Klasifikasi dengan Naive Bayes

Business Intelligent

Naive Bayes

- Simple Naive Bayesian Classifier merupakan salah satu metode pengklasifikasi berpeluang sederhana yang berdasarkan pada penerapan Teorema Bayes dengan asumsi antar variabel penjelas saling bebas (independen).
- Algoritma ini memanfaatkan metode probabilitas dan statistik yang dikemukakan oleh ilmuwan Inggris Thomas Bayes, yaitu memprediksi probabilitas di masa depan berdasarkan pengalaman di masa sebelumnya.

...

- Dua kelompok peneliti, satu oleh Pantel dan Lin, dan yang lain oleh Microsoft Research memperkenalkan metode statistik Bayesian ini pada teknologi anti spam filter.
- Tetapi yang membuat algoritma Bayesian filtering ini popular adalah pendekatan yang dilakukan oleh Paul Graham. Dasar dari teorema naive digunakan dalam pemrograman adalah rumus Bayes berikut ini:

  $P(A|B) = (P(B|A) * P(A))/P(B)$

  Artinya Peluang kejadian A sebagai B ditentukan dari peluang B saat A, peluang A, dan peluang B.

Penerapan Naive Bayes

- Untuk klasifikasi Dokumen
- Untuk deteksi SPAM atau fitering SPAM

Contoh Kasus (1)

- Misalnya terdapat ingin diketahui apakah suatu objek masuk dalam ketegori dipilih untuk perumahan atau tidak dengan algoritma Naive Bayes Classifier. Untuk menetapkan suatu daerah akan dipilih sebagai lokasi untuk mendirikan perumahan, telah dihimpun 10 aturan.

...

- Ada 4 atribut yang digunakan, yaitu:
  - harga tanah per meter persegi (C1),
  - jarak daerah tersebut dari pusat kota (C2),
  - ada atau tidaknya angkutan umum di daerah tersebut (C3), dan
  - keputusan untuk memilih daerah tersebut sebagai lokasi perumahan (C4).

...

Aturan ke-	Harga tanah (C1)	Jarak dari pusat kota (C2)	Ada angkutan umum (C3)	Dipilih untuk perumahan (C4)
1	Murah	Dekat	Tidak	Ya
2	Sedang	Dekat	Tidak	Ya
3	Mahal	Dekat	Tidak	Ya
4	Mahal	Jauh	Tidak	Tidak
5	Mahal	Sedang	Tidak	Tidak
6	Sedang	Jauh	Ada	Tidak
7	Murah	Jauh	Ada	Tidak
8	Murah	Sedang	Tidak	Ya
9	Mahal	Jauh	Ada	Tidak
10	Sedang	Sedang	Ada	Ya

...

- Probabilitas kemunculan setiap nilai untuk atribut Harga Tanah (C1)

Harga tanah	Jumlah kejadian "Dipilih"		Probabilitas	
	Ya	Tidak	Ya	Tidak
Murah	2	1	2/5	1/5
Sedang	2	1	2/5	1/5
Mahal	1	3	1/5	3/5
Jumlah	*5*	*5*	*1*	*1*

...

- Probabilitas kemunculan setiap nilai untuk atribut Jarak dari Pusat Kota (C2)

Jarak Dari Pusat Kota	Jumlah kejadian "Dipilih"		Probabilitas	
	Ya	Tidak	Ya	Tidak
Dekat	3	0	3/5	0
Sedang	2	1	2/5	1/5
Jauh	0	4	0	4/5
Jumlah	*5*	*5*	*1*	*1*

...

- Probabilitas kemunculan setiap nilai untuk atribut Ada Angkutan Umum (C3)

Angkutan Umum	Jumlah kejadian "Dipilih"		Probabilitas	
	Ya	Tidak	Ya	Tidak
Ada	1	3	1/5	3/5
Tidak	4	2	4/5	2/5
Jumlah	*5*	*5*	*1*	*1*

...

- Probabilitas kemunculan setiap nilai untuk atribut Dipilih untuk perumahan (C4)

Dipilih Untuk Perumahan	Jumlah kejadian "Dipilih"		Probabilitas	
	Ya	Tidak	Ya	Tidak
Jumlah	*5*	*5*	*1/2*	*1/2*

...

- Test Set: Berdasarkan data tersebut, apabila diketahui suatu daerah dengan harga tanah MAHAL, jarak dari pusat kota SEDANG, dan ADA angkutan umum, maka dapat dihitung:

YA = P(Ya|Tanah=MAHAL) . P(Ya|Jarak=SEDANG) .
P(Ya|Angkutan=ADA) . P(Ya)
= 1/5 x 2/5 x 1/5 x 5/10 = 2/125 = 0,008

TIDAK = P(Tidak| Tanah=MAHAL) . P(Tidak|Jarak=SEDANG) .
P(Tidak|Angkutan=ADA) . P(Tidak)
= 3/5 x 1/5 x 3/5 x 5/10 = 2/125 = 0,036

...

- Nilai probabilitas dapat dihitung dengan melakukan normalisasi terhadap likelihood tersebut sehingga jumlah nilai yang diperoleh = 1

Probabilitas Ya = $\frac{0,008}{0,008+0,036} = 0,182.$

Probabilitas Tidak = $\frac{0,036}{0,008+0,036} = 0,818.$

Klasifikasi : TIDAK

Contoh Kasus (2)

- Untuk jenis data harga tanah dan jarak pusat kota yang kontinue, misalnya :

Aturan ke-	Harga tanah (C1)	Jarak dari pusat kota (C2)	Ada angkutan umum (C3)	Dipilih untuk perumahan (C4)
1	100	2	Tidak	Ya
2	200	1	Tidak	Ya
3	500	3	Tidak	Ya
4	600	20	Tidak	Tidak
5	550	8	Tidak	Tidak
6	250	25	Ada	Tidak
7	75	15	Ada	Tidak
8	80	10	Tidak	Ya
9	700	18	Ada	Tidak
10	180	8	Ada	Ya

...

- Namun jika atribut ke-i bersifat kontinu, maka P(xi|C) diestimasi dengan fungsi densitas Gauss.
- Distribusi normal adalah distribusi dari variabel acak kontinu. Kadang-kadang distribusi normal disebut juga dengan distribusi Gauss. Distribusi ini merupakan distribusi yang paling penting dan paling banyak digunakan di bidang statistika.

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

e = 2,7183

...

- Probabilitas kemunculan setiap nilai untuk atribut Harga Tanah (C1)

	Ya	Tidak
1	100	600
2	200	550
3	500	250
4	80	75
5	180	700
Mean (μ) Deviasi standar (σ)	212 168,8787	435 261,9637

...

- Probabilitas kemunculan setiap nilai untuk atribut Jarak dari Pusat Kota (C2)

	Ya	Tidak
1	2	20
2	1	8
3	3	25
4	10	15
5	8	18
Mean (μ) Deviasi standar (σ)	4,8 3,9623	17,2 6,3008

...

- Probabilitas kemunculan setiap nilai untuk atribut Angkutan Umum (C3)

Angkutan Umum	Jumlah kejadian "Dipilih"		Probabilitas	
	Ya	Tidak	Ya	Tidak
Ada	1	3	1/5	3/5
Tidak	4	2	4/5	2/5
Jumlah	*5*	*5*	*1*	*1*

...

- Probabilitas kemunculan setiap nilai untuk atribut Dipilih untuk Perumahan (C4)

Dipilih Untuk Perumahan	Jumlah kejadian "Dipilih"		Probabilitas	
	Ya	Tidak	Ya	Tidak
Jumlah	*5*	*5*	*1/2*	*1/2*

=NORMDIST(300,212,168.8787,0)

- Apabila diberikan C1 = 300, C2 = 17, C3 = Tidak, maka:

$$f(C1=300\mid ya)=\frac{1}{\sqrt{2\pi}(168{,}8787)}e^{\frac{-(300-212)^2}{2(168{,}8787)^2}}=0{,}0021.$$

$$f(C1=300\mid tidak)=\frac{1}{\sqrt{2\pi}(261.9637)}e^{\frac{-(300-435)^2}{2(261.9637)^2}}=0{,}0013.$$

$$f(C2=17\mid ya)=\frac{1}{\sqrt{2\pi}(3.9623)}e^{\frac{-(17-4{,}8)^2}{2(3.9623)^2}}=0{,}0009.$$

$$f(C2=17\mid tidak)=\frac{1}{\sqrt{2\pi}(6{,}3008)}e^{\frac{-(17-17{,}2)^2}{2(6{,}3008)^2}}=0{,}0633.$$

...

Likelihood Ya = (0,0021) x (0,0009) x 4/5 x 5/10

= 0,000000756.

Likelihood Tidak = (0,0013) x (0,0633) x 2/5 x 5/10

= 0,000016458.

...

- Nilai probabilitas dapat dihitung dengan melakukan normalisasi terhadap likelihood tersebut sehingga jumlah nilai yang diperoleh = 1

Probabilitas Ya = $$\frac{0{,}000000756}{0{,}000000756 + 0{,}000016458} = 0{,}0439.$$

Probabilitas Tidak = $$\frac{0{,}000016458}{0{,}000000756 + 0{,}000016458} = 0{,}9561.$$

Klasifikasi : TIDAK

Terima Kasih ☺

Artificial Neural Network

Sejarah *Art. Neural Network*

- 1940-an, *McCulloch* dan *Pitt* memulai riset *ANN*
- 1960-an,
  - *Rosenblatt* menemukan teknik *perceptron*
  - *Minsky* dan *Papert* membuktikan kelemahan *perceptron* sederhana yg ditemukan *Rosenblatt*

Jenis2 design ANN

- *Back Propagation*
- *Recurrent Network*
- *Self Organizing Map*
- *Bayesian Network*
- dll

Manfaat

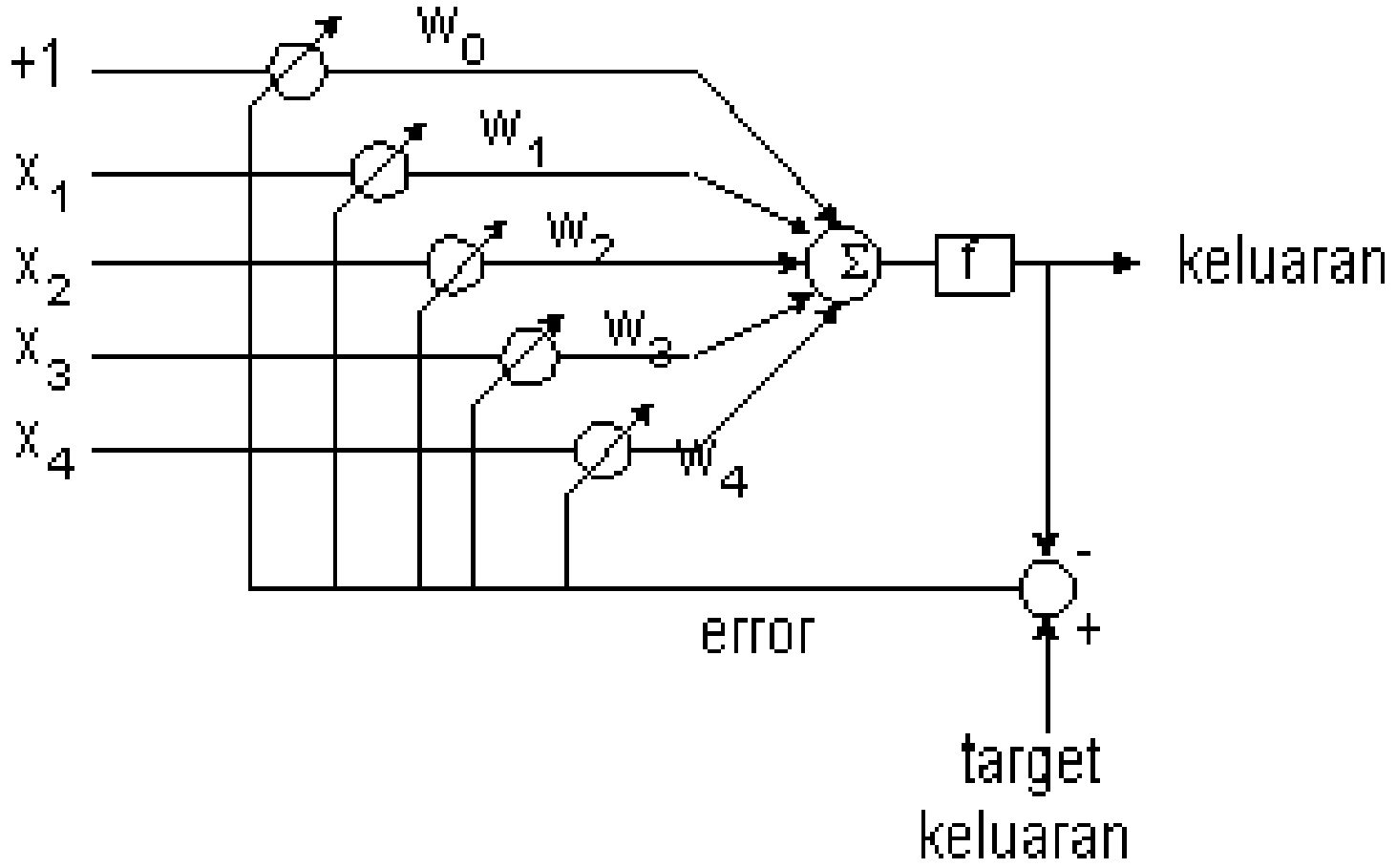
- Peramalan curah hujan
- Pendeteksian tornado
- Pendeteksian pemalsuan kartu kredit
- dll

Pengertian

- Jaringan syaraf tiruan merupakan algoritma komputasi yang meniru cara kerja sel syaraf.
- Semua sinyal yang masuk dikalikan dengan bobot yang ada pada tiap masukan, oleh sel neuron, semua sinyal yang sudah dikalikan dengan bobot dijumlahkan kemudian ditambah lagi dengan bias.
- Hasil penjumlahan ini diinputkan ke suatu fungsi (fungsi aktifasi) menghasilkan keluaran dari neuron (di sini digunakan fungsi aktifasi linier).

- Selama proses pembelajaran, bobot-bobot dan bias selalu diperbaharui menggunakan algoritma belajar jika ada error pada keluaran.
- Untuk proses identifikasi, bobot-bobot yang secara langsung memboboti masukan inilah yang dinamakan sebagai parameter yang dicari.
- Pada Gambar 1, parameter yang dicari adalah harga w_1, w_2, w_3 dan w_4.
- Dalam identifikasi secara on-line, neuron ataupun jaringan neuron akan selalu ‘belajar’ setiap ada data masukan dan keluaran.

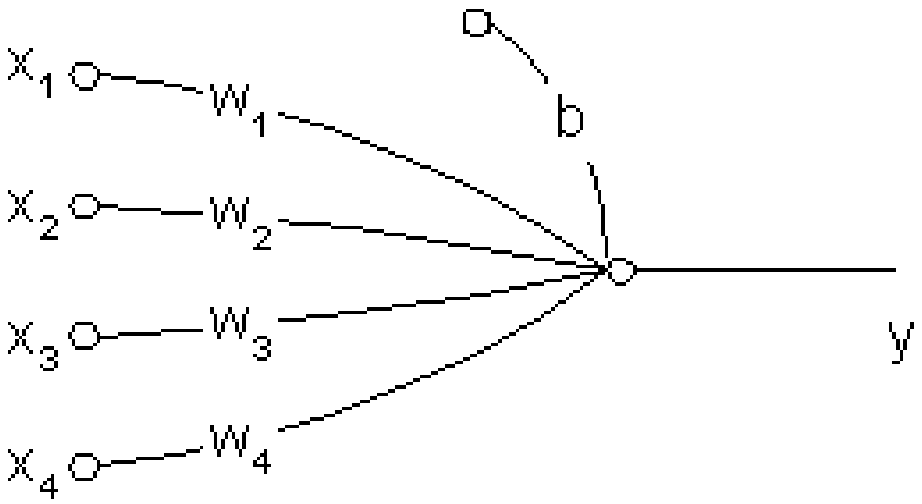
Gambar 1 : Sel neuron ketika sedang melakukan proses belajar



- Algoritma untuk memperbaharui bobot pada neuron satu lapis adalah seperti pada bagian algoritma pemrograman JST satu lapis langkah ke-7. Sedangkan untuk JST dua lapis adalah seperti pada bagian algoritma pemrograman JST dua lapis langkah ke-8 dan 9.

Algorithm Pemrograman

- **Untuk Neuron Satu Lapis**



- Algoritma pemrograman untuk neuron satu lapis didasarkan pada Gambar 2, dimana fungsi aktifasinya linier f(x) = x, data masukan dinyatakan dengan matrik berikut: X = [x1, x2, x3, x4]
- bobot-bobot link neuron adalah: W = [w1, w2, w3, w4]
- bias = b
- maka $y = X*W^T + b$, atau $y = x_1w_1 + x_2w_2 + x_3w_3 + x_4w_4 + b$, dengan demikian parameternya adalah q = W.

Algoritma pemrogramannya adalah:

1. Inisialisasi bobot-bobot (termasuk juga bias), termasuk perubahan bobot awal.
2. Mengambil nilai x_1, x_2, x_3 dan x_4 juga nilai target.
3. Menghitung keluaran jaringan neuron

   $y = X * W^T + b$
4. Menghitung parameter

   $\theta = W$
5. Menghitung error keluaran

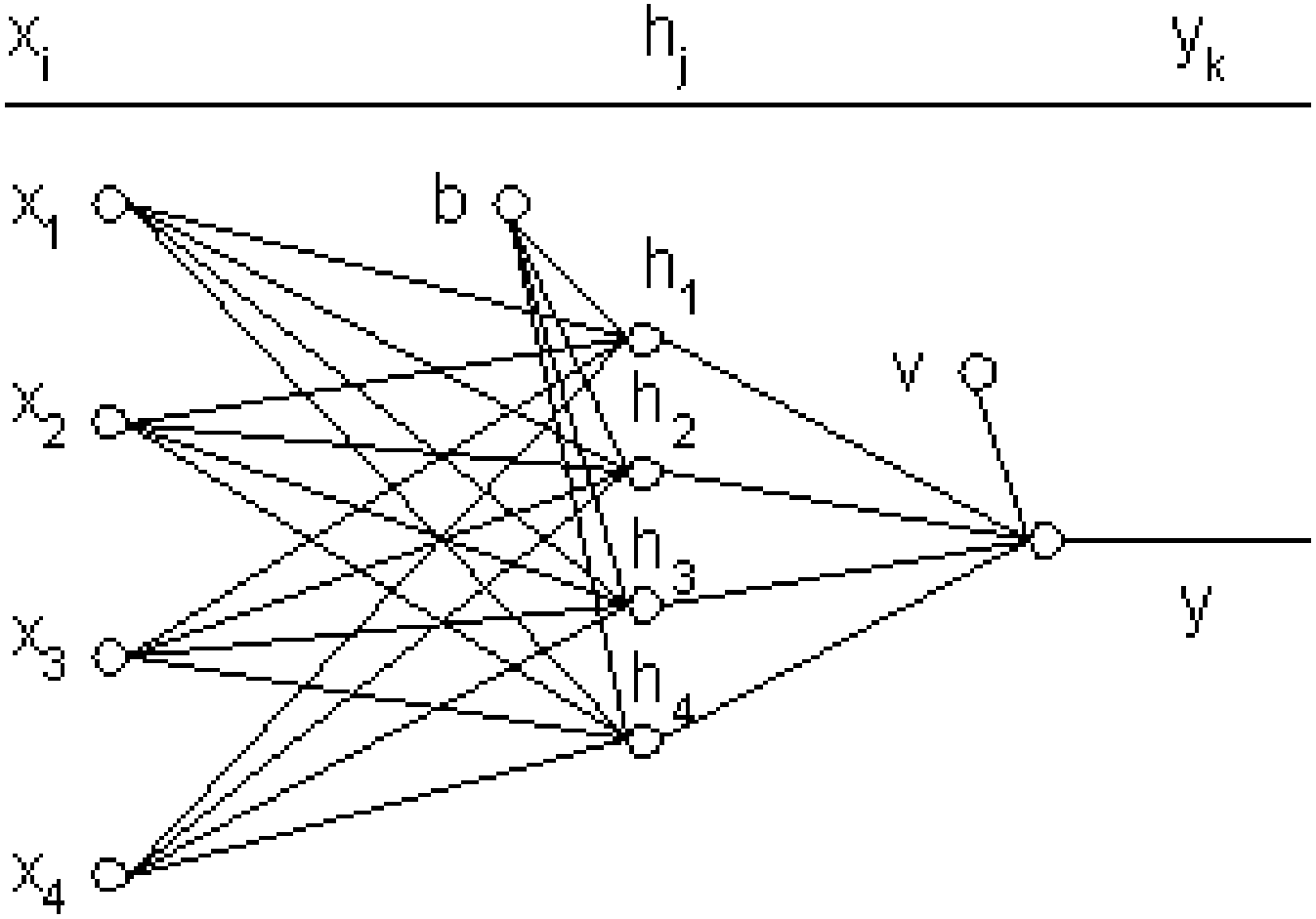
   $e = target - y$
6. Menyimpan bobot-bobot ke dalam variabel bobot lama
7. Menghitung perubahan bobot-bobot pada lapisan keluaran

   $$\Delta w_i = \eta e x_i \quad , \quad \Delta b = \eta e$$

   $$w_i(baru) = w_i(lama) + \Delta w_i(baru) + \alpha \Delta w_i(lama)$$

   $$b(baru) = b(lama) + \Delta b(baru) + \alpha \Delta b(lama)$$
8. Menyimpan perubahan-perubahan bobot dan bias ke variabel perubahan lama.
9. Kembali ke langkah 2.

Untuk Neuron Dua Lapis



- Algoritma pemrograman untuk neuron dua lapis didasarkan pada Gambar 3, dimana fungsi aktifasinya linier f(x) = x, data masukan dinyatakan dengan matrik:

  $X = [x1, x2, x3, x4]$

- Bobot-bobot link neuron masukan adalah a_{ij}, sehingga dalam bentuk matrik menjadi:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

- Keluaran dari tiap-tiap neuron pada lapisan masukan adalah: H = [h1, h2, h3, h4]
- Bobot-bobot bias pada lapisan masukan yaitu: B = [b1, b2, b3, b4]
- Bobot-bobot link neuron pada lapisan keluaran yaitu: W = [w1, w2, w3, w4]
- bobot bias pada lapisan keluaran = v, keluaran NN adalah, $y = (X*A+B)*W^T + v$.

Algoritma pemrogramannya adalah:

1. Inisialisasi bobot-bobot (termasuk juga bias), termasuk perubahan-perubahan bobot awal.
2. Mengambil nilai x_1, x_2, x_3 dan x_4 juga nilai target.
3. Menghitung keluaran jaringan neuron

   $y = (X * A + B) * W^T + v$
4. Menghitung parameter

   $\theta = \Delta * W^T$
5. Menghitung error keluaran

   $e = target - y$

6. Menyimpan bobot-bobot ke dalam variabel bobot lama
7. Menghitung matrik H

   H = X*A + b
8. Menghitung error propagasi pada lapisan keluaran

$$\delta_k = e * f'(y\_in_k)$$

Menghitung perubahan bobot-bobot pada lapisan keluaran

$$\Delta w_{jk} = \eta \delta_k h_j \qquad \Delta v_k = \eta \delta_k$$

$$w_{jk}(\text{baru}) = w_{jk}(\text{lama}) + \Delta w_{jk}(\text{baru}) + \alpha \Delta w_{jk}(\text{lama})$$

$$v_k(\text{baru}) = v_k(\text{lama}) + \Delta v_k(\text{baru}) + \alpha \Delta v_k(\text{lama})$$

9. Menghitung error propagasi pada lapisan masukan

$$\delta_j = f'(hc\_in_j)\sum_{1}^{j}\delta_k w_{jk}$$

$$\Delta a_{ij} = \eta\delta_j x_i$$

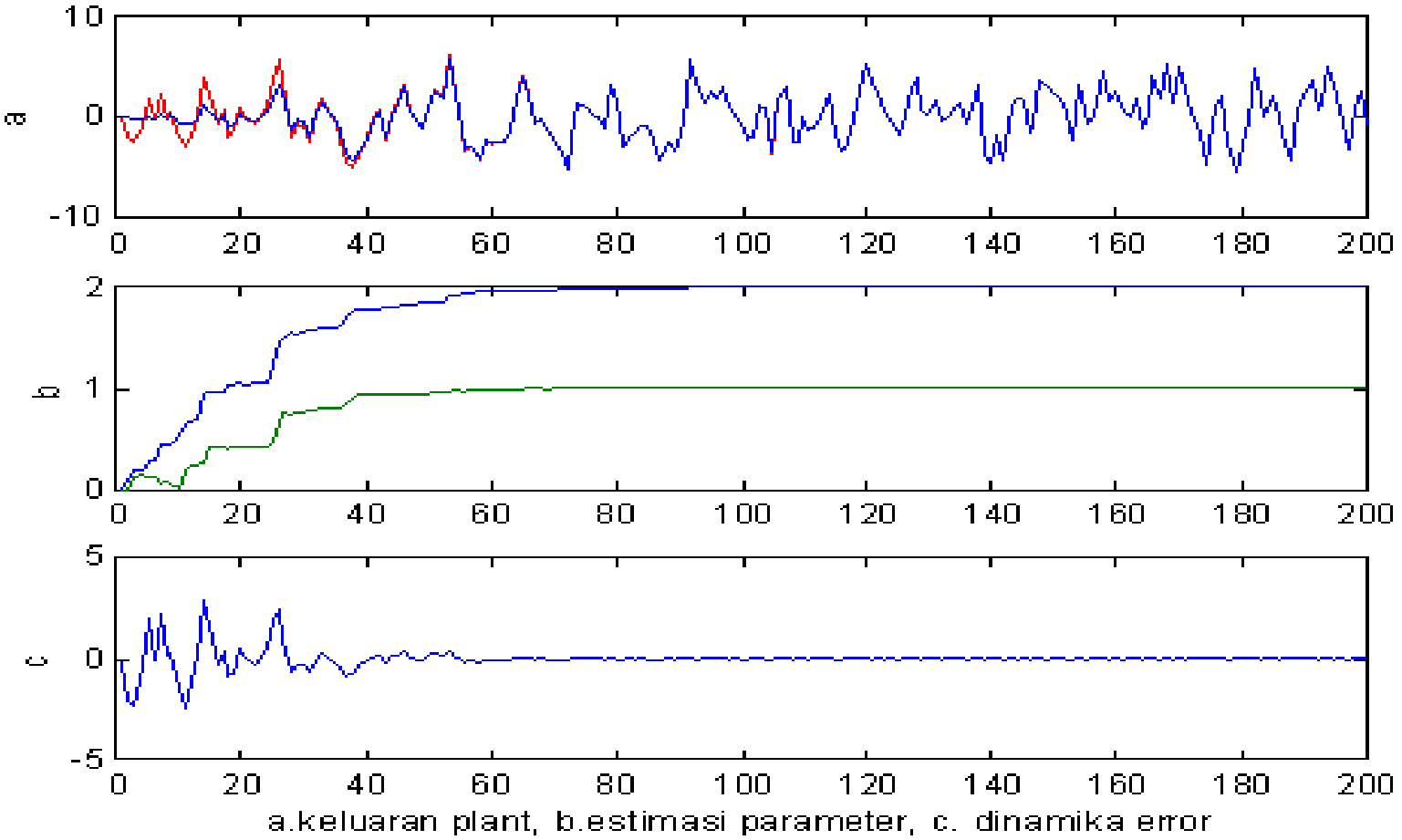
$$a_{ij}(baru) = a_{ij}(lama) + \Delta a_{ij}(baru) + \alpha\Delta a_{ij}(lama)$$

$$b_j(baru) = b_j(lama) + \Delta b_j(baru) + \alpha\Delta b_j(lama)$$

- 10. Menyimpan perubahan bobot-bobot dan bias kedalam variabel perubahan lama.
- 11. Kembali ke langkah 2.

Simulasi

- *Plant* ke-1 yaitu y(k) = 2u(k) + u(k-1), konstanta pembelajaran (delta *rule* biasa) b=0,05; a=0, bobot awal 0, $q_1 = 2$ dan $q_2 = 1$



a.keluaran plant, b.estimasi parameter, c. dinamika error

Neural Network dengan Algoritma Backpropagation

DEFINISI MASALAH

- Di bawah ini diberikan data hasil ujian suatu mata kuliah yang diikuti oleh 50 mahasiswa. Nilai suatu mata kuliah ditentukan oleh 5 nilai ujian atau tugas yang diperoleh setiap mahasiswa. Misalkan 5 nilai ujian yaitu X1, X2, X3, X4, X5 dan Y merupakan nilai akhir. Untuk memprediksi nilai akhir kita harus membuat program komputer yang dapat mengetahui bobot masing-masing nilai (X1, X2, X3, X4, X5).

NO	X1	X2	X3	X4	X5	Nilai
1	50	95	95	50	80	71
2	75	95	80	74	77	78
3	80	85	90	70	78	78
4	80	85	60	80	70	75
5	85	85	90	77	70	77
6	85	90	90	78	66	76
7	90	85	60	76	67	73
8	85	95	90	81	62	76
9	85	85	90	77	65	75
10	80	90	57	70	70	72
11	70	80	75	71	78	75
12	90	85	78	70	66	73
13	85	85	78	78	61	73
14	80	85	76	71	66	72
15	85	90	67	72	60	70
16	85	90	65	76	57	70
17	85	90	45	74	54	66
18	80	85	85	71	60	70
19	80	85	45	70	58	65
20	85	85	87	73	54	69
21	70	20	40	70	30	46
22	80	85	90	80	70	78
23	85	85	88	77	70	77
24	85	90	87	78	66	76
25	80	85	92	76	67	75

NO	X1	X2	X3	X4	X5	Nilai
26	85	95	90	81	62	76
27	85	85	60	77	65	72
28	80	90	90	70	70	75
29	70	80	75	71	78	75
30	90	85	81	70	66	73
31	80	90	90	70	70	75
32	70	80	75	71	78	75
33	90	85	85	70	66	73
34	85	85	20	78	61	67
35	80	85	85	71	66	73
36	85	90	85	72	60	72
37	85	90	90	76	57	72
38	85	90	85	74	54	70
39	80	85	95	71	60	71
40	80	85	85	70	58	69
41	85	85	90	73	54	70
42	70	43	80	45	30	45
43	80	85	85	80	70	77
44	85	85	90	77	70	77
45	85	90	80	78	66	75
46	90	85	90	76	67	76
47	85	95	85	81	62	76
48	85	85	85	77	65	75
49	80	90	80	70	70	74
50	70	80	70	71	78	75

Untuk melakukan uji prediksi, digunakan 50 data di bawah ini.

NO	X1	X2	X3	X4	X5	Y
1	90	85	90	70	66	74
2	85	85	85	78	61	73
3	80	85	80	71	66	72
4	85	90	85	72	60	72
5	85	90	90	76	57	72
6	85	90	80	78	66	75
7	90	85	85	76	67	76
8	85	95	85	81	62	76
9	85	85	85	77	65	75
10	80	90	90	70	70	75
11	70	80	75	71	78	75
12	90	85	90	70	66	74
13	85	85	78	78	61	73
14	80	85	89	71	66	73
15	85	90	76	72	60	71
16	85	90	65	76	57	70
17	85	90	56	78	66	73
18	90	85	87	76	67	76
19	85	95	45	81	62	72
20	85	85	45	77	65	71
21	80	90	98	70	70	76
22	70	80	75	71	78	75
23	90	85	90	70	66	74
24	85	85	90	78	61	74
25	80	85	43	71	66	69

NO	X1	X2	X3	X4	X5	Y
26	85	90	34	72	60	67
27	85	90	56	76	57	69
28	85	90	49	74	54	66
29	80	85	85	71	60	70
30	80	85	75	70	58	68
31	85	90	78	78	66	75
32	90	85	56	76	67	73
33	85	95	89	81	62	76
34	85	85	93	77	65	75
35	80	90	76	70	70	74
36	70	80	75	71	78	75
37	90	85	96	70	66	75
38	85	85	78	78	61	73
39	80	85	66	71	66	71
40	85	90	76	72	60	71
41	85	90	89	76	57	72
42	85	90	90	74	54	70
43	80	85	85	71	60	70
44	80	85	67	70	58	67
45	85	85	78	73	54	68
46	70	20	87	70	30	51
47	80	85	90	80	70	78
48	85	90	78	78	66	75
49	91	85	74	76	67	75
50	85	95	58	81	62	73

IDE PENYELESAIAN MASALAH

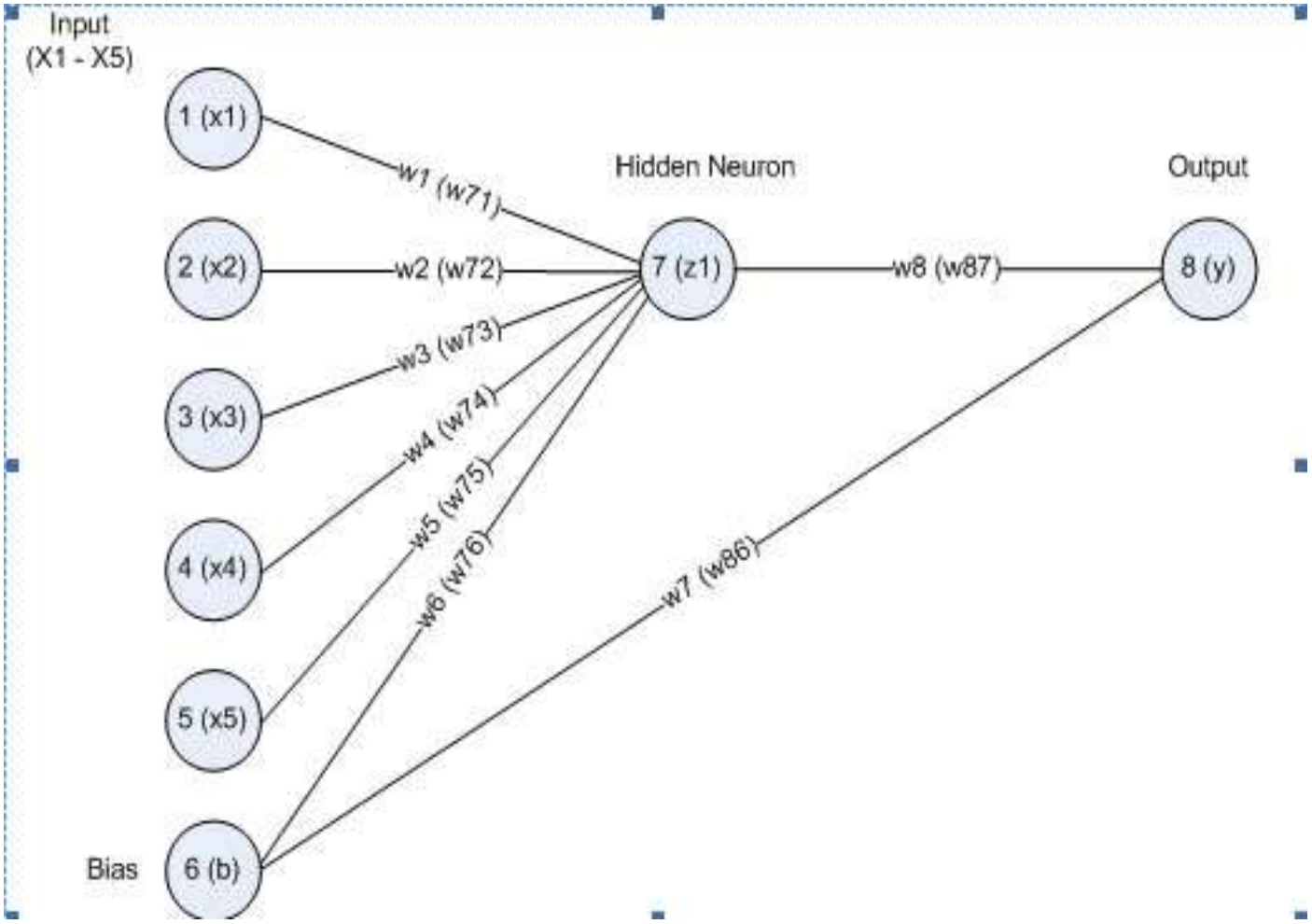
- Untuk menyelesaikan permasalahan ini, kita akan menggunakan algoritma *Backpropagation*.
- *Backpropagation* (perambatan galat mundur) adalah Algoritma pembelajaran yang terawasi dan biasanya digunakan oleh *perception* dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan neuron-neuron yang ada pada lapisan tersembunyi sebuah metode sistematik untuk pelatihan multiple layer jaringan saraf tiruan.
- Metode ini memiliki dasar matematis yang kuat, obyektif dan algoritma ini mendapatkan bentuk persamaan dan nilai koefisien dalam formula dengan meminimalkan jumlah kuadrat galat error melalui model yang dikembangkan (training set).

Algoritma pelatihan back propagation pada dasarnya terdiri:

1. Input nilai data pelatihan sehingga diperoleh nilai output
2. Propagasi balik dari nilai error yang diperoleh
3. Penyesuaian bobot koneksi untuk meminimalkan nilai error

Selama pelatihan, tiap unit keluaran membandingkan perhitungan aktivasinya Y_k dengan nilai targetnya T_k untuk menentukan kesalahan pola tersebut dengan unit itu.

Dalam pengecekan selalu ada bias yang nilainya sudah ditentukan yaitu 1. Pada bias juga dikalikan bobot yaitu $W_{7b,}$ tetapi langsung menuju output, tanpa melewati hidden layer.

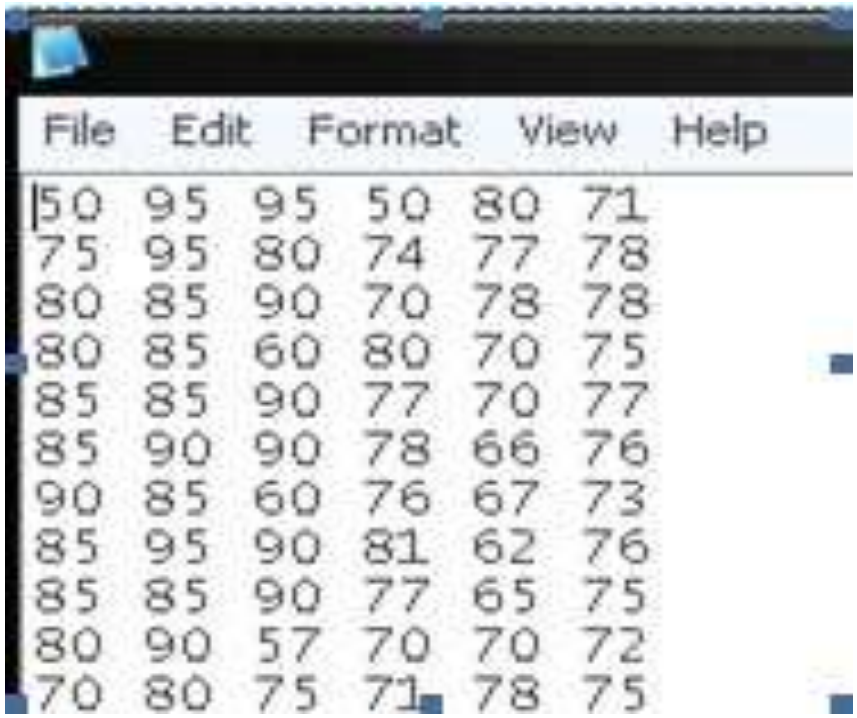


ALGORITMA

- Berikut ini algoritma program berdasarkan algoritma backpropagation
- N = neuron unit
- N1 = O1 = x1
- N1 = O2 = x2
- N1 = O3 = x3 input
- N1 = O4 = x4
- N1 = O5 = x5
- N1 = O6 = b (bias)
- N1 = O7 = z1 (hidden unit)
- N1 = O8 = y (output unit)
- Langkah 1 : Kita perlu membuat file bertipe teks (dari soal untuk proses training dan untuk roses uji prediksi).

Langkah 1

- Langkah 1 : Kita perlu membuat file bertipe teks (dari soal untuk proses training dan untuk roses uji prediksi).



- Langkah 2 : Menentukan Learning Rate (η) dengan kisaran nilai antara 0 sampai 1 yang ditentukan oleh user.
- Langkah 3 : Menentukan Epoch untuk training dari program tersebut. Epoch merupakan perulangan komputasi program untuk menentukan bobot dari masing-masing input. Epoch dimasukkan secara manual oleh user.
- Langkah 4 : Membuat procedure “bacadata” untuk membaca file teks, yang akan membaca teks nilai.txt, yang kemudian digunakan dalam training untuk input neuron dan target.
- Langkah 5 : Menginisialisasi bobot yang akan masuk ke hidden layer secara random, antara 0-1.

- Langkah 6 : Membuat fungsi sigmoid unipolar
- Dengan rumus : $Oj = F (\Sigma Wji\ Oi - \theta j)$
- Karena menggunakan neuron Bias, maka treshold $\theta j = 0$
- Fungsi aktivasi
- Langkah 7 : Membuat procedure training.

Dalam prosedur training terdapat 4 tahap

i. **Melakukan kalkulasi fungsi aktivasi terhadap tiap bobot dan unit.**

N7 = F(W1.N1 + W2.N2 + W3.N3 + W4.N4 + W5.N5 + W6.N6) (N6 = O6 = bias = selalu 1)

N8 = F(W8.N7 + W7.N6) (N6 = O6 = bias = selalu 1)

ii. **Menghitung error gradien untuk tiap hidden dan output unit**

§ Output unit :

d8 = N8.(1 - N8).(y - N8)

Oj = {nilai output} dan Tj = {nilai target}

§ Hidden unit :

d7 = N7.(1 - N7).(d8.W8)

iii. Menjalankan training bobot / weight training, yaitu bobot baru = bobot lama + perubahan bobot.

Perubahan bobot :

Perhitungan bobot :

w1(baru) = w1(lama) + ή.δ7.N1

w2(baru) = w2(lama) + ή.δ7.N2

w3(baru) = w3(lama) + ή.δ7.N3

w4(baru) = w4(lama) + ή.δ7.N4

w5(baru) = w5(lama) + ή.δ7.N5

w6(baru) = w6(lama) + ή.δ7.N6

w7(baru) = w8(lama) + ή.δ8.N6

w8(baru) = w8(lama) + ή.δ8.N7

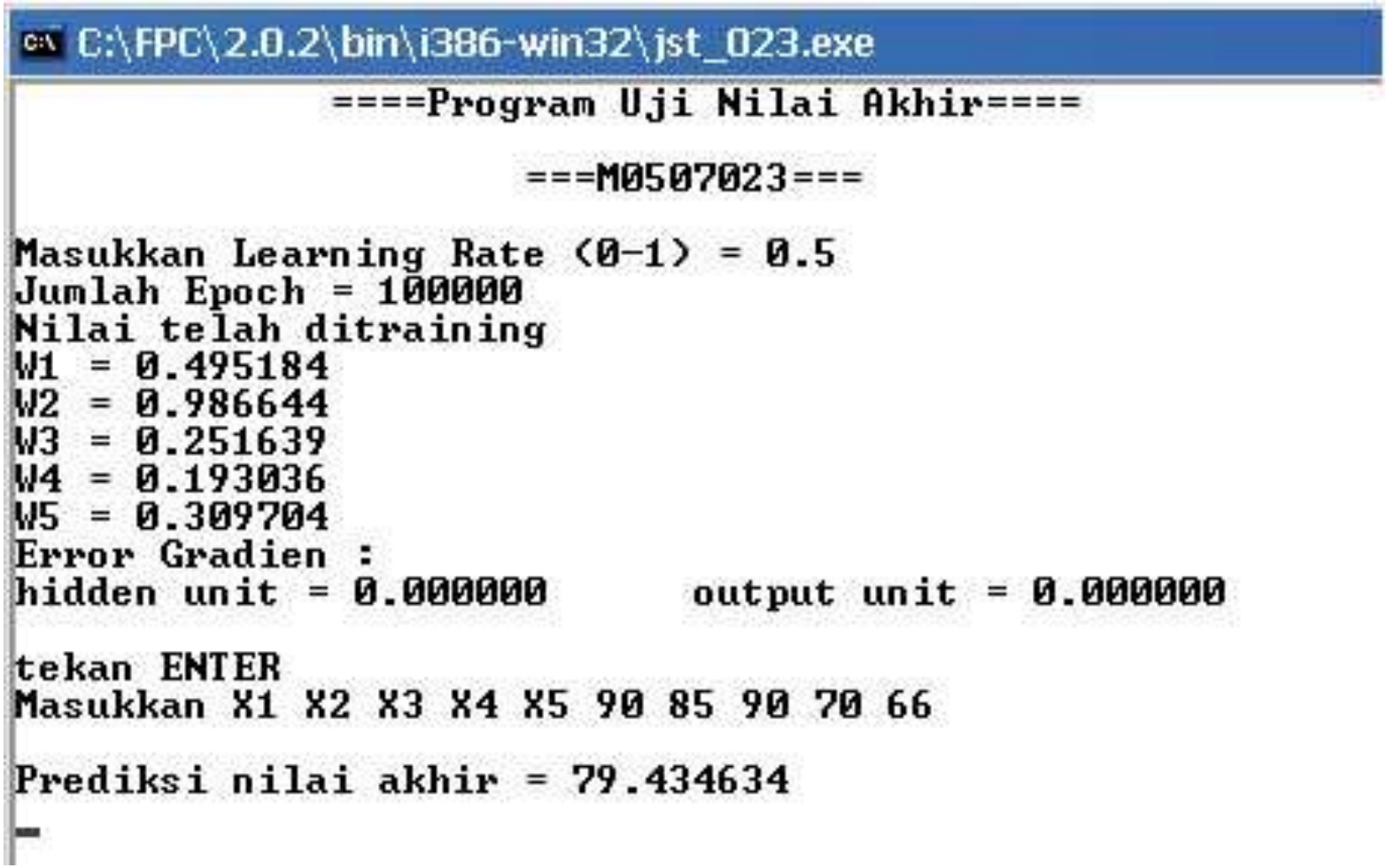
iv. Lanjutkan ke iterasi berikutnya sampai iterasi = epoch yang diinginkan.

Langkah 8 : Membuat fungsi uji, untuk menguji hasil training dengan nilai tes yang sudah disediakan.

Menghitung hidden unit menggunkan fungsi akivasi yaitu mengalikan bobot hasil training dengan nilai yang diinputkan.

$N7 = F(W1.N1 + W2.N2 + W3.N3 + W4.N4 + W5.N5 + W6.N6)$

$y = F(W8.N7 + W7.N6)$



- Contoh Output program untuk input Learning rate : 0.5
- Dengan selisih dengan hasil asli 2.434634

Selesai