 nomor kata yang
Information Gain-nya paling besar

No. Urut	Nomor kata
1	9
2	35
3	73
4	206
5	1.190
6	2.657
7	13.890
8	16.176
...	
60	199.576

- Jumlah kata?
- Trial & error
- Misal: 60 kata

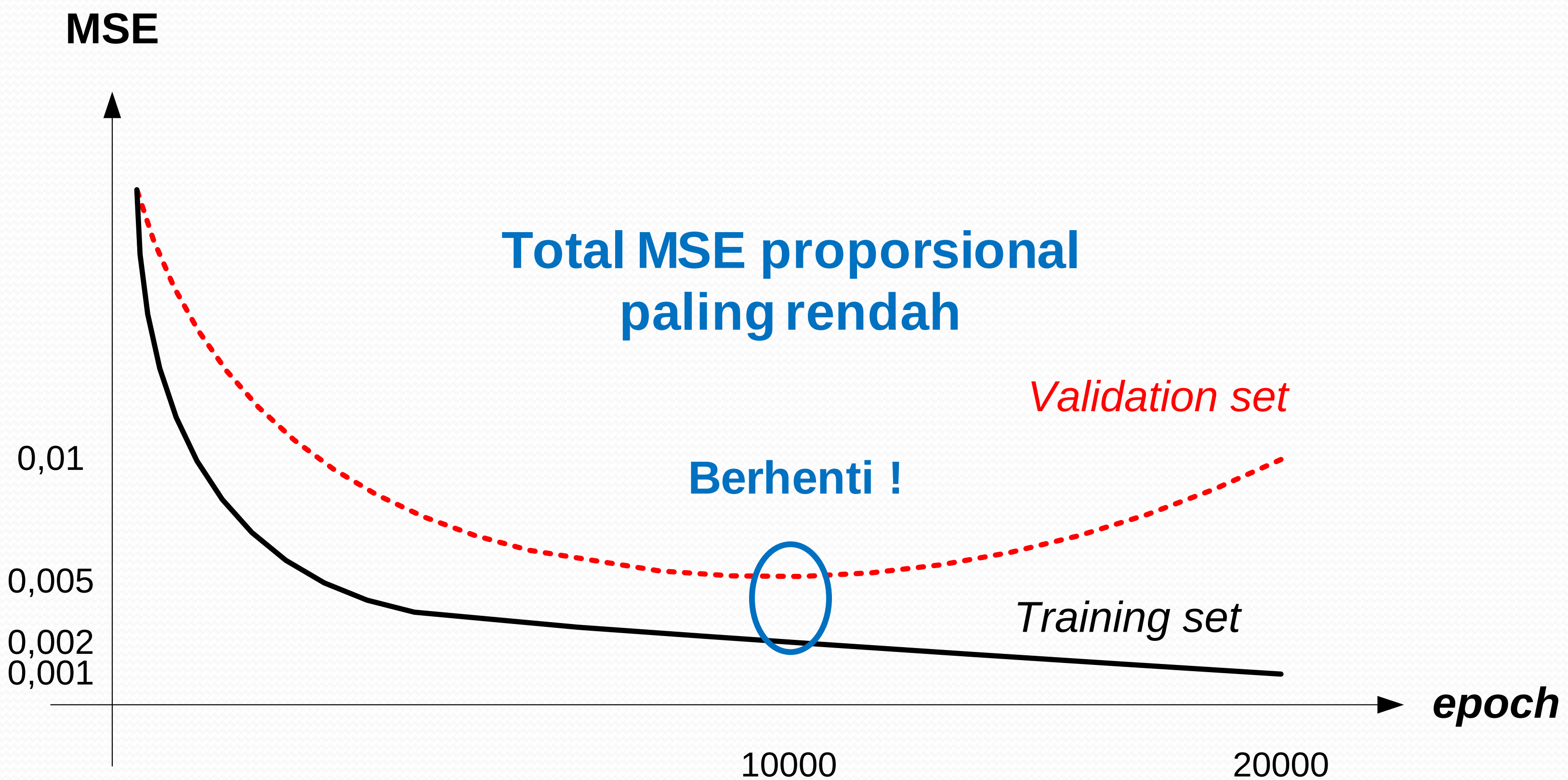
Feature yang dihasilkan:

0	11	5	12	0	0	0	6	0	0	...	0
1	2	3	4	5	6	7	8	9	10	...	60

PCA untuk Ekstraksi Fitur?

- 4.000 records dan 200.000 kolom
- Data: 4.000 x 200.000
- Matriks Covarian: 200.000 x 200.000 = 40 milyar
- Jika 1 elemen = 2 byte, maka memory = 80 GB
- PCA dengan cara matematis
  - Deterministik
  - Prosesnya lama dan perlu memory besar
- Hebbian Learning
  - Hasilnya berupa pendekatan (tidak pasti)
  - Lebih cepat dan memory kecil

Kapan Menghentikan *Learning*?



Kasus 2: Optimasi Pembelian Buku

- Data Mining Cup 2009
- Di Jerman: 96 ribu judul buku per tahun
- Pembelian yang optimal
  - Judul mana yang sebaiknya dibeli?
  - Berapa kuantitasnya untuk setiap judul tersebut?
- Kasus riil di perusahaan Libri (Jerman)
  - 2.418 cabang
  - Untuk 8 judul tertentu, berapa jumlah yang sebaiknya dibeli oleh setiap cabang?
  - Learning berdasarkan data histori?

Karakteristik Data

Feature	Type	Description	Attributes
ID	Integer	• Unique location id	random unique key
WGxxxxx	Integer	• Number of total items sold within 12 months within a category • Categories with 5 digits • First digit: Information about the type of product e.g. hardcover vs. paperback • Second to fifth digit: Hierarchical information about the type of content e.g. second digit Fiction and third digit subcategory Science-Fiction	independent variables
T1…T8	Integer	• Number of items sold within 12 months per title	target value

Contoh Data Training

ID	WG00000	WG23000	WG23110	…	T1	T2	…	T8
3377	0	0	10	…	3	0	…	2
4355	0	0	0	…	0	4	…	0
1119	0	0	7	…	4	0	…	0
4429	0	0	0	…	0	0	…	0
2428	3	0	3233	…	1	96	…	36
1866	6	8	1451	…	5	251	…	44
2458	0	6	347	…	0	45	…	14
4531	7	12	1302	…	34	20	…	47
1884	0	0	339	…	198	9	…	25
…	…	…	…	…	…	…	…	…

Teknik Klasifikasi?

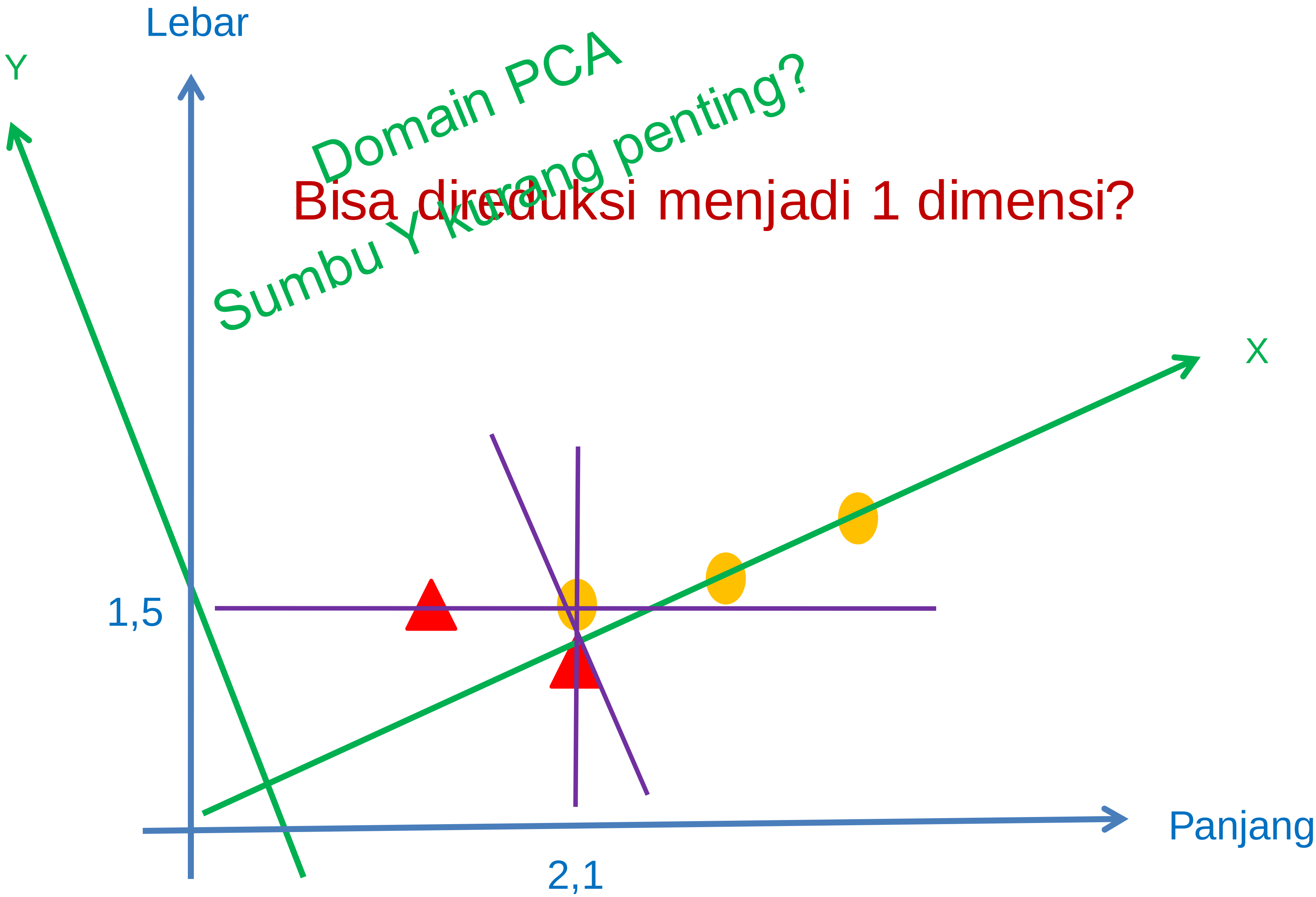
- Fuzzy
- ANN: MLP, CNN, …
- EAs: GA, ES, GE, …
- SVM
- …

Feature Extraction?

- Information Gain
- PCA
- ICA
- …

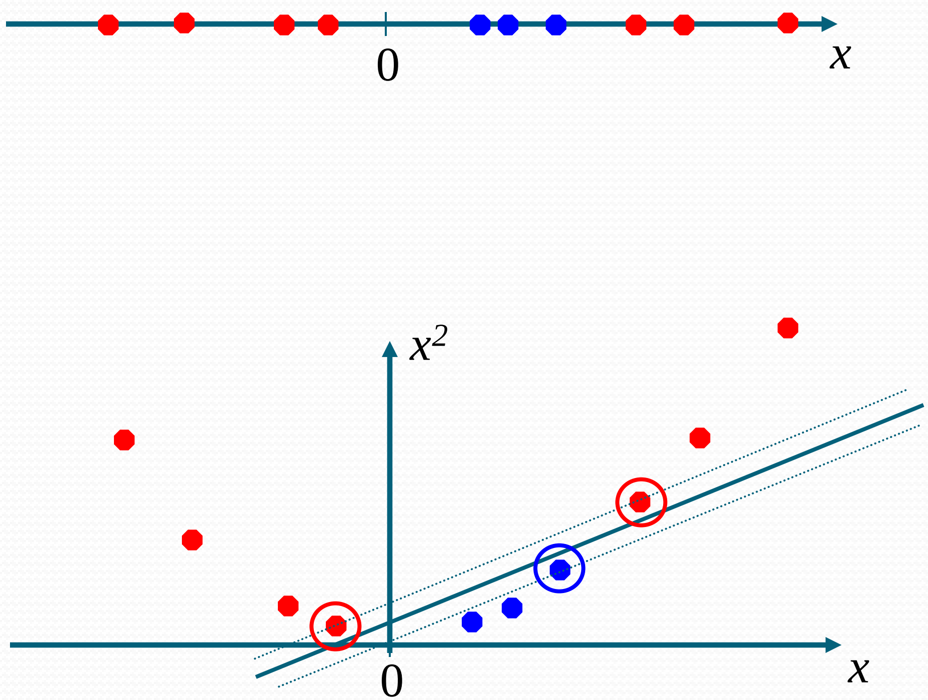
IG atau PCA?

- IG
  - Dunia Nyata
  - Menemukan atribut yang paling efektif
- PCA
  - Konversi ke dunia lain
  - Menemukan atribut yang paling efektif
- *Feature Extraction* yang baik:
  - Inter-class → jaraknya diperlebar
  - Intra-class → jaraknya dipersempit



Lebar
Y
Domain PCA
Sumbu Y Kurang penting?
Bisa direduksi menjadi 1 dimensi?
X
1,5
Panjang
2,1

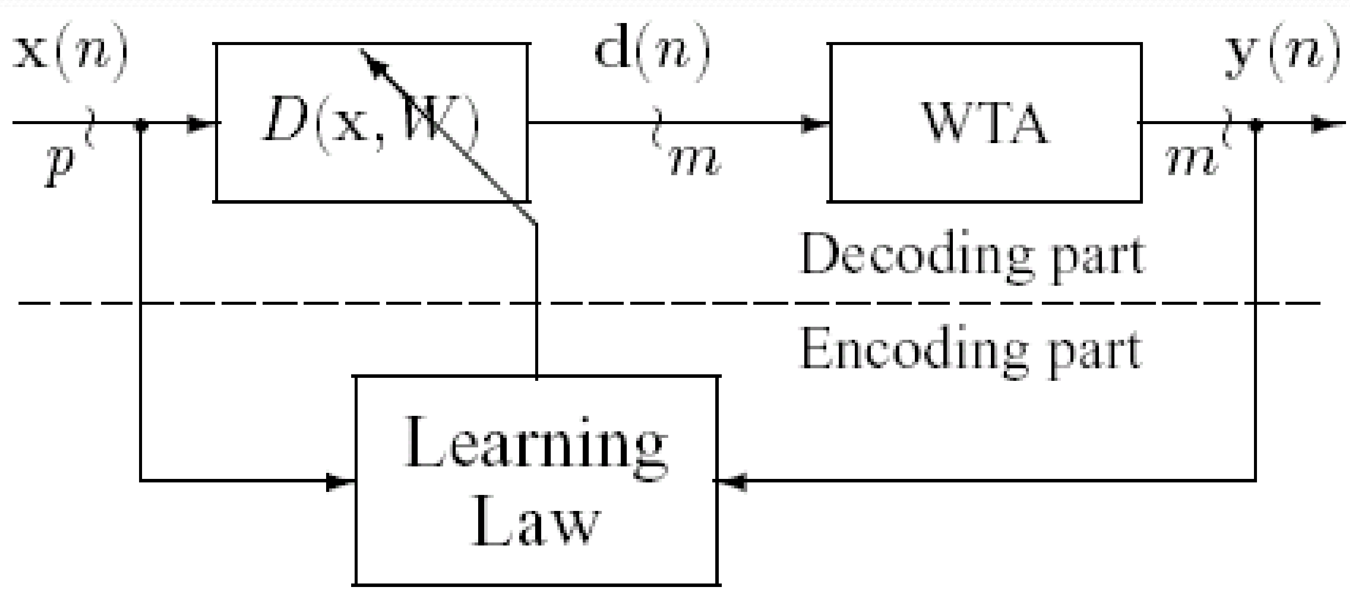
SVM



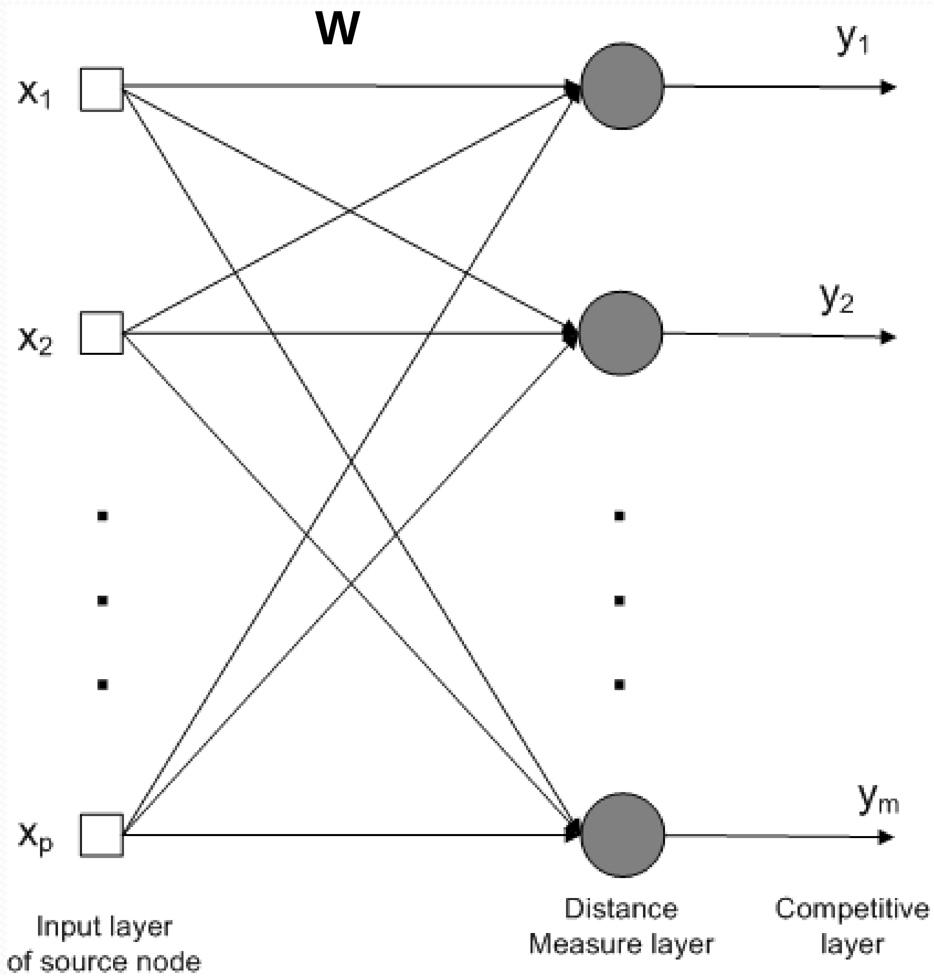
Competitive Neural Networks (CNN)

CNN terdiri dari dua lapis neurons:

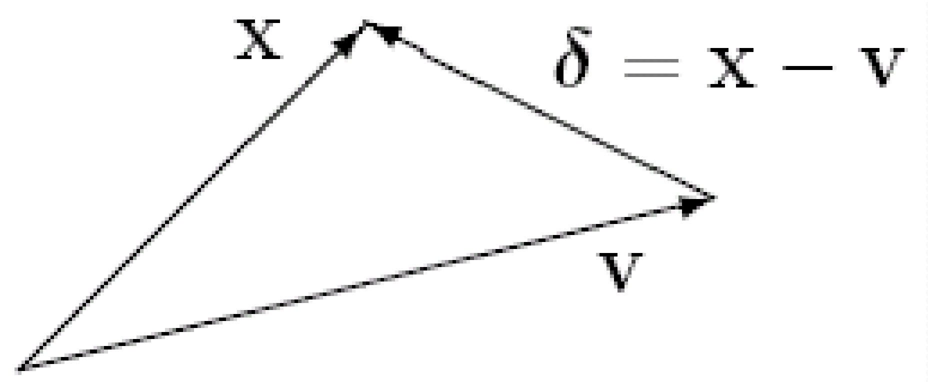
- Distance-measure layer
- Competitive layer atau "Winner-Takes-All" (WTA)



Arsitektur WTA Network



Euclidean norm



$$d = ||\mathbf{x} - \mathbf{v}|| = ||\boldsymbol{\delta}|| = \sqrt{\delta_1^2 + \ldots + \delta_p^2} = \sqrt{\boldsymbol{\delta}^T \cdot \boldsymbol{\delta}}$$

The Competitive Layer

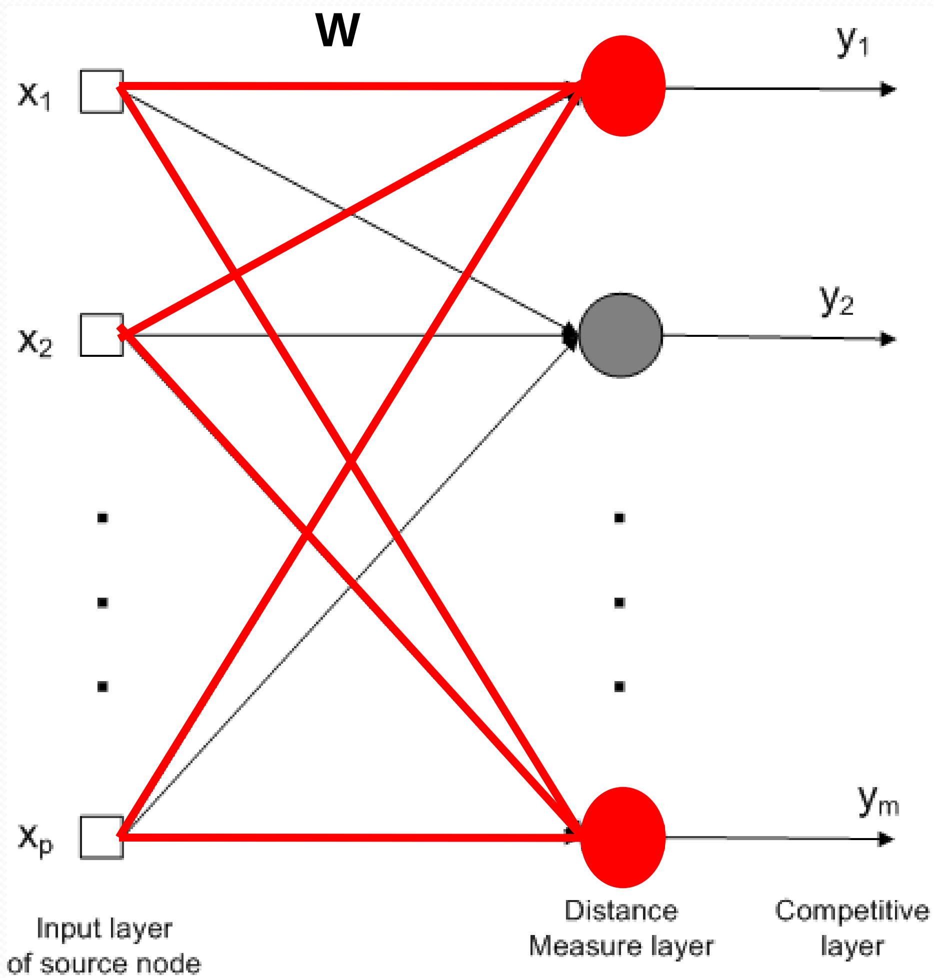
$$y_j = \begin{cases} 1 & \text{if } \; j = \arg\min_k D(\mathbf{x}, \mathbf{w}_k) \\ 0 & \text{otherwise} \end{cases}$$

Training

Inisialisasi: random

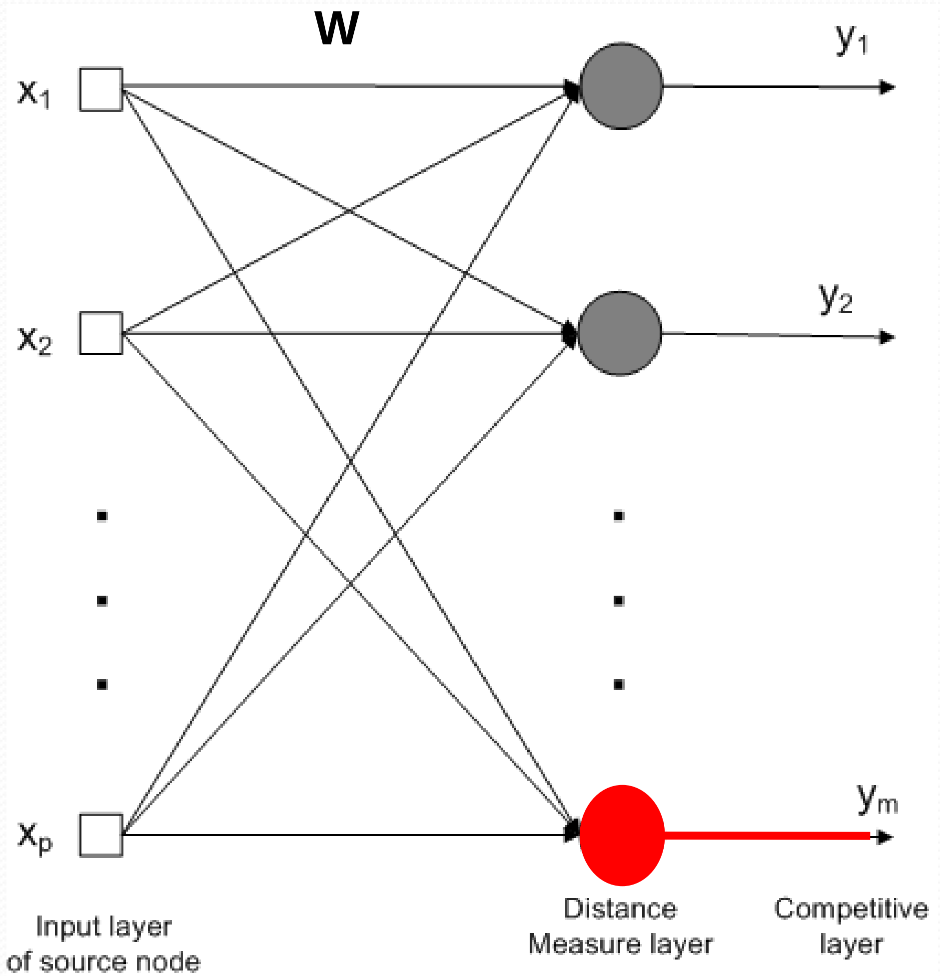
Hasil Training CNN
W → titik pusat cluster yang "TEPAT"

Pola N



Testing

Pola X



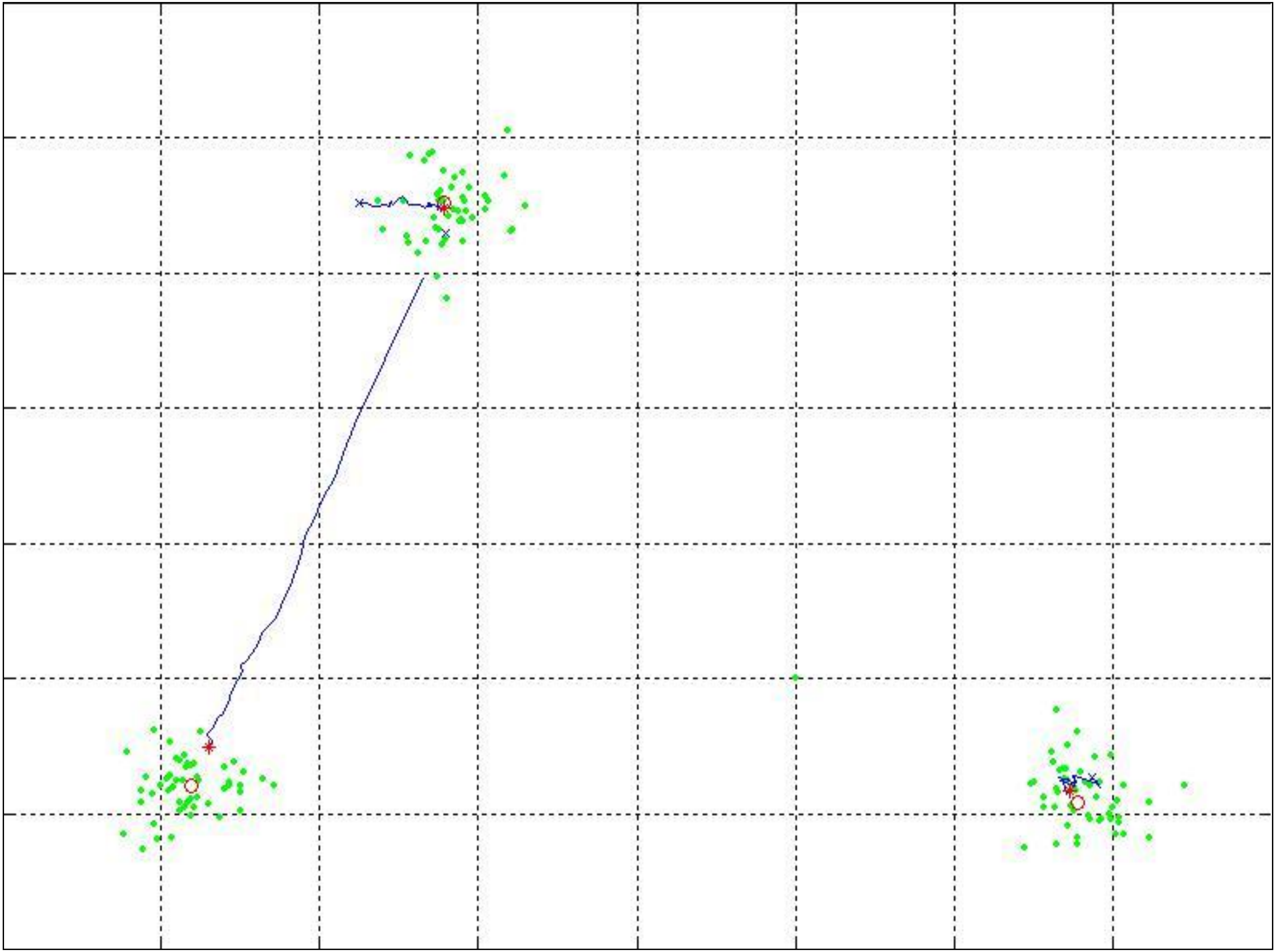
Jika pola Y bukan termasuk dalam cluster yang ada, bagaimana cara menolak?

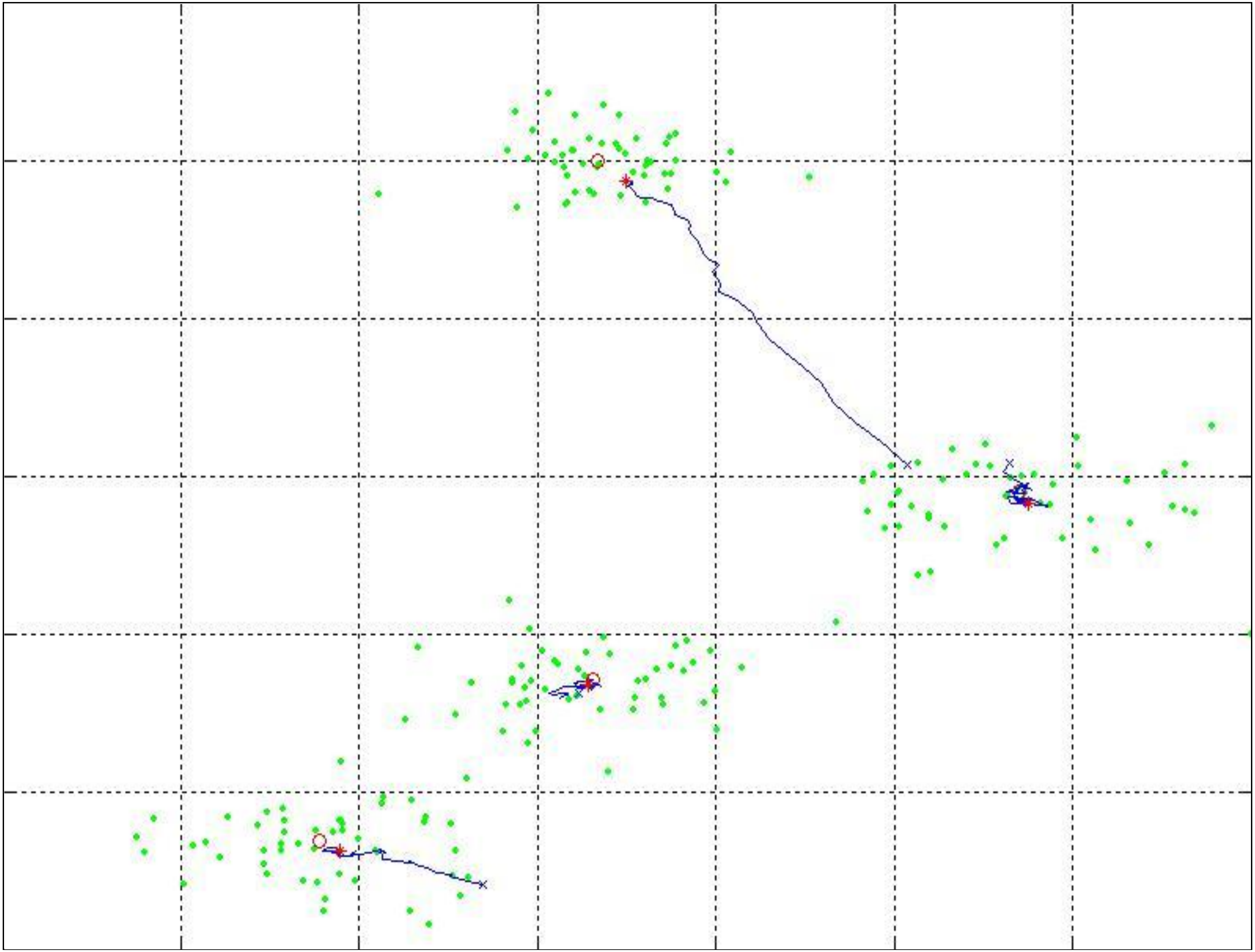
% Initialisation of competitive learning

```
p = 2;                                   % Dimension of input = p
m = 3;                                   % Number of clusters = m
clst = randn(p, m);                      % cluster centroids
Nk = 50;                                 % points per cluster
N = m*Nk;                                % total number of points
sprd = 0.1;                              % a relative spread of the Gaussian "blobs"
X = zeros(p,N+m);                        % X is p by m+N input data matrix
wNk = ones(1, Nk);
 for k = 1:m                             % generation of m Gaussian "blobs"
    xc = clst(:,k);
    X(:,(1+(k-1)*Nk):(k*Nk))=sprd*randn(p,Nk)+xc(:,wNk);
End

[xc k] = sort(rand(1,N+m));
X = X(:, k);                             % input data is shuffled randomly
winit = X(:,1:m)';                       % Initial values of weights
X = X(:,m+1:N+m);                        % Data matrix is p by N
```

% Competitive learning

```
W = winit;
V = zeros(N, m, p);          % to store all weights
V(1,:,:) = W ;
wnm = ones(m,1) ;
eta = 0.08 ;                 % learning gain
deta = 1-1/N ;               % learning gain decaying factor
for k = 1:N                  % main training loop
    xn = X(:,k)';
    xmw = xn(wnm,:)-W;          % the current vector is compared to all weight vectors
    [win jwin] = min(sum((xmw.^2),2));
    W(jwin,:) = W(jwin,:) + eta*xmw(jwin,:);   % update weights of the winning neurons
    V(k,:,:) = W;
    eta = eta*deta ;
End

plot (X(1,:),X(2,:),'g.',  clst(1,:),clst(2,:),'ro',  winit(:,1),winit(:,2),'bx' , ...
V(:,:,1), V(:,:,2), 'b', W(:,1),W(:,2)  , 'r*'), grid
```





Data Training 1

- Jumlah data: 10 ribu records
- Dimensi: 3 ribu
- Jumlah kelas: 4
- Semua data memiliki kelas yang valid

- **Metode**: Supervised atau Unsupervised?
- Bisa Supervised maupun Unsupervised

Data Training 2

- Jumlah data: 60 juta records
- Dimensi: 200
- Jumlah kelas: tidak diketahui

- **Metode**: Supervised atau Unsupervised?
- Unsupervised
- Supervised tidak bisa dipakai

Supervised vs Unsupervised

- **Supervised → Klasifikasi** → User mengajari ANN
  - Jumlah kelas diketahui
  - Tersedia data latih yang VALID
- **Unsupervised → Clustering** → ANN memberitahu user
  - Jumlah kelas bisa tidak diketahui
  - Tidak tersedia data latih yang VALID


Menangis?
Berjalan?
Arah?
Berbahasa?
Logika?
Multiple Intelligence!
Supervised dan Unsupervised!!

Molecular Memory

- Setiap sel punya memory?
- Transplantasi organ tubuh???
- “The Eye”

Referensi

[SUY08] Suyanto, 2008, “Soft Computing: Membangun Mesin Ber-IQ Tinggi”, Informatika, Bandung Indonesia. ISBN: 978-979-1153-49-2.

[TET01] Tettamanzi A., Tomassini M., ”Soft Computing”. Springer-Verlag Berlin Heidelberg, 2001. Printed in Germany.

[MIT97] Mitchell M. Tom. 1997. ”Machine Learning”. McGraw-Hill International Editions. Printed in Singapore.