MULTI LAYER PERCEPTRON BACKPROPAGATION

Pengantar MLP

- **Multi Layer Perceptron** (MLP) merupakan perkembangan dari perceptron
  - Kemampuan klasifikasi yang jauh lebih baik (klasifikasi data yang ***non-linearly separable***).
  - Mampu digunakan untuk keperluan ***classification*** dan ***regression***.
- **MLP** merupakan salah satu model JST yang paling sering digunakan, karena:
  - Arsitekturnya simpel
  - Algoritma pembelajarannya tergolong mudah diimplementasikan
  - Cenderung memiliki akurasi yang baik

Implementasi MLP

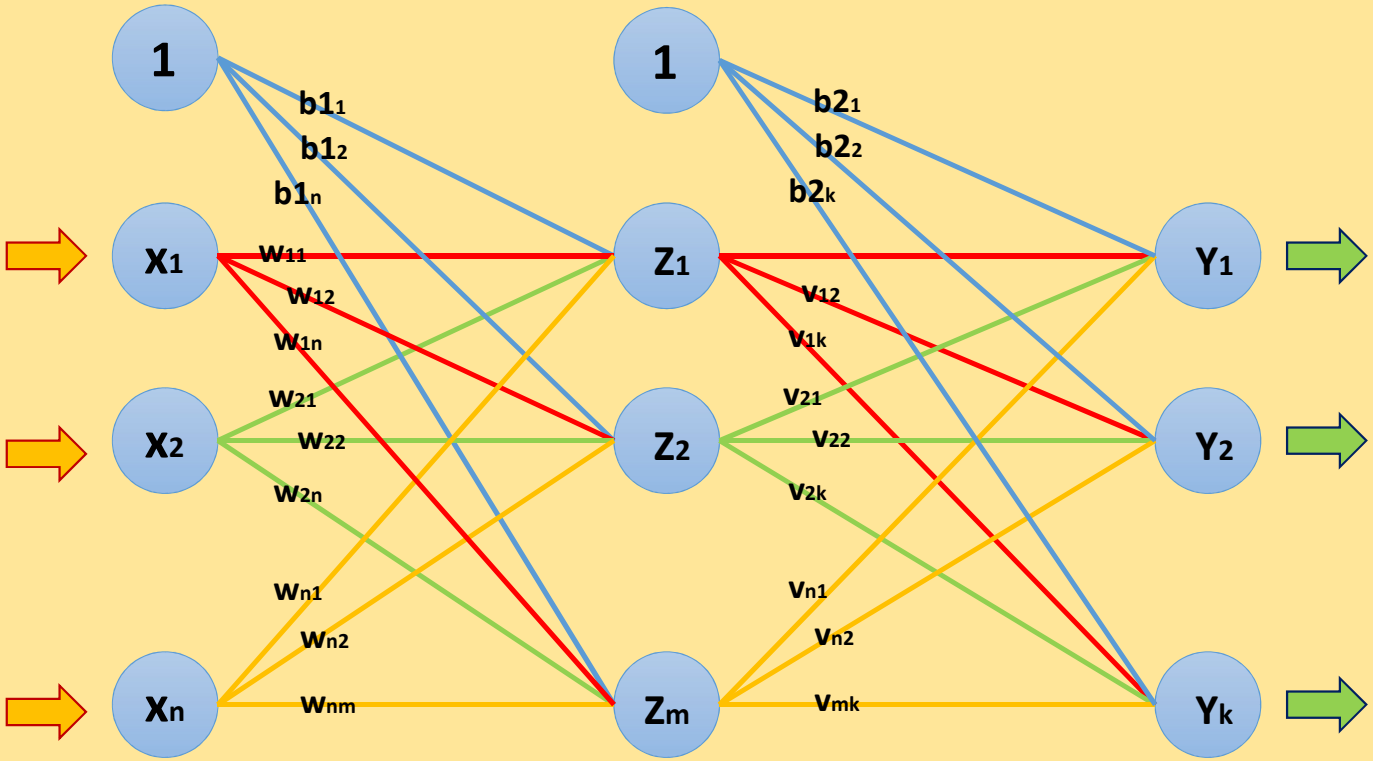
Classification

- Email classification
- Object Detection
- Object Classification
- Speech Recognition
- Face Recognition

Regression

- Gold Price forecasting
- Weather prediction
- Rate of exchange prediction

Arsitektur MLP

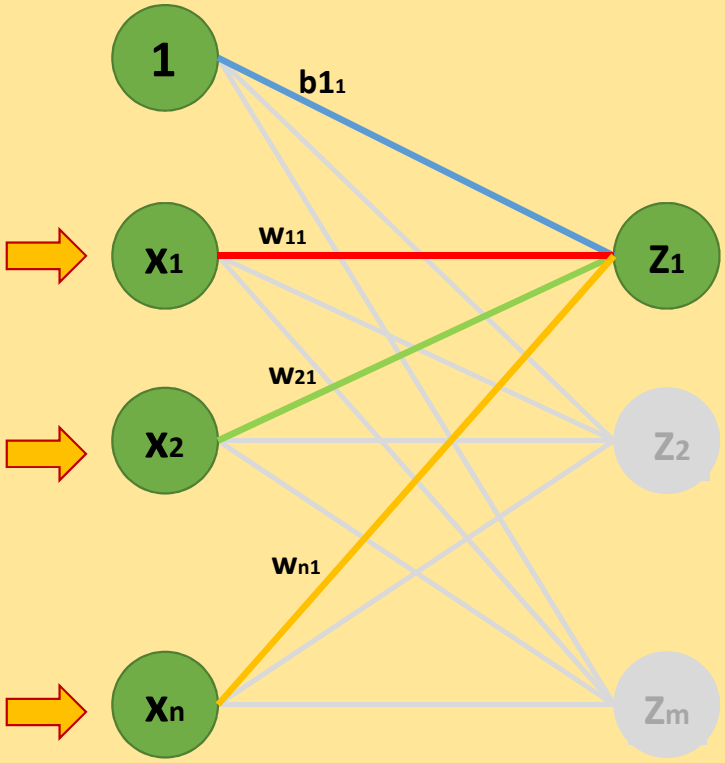


MLP - Perhitungan Maju

- Prosedur perhitungan maju pada **MLP** sama dengan perhitungan maju pada **perceptron**.
- Perbedaannya hanya terletak pada fungsi aktivasi yang digunakan.
- MLP menggunakan fungsi aktivasi **non-linear** yang disebut **fungsi sigmoid**.

$$y = \frac{1}{1 + e^{-net}}$$

MLP – Perhitungan Maju

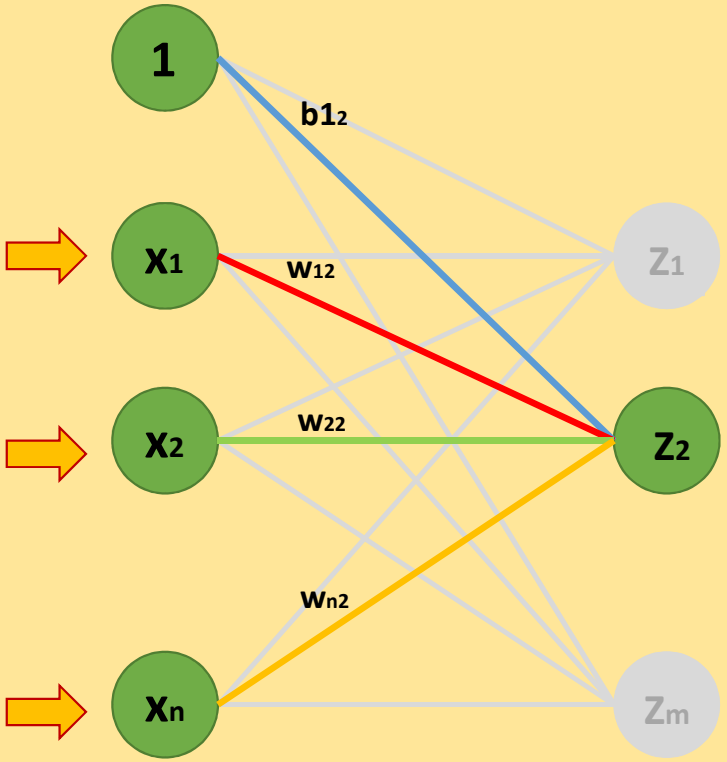


$$z = \frac{1}{1 + exp^{-net}}$$

dimana:

$$net = \sum_{i}^{N} w_{i1} + b1_1$$

MLP – Perhitungan Maju

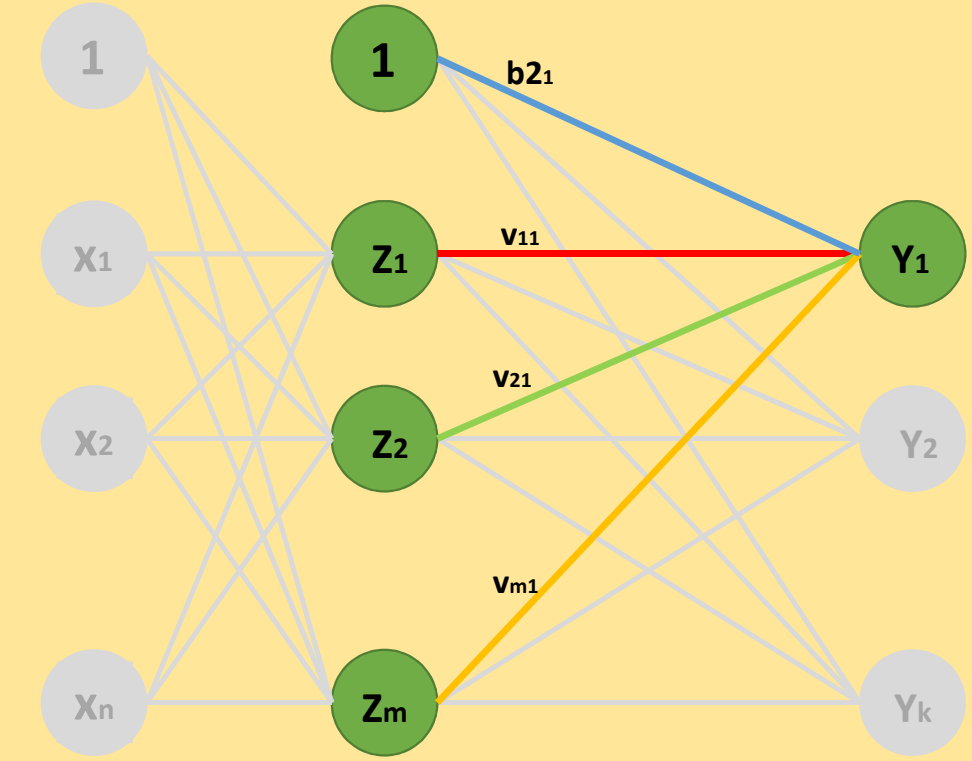


$$z = \frac{1}{1 + exp^{-net}}$$

dimana:

$$net = \sum_{i}^{N} w_{i2} + b1_1$$

MLP – Perhitungan Maju

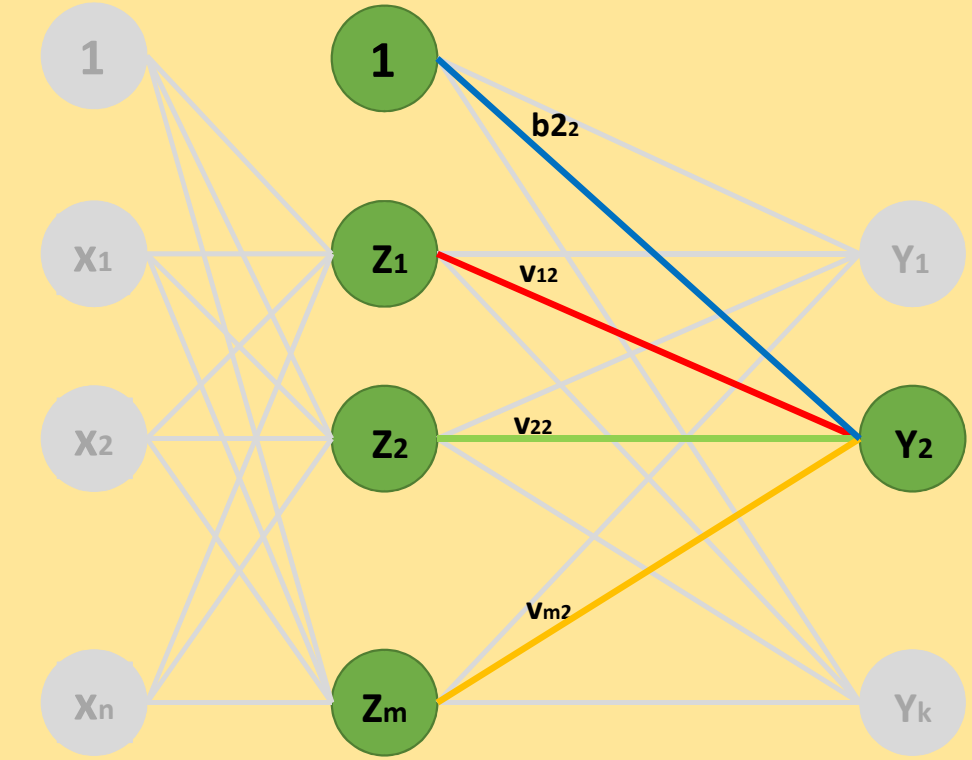


$$y = \frac{1}{1 + exp^{-net}}$$

dimana:

$$net = \sum_{i=1}^{m} z_i v_{i1} + b_{21}$$

MLP – Perhitungan Maju



$$y = \frac{1}{1 + exp^{-net}}$$

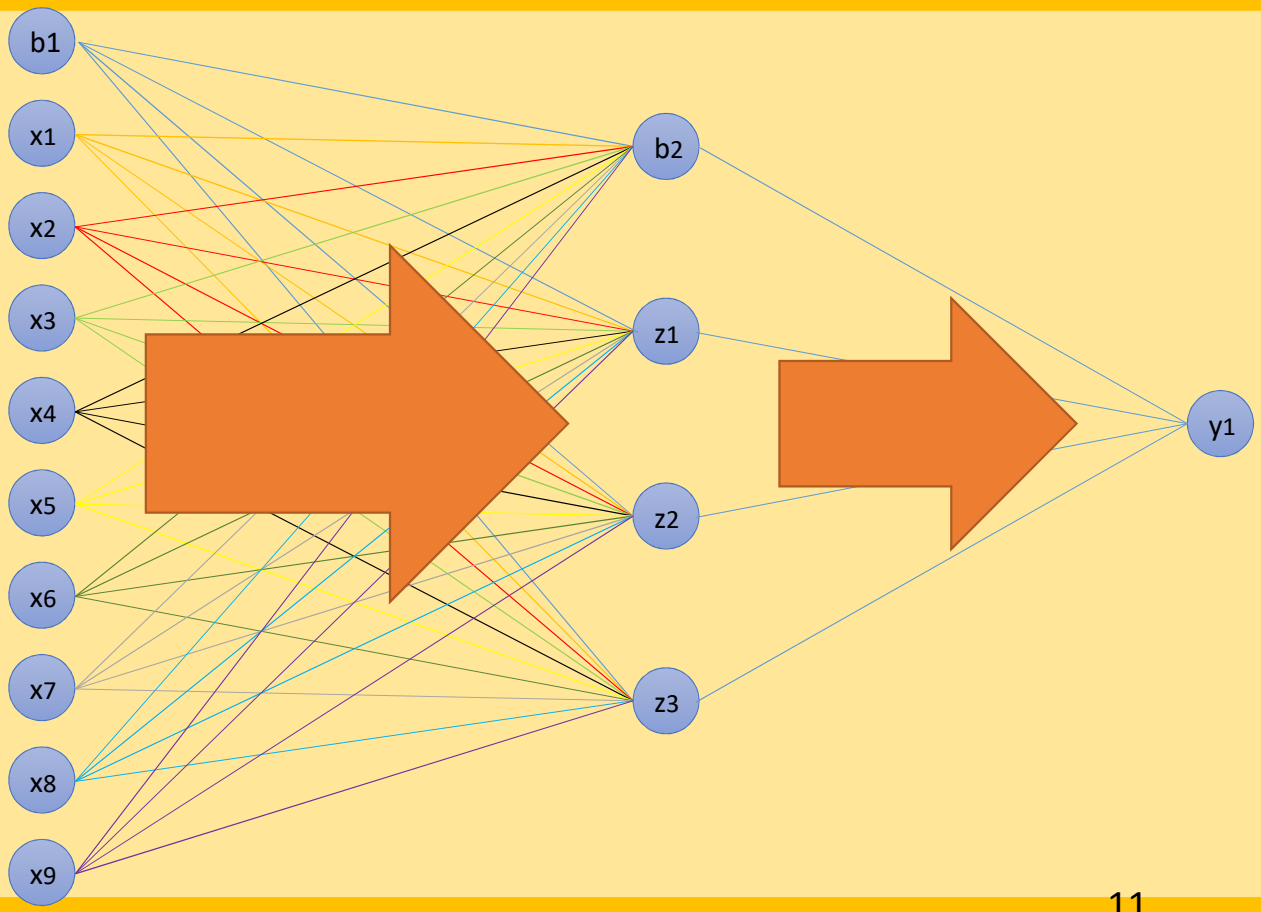
dimana:

$$net = \sum_{i=1}^{m} z_i v_{i2} + b_{22}$$

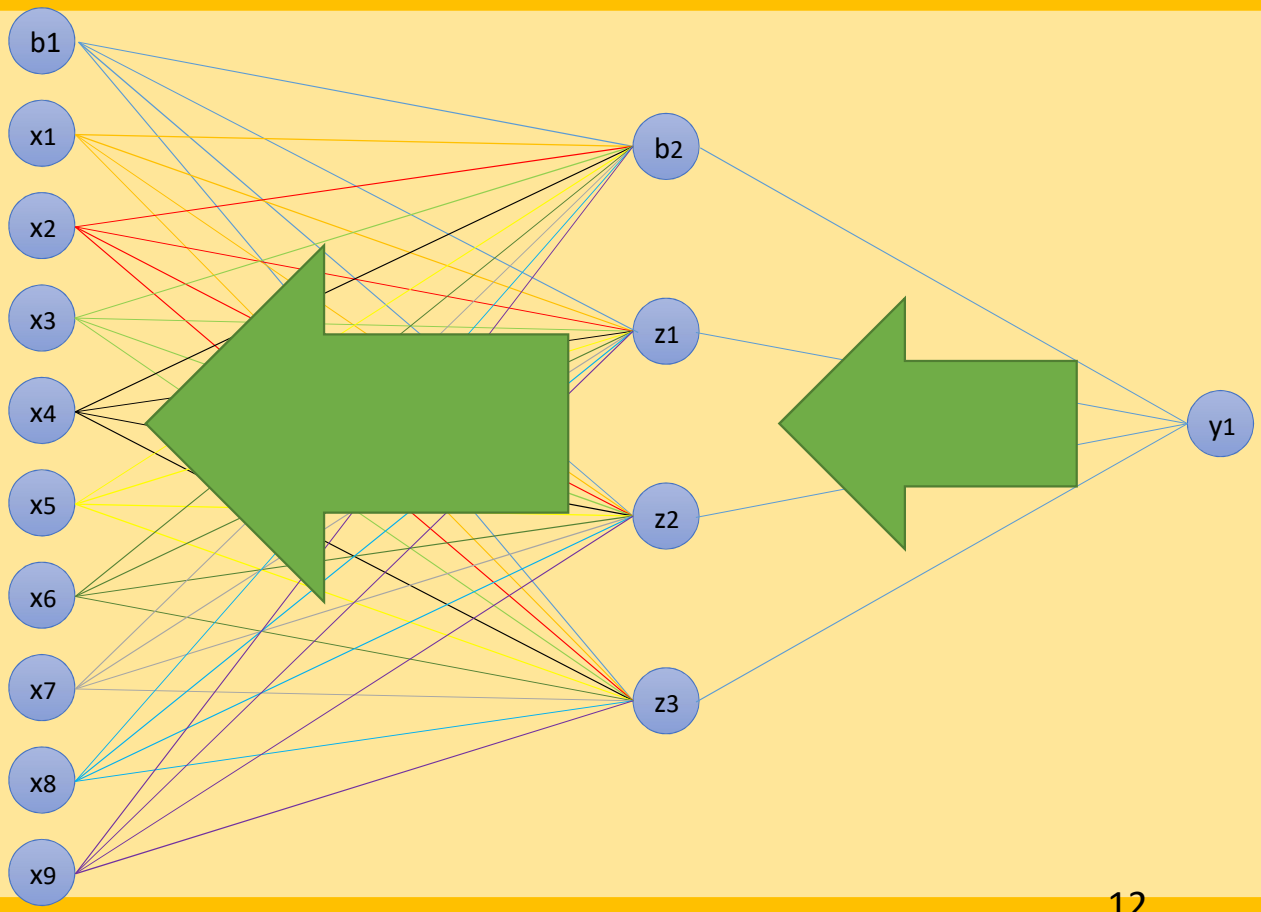
Algoritma Learning MLP

- **Algoritma *learning* MLP** disebut dengan ***Backpropagation*** yang merupakan pengembangan dari algoritma *learning* pada Perceptron.
- Ciri dari metode ini adalah meminimalkan error pada output yang dihasilkan oleh jaringan.
- Pada algoritma *Backpropagation*, terjadi dua proses perhitungan, yakni:
  1. Proses perhitungan maju (***feedforward***)
  2. Proses perhitungan mundur (***backpropagation***)

Backpropagation – Perhitungan Maju



Backpropagation – Perhitungan Mundur



Proses belajar & Pengujian

- Penggunaan Back Propagation Network terdiri dari 2 tahap:
  - Tahap belajar atau pelatihan, di mana pada tahap ini pada BPN diberikan sejumlah data pelatihan dan target
  - Tahap pengujian atau penggunaan, pengujian dan penggunaan dilakukan setelah BPN selesai belajar

Tahap Belajar atau Pelatihan

- Pada intinya, pelatihan dengan metode backpropagation terdiri dari tiga langkah, yaitu:
  - Data dimasukkan ke input jaringan (feedforward)
  - Perhitungan dan propagasi balik dari error yang bersangkutan (*backpropagation)*
  - Pembaharuan (adjustment) bobot dan bias.

Tahap Belajar atau Pelatihan

- Saat umpan maju (feedforward), setiap unit input (X_i) akan menerima sinyal input dan akan menyebarkan sinyal tersebut pada tiap hidden unit (Z_j).
- Setiap hidden unit kemudian akan menghitung aktivasinya dan mengirim sinyal (z_j) ke tiap unit output.
- Kemudian setiap unit output (Y_k) juga akan menghitung aktivasinya (y_k) untuk menghasilkan respons terhadap input yang diberikan jaringan.

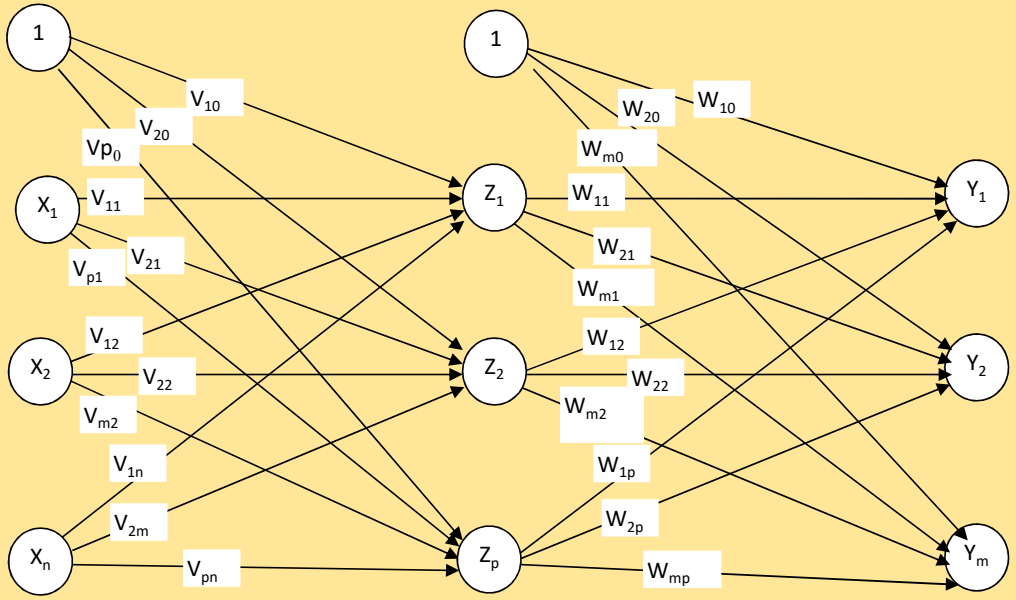
Tahap Belajar atau Pelatihan

- Saat proses pelatihan (training), setiap unit output membandingkan aktivasinya (y_k) dengan nilai target (t_k) untuk menentukan besarnya error.
- Berdasarkan error ini, dihitung faktor δ_k, di mana faktor ini digunakan untuk mendistribusikan error dari output ke layer sebelumnya.
- Dengan cara yang sama, faktor δ_j juga dihitung pada hidden unit Z_j, di mana faktor ini digunakan untuk memperbaharui bobot antara hidden layer dan input layer.
- Setelah semua faktor δ ditentukan, bobot untuk semua layer diperbaharui.

Arsitektur JST Backpropagation

Contoh Arsitekstur JST Backpropagation dengan:

- n unit masukkan
- p unit layer tersembunyi
- m unit keluaran

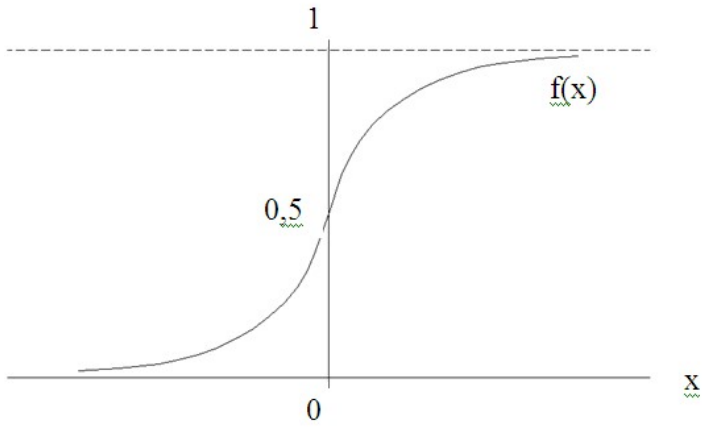


Layer Masukkan Layer tersembunyi Layer Keluaran

Fungsi Aktifasi

Fungsi aktifasi yang digunakan pada backpropagation yaitu sigmoid biner dan sigmoid bipolar

Fungsi sigmoid biner
Grafik

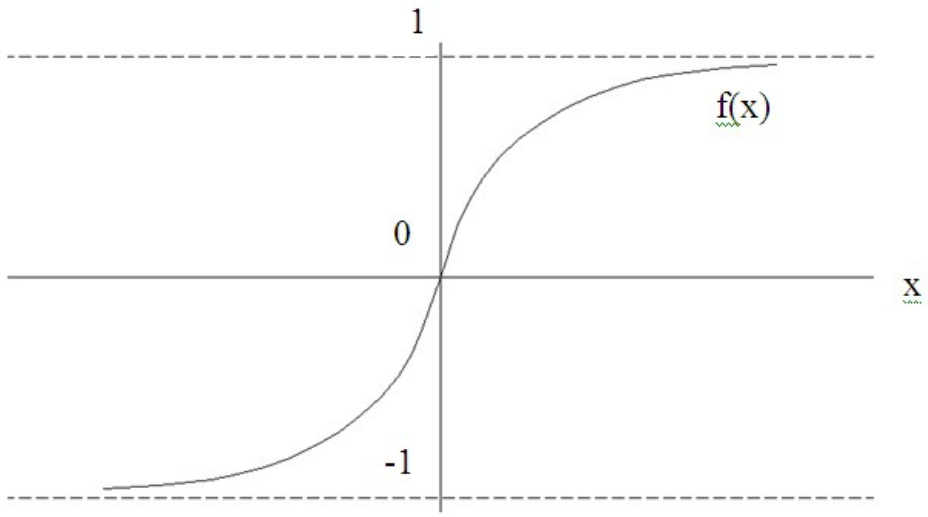


Mempunyai Range 0 sampai dengan 1
Fungsi:

$$f(x) = \frac{1}{1 + e^{-x}}$$

Fungsi Aktifasi

Fungsi sigmoid bipolar
Grafik

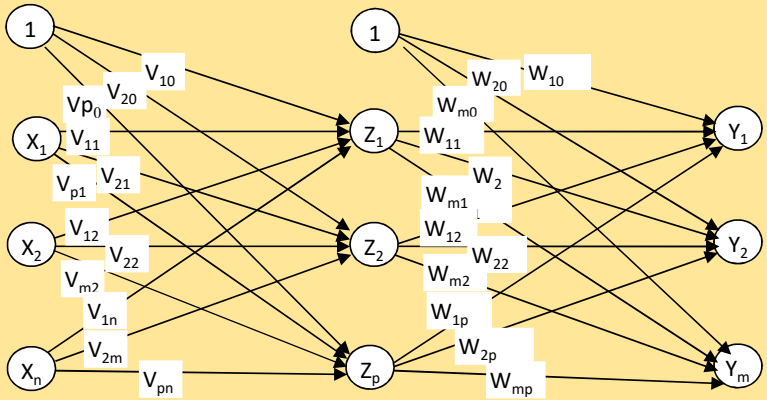


Mempunyai Range -1 sampai dengan 1
Fungsi:

$$f(x) = \frac{2}{1 + e^{-x}} - 1$$

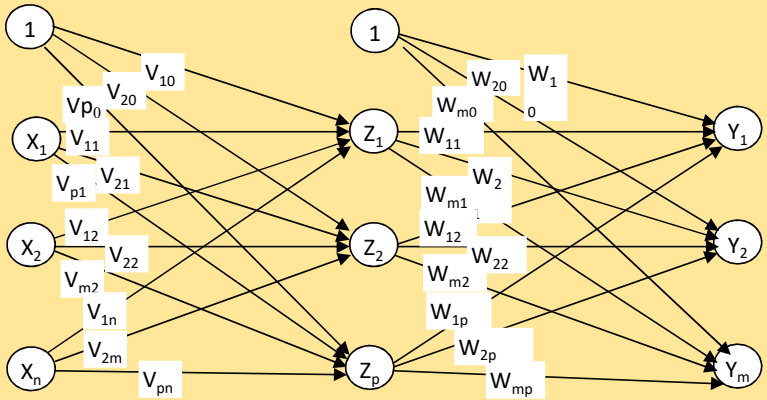
Algoritma Pelatihan Backpropagation

Algoritma Pelatihan Backpropagation dengan satu layer tersembunyi dan dengan menggunakan fungsi aktivasi sigmoid biner



- **Langkah 0 :** Inisialisasi semua bobot dengan bilangan acak kecil.
- **Langkah 1 :** Jika kondisi penghentian belum terpenuhi, lakukan langkah 2 sampai dengan 8.
- **Langkah 2:** Untuk setiap pasang data pelatihan, lakukan langkah 3 sampai dengan 8

Algoritma Pelatihan Backpropagation



Fase I: Propagasi Maju

- **Langkah 3**

Tiap unit masukkan menerima sinyal dan meneruskan ke unit tersembunyi

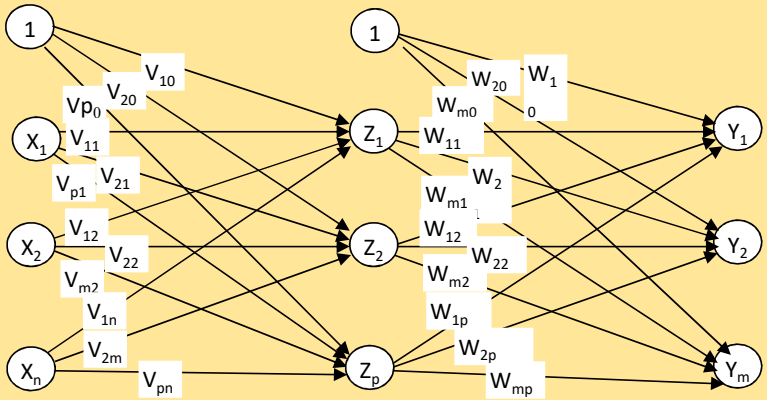
- **Langkah 4**

Hitung semua keluaran di unit tersembunyi (Z_j):

$$z\_net_j = v_{j0} + \sum_{i=1}^{n} x_i v_{ji}$$

$$z_j = f\left(z_{net_j}\right) = \frac{1}{1 + e^{-z\_net_j}}$$

Algoritma Pelatihan Backpropagation



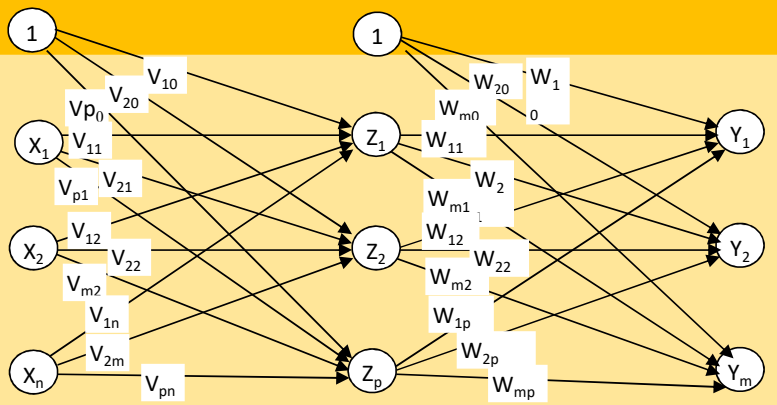
- **Langkah 5**

  Hitung semua jaringan di unit keluaran (y_k)

$$y\_net_k = w_{k0} + \sum_{j=1}^{p} x_j v_{kj} \qquad y = w0 + \sum zw$$

$$y_k = f(y\_net_k) = \frac{1}{1 + e^{-y\_net_k}}$$

Algoritma Pelatihan Backpropagation



Fase II : Propagasi Mundur

- **Langkah 6**

  Hitung factor δ unit keluaran berdasarkan kesalahan setiap unit keluaran y_k (k=1,2,3….)

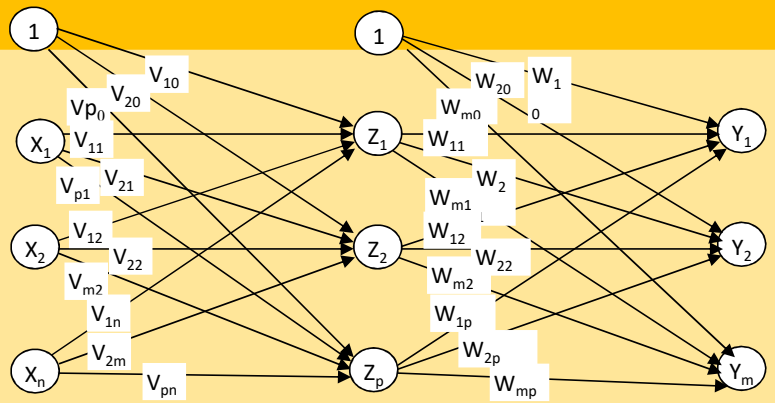
  $$\delta_k=(t_k-y_k)\, f'(y\_net_k)= (t_k-y_k)\, y_k\, (1-y_k)$$

  δ_k meruapakan unit kesalahan yang akan dipakai dalam perubahan bobot layer dibawahnya (lankah 7)

  Hitung suku perubahan bobot W_{kj} dengan laju perubahan α

  $$\Delta w_{kj}= \alpha\, \delta_k\, z_j \quad ; k=1,2,3,\ldots,m \; ; j=0,1,2,\ldots,p$$

Algoritma Pelatihan Backpropagation



• **Langkah 7**

Hitung factor δ unit tersembunyi berdasarkan kesalahan di setiap unit tersembunyi z_j (j=1,2,3,…,p)

$$\delta\_net_j = \sum_{k=1}^{m} \delta_k w_{kj}$$

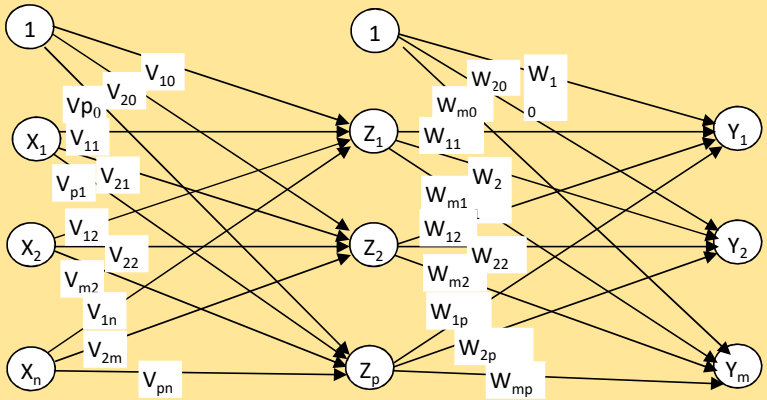
Faktor unit tersembunyi

$$\delta_j = \delta\_net_j\, f'(z\_net_j) = \delta\_net_j\, z_j\, (1-z_z)$$

Hitung suku perubahan bobot v_{ji}

$$\Delta v_{ji} = \alpha\, \delta_j x_i \quad ; j=1,2,\ldots,p \quad ; \quad i=0,1,2,\ldots,n$$

Algoritma Pelatihan Backpropagation



Fase III : Perubahan Bobot

- **Langkah 8**

Perubahan bobot garis yang menuju unit keluaran

$$\mathbf{w_{kj}\ (baru) = w_{kj}\ (lama) + \Delta w_{kj}}$$

Perubahan bobot garis yang menuju ke unit tersembunyi

$$\mathbf{V_{ji}\ (baru) = v_{ji}\ (lama) + \Delta v_{ji}}$$

Proses belajar secara detail

- Step 9 : Memeriksa stopping condition
  - Jika stop condition telah terpenuhi, maka pelatihan jaringan dapat dihentikan.

Stopping Condition

- Untuk menentukan stopping condition terdapat dua cara yang biasa dipakai, yaitu:
  - Membatasi iterasi yang ingin dilakukan.
    - Misalnya jaringan akan dilatih sampai iterasi yang ke-500.
    - Yang dimaksud dengan satu iterasi adalah perulangan step 3 sampai step 8 untuk semua training data yang ada.
  - Membatasi error.
    - Misalnya menentukan besar Mean Square Error antara output yang dikehendaki dan output yang dihasilkan oleh jaringan.

Mean Square Error

- Jika terdapat sebanyak m training data, maka untuk menghitung Mean Square Error digunakan persamaan berikut:
  - $MSE=0,5 \times \{(t_{k1}-y_{k1})^2+ (t_{k2}-y_{k2})^2+\ldots+ (t_{km}-y_{km})^2\}$

Tahap pengujian & Penggunaan

- Setelah pelatihan selesai, BP dianggap telah memahami jaringan sehingga apabila jaringan diberi input tertentu, jaringan akan menghasilkan output seperti yang diharapkan.
- Cara mendapatkan output tersebut adalah dengan mengimplementasikan metode backpropagation yang sama seperti proses belajar, tetapi hanya pada bagian umpan majunya saja, yaitu dengan langkah-langkah sebagai berikut:

Penghitungan output jaringan

- Step 0: Inisialisasi bobot sesuai dengan bobot yang telah dihasilkan pada proses pelatihan di atas.
- Step 1: Untuk setiap input, lakukan step 3-5.
- Step 2: Untuk setiap input i=1,…,n skalakan bilangan dalam range fungsi aktivasi seperti yang dilakukan pada proses pelatihan di atas.

Penghitungan output jaringan

- Step 4: untuk j=1,…,p:

$$z\_net_j = v_{j0} + \sum_{i=1}^{n} x_i v_{ji}$$

$$z_j = f\left(z_{net_j}\right) = \frac{1}{1 + e^{-z\_net_j}}$$

Penghitungan output jaringan

- Step 5 : Untuk k=1,…,m:

$$y\_net_k = w_{k0} + \sum_{j=1}^{p} x_j v_{kj}$$

$$y_k = f(y\_net_k) = \frac{1}{1 + e^{-y\_net_k}}$$

Penghitungan output jaringan

- Variabel y_k adalah output yang masih dalam skala menurut range fungsi aktivasi.
- Untuk mendapatkan nilai output yang sesungguhnya, y_k harus dikembalikan seperti semula.

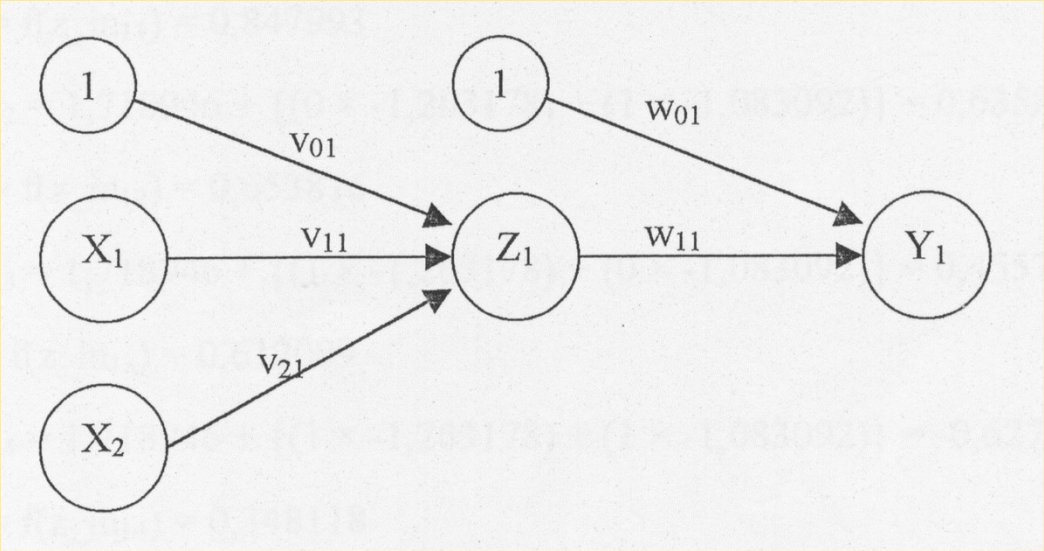
Contoh aplikasi BP

- Misalkan, jaringan terdiri dari 2 unit input, 1 hidden unit (dengan 1 hidden layer), dan 1 unit output.
- Jaringan akan dilatih untuk memecahkan fungsi XOR.
- Fungsi aktivasi yang digunakan adalah sigmoid biner dengan nilai learning rate (α) = 0,01 dan nilai σ =1.

$$y = f(x) = \frac{1}{1 + e^{-\sigma x}}$$

dengan $f' = \sigma f(x)[1 - f(x)]$

- Arsitektur jaringan yang akan dilatih adalah sebagai berikut:



- Training data yang digunakan terdiri dari 4 pasang input-output, yaitu:

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	0

- Sebelum pelatihan, harus ditentukan terlebih dahulu stopping conditionnya.
- Misalnya dihentikan jika error telah mencapai 0,41.

Langkah-langkah pelatihan

- Step 0: Misalnya inisialisasi bobot dan bias adalah:

  v_{01}=1,718946

  v_{11}=-1,263178

  v_{21}=-1,083092

  w_{01}=-0,541180

  w_{11}=0,543960

- Step 1: Dengan bobot di atas, tentukan error untuk training data secara keseluruhan dengan Mean Square Error:
  - z_in_{11}=1,718946+{(0 x -1,263178)+(0 x-1,083092)}=1,718946
  - z_{11}=f(z_in_{11})=0,847993
  - z_in_{12}=1,718946+{(0 x-1,263178)+(1 x -1,083092)}=0,635854
  - z_{12}=f(z_in_{12})=0,653816
  - z_in_{13}=1,718946+{(1 x- 1,263178)+(0 x- 1,083092)}=0,455768
  - z_{13}=f(z_in_{13})=0,612009
  - z_in_{14}=1,718946+{(1 x -1,263178)+(1 x -1,083092)=-0,627324
  - z_{14}=f(z_in_{14})=0,348118
  - di mana indeks z_{jn} berarti hidden unit ke-j dan training data ke-n.

- y_in_{11}=-0,541180+(0,847993 x 0,543960)=0,079906
- $y_{11}=f(y\_in_{11})$=0,480034
- y_in_{12}=-0,541180+(0,653816 x 0,543960)=-0,185530
- $y_{12}=f(y\_in_{12})$=0,453750
- y_in_{13}=-0,541180+(0,612009 x 0,543960)=0,208271
- $y_{13}=f(y\_in_{13})$=0,448119
- y_in_{14}=-0,541180+(0,348118 x 0,543960)=-0,351818
- $y_{14}=f(y\_in_{14})$=0,412941

- Maka $E=0,5 \times \{(0-0,480034)^2 + (1-0,453750)^2) + (1-0,448119)^2 + (0-0,412941)^2\}=0,501957$

- Step2. Karena error masih lebih besar dari 0,41 maka step 3-8 dijalankan.
- Step 3.

  x1=0; x2=0 (iterasi pertama, pada data pertama)
- Step 4.
  - z_in_1=1,718946+{(0x-1,263126)+(0x-1,083049)}=1,718946.
  - z_1=f(z_in_1)=0,847993
- Step 5.
  - y_in_{11}=-0,541180+(0,847993x0,543960)=0,079906
  - y_{11}=f(y_in_{11})=0,480034
- Step 6.
  - δ_1=(0-0,480034)f ’(0,079906)=-0,119817
  - Δw_{11}=0,01x-0,119817x0,847993=-0,001016
  - Δw_{01}=0,01x-0,119817=-0,00119817

$$y\_net_k = w_{k0} + \sum_{j=1}^{p} x_j v_{kj}$$

$$y_k = f(y\_net_k) = \frac{1}{1 + e^{-y\_net_k}}$$

$\delta_k=(t_k-y_k)\, f'(y\_net_k)= (t_k-y_k)\, y_k\, (1-y_k)$

$\Delta w_{kj}= \alpha\, \delta_k\, z_j$

$$\delta\_net_j = \sum_{k=1}^{m} \delta_k w_{kj}$$

- Step 7.
  - δ_in_1=-0,00119817x0,543960=-0,00065176
  - δ_1=-0,00065176xf'(1,718946) = δ_in_1 x z x (1-z)=-0,00008401
  - Δv_{11}=0,01x-0,00008401x0=0
  - Δv_{21}=0,01x-0,00008401x0=0
  - Δv_{01}=0,01x-0,00008401=-0,0000008401

$\delta_j = \delta\_net_j\ f'(z\_net_j) = \delta\_net_j\ z_j\ (1-z_z)$

- Step 8.
  - w_{01}(baru)=-0,541180+(-0,00119817)=-0,542378
  - w_{11}(baru)=0,543960+(-0,001016)=0,542944
  - v_{01}(baru)=1,718946+(-0,0000008401)=1,718862
  - v_{11}(baru)=-1,263178+0=-1,263178
  - v_{21}(baru)=-1,083092+0=-1,083092

  - Saat ini v_{11} dan v_{12} masih belum berubah karena kedua inputnya =0. Nilai v_{01} dan v_{02} baru berubah pada iterasi pertama untuk training data yang kedua

- Setelah step 3-8 untuk training data pertama dijalankan, selanjutnya kembali lagi ke step 3 untuk training data yang kedua (x_1=0 dan x_2=1).
- Langkah yang sama dilakukan sampai pada training data yang keempat.
- Bobot yang dihasilkan pada iterasi pertama, training data ke-2,3, dan 4 adalah:

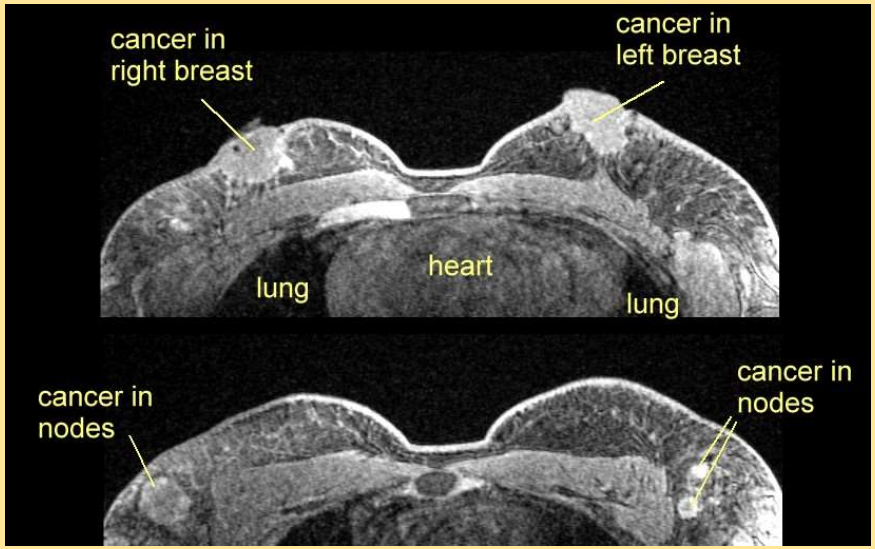
- Training pada data ke-2:
  - w_{01}=-0,541023
  - w_{11}=0,543830
  - v_{01}=1,718862
  - v_{11}=-1,263178
  - v_{21}=-1,083092
- Training pada data ke-3:
  - w_{01}=-0,539659
  - w_{11}=0,544665
  - v_{01}=1,719205
  - v_{11}=-1,263002
  - v_{21}=-1,082925
- Training pada data ke-4:
  - w_{01}=-0,540661
  - w_{11}=0,544316
  - v_{01}=1,719081
  - v_{11}=-1,263126
  - v_{21}=-1,083049

- Setelah sampai pada training data ke-4, maka iterasi pertama selesai.
- Berikutnya, pelatihan sampai pada step9, yaitu memeriksa stopping condition dan kembali pada step 2.
- Demikian seterusnya sampai stopping condition yang ditentukan terpenuhi.
- Setelah pelatihan selesai, bobot yang didapatkan adalah:
  - v_{01}=12,719601
  - v_{11}=-6,779127
  - v_{21}=-6,779127
  - w_{01}=-5,018457
  - w_{11}=5,719889

- Jika ada input baru (data uji), misalnya x_1=0,2 dan x_2=0,9 maka outputnya dapat dicari dengan langkah umpan maju sebagai berikut:
- Step 0. Bobot yang dipakai adalah bobot hasil pelatihan di atas.
- Step 1. Perhitungan dilakukan pada step 2-4
- Step 2. Dalam contoh ini, bilangan telah berada dalam interval 0 sampai dengan 1, jadi tidak perlu diskalakan lagi.
- Step 3.
  - z_in_1=12,719601+{(0,2x-6,779127)+(0,9x-6,779127)}=5,262561
  - z_1=f(5,262561)=0,994845
- Step 4.
  - y_in_1=-5,018457+(0,994845x5,719889)=0,671944
  - y_1=f(0,671944)=**0,661938**
- Jadi jika input x_1=0,2 dan x_2=0,9; output yang dihasilkan jaringan adalah 0,661938

Klasifikasi Penderita Kanker

- Sebuah kanker biasanya diidentifikasi/ dianalisis melalui gambar hasil scan MRI.
- Kemudian, diambil beberapa aspek (*feature*) yang menjadi indikasi adanya kanker atau tidak.



Klasifikasi Penderita Kanker

- Data yang digunakan untuk melatih MLP diperoleh dari website http://archive.ics.uci.edu/ml/
- Data tersebut berjumlah 300 buah, dan akan dibagi menjadi 2 bagian, yakni:
  1. Data latih berjumlah 200 buah
  2. Data uji berjumlah 100 buah
- Data tersebut terdiri dari dua kelas, yakni **malignant** (positif kanker) dan **benign** (negatif kanker).
- **Malignant** disimbolkan dengan biner 1, sedangkan **benign** disimbolkan dengan biner 0.

Klasifikasi Penderita Kanker

No.	Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses	C LASS (TARGET)
1	5	8	7	7	10	10	5	7	1	**0**
2	8	7	6	4	4	10	5	1	1	**0**
3	4	2	3	5	3	8	7	6	1	**0**
4	5	10	10	3	7	3	8	10	2	**0**
5	5	10	10	5	4	5	4	4	1	**0**
6	1	5	8	6	5	8	7	10	1	**0**
⋮										
195	1	1	1	1	1	1	2	1	1	**1**
196	5	1	1	1	2	1	1	1	1	**1**
197	3	1	1	1	2	1	2	1	1	**1**
198	2	1	1	1	2	1	3	1	1	**1**
199	1	3	1	2	2	2	5	3	2	**1**
200	3	2	1	2	2	1	3	1	1	**1**

Data Latih Kanker Payudara

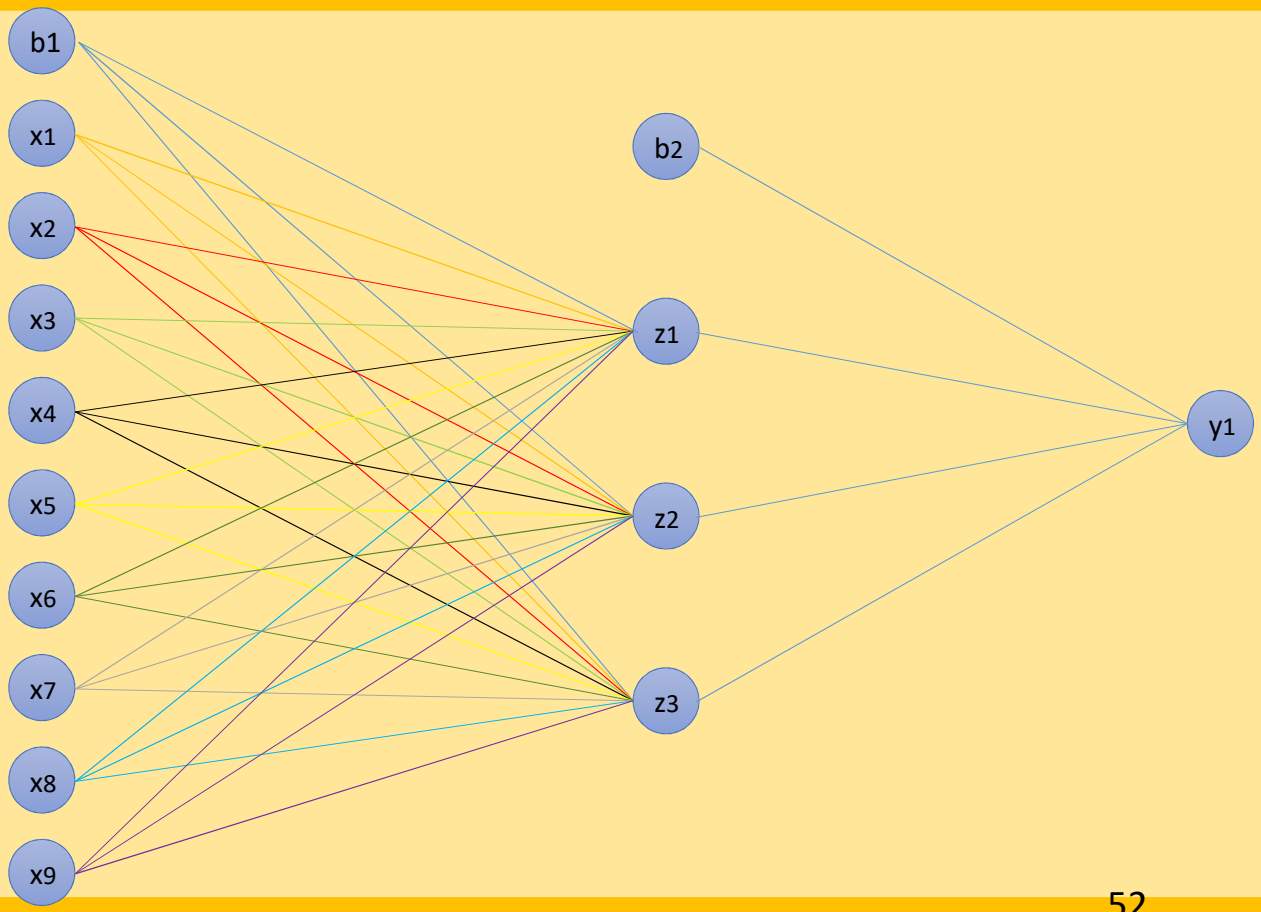
No.	Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses	C LASS (TARGET)
1	5	8	7	7	10	10	5	7	1	**0**
101	1	1	1	1	2	1	3	1	1	**1**
2	8	7	6	4	4	10	5	1	1	**0**
102	2	1	1	1	3	1	2	1	1	**1**
3	4	2	3	5	3	8	7	6	1	**0**
103	5	1	1	1	2	1	1	1	1	**1**
⋮										
98	5	3	3	3	2	3	4	4	1	**0**
198	2	1	1	1	2	1	3	1	1	**1**
99	5	8	8	8	5	10	7	8	1	**0**
199	1	3	1	2	2	2	5	3	2	**1**
100	7	5	10	10	10	10	4	10	3	**0**
200	3	2	1	3	2	1	3	1	1	**1**

Desain Arsitektur MLP

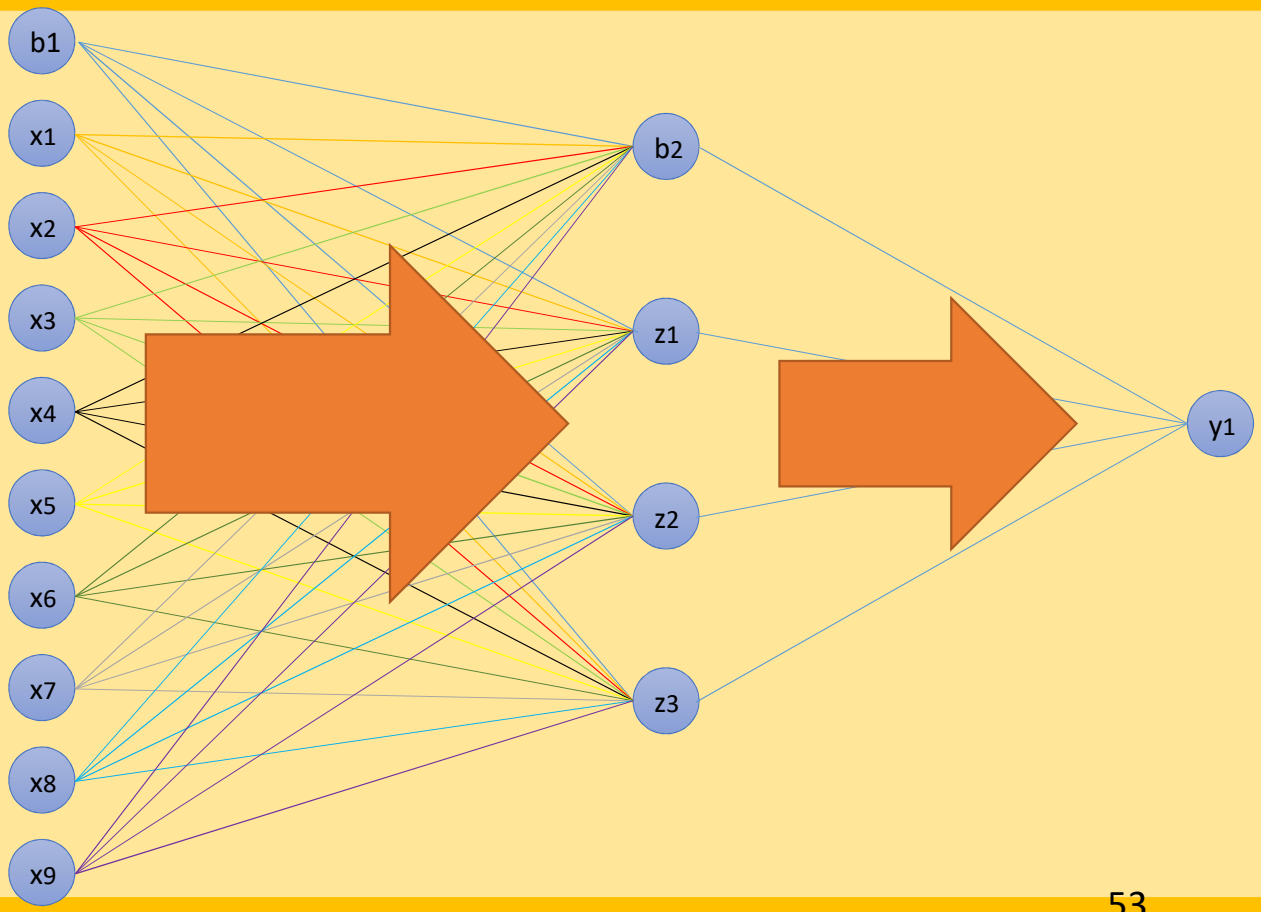
Contoh :

- Input layer : **9 neuron**
- Output layer : **1 neuron**
- Hidden layer :
  a. Skenario 1 → **3 neuron**
  b. Skenario 2 → **6 neuron**
  c. Skenario 3 → **9 neuron**
  d. Skenario 4 → **12 neuron**

Desain Arsitektur MLP – Skenario 1



Backpropagation – Perhitungan Maju



Backpropagation – Perhitungan Mundur