BAB 3
ANALISIS DAN PERANCANGAN SISTEM

Analisis merupakan proses penguraian konsep ke dalam bagian-bagian yang lebih sederhana, sehingga struktur logisnya menjadi jelas. Metode untuk menguji, menilai, dan memahami sistem pemikiran yang kompleks dengan memecahnya ke dalam unsur-unsur yang lebih sederhana sehingga hubungan antarunsur-unsur itu menjadi jelas.

3.1 Analisis Sistem

Jaringan Saraf Tiruan (JST) adalah paradigma pemrosesan suatu informasi yang terinspirasi oleh sistem sel saraf biologi, sama seperti otak manusia yang memproses suatu informasi.

Ada banyak metode JST yang dapat diimplementasikan untuk prediksi. Masing-masing metode JST memiliki kelebihan dan kekurangan. Dalam lingkup prediksi jumlah pencari kerja, tidak dapat dikatakan langsung metode mana yang sesuai untuk diaplikasikan, karena suatu metode yang baik untuk suatu kasus belum tentu baik pula untuk kasus yang berbeda.

Metode *Adaline* adalah algoritma pembelajaran terawasi lapis tunggal di mana *layer* input langsung berhubungan dengan *layer* output. Pembelajaran *Adaline* menggunakan aturan delta yang mengatur bobot-bobot untuk mengurangi selisih antara input jaringan ke unit output dengan output yang diinginkan. Metode ini memiliki keunggulan diantaranya cepat dalam mendapatkan output. Sedangkan metode *Backpropagation* adalah algoritma pembelajaran terawasi yang menggunakan pola penyesuaian bobot untuk mencapai nilai kesalahan minimum antara keluaran hasil prediksi dengan keluaran yang nyata dan memiliki satu atau lebih *layer* tersembunyi. Metode *Backpropagation* banyak digunakan untuk peramalan.

Dalam pembelajaran terawasi, terdapat sejumlah pasangan data (masukan-target keluaran) yang dipakai untuk melatih jaringan hingga diperoleh bobot yang diinginkan. Pasangan data tersebut berfungsi sebagai “guru” untuk melatih

jaringan hingga diperoleh bentuk yang terbaik. “Guru” akan memberikan informasi yang jelas tentang bagaimana sistem harus mengubah dirinya untuk meningkatkan unjuk kerjanya. Berbeda dengan pembelajaran yang tak terawasi di mana pada pembelajaran tak terawasi tidak memiliki target keluaran

Penerapan kedua metode JST ini sebagai komparasi untuk menemukan metode yang memiliki akurasi yang lebih baik dalam memprediksi jumlah pencari kerja di Jawa Barat yang belum memiliki sistem peramalan jumlah pencari kerja.

3.2 Analisis Data

Secara administratif, para pencari kerja yang meminta kartu pencari kerja (kartu kuning) akan dicatat di dinas kependudukan sesuai dengan tempat domisili yang tertera di Kartu Tanda Penduduk (KTP) pencari kerja yang bersangkutan. Syarat untuk membuat kartu kuning diantaranya adalah menyerahkan foto *copy* ijazah terakhir dan KTP. Dari pencatatan kartu pencari kerja tersebut, dapat diketahui jumlah pencari kerja berdasarkan pendidikan yang ditamatkan di Jawa Barat.

Secara umum data yang akan menjadi masukan dalam sistem peramalan adalah data historis. Data historis digunakan sebagai variabel masukan untuk menentukan atau meramalkan jumlah pencari kerja pada tahun berikutnya. Data historis dibagi menjadi dua yaitu data pelatihan dan data pengujian. Data historis diambil 6 tahun ke belakang dengan 5 variabel masukan yaitu jumlah penduduk, jumlah lulusan, jumlah angkatan kerja, jumlah lowongan kerja, dan jumlah pengangguran terbuka. Dan terdapat 1 variabel sebagai target. Dengan demikian ada 30 data masukan (5 x 6) sebagai historis untuk pelatihan. Data yang dikumpulkan adalah data historis dari tahun 2004 sampai tahun 2009. Data bersumber dari buku Jawa Barat dalam Angka yang diterbitkan oleh Badan Pusat Statistika (BPS) Propinsi Jawa Barat tahun 2005-2010.

Data pertama yang dijadikan variabel masukan adalah data jumlah penduduk yang merupakan data primer yang menjadi rujukan data statistika lainnya. Tabel 3.1 memperlihatkan data jumlah penduduk.

Tabel 3.1 Jumlah Penduduk

Tahun	Jumlah
2004	38.472.185
2005	39.956.160
2006	40.737.594
2007	41.483.729
2008	42.194.869
2009	42.693.951

Data kedua yang dijadikan variabel masukan adalah data jumlah lulusan. Data jumlah lulusan merupakan akumulasi dari jumlah lulusan SD, SMP, SMA, D3, dan S1. Tabel 3.3 memperlihatkan jumlah lulusan.

Tabel 3.2 Jumlah Lulusan

Tahun	Jumlah
2004	4.219.610
2005	4.517.615
2006	4.938.977
2007	8.685.830
2008	9.457.125
2009	12.327.771

Data ketiga yang dijadikan variabel masukan adalah jumlah angkatan kerja yakni penduduk usia produktif yang mencari pekerjaan. Tabel 3.3 memperlihatkan jumlah angkatan kerja.

Tabel 3.3 Jumlah Angkatan Kerja

Tahun	Jumlah
2004	16.636.057
2005	17.040.084
2006	17.340.593
2007	18.240.036
2008	18.743.979
2009	18.981.260

Data keempat yang dijadikan variabel masukan adalah jumlah lowongan kerja yang membuka peluang bagi para pencari kerja untuk berkompetisi mengisi bidang-bidang yang sesuai. Umumnya jumlah lowongan kerja selalu lebih kecil dari jumlah pencari kerja yang membutuhkan sehingga berdampak pada pengangguran terbuka. Tabel 3.4 memperlihatkan jumlah lowongan kerja.

Tabel 3.4 Jumlah Lowongan Kerja

Tahun	Jumlah
2004	25.026
2005	35.003
2006	28.299
2007	132.593
2008	106.572
2009	147.927

Data kelima yang dijadikan variabel masukan adalah jumlah pengangguran terbuka yang merupakan jumlah penduduk usia produktif yang tidak mendapatkan kesempatan kerja. Tabel 3.5 memperlihatkan jumlah pengangguran terbuka.

Tabel 3.5 Jumlah Pengangguran Terbuka

Tahun	Jumlah
2004	2.037.917
2005	2.029.474
2006	1.898.795
2007	2.386.214
2008	2.263.584
2009	2.079.830

Sedangkan data target adalah data pencari kerja. Diperoleh dari akumulasi jumlah pencari kerja lulusan SD, SMP, SMA, D3, dan S1. Jumlah pencari kerja jauh lebih besar dibanding jumlah lowongan kerja yang tersedia. Tabel 3.6 memperlihatkan jumlah pencari kerja.

Tabel 3.6 Jumlah Pencari Kerja

Tahun	**Jumlah**
2004	4.219.610
2005	1.573.929
2006	938.977
2007	2.841.321
2008	10.423.795
2009	12.267.323

Lima data input dan satu data target tesebut disatukan menjadi data master yang diperlihatkan di tabel 3.7. Tiap baris dari data master disebut pola. Dengan demikian dapat diketahui bahwa data master memiliki 6 pola.

Tabel 3.7 Data Master

No	x_1	x_2	x_3	x_4	x_5	t
1	38472185	4219610	16636057	25026	4059736	4219610
2	39956160	4517615	17040084	35003	5830865	1573929
3	40737594	4938977	17340593	28299	987366	938977
4	41483729	8685830	18240036	132593	11103301	2841321
5	42194869	9457125	18743979	106572	8814949	10423795
6	42693951	12327771	18981260	147927	21697170	12267323

Keterangan:

x_1 : Jumlah penduduk

x_2 : Jumlah lulusan

x_3 : Jumlah angkatan kerja

x_4 : Jumlah lowongan kerja

x_5 : Jumlah pengangguran terbuka

t : target

Sebelum diproses, data master dinormalisasi terlebih dahulu. Normalisasi data adalah proses penskalaan data input sesuai dengan *range* fungsi aktifasi yang digunakan. Normalisasi data dilakukan apabila data input bernilai besar (melebihi nilai maksimum fungsi aktifasi yang digunakan) dan prosesnya sebelum

perhitungan keluaran jaringan. Jadi, keluaran jaringan yang merupakan jumlah perkalian antara bobot dan data input menggunakan data input yang sudah melalui normalisasi data. Sedangkan denormalisasi data adalah proses pengembalian data yang diskala menjadi data yang sebenarnya. Denormalisasi data dilakukan pada data hasil keluaran jaringan.

Persamaan 3.1, 3.2 dan 3.3 adalah persamaan untuk normalisasi data, sedangkan persamaan 3.4 untuk denormalisasi data.

$$x = \alpha * x + \beta \quad (3.1)$$

$$\alpha = \frac{Upperbound - Lowerbound}{\max(x) - \min(x)} \quad (3.2)$$

$$\beta = Upperbound - (\alpha * \max(x)) \quad (3.3)$$

Keterangan:

x' : Data diskala

x : Data asli

Upperbound : Batas atas nilai data

Lowerbound : Batas bawah nilai data

$$x = \frac{(x - \beta)}{\alpha} \quad (3.4)$$

Dari data master diketahui data minimum=25.026 dan data maksimum=42.693.951. *Adaline* menggunakan fungsi aktifasi sigmoid bipolar dengan interval [-1,1] di mana *Upperbound*=-1 dan *Lowerbound*=1. Sedangkan pada *Backpropagation* menggunakan fungsi aktifasi sigmoid biner dengan interval [0,1] di mana *Upperbound*=1 dan *Lowerbound*=0.

a. Normalisasi data untuk metode *Adaline*

1) Cari nilai α dengan rumus (3.2)

$$\alpha = \frac{1-(-1)}{42693951-25026} = \frac{2}{42668925} = 0.0000000469$$

2) Cari nilai β dengan rumus (3.4)

$$\beta = 1 - (0.0000000469 * 42693951)$$
$$\beta = 1 - 2.00117$$
$$\beta = -1{,}00117$$

3) Cari nilai x' dengan rumus (3.1)

$$x' = 0.0000000469 * 38472185 + (-1.00117)$$
$$x' = \mathbf{0.8021152}$$

0.8021152

Jika perhitungan normalisasi data untuk metode *Adaline* tersebut diteruskan sampai selesai, hasilnya diperlihatkan di tabel 3.8.

Tabel 3.8 Data Hasil Normalisasi Metode *Adaline*

No	x1	x2	x3	x4	x5	t
1	**0.8021152**	-0.8033893	-0.2213991	-1.0000000	-0.9056507	-0.8033893
2	0.8716728	-0.7894210	-0.2024614	-0.9995324	-0.9060465	-0.9273990
3	0.9083006	-0.7696707	-0.1883758	-0.9998466	-0.9121717	-0.9571608
4	0.9432738	-0.5940463	-0.1462166	-0.9949581	-0.8893252	-0.8679932
5	0.9766068	-0.5578938	-0.1225955	-0.9961777	-0.8950731	-0.5125835
6	1.0000000	-0.4233394	-0.1114736	-0.9942393	-0.9036862	-0.4261727

b. Normalisasi data untuk metode *Backpropagation*

1) Cari nilai α dengan rumus (3.2)

$$\alpha = \frac{1-0}{42693951-25026} = \frac{1}{42668925} = 0.0000000234$$

2) Cari nilai β dengan rumus (3.4)

$$\beta = 1 - (0.0000000234 * 42693951)$$
$$\beta = 1 - 1.000586516$$
$$\beta = -0.000586516$$

3) Cari nilai x’ dengan rumus (3.1)

$$x' = 0.0000000234 * 38472185 + (-0.000586516)$$
$$x' = \mathbf{0.9010576}$$

Jika perhitungan normalisasi data untuk metode *Backpropagation* tersebut diteruskan sampai selesai, hasilnya diperlihatkan di tabel 3.9.

Tabel 3.9 Data Hasil Normalisasi

No	x_1	x_2	x_3	x_4	x_5	t
1	**0.9010576**	0.0983054	0.3893004	0.0000000	0.0471746	0.0983054
2	0.9358364	0.1052895	0.3987693	0.0002338	0.0469768	0.0363005
3	0.9541503	0.1151646	0.4058121	0.0000767	0.0439141	0.0214196
4	0.9716369	0.2029769	0.4268917	0.0025210	0.0553374	0.0660034
5	0.9883034	0.2210531	0.4387022	0.0019111	0.0524634	0.2437082
6	1.0000000	0.2883303	0.4442632	0.0028803	0.0481569	0.2869136

3.3 Analisis Metode JST

JST merupakan salah satu sistem pemrosesan informasi atau data yang didisain dengan menirukan cara kerja otak manusia dalam menyelesaikan suatu masalah dengan melakukan proses belajar melalui perubahan bobot sinapsisnya. Ada banyak metode JST sebagaimana disebutkan dalam landasan terori, namun dalam laporan ini pembahasan difokuskan pada dua metode JST yaitu *Adaline* dan *Backpropagation*.

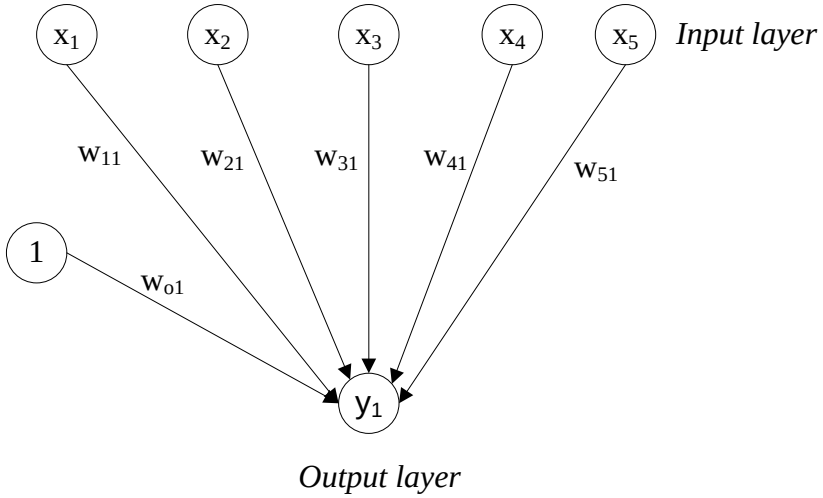
3.3.1 Analisis Metode *Adaline*

Metode *Adaline* ditemukan oleh Widrow dan Hoff (1960). Beberapa masukan dan sebuah bias (unit masukan tambahan) dihubungkan langsung dengan sebuah *neuron* keluaran. Bobot dimodifikasi dengan aturan delta (sering juga disebut *least mean square*).

a. Arsitektur Jaringan Metode *Adaline*

Adaline termasuk jaringan *single-layer* yang merupakan bagian dari jaringan *feedforward*, di mana sinyal datang dari input mengalir ke output. Jaringan *single*

layer hanya mempunyai satu lapisan koneksi. Dalam sistem *single layer*, unit dapat dibedakan sebagai unit input dan unit output secara jelas. Biasanya dalam model *single layer* setiap unit input terhubung ke unit output tetapi tidak terhubung ke unit input lainnya. Gambar 3.1 memperlihatkan arsitektur jaringan metode *Adaline* yang akan diimplementasikan.

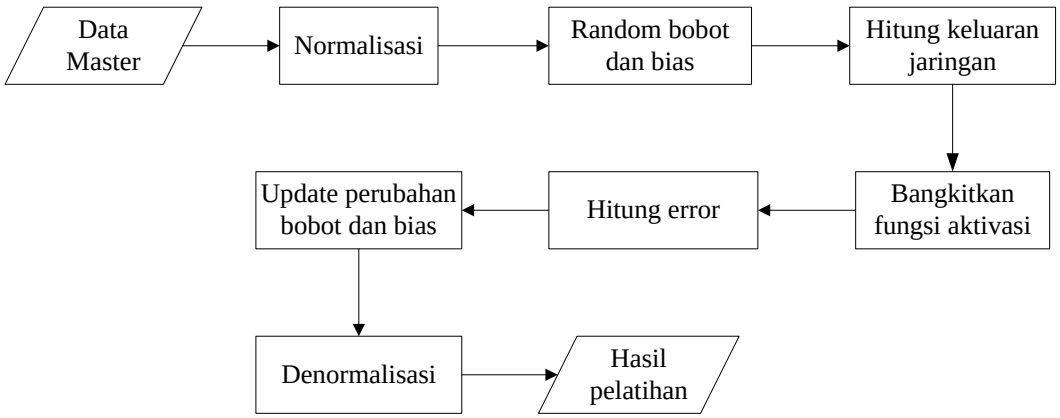


Gambar 3.1 Arsitektur Jaringan Metode *Adaline*

b. Algoritma Pelatihan Metode *Adaline*

Berikut ini algoritma pelatihan metode *Adaline*:

1. Inisialisasi semua bobot dan bias (umumnya w= b = 0)
2. Tentukan laju pemahaman (α). Untuk penyederhanaan biasanya α diberi nilai kecil (umumnya α =0.1)
3. Tentukan batas toleransi (toleransi *error* dan jumlah *epoch*)
4. Selama batas toleransi *false*, lakukan*:*
   a. Set aktifasi unit masukan $x_i = s_i$ (i=1,…,n)
   b. Hitung respon unit keluaran dengan persamaan (2.16) dan (2.17)
   c. Perbaiki bobot dan bias yang mengandung kesalahan (y≠t) menurut persamaan (2.20) dan (2.21)



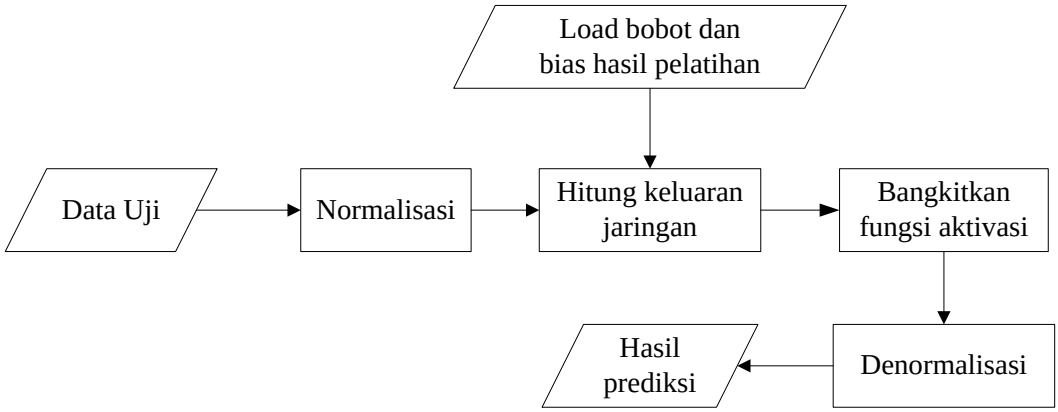
Gambar 3.2 Proses Pelatihan Metode *Adaline*

Secara lebih rinci, tiap bagian pada proses pelatihan dan pengujian *Adaline* dijelaskan di bagian analisis proses *Adaline*.

c. Algoritma Pengujian Metode *Adaline*

Berikut ini algoritma pengujian metode *Adaline*:

1. Inisialisasi semua bobot dan bias dengan bobot dan bias hasil pelatihan.
2. Untuk setiap input masukan, lakukan :
   a. Set aktifasi unit masukan $x_1 = s_1$ (i=1, …,n).
   b. Hitung respon *net* keluaran denga persamaan (2.16)
   c. Kenakan fungsi aktifasi :



Gambar 3.3 Alur Pengujian Metode *Adaline*

d. Contoh Perhitungan Metode *Adaline*

Contoh kasus menggunakan data master yang sudah dinormalisasi sebagaiman dapat dilihat tabel 3.7. Berikut ini studi kasus perhitungan pelatihan metode Adaline:

1. Inisialisasi semua bobot dan bias.

$$w_1=w_2=w_3=w_4=w_5=b=0$$

2. Tentukan laju pemahaman (α)

$$\alpha = 0.1$$

3. Selama kriteria belum berhenti:
   a. Set aktifasi unit masukan $x_i = s_i$
   b. Hitung respon unit keluaran (*net*) dan fungsi aktifasi dengan persamaan (2.16) dan (2.17)

Pola 1----------

y_1 = 0*0,8021152+0*(-0.8033893)+0*(-0.2213991)+
0*(-1.0000000)+0*(-0.8033893)+0 = 0

Hasil aktifasi y=1
Error = $(t_1 - y_1)$= (-0.0988339-1)= -1.0988339

Pola 2----------

y_2=0*0.8716728+0*(-0.7894210)+0*(-0.2024614)+

0*0.9995324+0*0.9060465 +0= 0

Hasil aktifasi y=1
Error = ($t_2 – y_2$)= (-0. 927399-1)= -1.927399

Pola 3----------

y_3=0*0.9083006+0*(-0.7696707)+0*(-0.1883758)+
0*(-0.9998466)+0*(-0.9548927)+0= 0

Hasil aktifasi y=1
Error = ($t_3 – y_3$)= (-0.9571608-1)= -1.9571608

Pola 4----------

y_4=0*0,9432738+0*(-0,5940463)+0*(-0,1462166)+
0*(-0,9949581)+0*(-0.4807333)+0 =0

Hasil aktifasi y=1
Error = ($t_2 – y_3$)= (-0.8679932-1)= -1.8679932

Pola 5----------

y_5=0*0.9766068+0*(-0.5578938)+0*(-0.1225955)+
0*(-0.9961777)+0*(-0.5879942)+0=0

Hasil aktifasi y=1
Error = ($t_5 – y_5$)= (-0.5125835-1)= -1.5125835

Pola 6----------

y_6=0*1.0000000+0*(-0.4233394)+0*(-0.1114736)+
0*(-0.9942393)+0*(-0.0158280)+0=0

Hasil aktifasi y=1
Error = ($t_6 – y_6$)= (-0.4261727-1)= -1.4261727

c. Perbaiki bobot yang mengandung kesalahan ($y \neq t$) menurut persamaan (2.20)

w_1(baru)= w_1(lama)+α(t_1-y_1)x_1

w_1(baru)= 0+0.1*(-1.0988339)*0.8021152 = -0.0881

w_2(baru)= w_2(lama)+α(t_2-y_2)x_2

w_2(baru)= 0+0.1*(-1.927399)* (-0.8033893) =0.15485

w_3(baru)= w_3(lama)+α(t_2-y_3)x_3

w_3(baru)=0+0.1*(-1.9571608)* (-0.2213991) = 0.04333136

w_4(baru)= w_4(lama)+α(t_3-y_4)x_4

w_4(baru)=0+0.1*(-1.8679932)* (-1.0000000) = 0.18679932

w_5(baru)= w_4(lama)+α(t_4-y_5)x_5

w_5(baru)=0+0.1*(-1.8679932)* (-0.8679932) = 0.16214054

w_6(baru)= w_6(lama)+α(t_5-y_4)x_5

w_6(baru)=0+0.1*(-1.5125835)* (-0.8108830) = 0.12265282

d. Perbaiki bias menurut persamaan (2.21)

b(baru) = b(lama)+ α*(t_1-y_1)

= 0+0,1*(-1.0988339) = -0.1098834

Perhitungan diteruskan sampai iterasi maksimum. Misalnya pada ahir pelatihan *Adaline*, diperoleh bobot w_1=-0.02, w_2=0.15, w_3=0.01, w_4=0.11, w_5=0.13, dan bias b=-0.01

Setelah proses pelatihan selesai, *Adaline* dapat dipakai untuk pengujian. Proses pengujian dilakukan dengan mengambil data-data pelatihan. Berikut ini pangujian algoritma metdoe *Adaline.*

1. Inisialisasi semua bobot dan bias dengan bobot dan bias hasil pelatihan.

   w_1=-0.1, w_2=0.5, w_3=0.2, w_4= 0.3, w_5= 0.1 dan bias b=-0.2

2. Untuk setiap input masukan x, lakukan :

   a. Hitung respon *net* keluaran dengan persamaan (2.16)

      y=((-0.1)*0.9432738+0.5*(-0.5940463)+0.2*(-0.1462166)+ 0.3*(-0.9949581)+ (0.1)+(-0.04262))+(-0.2) = -0.8617

Berdasarkan pola data yang diujikan (pola data pada tahun 2007) dapat diketahui nilai min=132593 dan max=41483729. Hasil pengujian berupa data yang diskla x'=-0.8617. Sedangkan berdasarkan fungsi identitas pada pengujian yang memiliki interval [-1,1] dapat diketahui *Upperbound*=1 dan *Lowerbound*=-1. Sehingga perhitungan denormalisasinya adalah sebagai berikut:

1) Cari nilai α dengan persamaan (3.2)

$$\alpha = \frac{1-(-1)}{41483729-132593} = \frac{2}{42668925} = 0.0000000469$$

2) Cari nilai β dengan persamaan (3.3)

$$\beta = 1 - (0.0000000469 * 41483729)$$
$$\beta = 1 - 0.94559$$
$$\beta = -0.94559$$

3) Cari nilai x dengan persamaan (3.4)

$$x = \frac{(-0.8617-(-0.94559))}{0.0000000469} = \mathbf{1.788.699}$$

Dari hasil denormalisasi dapat diketahui bahwa data hasil prediksi jumlah pencari kerja adalah **1.788.699**

3.3.2 Analisis Metode *Backpropagation*

Backpropagation adalah algoritma pembelajaran untuk memperkecil tingkat *error* dengan cara menyesuaikan bobotnya berdasarkan perbedaan *output* dan target yang diinginkan. *Backpropagation* termasuk *multilayer network* yang merupakan perkembangan dari *single layer network.*

a. Arsitektur Jaringan Metode *Backpropagation*

Arsitektur metode *Backpropagation* terdiri dari tiga *layer* dalam proses pembelajarannya, yaitu *input layer*, *hidden layer* dan *output layer.*

1) *Input layer*

Input layer berfungsi menerima nilai dari ciri pola input di mana nilai dari *neuron* pada *input layer* adalah numerik. Banyaknya neuron pada *input layer* dipengaruhi oleh banyaknya pengambilan pola input yang akan dimasukan kedalam JST. Pada *input layer* tidak terjadi proses komputasi, namun terjadi pengiriman sinyal input x ke *hidden layer*.

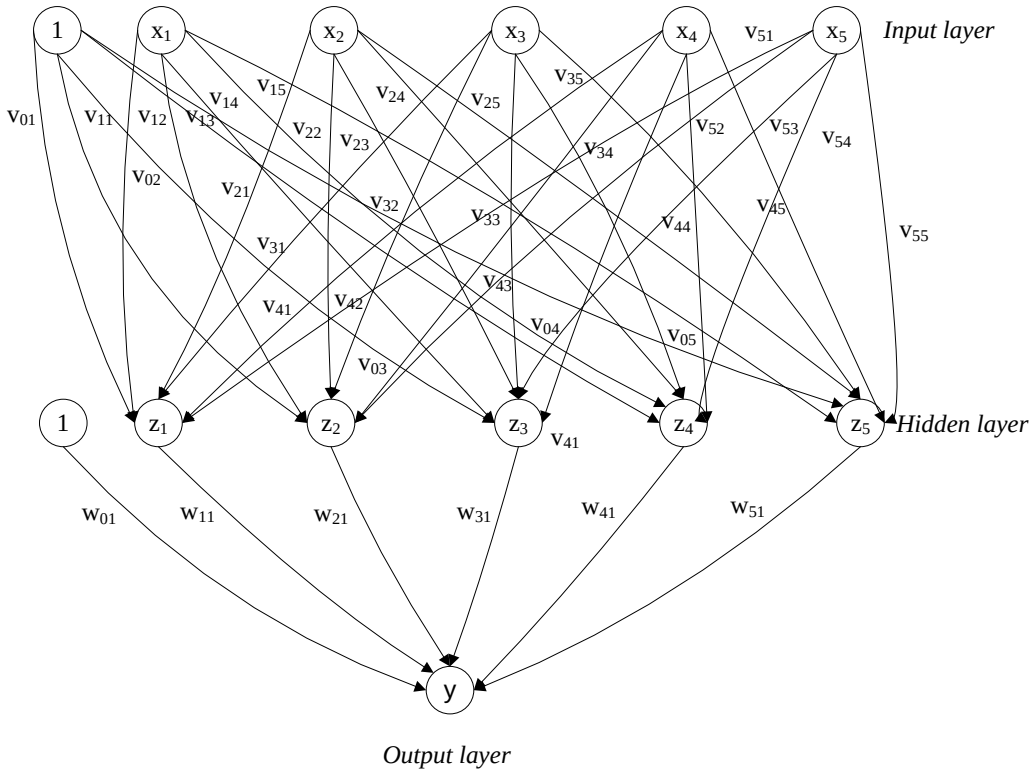
2) *Hidden layer*

Pada *hidden layer* terjadi proses komputasi terhadap bobot dan bias dan dihitung pula besarnya output dari *hidden layer* tersebut berdasarkan fungsi aktifasi tertentu. Untuk menentukan jumlah *hidden layer* tidak ada ketentuan yang pasti. Namun jumlah *hidden layer* berpengaruh terhadap akurasi dan kecepatan. Dengan adanya *hidden layer* dapat menyebabkan tingkat *error* pada *Backpropagation* lebih kecil dibanding tingkat *error* pada *single layer network*. karena *hidden layer* pada *Backpropagation* berfungsi sebagai tempat untuk meng-*update* dan menyesuaikan bobot, sehingga didapatkan nilai bobot yang baru yang bisa diarahkan mendekati dengan target *output* yang diinginkan.

3) *Output layer*

Banyaknya *layer* yang digunakan adalah 1 *layer*. Nilai *neuron* output merupakan bilangan *sigmoid biner*. *Output layer* terdiri dari beberapa *neuron output*. Kombinasi dari semua *neuron* tersebut dapat digunakan untuk mengidentifikasi dan juga *learning* sebagai *output* yang seharusnya.

Gambar 3.4 memperlihatkan arsitektur jaringan metode *Backpropagation* yang akan diimplementasikan.



Gambar 3.4 Arsitektur Jaringan Metode *Backpropagation*

Keterangan:

x_i : Input terdiri dari 5 *neuron* yaitu jumlah penduduk (x_1), jumlah lulusan (x_2), jumlah angkatan kerja (x_3), jumlah lowongan kerja (x_4), dan jumlah pengangguran terbuka (x5).

v_{ij} : Bobot pada lapisan tersembunyi.

w_{ij} : Bobot pada lapisan keluaran.

v_{0j} : Bias pada lapisan tersembunyi

w_{0j} : Bias pada lapisan lapisan keluaran.

i,j : 1,2,3,…..,n.

n : Jumlah *neuron* dalam suatu lapisan.

y : Output terdiri dari 1 *neuron* yaitu jumlah pencari kerja

1 : Konstanta bias.

b. Algoritma Pelatihan Metode *Backpropagation*

Berikut ini algoritma pelatihan metode *Backpropagation*:

1. Insialisasi bobot (ambil bilai random yang cukup kecil)
2. Selama kondisi berhenti bernilai salah, kerjakan:

   Tahap 1: Peramabatan Maju (*Feed Forward*)

   a. Setiap unit input (x_i, i=1,2,3,…,n) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua unit pada lapisan tersembunyi.
   b. Setiap unit tersembunyi (z_j, j=1,2,3,…,p) menjumlahkan bobot sinyal input dengan persamaan (2.22) Dan menerapkan fungsi aktifasi untuk menghitung sinyal output-nya dengan persamaan (2.23) kemudian mengirimkan sinyal tersebut ke semua unit output.
   c. Setiap unit output (y_k, k=1,2,3,…,m) menjumlahkan bobot sinyal input dengan persamaan (2.24)

      Dan menerapkan fungsi aktifasi untuk menghitung sinyal outputnya dengan persamaan (2.25)

   Tahap 2: Perambatan Balik *(Back Forward)*

   a. Setiap unit output (y_k, k=1,2,3,…,m) menerima pola target yang sesuai dengan pola input pembelajaran, kemudian hitung *error* dengan persamaan (2.26) kemudian kemudian hitung koreksi bobot dengan persamaan (2.27)

   b. Setiap unit tersembunyi (z_j, j=1,2,3,…,p) menjumlahkan delta input-nya dengan persamaan (2.28)

      Untuk menghitung informasi *error*, kalikan nilai ini dengan turunan dari fungsi aktifasinya dengan persamaan (2.29) kemudian hitung koreksi bobot dengan persamaan (2.30)

Setelah itu, hitung koreksi bias dengan persamaan (2.31)

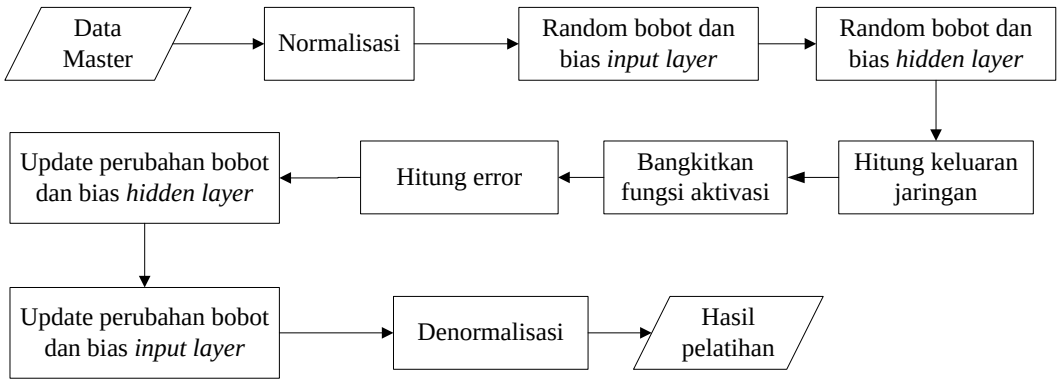
Tahap 3: Perubahan Bobot dan Bias

a. Setiap unit output (y_k, k=1,2,3,…,m) dilakukan perubahan bobot dan bias (j=0,1,2,…,p) dengan persamaan (2.32)

   Setiap unit tersembunyi (z_j, j=1,2,3,…,p) dilakukan perubahan bobot dan bias (i=0,1,2,…,n) dengan persamaan (2.33)

b. Tes kondisi berhenti.

Untuk lebih jelas, proses pelatihan metode *Backpropagation* diperlihatkan dalam alur sebagai berikut:



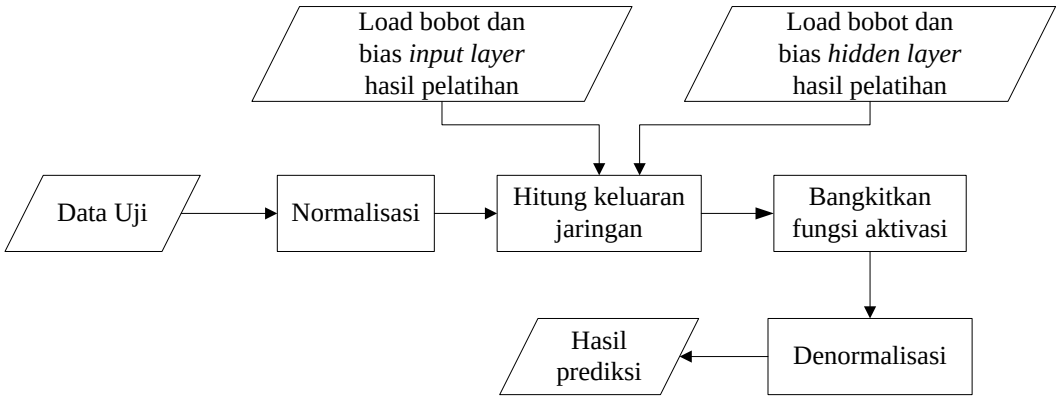
Gambar 3.5 Proses Pelatihan Metode *Backpropagation*

c. Algoritma Pengujian Metode *Backpropagation*

1. Load bobot dan bias *input layer* (v_{ij}) dan *hidden layer* (w_{ij}) hasil pelatihan.
2. Setiap unit input (x_i, i=1,2,3,…,n) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua unit pada lapisan tersembunyi.

3. Setiap unit tersembunyi (z_j, j=1,2,3,…,p) menjumlahkan bobot sinyal input dengan persamaan (2.22) dan menerapkan fungsi aktifasi untuk menghitung sinyal output-nya dengan persamaan (2.23)

4. Setiap unit output (y_k, k=1,2,3,…,m) menjumlahkan bobot sinyal input dengan persamaan (2.24) dan menerapkan fungsi aktifasi untuk menghitung sinyal output-nya dengan persamaan (2.25)



Gambar 3.6 Proses Pengujian Metode *Backpropagation.*

Secara lebih rinci, tiap bagian dari proses pelatihan dan pengujian *Backpropagation* dijelaskan di bagian analisis proses *Backpropagation*.

d. Contoh Perhitungan Pelatihan Metode *Backpropagation*

Berikut ini perhitungan pelatihan metode *Backpropagation*:

1. Insialisasi bobot (ambil nilai *random* yang cukup kecil)

   Bobot awal input ke *hidden layer* :

   v_{11}=0.1, v_{12}=0.1, v_{13}=0.2, v_{14}=0.2, v_{15}=0.3

   v_{21}=0.1, v_{22}=0.1, v_{23}=0.2, v_{24}=0.2, v_{25}=0.3

   v_{31}=0.2, v_{32}=0.2, v_{33}=0.2, v_{34}=0.2, v_{35}=0.3

   v_{41}=0.2, v_{42}=0.2, v_{43}=0.2, v_{44}=0.2, v_{45}=0.3

   v_{51}=0.3, v_{52}=0.3, v_{53}=0.2, v_{54}=0.2, v_{55}=0.3

Bobot awal bias ke *hidden layer* :

v_{01}=0.1, v_{02}=0.1, v_{03}=0.2, v_{04}=0.2, v_{05}=0.3

Bobot awal *hidden layer* ke *output layer*:

w_1=0.1, w_2=0.1, w_3=0.2, w_4=0.2, w_5=0.3

Bobot awal bias ke *output layer*:

w_0=0.1

Learning rate(α)=0.1

Maksium *Epoch*=1000

Error Rate = 0.01

2. Selama kondisi berhenti bernilai salah, kerjakan:

Tahap Perambatan Maju (*Forwar Propagation*)

a. Operasi pada *hidden layer* dengan persamaa (2.22)

$z_in_1 = v_{o1}+v_{11}*x_1+v_{21}*x_2*v_{31}*x_3+v_{41}*x_4+v_{51}*x_5$

= 0.1+0.1*0.9010576+0.1*0.0983054+0.2*0.3893004+ 0.2*0.0000000+0.3*0.0471746=0.29195

$z_in_2 = v_{o2}+v_{12}*x_1+v_{22}*x_2*v_{32}*x_3+v_{42}*x_4+v_{52}*x_5$

= 0.1+0.1*0.9358364+0.1*0.1052895+0.2*0.3987693+ 0.2*0.0002338+0.3*0.0469768=0.298006

$z_in_3 = v_{o3}+v_{13}*x_1+v_{23}*x_2*v_{33}*x_3+v_{43}*x_4+v_{53}*x_5$

= 0.2+0.2*0.9541503+0.2*0.1151646+0.2*0.4058121+ 0.2*0.0000767+0.2*0.0439141=0.503824

$z_in_4 = v_{o4}+v_{14}*x_1+v_{24}*x_2*v_{34}*x_3+v_{44}*x_4+v_{54}*x_5$

= 0.2+0.2*0.9716369+0.2*0.2029769+0.2*0.4268917+ 0.2*0.0025210+0.2*0.0553374=0.531873

$z_in_5 = v_{o5}+v_{15}*x_1+v_{25}*x_2*v_{35}*x_3+v_{45}*x_4+v_{55}*x_5$

= 0.3+0.3*0.9883034 +0.3*0.2210531+0.3*0.4387022+

0.3*0.0019111+0.3*0.0524634=0.81073

b. Fungsi aktifasi pada *hidden layer* dengan persamaan (2.24)

$$z_1 = \frac{1}{1+e^{-0.291950}} = 0.572473$$
$$z_2 = \frac{1}{1+e^{-0.298006}} = 0.573955$$
$$z_3 = \frac{1}{1+e^{-0.503824}} = 0.623357$$
$$z_4 = \frac{1}{1+e^{-0.531873}} = 0.629920$$
$$z_5 = \frac{1}{1+e^{-0.810730}} = 0.692265$$

c. Operasi pada output layer dengan persamaan (2.25)

$y_ink = w_0+w_1*z_1+w_2*z_2*w_3*z_3+w_4*z_4+w_5*z_5$

$= 0.1+0.1*0.572473+0.1*0.573955 +0.2*0.623357+$
$0.2*0.62992 +0.3*0.692265 =0.672978$

d. Fungsi aktifasi pada *output layer* dengan persamaan (2.26)

$$y = \frac{1}{1+e^{-0.672978}} = 0.66217$$

e. Cek *error* (iterasi berhenti bila |error|<0.01)

Error= (t-y) = 0.0983054 - 0.66217=-0.56386
Jumlah kuadrat *error* $=(-0.56386)^2=0.317943$

Tahap Perambatan-Balik (*Backpropagation*):

a. Setiap unit output menerima pola target yang sesuai dengan pola input pelatihan, kemudian hitung *error* dengan dengan persamaan (2.26)

$$\delta = (0.0983054 - 0.66217) * \frac{1}{1+e^{-0.672978}} + [1 - (\frac{1}{1+e^{-672978}})]$$

$= 0.288796$

b. Hitung koreksi bobot dengan persamaan (2.27)

Δw_1=0.1*0.288796*0.291950 = 0.008430

Δw_2=0.1*0.288796*0.298006 = 0.008606

Δw_3=0.1*0.288796*0.503824 = 0.014550

Δw_4=0.1*0.288796*0.291950 = 0.008431

Δw_5=0.1*0.288796*0.692265 = 0.019992

Δw_0=0.1*0.288796=0.028880

c. Setiap unit tersembunyi menjumlahkan delta input-nya dengan dengan persamaan (2.28)

δ_{in1}=0.288796*0.008430 = 0.002435

δ_{in2}=0.288796*0.008606 = 0.002485

δ_{in3}=0.288796*0.014550 = 0.004202

δ_{in4}=0.288796*0.008431 = 0.002435

δ_{in5}=0.288796*0.008431 = 0.002435

d. Hitung informasi *error* dengan persamaan (2.29)

$$\delta_1 = \delta_{in1} * \frac{1}{1+e^{-0.572473}} + [1 - (\frac{1}{1+e^{-0.572473}})]$$
$$= 0.364475$$
$$\delta_2 = \delta_{in2} * \frac{1}{1+e^{-0.573955}} + [1 - (\frac{1}{1+e^{-0.573955}})]$$
$$= 0.364210$$
$$\delta_3 = \delta_{in3} * \frac{1}{1+e^{-0.623357}} + [1 - (\frac{1}{1+e^{-0.623357}})]$$
$$= 0.355473$$
$$\delta_4 = \delta_{in4} * \frac{1}{1+e^{-0.62992}} + [1 - (\frac{1}{1+e^{-0.62992}})]$$
$$= 0.351261$$
$$\delta_5 = \delta_{in5} * \frac{1}{1+e^{-0.692265}} + [1 - (\frac{1}{1+e^{-0.692265}})]$$
$$= 0.337183$$

e. Hitung koreksi bobot dengan dengan persamaan (2.30)

Δv_{11}=0.1* 0.364475*0.9010576 = 0.032841

Δv_{12}=0.1* 0.364210*0.0983054 = 0.003580

Δv_{13}=0.1* 0.355473*0.3893004 = 0.013839

Δv_{14}=0.1* 0.351261*0.0000000 = 0.000000

Δv_{15}=0.1* 0.337183*0.0471746 = 0.001591

Δv_{21}=0.1* 0.364475*0.9358364 = 0.034109

Δv_{22}=0.1* 0.364210*0.1052895 = 0.003835

Δv_{23}=0.1* 0.355473*0.3987693 = 0.014175

Δv_{24}=0.1* 0.351261*0.0002338 = 0.000008

Δv_{25}=0.1* 0.337183*0.0469768 = 0.001584

Δv_{31}=0.1* 0.364475*0.9541503 = 0.034776

Δv_{32}=0.1* 0.364210*0.1151646 = 0.004194

Δv_{33}=0.1* 0.355473*0.4058121 = 0.014426

Δv_{34}=0.1* 0.351261*0.0000767 = 0.000003

Δv_{35}=0.1* 0.337183*0.0439141 = 0.001481

Δv_{41}=0.1* 0.364475*0.9716369 = 0.035414

Δv_{42}=0.1* 0.364210*0.2029769 = 0.007393

Δv_{43}=0.1* 0.355473*0.4268917 = 0.015175

Δv_{44}=0.1* 0.351261*0.0025210 = 0.000089

Δv_{45}=0.1* 0.337183*0.0553374 = 0.001866

Δv_{51}=0.1* 0.364475*0.9883034 = 0.036021

Δv_{52}=0.1* 0.364210*0.2210531 = 0.008051

Δv_{53}=0.1* 0.355473*0.4387022 = 0.015595

Δv_{54}=0.1* 0.351261*0.0019111= 0.000067

Δv_{55}=0.1* 0.337183*0.0524634 = 0.001769

f. Hitung koreksi bias dengan persamaan (2.31)

Δv_{01}=0.1* 0.364475 = 0.036448

Δv_{02}=0.1* 0.364210 = 0.036421

Δv_{03}=0.1* 0.355473 = 0.035547

Δv_{04}=0.1* 0.351261 = 0.035126

Δv_{05}=0.1* 0.337183 = 0.033718

g. Hitung perubahan bobot dan bias dengan persamaan (2.32) dan (2.33)

$v_{11(baru)}$=0.1+0.032841 = 0.132841

$v_{12(baru)}$=0.1+0.003580 = 0.103580

$v_{13(baru)}$=0.1+0.013839 = 0.113839

$v_{14(baru)}$=0.1+0.000000 = 0.100000

$v_{15(baru)}$=0.1+0.001591 = 0.101591

$v_{21(baru)}$=0.1+0.034109 = 0.134109

$v_{22(baru)}$=0.1+0.003835 = 0.103835

$v_{23(baru)}$=0.1+0.014175 = 0.114175

$v_{24(baru)}$=0.1+0.000008 = 0.100008

$v_{25(baru)}$=0.1+0.001584 = 0.101584

$v_{31(baru)}$=0.1+0.034776 = 0.134776

$v_{32(baru)}$=0.1+0.004194 = 0.104194

$v_{33(baru)}$=0.1+0.014426 = 0.114426

$v_{34(baru)}$=0.1+0.000003 = 0.100003

$v_{35(baru)}$=0.1+0.001481 = 0.101481

$v_{41(baru)}$=0.1+0.035414 = 0.135414

$v_{42(baru)}$=0.1+0.007393 = 0.107393

$v_{43(baru)}$=0.1+0.015175 = 0.115175

$v_{44(baru)}$=0.1+0.000089 = 0.100089

$v_{45(baru)}$=0.1+0.001866 = 0.101866

$v_{51(baru)}$=0.1+0.036021= 0.136021

$v_{52(baru)}$=0.1+0.008051= 0.108051

$v_{53(baru)}$=0.1+0.015595= 0.115595

$v_{54(baru)}$=0.1+0.000067= 0.100067

$v_{55(baru)}$=0.1+0.001769= 0.101769

$v_{01(baru)}$=0.1+0.036448= 0.136448

$v_{02(baru)}$=0.1+0.036421= 0.136421

$v_{03(baru)}$=0.1+0.035547= 0.135547

$v_{04(baru)}$=0.1+0.035126= 0.135126

$v_{05(baru)}$=0.1+0.033718= 0.133718

$w_{1(baru)}$=0.1+0.008430 = 0.108430

$w_{2(baru)}$=0.1+0.008606 = 0.108606

$w_{3(baru)}$=0.1+0.014550 = 0.114550

$w_{4(baru)}$=0.1+0.008431 = 0.108431

$w_{5(baru)}$=0.1+0.019992 = 0.119992

$w_{0(baru)}$=0.1+0.028880=0.128880

Perhitungan diteruskan sampai iterasi maksimum. Misalnya pada ahir pelatihan *Backpropagation*, diperoleh bobot input ke *hidden layer*.

v_{11}=0.01, v_{12}=0.02, v_{13}=0.03, v_{14}=0.04, v_{15}=0.05

v_{21}=0.01, v_{22}=0.02, v_{23}=0.03, v_{24}=0.04, v_{25}=0.05

v_{31}=0.01, v_{32}=0.02, v_{33}=0.03, v_{34}=0.04, v_{35}=0.05

v_{41}=0.01, v_{42}=0.02, v_{43}=0.03, v_{44}=0.04, v_{45}=0.05

v_{51}=0.01, v_{52}=0.02, v_{53}=0.03, v_{54}=0.04, v_{55}=0.05

Bobot awal bias ke *hidden layer*:

v_{01}=0.1, v_{02}=0.2, v_{03}=0.3, v_{04}=0.4, v_{05}=0.5

Bobot awal *hidden layer* ke *output layer*:

w_1=-0.2, w_2=-0.5, w_3=-0.1, w_4=0.01, w_5=-0.1

Bobot awal bias ke *output layer*:

w_0=-0.3

Setelah proses pelatihan selesai, *Backpropagation* dapat dipakai untuk pengujian. Berikut ini pangujian algoritma metdoe *Backpropagation.*

a. Operasi pada *Hidden Layer* dengan persamaan (2.22)

z_in_1=0.1+(0.01+**0.9010576**)+(0.01*0.0983054)+(0.01*0.3893004)+
(0.01*0.0000000)+ (0.01*0.0471746)

z_in_1=1.016405

b. Fungsi aktifasi pada *hidden layer* dengan persamaan (2.23)

$$z_1 = \frac{1}{1+e^{-1.016405}} = 0.734272$$

c. Operasi pada *output layer* dengan persamaan (2.24)

$y_ink=w_0+w_1*z_1+w_2*z_2*w_3*z_3+w_4*z_4+w_5*z_5$

y_ink = -0.3+(-0.2*0.572473)+(-0.5*0.573955)+(-0.1*0.623357)+
(0.01*0.62992)+(-0.1*0.692265)

= -0.8267

Berdasarkan pola data yang diujikan (pola data pada tahun 2007) dapat diketahui nilai min=132.593 dan max=41.483.729. Hasil pengujian berupa data yang diskla x'=-0.8267. Sedangkan berdasarkan fungsi identitas pada pengujian yang memiliki interval [-1,1] dapat diketahui *Upperbound*=1 dan *Lowerbound*=-1. Sehingga perhitungan denormalisasinya adalah sebagai berikut:

1) Cari nilai α dengan persamaan (3.2)

$$\alpha = \frac{1-(-1)}{41483729-132593} = \frac{2}{42668925} = 0.0000000469$$

2) Cari nilai β dengan persamaan (3.3)

$$\beta = 1 - (0.0000000469 * 41483729)$$
$$\beta = 1 - 0.94559$$
$$\beta = -0.94559$$

3) Cari nilai x dengan persamaan (3.4)

$$x = \frac{(-0.8267-(-0.94559))}{0.0000000469} = \mathbf{2.534.968}$$

Dari hasil denormalisasi dapat diketahui bahwa data hasil prediksi jumlah pencari kerja adalah **2.534.968**

Berdasarkan contoh perhitungan kedua metode JST tersebut, perbandingan hasilnya dapat perlihatkan di tabel berikut:

Tabel 3.10 Perbandingan Hasil Prediksi *Adaline* dan *Backpropation*

Tahun	Target	Hasil Prediksi		Selisih dengan Target	
		Adaline	Backpropagation	Adaline	Backpropagation
2007	2.841.321	1.788.699	2.534.968	1.052.622	306.353

Perhitungan akurasi *Adaline* dan *Backpropagation* terhadap target dihtung dengan rumus :

$$\left|\frac{\text{Hasil prediksi}-\text{Nilai sebenarnya}}{\text{Nilai sebenarnya}}\right|*100\%$$

Dengan demikian, akurasi hasil prediksi *Adaline* terhadap target dihitung sebagai berikut:

$$\left|\frac{1.788.699-2.841.321}{2.841.321}\right|*100\% = 37.05\%$$

$$100\%-37\%=62.95\%$$

Sedangkan akurasi hasil prediksi *Backpropagation* terhadap target dihitung sebagai berikut:

$$\left|\frac{2.534.968-2.841.321}{2.841.321}\right|*100\% =10.78\ \%$$

$$100\%-10.78\%=89.22\%$$

Hasil perbandingan akurasi metode *Adaline* dan *Backpropagation* terhadap target diperlihatkan di tabel 3.11

Tabel 3.11 Perbandingan Akurasi *Adaline* dan *Backporpagation*

Tahun	Target	Hasil Prediksi		Akurasi	
		Adaline	Backpropagation	Adaline	Backpropagation
2007	2.841.321	1.788.699	2.534.968	62.95%	89.22%

3.3.3 Komparasi Metode *Adaline* dan *Backpropagation*

Berdasarkan analisis metode *Adaline* dan *Backpropagation* yang meliputi algoritma dan studi kasus perhitungan pada tahap pelatihan dan pengujian, tabel berikut merangkum komparasi antara metode *Adaline* dan *Backpropagation*.

Tabel 3.12 Komparasi Metode *Adaline* dan *Backpropagation*

Parameter	***Adaline***	***Backpropagation***
Jenis Jaringan	*Single Layer*. Terdiri dari *input layer* dan *output layer*	*Multilayer*. Terdiri dari *input layer*, *hidden*, dan *output layer*
Alur Pelatihan	Menggunakan 1 alur yaitu alur maju (*feedforwad*)	Menggunakan 2 alur, yaitu alur maju (*forward propagation*) dan alur mundur (*backpropagation*)
Parameter Pelatihan	*Learning rate*, *error max*, dan iterasi maksimum	*Learning rate*, momentum, *error max*, iterasi maksimum, dan jumlah elemen *hidden layer*
Bobot Pelatihan	Memiliki 1 nilai bobot yaitu bobot antar *layer input* dan *layer output* (w)	Memiliki 2 nilai bobot yaitu bobot pada *hidden layer* (v) dan bobot pada *output layer* (w)
	Nilai awal setiap bobot diambil dari nilai *random*	Nilai awal setiap bobot diambil dari nilai *random*
	Perubahan bobot dihitung dengan aturan delta	Perubahan bobot dihitung dengan alur mundur (*backpropagation*), yaitu dengan mengubah nilai bobot pada *hidden layer* (w), kemudian mengubah nilai bobot pada *input layer* (v)
Perulangan Pelatihan	Perulangan dilakukan selama belum mencapai iterasi maksimum	Perulangan dilakukan selama belum mencapai iterasi maksimum
Fungsi Aktifasi	Menggunakan sigmoid bipolar	Menggunakan sigmoid binary
Kompleksitas Algoritma	Lebih sederhana karena hanya terdiri dari dua *layer*	Lebih kompleks karena terdiri dari tiga *layer*
Hasil Pelatihan	Berupa nilai bobot dan bias akhir *input layer* (w), *error max*, dan iterasi maksimum	Berupa nilai bobot dan bias pada *input layer layer* (v), bobot dan bias pada *hidden layer* (w), *error* max, dan

Parameter	*Adaline*	*Backpropagation*
		iterasi maksimum
Hasil Pengujian	Berupa angka jumlah pencari kerja yang sudah didenormalisasi	Berupa angka jumlah pencari kerja yang sudah didenormalisasi

3.4 Deskripsi Sistem

Artificial Intelligence (AI) merupakan salah satu bagian ilmu komputer yang membuat agar mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik manusia. AI dapat diterapkan untuk memprediksi data masa depan melalui proses pembelajaran terhadap data masa lalu. Pada topik ini, AI akan diimplementasikan untuk memprediksi data pencari kerja.

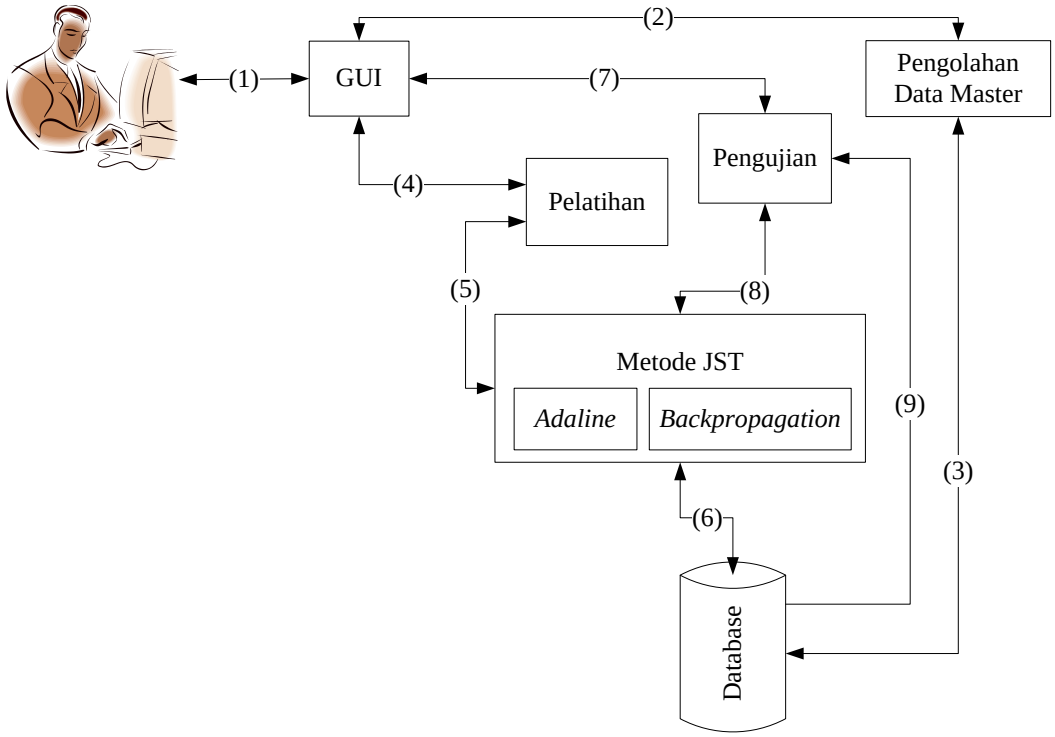
Terdapat 2 jenis metode pembelajaran pada JST, yaitu pembelajaran terawasi (*supervised learning*) dan pembelajaran tak terawasi (*unsupervised learning*). Karakteristik yang membedakaan antara pembelajaran terawasi dan tak terawasi dilihat dari ada atau tidaknya data target yang digunakan saat pelatihan. Untuk melakukan prediksi jumlah pencari kerja, pembelajaran terawasi lebih cocok karena menggunakan target keluaran, diantaranya yang termasuk metode pembelajaran terawasi adalah *Adaline* dan *Backpropagation*.

Metode *Adaline* adalah algoritma pembelajaran terawasi lapis tunggal di mana *layer* input langsung berhubungan dengan *layer* output. Aturan pembelajaran *Adaline* menggunakan aturan delta yang mengatur bobot-bobot untuk mengurangi selisih antara input jaringan ke unit output dengan output yang diinginkan. Hasilnya berupa *error* kuadarat pertengahan yang paling kecil.

Sedangkan metode *Backpropagation* adalah algoritma pembelajaran yang terawasi yang menggunakan pola penyesuaian bobot untuk mencapai nilai kesalahan yang minimum antara keluaran hasil prediksi dengan keluaran yang nyata dan memiliki satu atau lebih layer tersembunyi (*hidden layer*).

Sistem perbandingan metode *Adaline* dan *Backpropagation* dibutuhkan untuk dapat mengetahui dan merekomendasikan metode yang lebih baik dari segi akurasi dan waktu di antara kedua metode tersebut dalam memprediksi data jumlah pencari kerja. Metode *Adaline* dan *Backpropagation* memiliki langkah-

langkah yang sama dalam sistem prediksi. Data master (data yang digunakan untuk proses pelatihan) terlebih dahulu harus dipelajari untuk mendapatkan bobot yang kemudian disimpan kedalam *database* dan dapat digunakan pada proses pengujian data. Gambaran sistem secara umum dapat dilihat pada Gambar 3.7.



Gambar 3.7 Gambaran Umum Sistem

Gambar 3.7 mengilustrasikan alur aplikasi perbandingan metode *Adaline* dan *Backpopagation.* Pengguna membuka aplikasi (1) untuk melakukan pengolahan data master (2) seperti *insert*, *update*, dan *delete* data master kemudian disimpan ke database (3) dan ditampilkan ke pengguna melalui GUI pengolahan data master (2). Data master digunakan sebagai data pelatihan. Setelah data master tersedia, pengguna dapat melakukan pelatihan (4) dengan memilih jenis metode pelatihan JST (5). Hasil pelatihan disimpan ke database (6) dan ditampilkan ke pengguna melalui GUI pelatihan (4). Setelah melakukan pelatihan, pengguna dapat melakukan pengujian (7) dengan memilih jenis metode pengujian JST (8), sistem akan menge-*load* bobot hasil pelatihan (9) untuk melakukan pengujian.

Data hasil pengujian disimpan ke database (6) dan hasilnya ditampilkan ke pengguna melalui GUI pengujian (7). Semua hasil proses (pengolahan data master, pelatihan, dan pengujian) ditampilkan ke layar komputer (1) melalui GUI yang terkait.

3.5 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan *non fungsional* menjelaskan pendukung sistem yang akan dijalankan. Kebutuhan tersebut meliputi kebutuhan perangkat keras, kebutuhan perangkat lunak, dan kebutuhan user.

3.5.1 Kebutuhan Perangkat Keras

Komputer terdiri dari perangkat keras dan perangkat lunak yang saling berinteraksi. Perangkat lunak memberikan instruksi-instruksi kepada perangkat keras untuk melakukan suatu tugas tertentu, sehingga dapat menjalankan suatu sistem di dalamnya.

Pada aplikasi perbandingan metode JST ini, perangkat keras yang digunakan berupa sebuah laptop dengan spesifikasi sebagai yaitu:

1. Processor : 2.0 GHz
2. Memory : 3 GB
3. VGA : On board
4. Hardisk : 120 MB
5. Monitor : 11 inch
6. Mouse : Standar
7. Keyboard : Standar

Sedangkan untuk dapat menjalankan aplikasi perbandingan metode *Adaline* dan *Backrpopagation* dibutuhkan spesifikasi komputer pengguna yang direkomendasikan minimum sebagai berikut:

1. Processor : 1.2 GHz (lebih tinggi)
2. Memory : 512 MB (lebih tinggi)
3. VGA : On board (lebih tinggi)

4. Hardisk : 5 MB (*free*)
5. Monitor : 11 inch resulusi 1028 x 786
6. Mouse : Standar
7. Keyboard : Standar

3.5.2 Kebutuhan Perangkat Lunak

Perangkat lunak digunakan dalam sebuah sistem merupakan perintah-perintah yang diberikan kepada perangkat keras agar bisa saling berinteraksi di antara keduanya. Perangkat lunak yang dibutuhkan untuk membangun aplikasi ini adalah sebagai berikut :

1. Sistem Operasi Windows 7
2. Bahasa Pemrograman C#
3. Editor bahasa pemrograman Visual Studio 2010
4. Database MySQL 5.5.27, xampp 1.8.1 dan SQLYog v9.10 Beta 1.
5. *Library* ZedGraph 5.1.5
6. MySQL Connector/NET 6.6.5

Sedangkan kebutuhan perangkat lunak untuk pengguna direkomendasikan sebagai berikut:

1. Sistem Operasi Windows XP/Windows 7
2. Install aplikasi perbandingan metode *Adaline* dan *Backpropagation*

3.5.3 Kebutuhan User

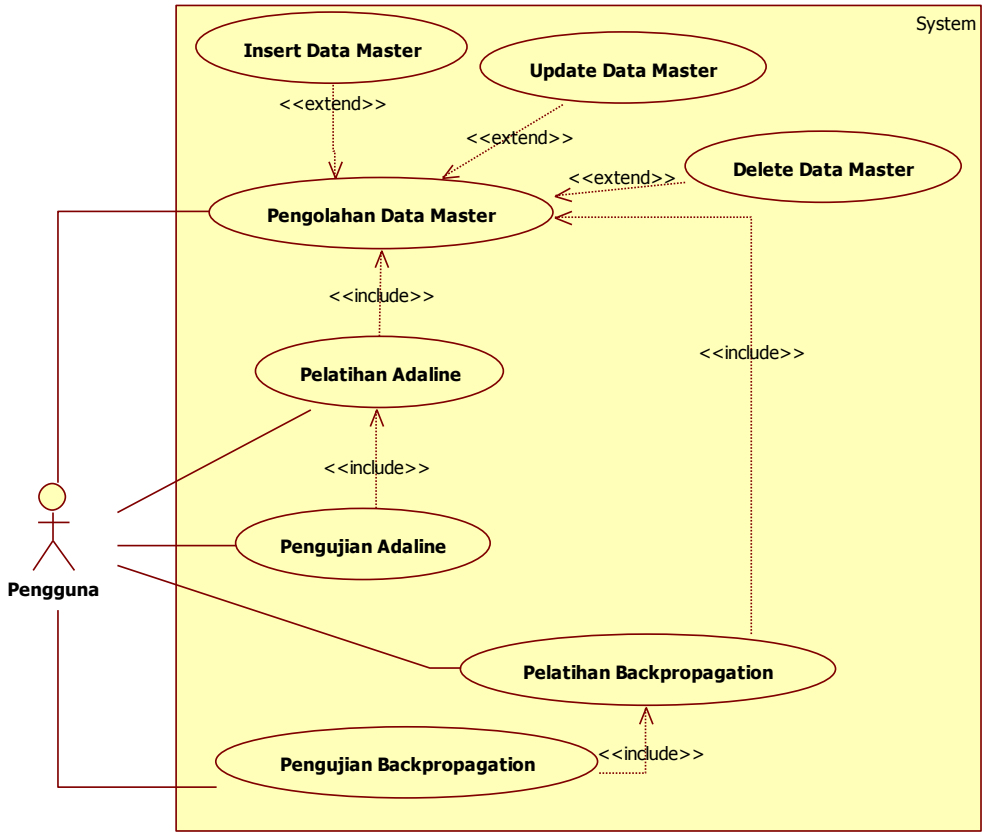
Untuk menjalankan aplikasi dibutuhkan spesifikasi user yang dapat menjalankan fungsi-fungsi dari aplikasi ini. Hanya terdapat satu jenis user dalam aplikasi ini, yaitu *Pengguna* dengan spesifikasi dapat mengoperasikan komputer, mengerti parameter input pelatihan, mengerti data masukan, dan dapat membaca data hasil pelatihan dan prediksi.

3.6 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menjelaskan proses-proses yang terjadi di dalam aplikasi ini. Proses-proses tersebut dijelaskan dalam *use case diagram*, *skenario diagram, activity diagram, sequence diagram* dan *class diagram*.

3.6.1 *Use Case Diagram*

Use case diagram menggambarkan proses yang ada di dalam aplikasi ini. *Use Case diagram* sistem diperlihatkan pada Gambar 3.8.



Gambar 3.8 Use Case Diagram Perbandingan *Adaline* dan *Backpropagation*

3.6.2 *Use Case* Skenario

Use Case skenario menjelaskan skenario dari setiap proses yang terdapat pada *use case diagram*. *Use case* skenario pada aplikasi ini dijelaskan pada Tabel 3.13 sampai 3.14

Tabel 3.13 *Use Case* Skenario Pengolahan Data Master

Nama *Use Case*	Pengolahan Data Master
Deskripsi	Pengguna dapat mengelolah data master
Aktor	Pengguna
Kondisi Awal	Sumber pola data input tersedia
Kondisi Akhir	Sistem menyimpan perubahan data master
Extend	Insert Data Master, Update Data Master, Delete Data Master
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pengguna berada pada tampilan sistem perbandingan metode JST	
2. Pengguna memilih menu data master	3. Menampilkan form menu data master
Alternatif	
3. Jika pengguna ingin mengelolah data master: a. Jika pengguna ingin menambah data master, <<*extend*>> Insert Data Master b. Jika pengguna ingin mengubah data master, <<*extend*>> Update Data Master c. Jika pengguna ingin menghapus data master, <<*extend*>> Delete Data Master	

Tabel 3.14 *Use Case* Skenario *Insert Data Master*

Nama *Use Case*	Insert Data Master
Deskripsi	Pengguna dapat mengentrikan pola data baru
Aktor	Pengguna
Kondisi Awal	Pola data baru belum ada
Kondisi Akhir	Menyimpan pola data baru
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Memilih insert pola data baru	
2. Mengisi pola data baru	
3. Memilih tombol **Simpan**	4. Melakukan validasi data
	5. Menyimpan pola data baru ke database.
	6. Menampilkan pesan "Data sudah disimpan".
Alternatif	
3. Pengguna dapat mengisi pola data baru tidak harus berurutan	
Eksepsi	

4. Jika data tidak valid: d. Jika data belum lengkap, menampilkan pesan "Data harus diisi", kemudian kembali ke tahap 2. e. Jika format data bukan angka, menampilkan pesan "Data harus angka", kemudian kembali ke tahap 2. f. Jika data tahun sudah tersedia, menampilkan pesan, "Data tahun sudah tersedia", kemudian kembali ke tahap 2.

Tabel 3.15 *Use Case* Skenario *Update Data Master*

Nama *Use Case*	Update Data Master	
Deskripsi	Pengguna dapat mengubah pola data master	
Aktor	Pengguna	
Kondisi Awal	Data master tersedia dan pola data tidak sesuai	
Kondisi Akhir	Menyimpan perubahan pola data master	
Skenario Utama		
Aksi Aktor		Reaksi Sistem
1. Memilih pola data yang ingin diubah		
2. Mengganti dengan pola data baru		
3. Memilih tombol **Ubah**		4. Melakukan validasi
		5. Menyimpan perubahan pola data baru ke database
		6. Menampilkan pesan "Data sudah diubah"
Kondisi Akhir	Perubahan pola data disimpan	
Alternatif		
2. Pengguna dapat mengubah pola data tidak harus berurutan		
Eksepsi		
4. Jika data tidak valid: a. Jika data belum lengkap, menampilkan pesan "Data harus diisi", kemudian kembali ke tahap 2. b. Jika format data bukan angka, menampilkan pesan "Data harus angka", kemudian kembali ke tahap 2. c. Jika data tahun sudah tersedia, menampilkan pesan, "Data tahun sudah tersedia", kemudian kembali ke tahap 2.		

Tabel 3.16 *Use Case* Skenario *Delete*

Nama *Use Case*	*Delete Data Master*
Deskripsi	Pengguna dapat menghapus pola data master
Aktor	Pengguna
Kondisi Awal	Data master tersedia
Kondisi Akhir	Menghapus pola data
Include	Data Master
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Melakukan *double click* pada data yang akan dihapus.	2. Menampilkan konfirmasi "Apakah yakin akan menghapus data?"
3. Memilih tombol **Yes**	4. Menampilkan pesan "Pola data telah dihapus".
Alternatif	
4. Jika pengguna memilih tombo **No**, pola data tidak dihapus dan kembali ke form data master	

Tabel 3.17 *Use Case* Skenario Pelatihan *Adaline*

Nama *Use Case*	*Pelatihan Adaline*
Deskripsi	Pengguna melakukan pelatihan dengan metode *Adaline*
Aktor	Pengguna
Kondisi Awal	Data master tersedia dan data pelatihan *Adaline* belum ada
Kondisi Akhir	Menampilkan data hasil pelatihan *Adaline*
Include	Pengolahan Data Master
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Memilih pelatihan *Adaline*	2. Menampilkan form pelatihan *Adaline*
3. Mengisi parameter pelatihan *Adaline*	
4. Memilih tombol **Pelatihan**	5. Melakukan validasi semua data parameter
	6. Load data master
	7. Normalisasi
	8. Random bobot input layer
	9. Hitung keluaran jaringan
	10. Bangkitkan fungsi aktifasi
	11. Hitung error
	12. Hitung perubahan bobot dan bias
	13. Denormalisasi
	14. Tampilkan data hasil pelatihan
	15. Tampilkan grafik hasil pelatihan

	16. Menampilkan pesan "Data pelatihan Adaline selesai"
17. Menyimpan hasil pelatihan *Adaline*	18. Menyimpan data hasil pelatihan *Adaline*
	19. Menampilkan pesan "Data sudah disimpan"
Alternatif	
2. Pengguna dapat mengisi data parameter tidak harus berurutan	
Eksepsi	
4. Jika parameter tidak valid: a. Jika data belum lengkap, menampilkan pesan "Data harus diisi", kemudian kembali ke tahap 3. b. Jika format data bukan angka, menampilkan pesan "Data harus angka", kemudian kembali ke tahap 3.	

Tabel 3.18 *Use Case* Skenario Grafik *Error Adaline*

Nama *Use Case*	Grafik *Error Adaline*
Deskripsi	Pengguna dapat melihat grafik *error* pelatihan *Adaline*
Aktor	Pengguna
Kondisi Awal	Pelatihan *Adaline* sudah dilakukan dan tombol **Lihat Grafik Error** *enabled*
Kondisi Akhir	Pengguna melihat grafik *error* pelatihan *Adaline*
Include	*Pelatihan Adaline*
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Berada pada form pelatihan *Adaline*	
2. Memilih tombol **Lihat Grafik Error**	3. *<<include>> Pelatihan Adaline*
	4. Menampilkan grafik *error* pelatihan *Adaline*
5. Melihat grafik *error* pelatihan *Adaline*	
6. Memilih tombol ***Close***	7. Menutup grafik *error* pelatihan *Adaline* dan kembali ke form pelatihan *Adaline*

Tabel 3.19 *Use Case* Skenario Pelatihan *Backpropagation*

Nama *Use Case*	Pelatihan *Backpropagation*
Deskripsi	Pengguna melakukan pelatihan dengan metode *Backpropagation*
Aktor	Pengguna
Kondisi Awal	Data master tersedia dan data pelatihan *Backpropagation* belum ada
Kondisi Akhir	Menampilkan data hasil pelatihan *Backpropagation*
Include	Pengolahan Data Master
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Berada pada form pelatihan *Backpropagation*	
2. Mengisi parameter pelatihan *Backpropagation*	
3. Memilih tombol **Proses**	4. Melakukan validasi semua data parameter.
	5. Random bobot *input layer*
	6. Random bobot *hidden layer*
	7. Normalisasi
	8. Hitung keluaran jaringan dari *input layer*
	9. Bangkitkan fungsi aktifasi
	10. Hitung keluaran jaringan dari *hidden layer*
	11. Bangkitkan fungsi aktifasi
	12. Hitung perubahan bobot dan bias *hidden layer*
	13. Hitung perubahan bobot dan bias *input layer*
	14. Denormalisasi
	15. Tampilkan data hasil pelatihan
	16. Tampilkan grafik hasil pelatihan
	17. Menampilkan pesan "Data pelatihan Adaline selesai"
18. Menyimpan hasil pelatihan *Backpropagation*	19. Menyimpan data hasil pelatihan *Backpropagation*
	20. Menampilkan pesan "Data sudah disimpan"
Alternatif	
2. Pengguna dapat mengisi data parameter tidak harus berurutan	
Eksepsi	
4. Jika parameter tidak valid: a. Jika data belum lengkap, menampilkan pesan "Data harus diisi",	

kemudian kembali ke tahap 2. b. Jika format data bukan angka, menampilkan pesan “Data harus angka”, kemudian kembali ke tahap 2.

Tabel 3.20 *Use Case* Skenario Grafik *Error Backpropagation*

Nama *Use Case*	Grafik *Error Backpropagation*	
Deskripsi	Pengguna dapat melihat grafik *error* pelatihan *Backpropagation*	
Aktor	Pengguna	
Kondisi Awal	Pelatihan *Backpropagation* sudah dilakukan dan tombol **Lihat Grafik Error** *enabled*	
Kondisi Akhir	Pengguna melihat grafik *error* pelatihan *Backpropagation*	
Skenario Utama		
Aksi Aktor		Reaksi Sistem
1. Berada pada form pelatihan *Backpropagation*		
2. Memilih tombol **Lihat Grafik Error**		3. <<*include*>> *Pelatihan Backpropagation*
		4. Menampilkan grafik *error* pelatihan *Backpropagation*
5. Melihat grafik *error* pelatihan *Backpropagation*		
6. Memilih tombol ***Close***		7. Menutup grafik *error* pelatihan *Backpropagation* dan kembali ke form pelatihan *Backpropagation*

Tabel 3.21 *Use Case* Skenario Pengujian *Adaline*

Nama *Use Case*	Pengujian *Adaline*	
Deskripsi	Pengguna dapat menguji metode *Adaline* untuk prediksi	
Aktor	Pengguna	
Kondisi Awal	Data pelatihan *Adaline* tersedia dan data prediksi *Adaline* belum ada	
Kondisi Akhir	Menampilkan hasil pengujian *Adaline*, *m*enyimpan hasil pengujian *Adaline*, dan menampilkan grafik hasil pengujian *Adaline*	
Skenario Utama		
Aksi Aktor		Reaksi Sistem
1. Memilih menu pengujian *Adaline*		
		2. Menampilkan form pengujian *Adaline*
3. Mengisi pola data master		
4. Memilih tombol **Proses**		5. Melakukan validasi data.

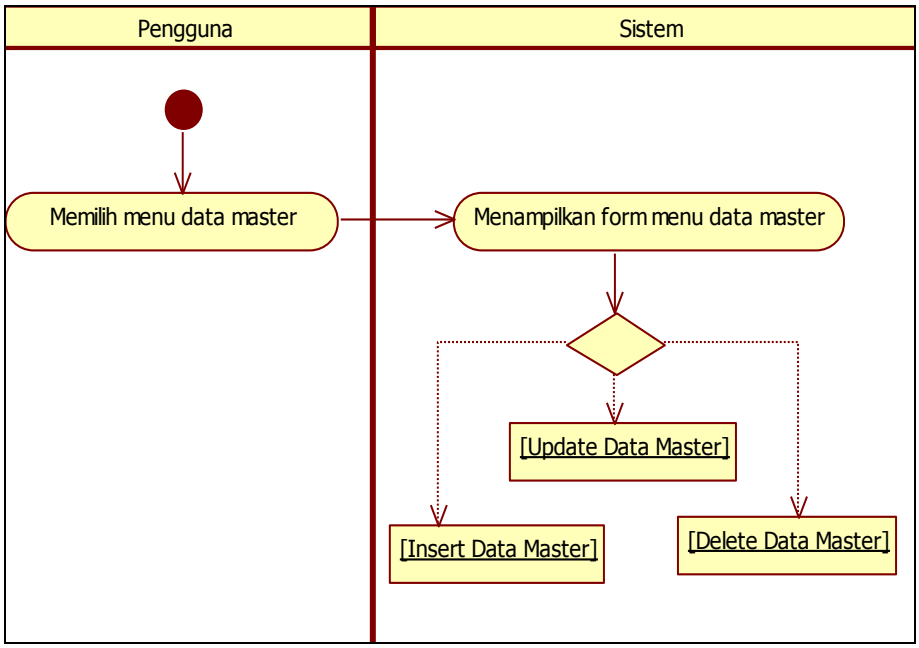
	6. Menampilkan data hasil pengujian *Adaline*
	7. Menampilkan grafik hasil pengujian *Adaline*
8. Memilih tombol **Simpan**	9. Menyimpan data hasil pelatihan *Adaline*
Eksepsi	
6. Melakukan validasi data: a. Jika data belum lengkap, menampilkan pesan "Data harus diisi", kemudian kembali ke tahap 3. b. Jika format data bukan angka, menampilkan pesan "Data harus angka", kemudian kembali ke tahap 3.	

Tabel 3.22 *Use Case* Skenario Pengujian *Backpropagation*

Nama *Use Case*	Pengujian *Backpropagation*
Deskripsi	Pengguna dapat menguji metode *Backpropagation* untuk prediksi
Aktor	Pengguna
Kondisi Awal	Data pelatihan *Backpropagation* tersedia dan data prediksi *Backpropagation* belum ada
Kondisi Akhir	Menampilkan hasil pengujian *Backpropagation, m*enyimpan hasil pengujian *Backpropagation,* dan menampilkan grafik hasil pengujian *Backpropagation*
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Memilih menu pengujian *Backpropagation*	
	2. Menampilkan form pengujian *Backpropagation*
3. Mengisi pola data master	
4. Memilih tombol **Proses**	5. Melakukan validasi data
	6. Menampilkan data hasil pengujian *Backpropagation*
	7. Menampilkan grafik hasil pengujian *Backpropagation*
8. Memilih tombol **Simpan**	9. Menyimpan data hasil pelatihan *Backpropagation*
Eksepsi	
7. Melakukan validasi data: a. Jika data belum lengkap, menampilkan pesan "Data harus diisi", kemudian kembali ke tahap 3. b. Jika format data bukan angka, menampilkan pesan "Data harus angka", kemudian kembali ke tahap 3.	

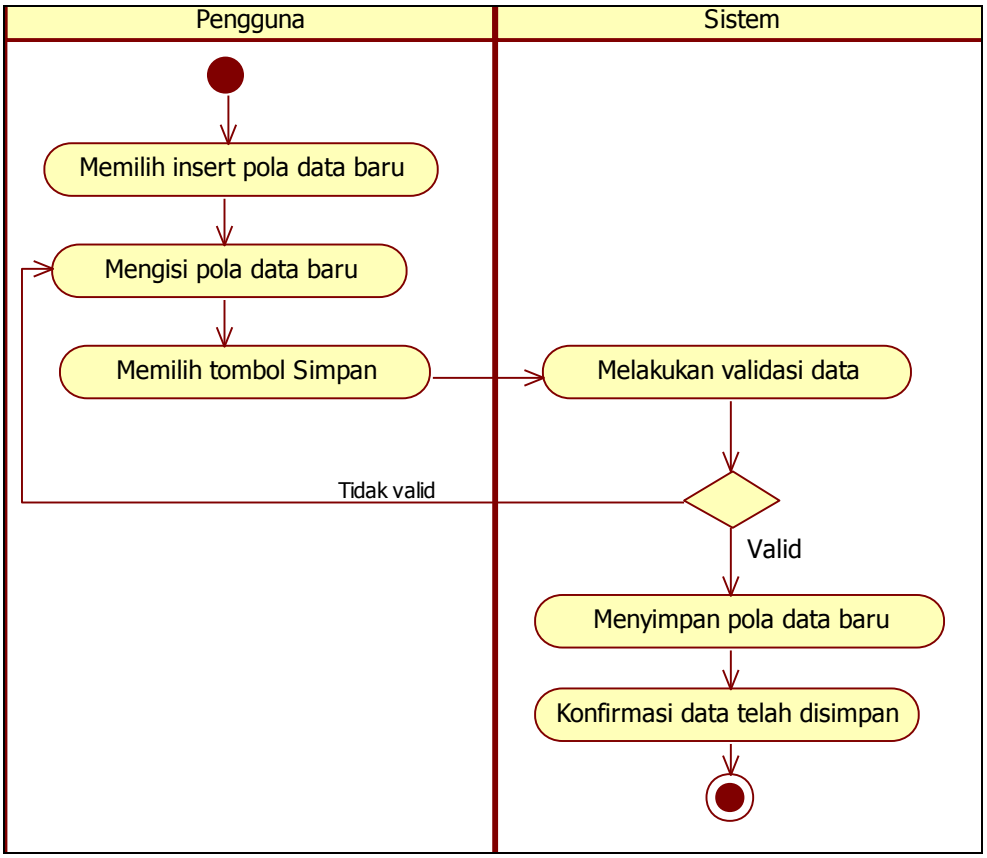
3.6.3 *Activity Diagram*

Secara umum *activity diagram* memiliki fungsi yang sama dengan *use case* spesifikasi yaitu untuk menjelaskan *use case* secara lebih rinci. Perbedaan keduanya terletak pada cara pandang dalam hal menjelaskan *use case* tersebut. Pada *use case* spesifikasi, *use case* dijelaskan dengan narasi yang tanggung jawabnya dibuat terpisah antara pengguna dan reaksi sitem. Skenario utama pada *use case* spesifikasi identik dengan penulisan algoritma dalam bentuk naratif. Sedangkan pada *activity diagram, use case* dijelaskan dalam bentuk diagram alir yang juga memisahkan tanggung jawab pengguna dan sistem. *Activity diagram* identik dengan penulisan algoritma dalam bentuk *flowchart*. Gambar 3.9 sampai 3.18 menujukan *activity diagram* dari sistem yang akan dibuat.



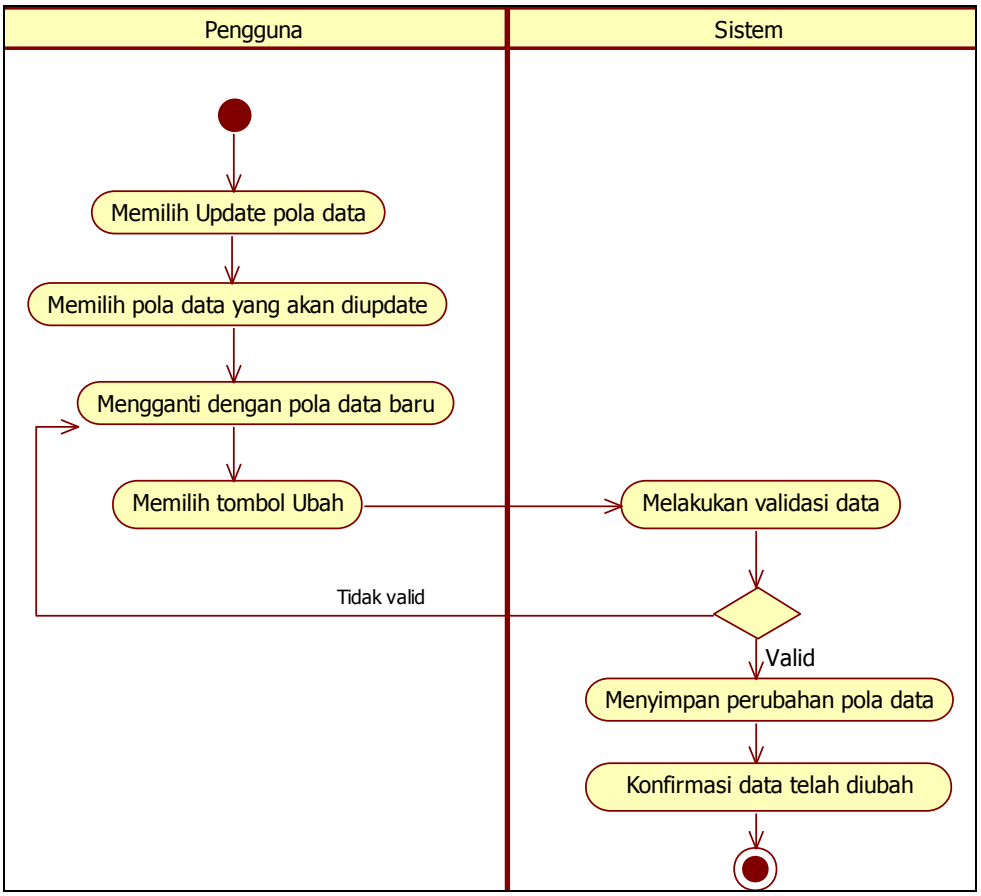
Gambar 3.9 *Activity Diagram* Data Master

Gambar 3.9 menunjukkan alur untuk mengelola data master. Pengguna memilih menu data master, sistem akan menampilkan form data master. Di dalam form data master, pengguna dapat memilih jenis kelola data master yang diinginkan dengan pilihan insert data master, update data master, dan delete data master. Jenis-jenis kelola tersebut, dijelaskan di *activtity diagram* tersendiri.



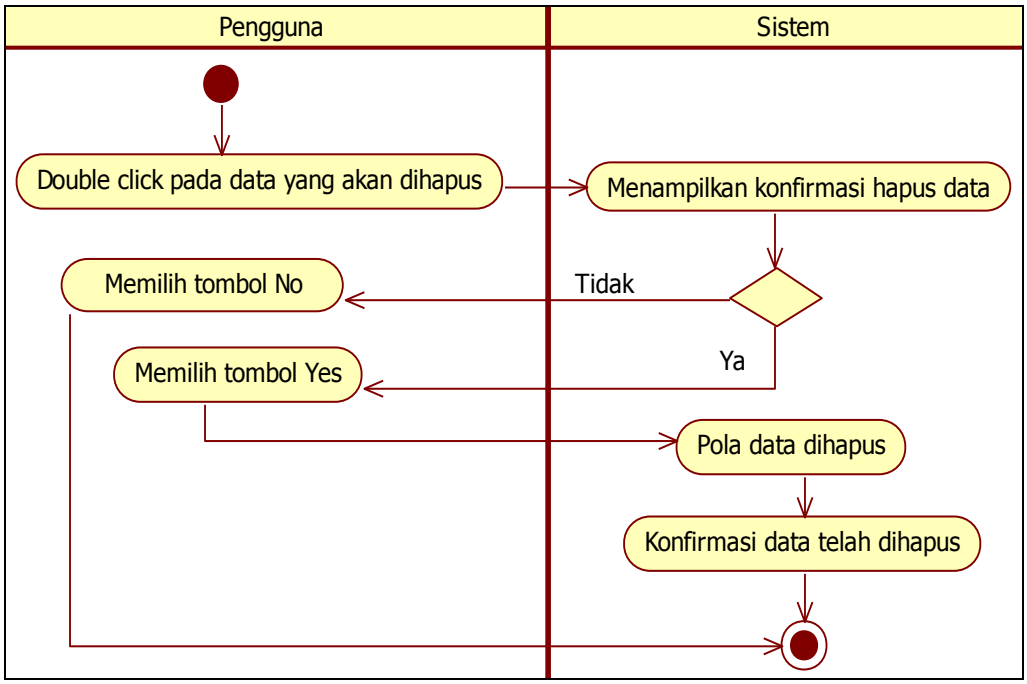
Gambar 3.10 *Activity Diagram Insert*

Gambar 3.10 menunjukkan alur untuk insert pola data baru ke data master. Pengguna memasukan pola data ke form data master kemudian memilih tombol **Simpan**. Sistem akan melakukan validasi terhadap data yang diinputkan oleh pengguna. Jika data tidak valid, pengguna memperbaiki pola data baru yang diinputkan. Jika data valid, sistem akan menyimpan pola data baru ke *database* lalu menampilkan pesan konfirmasi bahwa data sudah disimpan. Pengguna dapat melihat perubahan data di list data master.



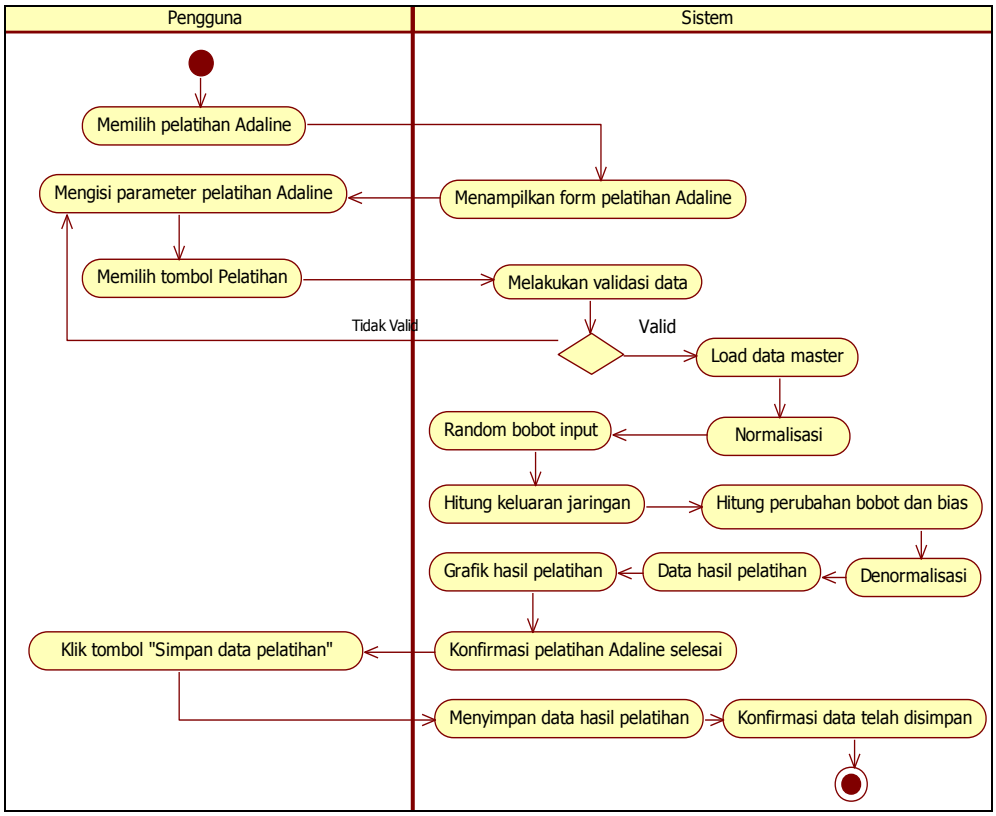
Gambar 3.11 *Activity Diagram Update*

Gambar 3.11 menunjukkan alur untuk mengubah pola data dari data master. Pengguna memilih pola data yang akan diupdate dari list data master. Data yang dipilih oleh pengguna akan ditampilkan di form data master yang sama dengan insert data master. Melalui form tersebut, pengguna mengubah pola data lama dengan pola data baru kemudian memilih tombol **Ubah**. Sistem akan melakukan validasi terhadap data yang diinputkan oleh pengguna. Jika data tidak valid, pengguna memperbaikin pola input data baru. Jika data valid, sistem akan menyimpan pola data baru ke *database* lalu menampilkan pesan konfirmasi bahwa data sudah diubah. Pengguna dapat melihat perubahan data di list data master.



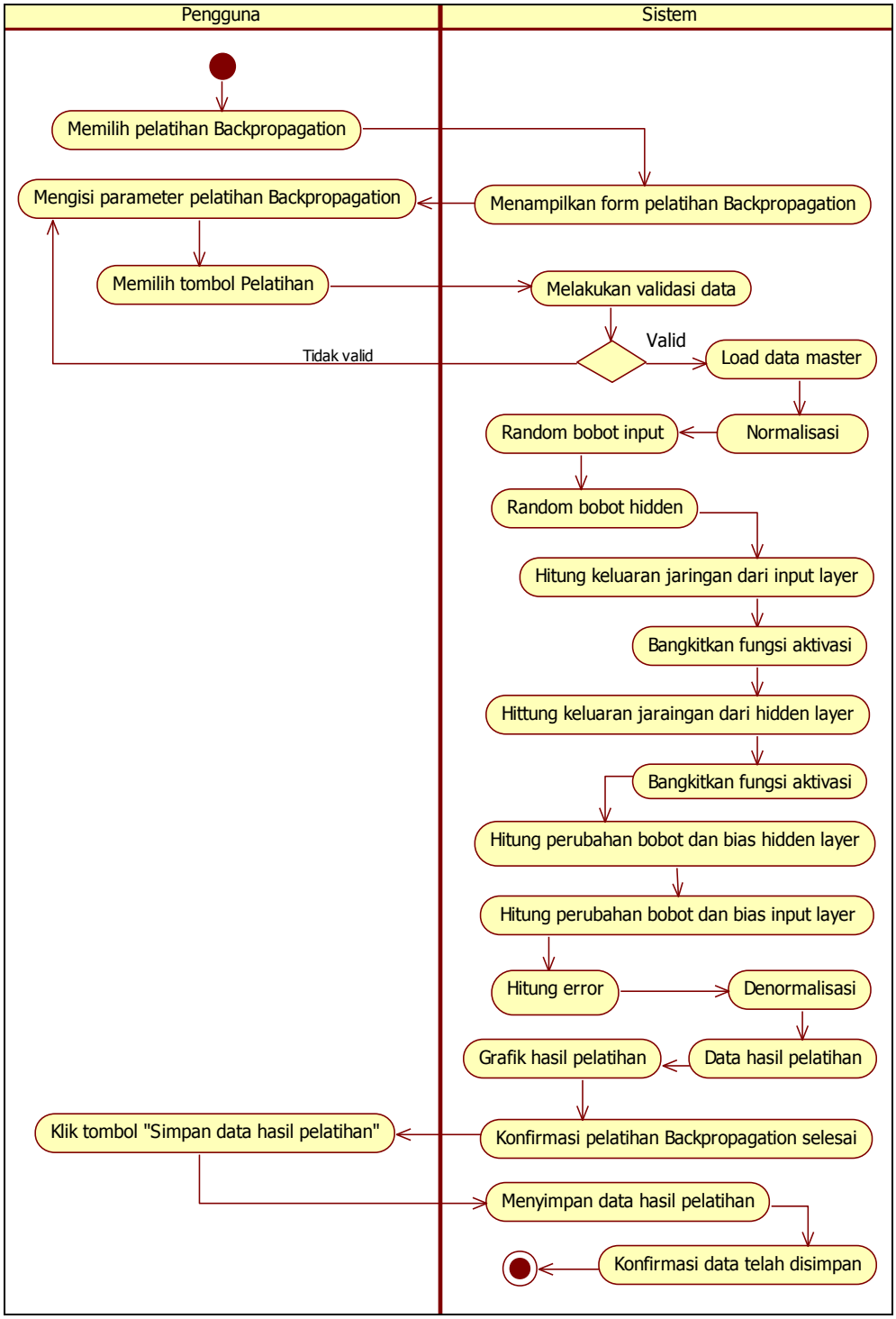
Gambar 3.12 *Activity Diagram Delete*

Gambar 3.12 menunjukkan alur untuk menghapus pola data dari data master. Pengguna memilih pola data yang akan dihapus dari list data master dengan cara *double click* pada data yang akan dihapus. Sistem akan menampilkan konfirmasi penghapusan data. Jika pengguna menekan tombol **Yes**, data akan dihapus dari *database*, jika memilih **No**, pola data tidak jadi dihapus.



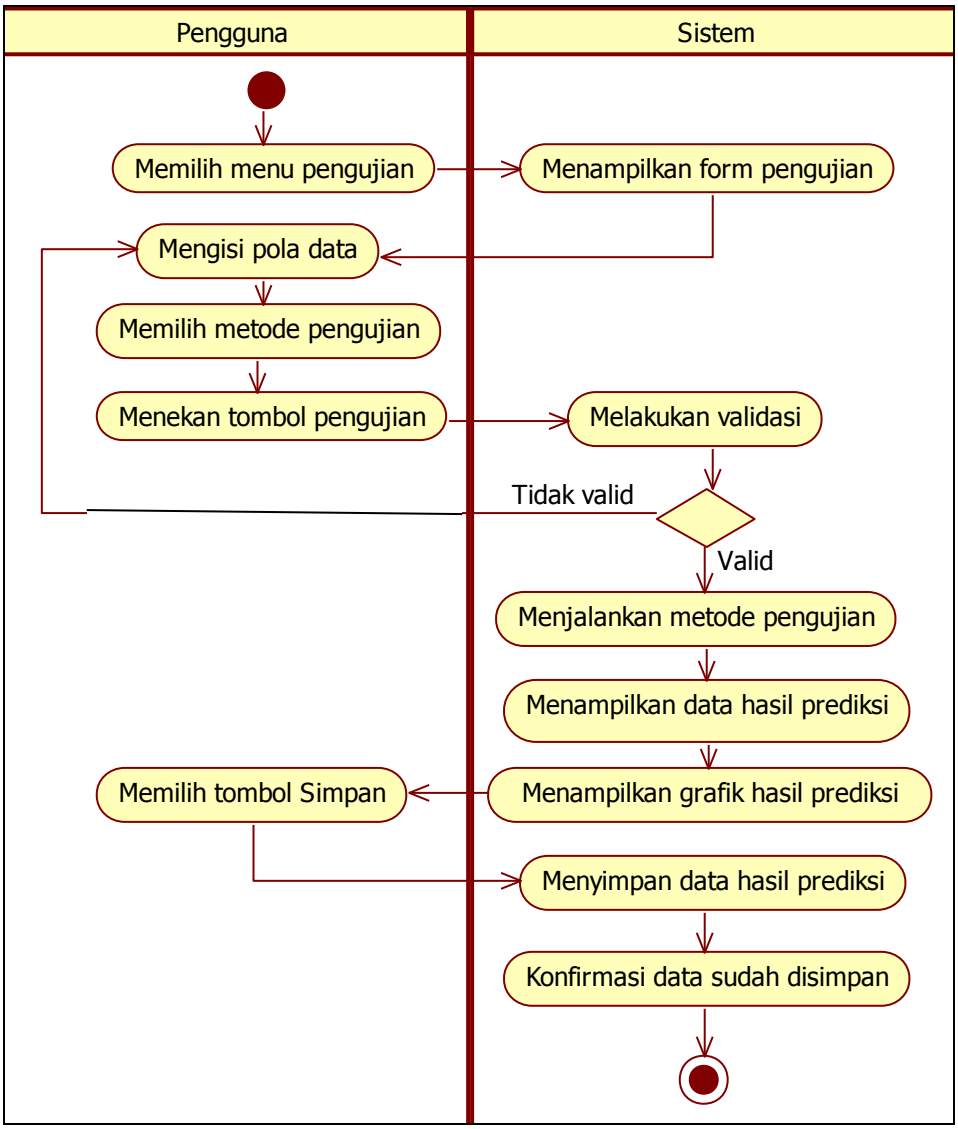
Gambar 3.13 *Activity Diagram* Pelatihan *Adaline*

Gambar 3.13 menunjukkan alur untuk pelatihan *Adaline*. Pengguna memilih metode pelatihan *Adaline*, sistem menampilan form pelatihan *Adaline*. Pengguna mengisi parameter pelatihan kemudian memilih tombol **Proses**. Sistem akan melakukan validasi terhadap data yang diinputkan oleh pengguna. Jika data tidak valid, sistem akan menampilkan pesan konfirmasi bahwa data yang dimasukkan oleh pengguna tidak valid agar pengguna memperbaikinya. Jika data valid, sistem akan menjalankan proses pelatihan *Adaline* yang meliputi *load* data master, normalisasi data master, random bobot input, hitung keluaran jaringan, bangkitkan fungsi aktifasi, hitung *error*, hitung perubahan bobot dan bias, denormalisasi, menampilkan data hasil pelatihan, dan menampilkan grafik pelatihan *Adaline*. Pengguna dapat menyimpan data hasil pelatihan untuk basis perhitungan pengujian *Adaline*. Sistem akan menampilkan konfirmasi bahwa data hasil pelatihan sudah disimpan.



Gambar 3.14 *Activity Diagram* Pelatihan *Backpropagation*

Gambar 3.15 menunjukkan alur untuk pelatihan *Backpropagation*. Pengguna memilih metode pelatihan *Backpropagation,* sistem menampilan form pelatihan *Backpropagation*. Pengguna mengisi parameter pelatihan kemudian memilih tombol **Proses**. Sistem akan melakukan validasi terhadap data yang diinputkan oleh pengguna. Jika data tidak valid, sistem akan menampilkan pesan konfirmasi bahwa data yang dimasukkan oleh pengguna tidak valid agar pengguna memperbaikinya. Jika data valid, sistem akan menjalankan proses pelatihan *Backpropagation* yang meliputi *load* data master, normalisasi data master, random bobot *input layer*, random bobot *hidden layer*, hitung keluaran jaringan *input layer*, bangkitkan fungsi aktifasi, hitung keluaran jaringan *hidden layer*, bangkitkan fungsi aktifasi, hitung *error*, hitung perubahan bobot dan bias *hidden layer*, hitung perubahan bobot dan bias *input layer*, denormalisasi, menampilkan data hasil pelatihan, dan menampilkan grafik pelatihan *Backpropagation.* Pengguna dapat menyimpan data hasil pelatihan untuk basis perhitungan pengujian *Backpropagation*. Sistem akan menampilkan konfirmasi bahwa data hasil pelatihan sudah disimpan.



Gambar 3.15 *Activity Diagram* Pengujian

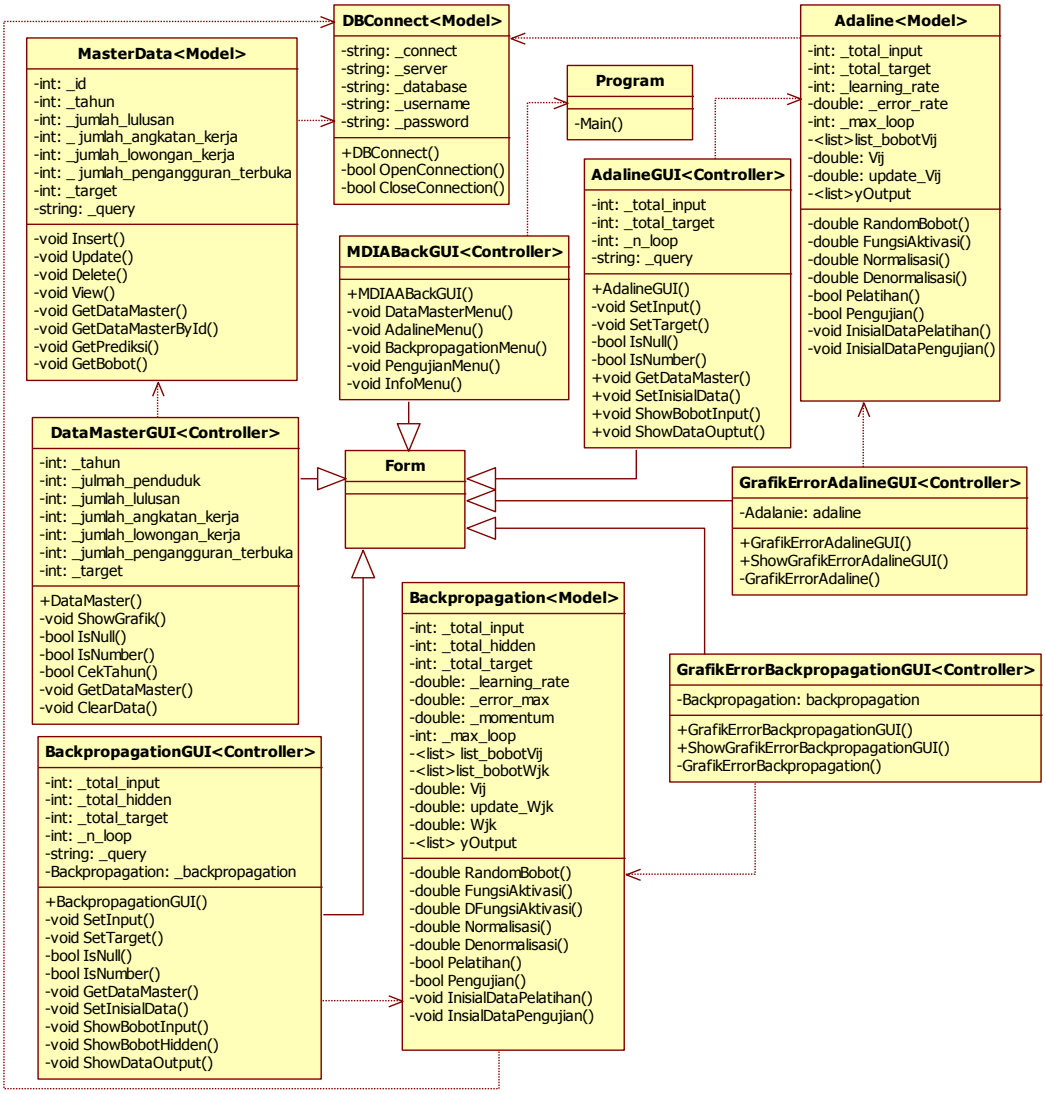
Gambar 3.15 menunjukkan alur untuk pengujian. Pengguna memilih menu pengujian, sistem menampilan form pengujian. Pengguna mengisi pola data input dan memilih metode pengujian kemudian menekan tombol **Proses**. Sistem akan menge-*load* bobot hasil pelatihan sesuai dengan metode pengujian yang dipilih. Sistem akan melakukan validasi terhadap data yang diinputkan oleh pengguna. Jika data tidak valid, sistem akan menampilkan pesan konfirmasi bahwa data yang dimasukkan oleh pengguna tidak valid agar pengguna memperbaikinya. Jika data valid, sistem akan menjalankan proses pengujian kemudian menampilkan data hasil prediksi. Pengguna dapat menyimpan data hasil pengujian dengan memilih

tombol **Simpan**. Sistem akan menyimpan data hasil prediksi di *database* kemudian ditampilkan di list data prediksi. Pengguna dapat menghapus data hasil prediksi dengan cara yang sama saat menghapus data master. Tetapi, data prediksi yang dapat dihapus hanya data prediksi yang dihasilkan dari form pengujian. Sedangkan data prediksi dari hasil pelatihan hanya dapat dihapus form data master.

3.6.4 *Class Diagram*

Penamaan *class diagram* pada dokumentasi ini diawali dengan huruf kapital. Jika lebih dari dua kata maka huruf pertama dari masing-masing kata tersebut menggunakan huruf kapital atau disingkat dengan mengambil huruf pertama dari kata tersebut kemdian digabungkan tanpa spasi. *Class* "Form" adalah *class* utama bawaan C# untuk aplikasi window (desktop) yang merupakan *parent* dari *class* yang berakhiran "GUI" atau dengan kata lain, *class* yang berakhiran "GUI" adalah turunan dari *class* Form. Tanda "mins (-)" pada atribut atau metode menunjukkan *visibility private* sedangkan tanda "plus (+)" menunjukkan *visibility public*. Nama dari *class diagram* akan menjadi nama *file* dalam aplikasi (kecuali *class* Form) tanpa menggunakan tipe *class.*

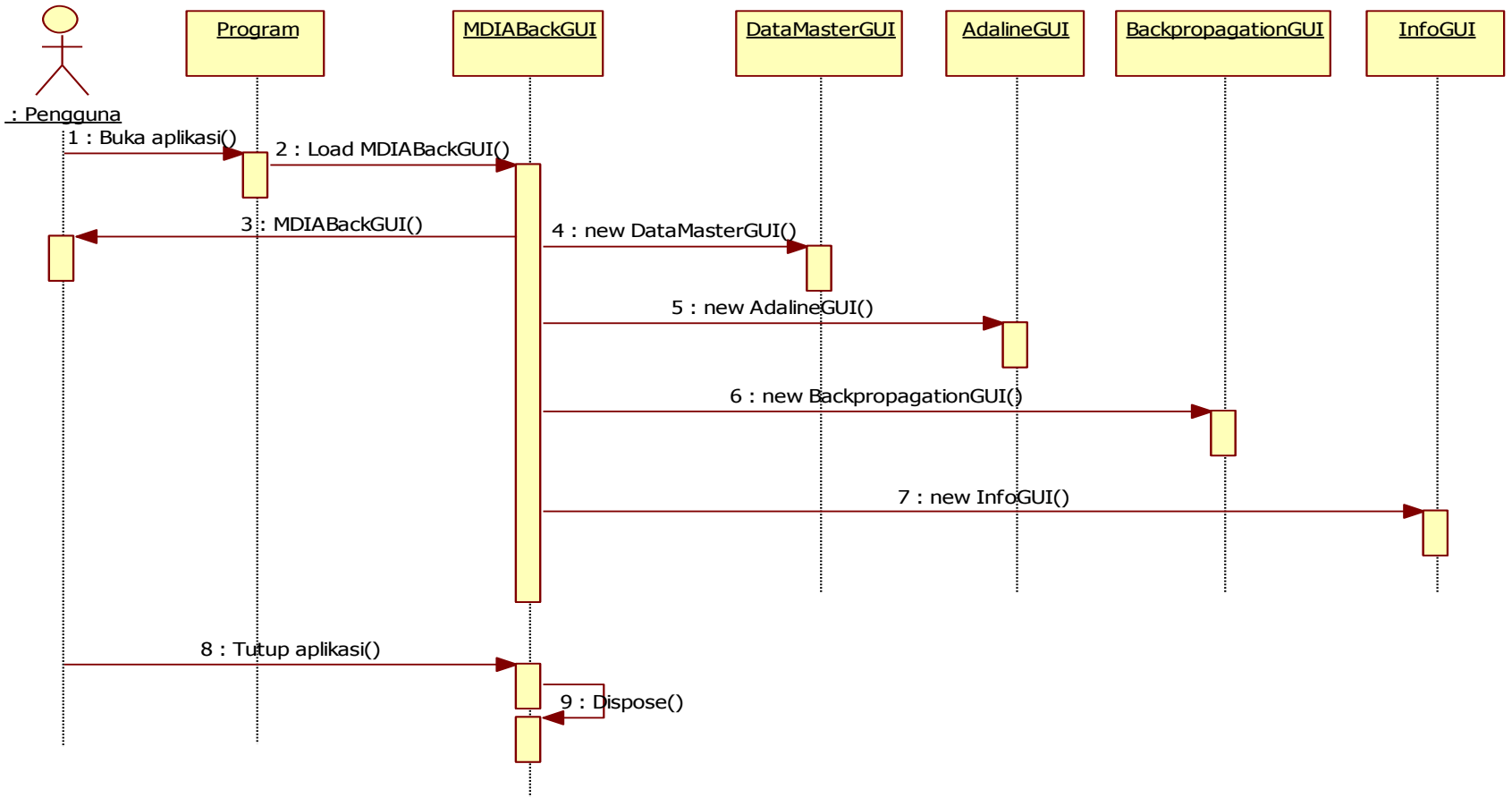
Class yang berakhiran "GUI" representasi dari *view* dan *controller* yang dalam C#, *view* adalah *user interface* sedangkan komponen-komponen yang ada di *view* memiliki metode tersendiri yang berfungsi sebagai *controller*. Dalam dokumentasi ini, *view* dan *controller* direpresentasikan dalam satu *class* yang berakhiran <C*ontroller*> sedangkan *class* yang berakhiran <*Model*> merepresentasikan *class* yang berinteraksi dengan *database*. *Model*, *view*, dan *controller* dalam dunia *object-oriented programming* seringkali disingkat MVC. Gambar 3.29 menunjukkan *class diagram*.



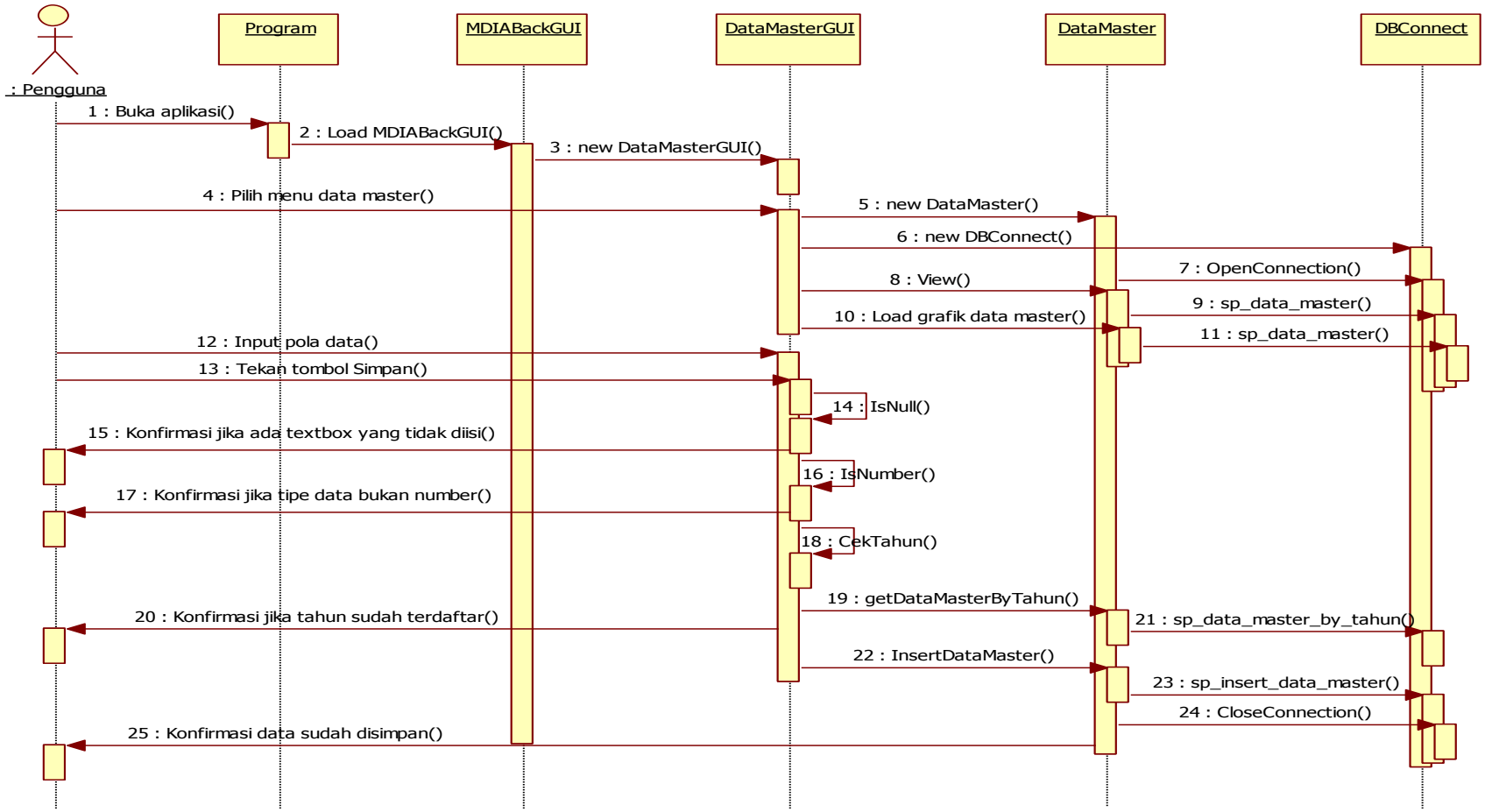
Gambar 3.16 *Class Diagram*

3.6.5 *Sequence Diagram*

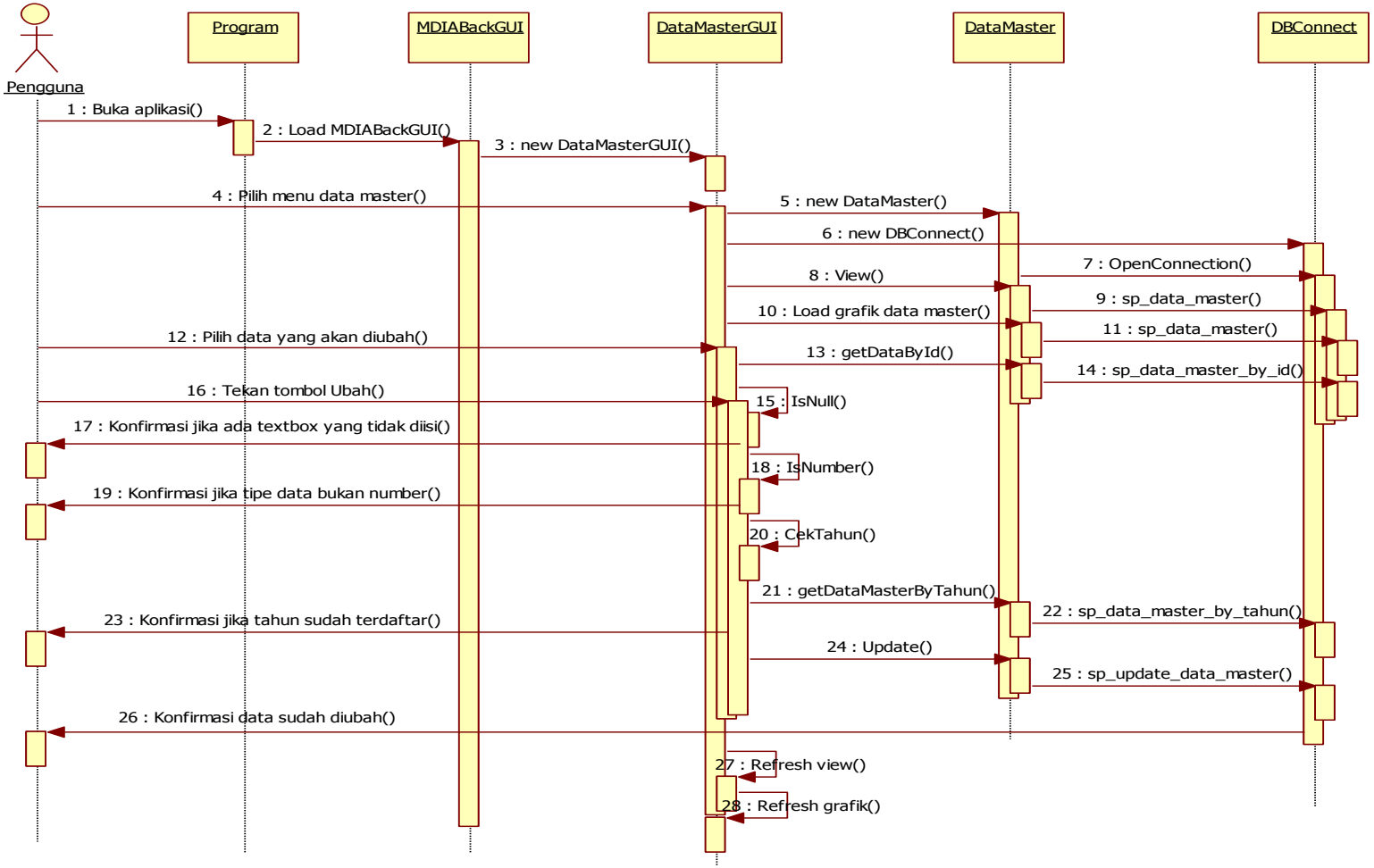
Sequence diagram menggambarkan interaksi antarobjek di dalam dan di sekitar sistem berupa pesan yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah kejadian untuk menghasilkan output tertentu. Berikut ini *sequence diagram* dari sistem yang akan dibuat:



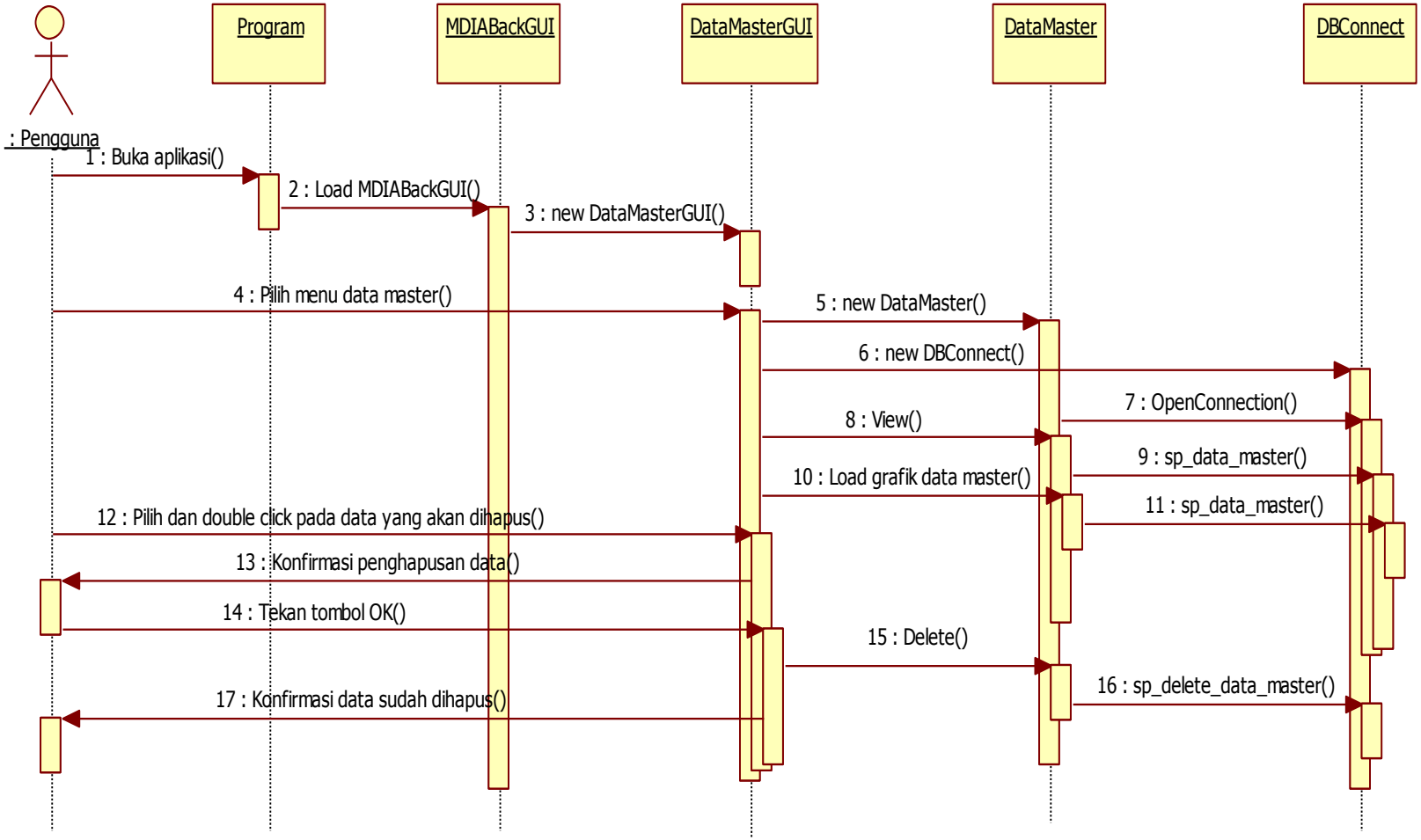
Gambar 3.17 *Sequence Diagram* Inisialisasi Aplikasi



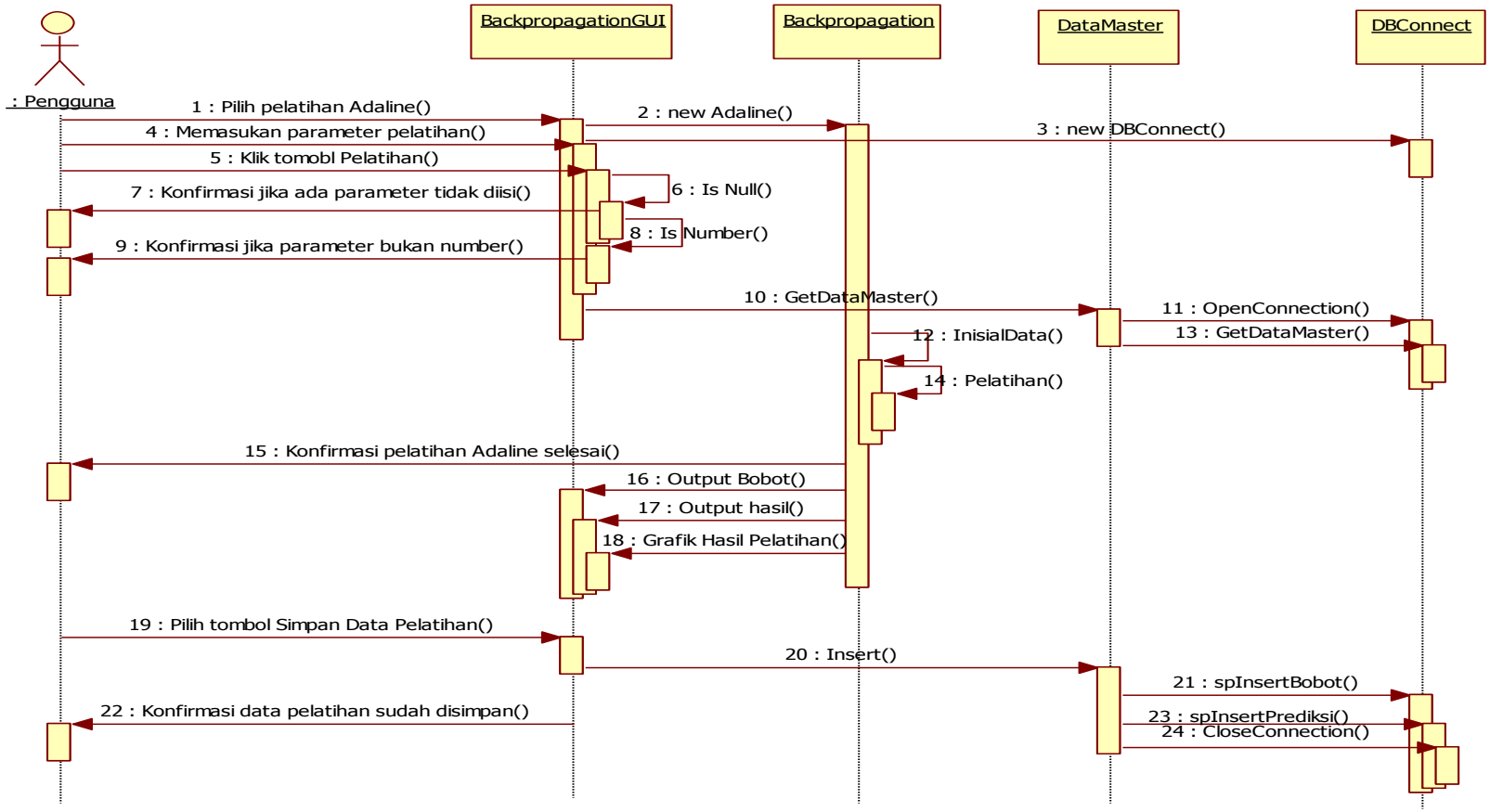
Gambar 3.18 *Sequence Diagram Insert* Data Master



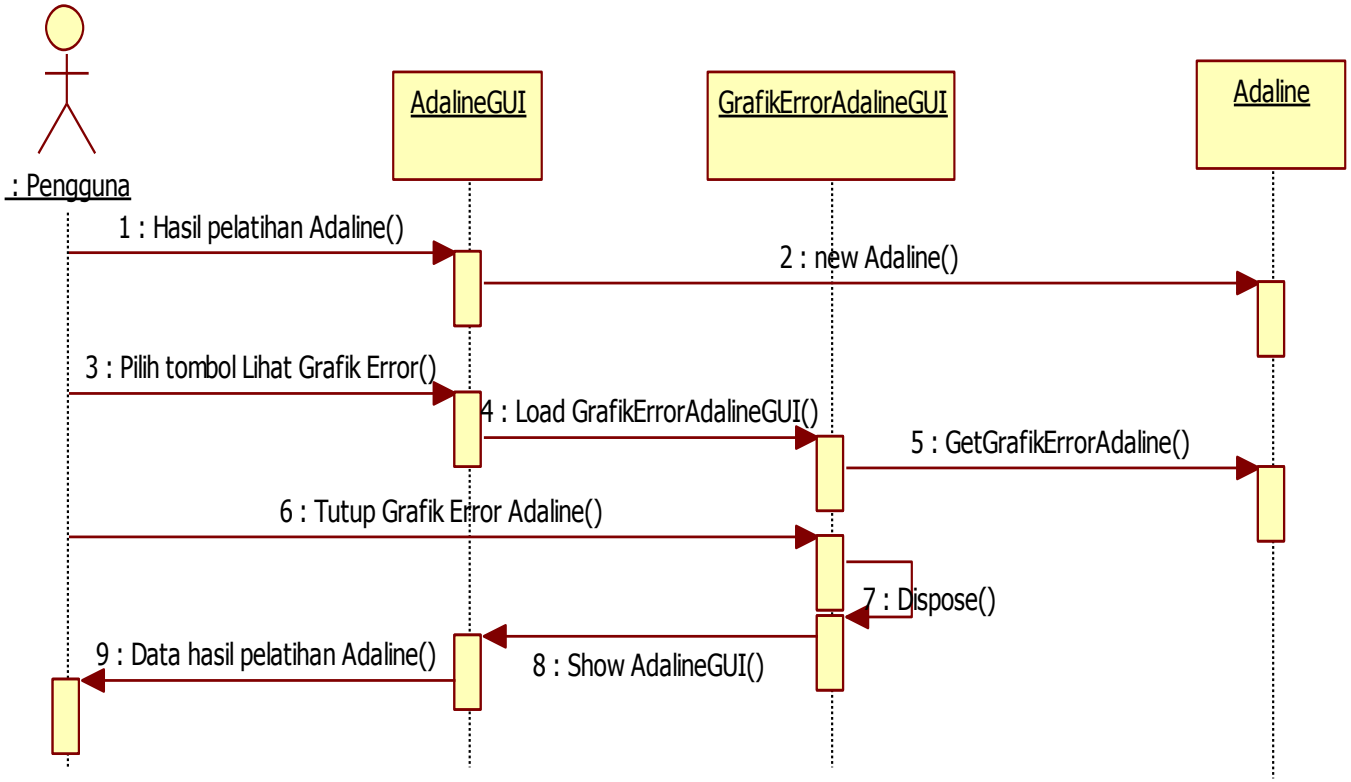
Gambar 3.19 *Sequence Diagram Update* Data Master



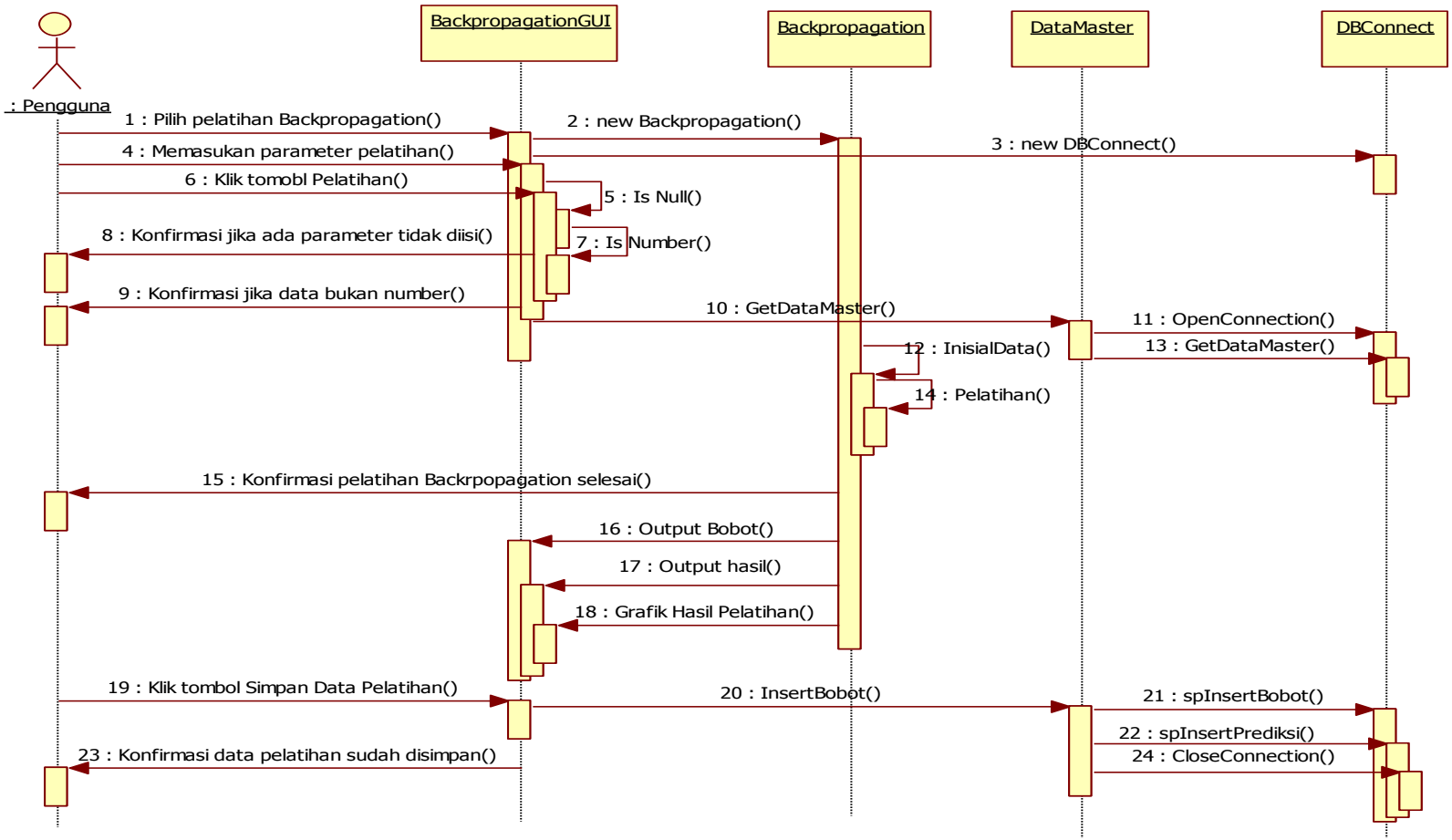
Gambar 3.20 *Sequence Diagram Delete* Data Master



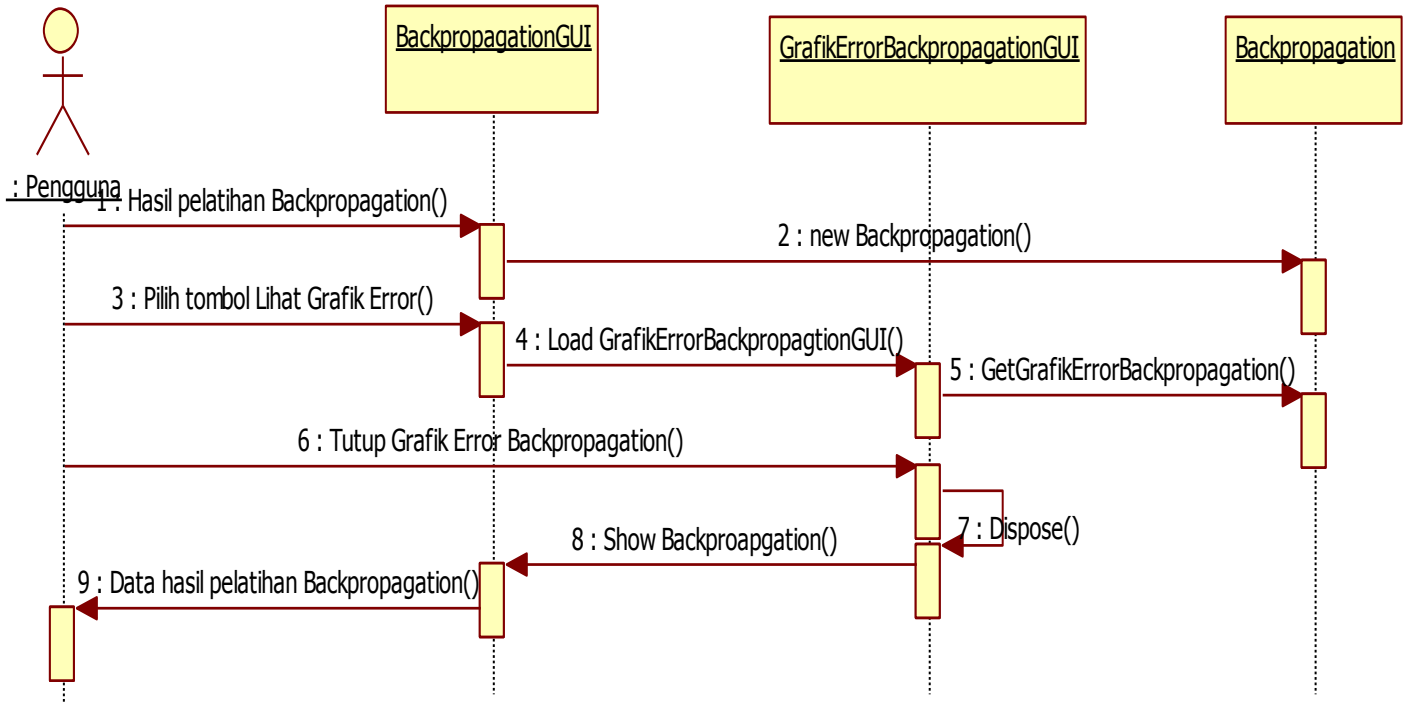
Gambar 3.21 *Sequence Diagram* Pelatihan *Adaline*



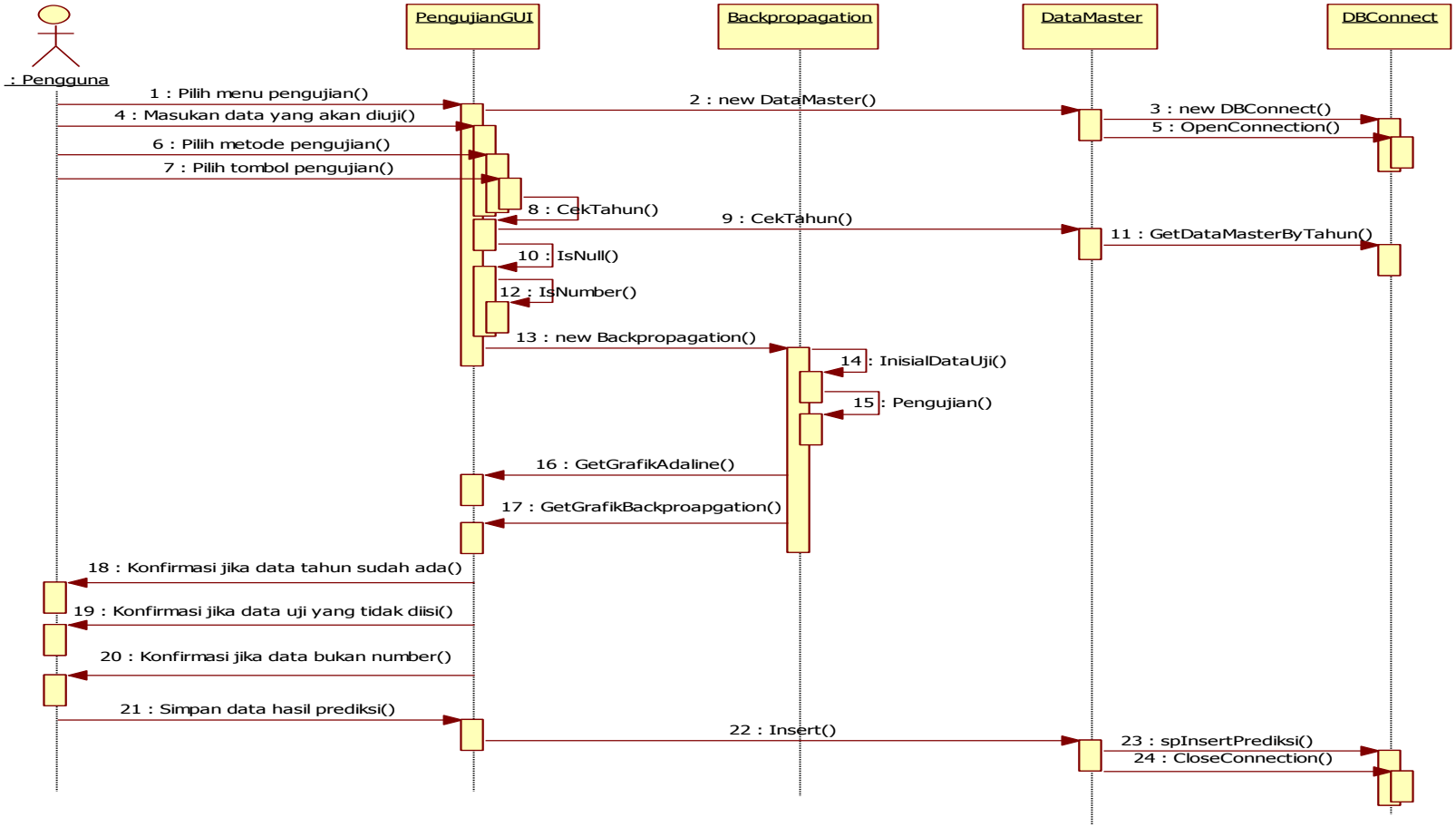
Gambar 3.22 *Sequence Diagram* Grafik *Error* Pelatihan *Adaline*



Gambar 3.23 *Sequence Diagram* Pelatihan *Backpropagation*



Gambar 3.24 *Sequence Diagram* Grafik *Error* Pelatihan *Backpropagation*

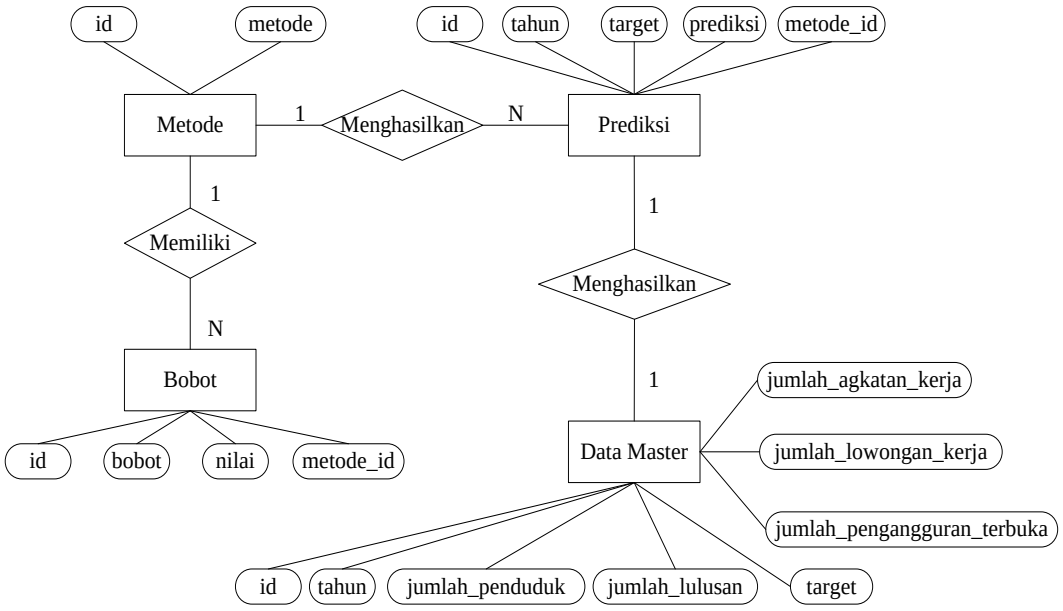


Gambar 3.25 *Sequence Diagram* Pengujian

3.7 Perancangan *Database*

Perancangan *database* dibutuhkan untuk menyimpan data terkait aplikasi yang dibuat. Perancangan *database* pada dokumentasi ini terdiri dari entity relational diagram (ERD), diagram relasional dan struktur diagram relasional.

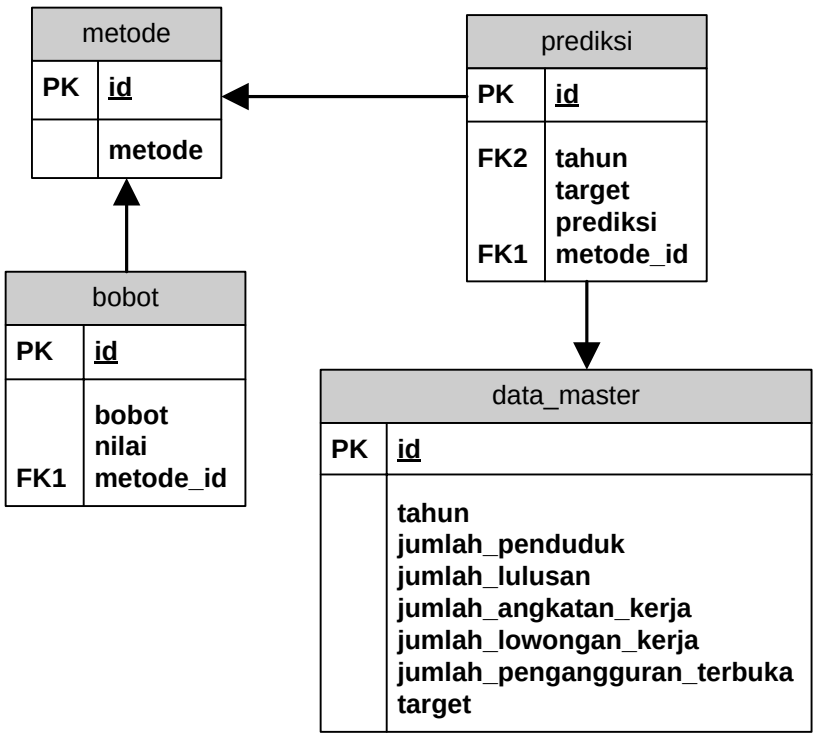
3.7.1 *Entity Relational Diagram (ERD)*



Gambar 3.26 ERD Perbandingan Metode *Adaline* dan *Backpropagation*

3.7.2 Diagram Relasional

Diagram relasional menggambarkan tabel-tabel (*relational*) dan hubungan antar tabel (*relationship*) untuk menyimpan dan memformat data yang digunakan oleh aplikasi. Gambar 3.27 menunjukkan diagram relasional.



Gambar 3.27 Diagram Relasional

3.7.3 Struktur Relasional

Data-data terkait prediksi jumlah pencari kerja akan disimpan di *database* yang dikelompokkan berdasarkan tabel-tabel yang sesuai dengan struktur data yang akan disimpan. Struktur relational (tabel) menjelaskan *constraint* tiap atribut pada diagram relational. Pada dokumentasi ini, penamaan tabel dan atributnya menggunakan huruf kecil. Jika lebih dari satu kata, diberi garis pemisah antara kata yang pertama dengan kata yang berikutnya dengan format atribut1_atribut2. Masing-masing tabel memiliki id sebagai *primary key*. *Primary key* yang menjadi tamu pada tabel lain (*foreign key*) diberi nama dengan format nama_tabel_id.

Tabel 3.23 Struktur Tabel Metode

KOLOM	TIPE	PK	FK	NOT NULL	KETERANGAN
id	char(1)	√		√	Id metode JST
metode	varchar(15)				Nama metode JST

Tabel 3.24 Struktur Tabel Data Master

KOLOM	TIPE	PK	FK	NOT NULL	KETERANGAN
id	integer(11)	√		√	Id data master
tahun	integer(11)			√	Tahun
jumlah_penduduk	integer(11)			√	Jumlah penduduk
jumlah_lulusan	integer(11)			√	Jumlah lulusan
jumlah_angkatan_kerja	integer(11)			√	Jumlah angkatan kerja
jumlah_lowongan_kerja	integer(11)			√	Jumlah lowongan kerja
jumlah_pengangguran_terbuka	integer(11)			√	Jumlah pengangguran terbuka
target	integer(11)			√	Target

Tabel 3.25 Struktur Tabel Bobot

KOLOM	TIPE	PK	FK	NOT NULL	KETERANGAN
id	integer(11)	√		√	Id pelatihan
bobot	integer(11)			√	Nama bobot
nilai	float			√	Nilai bobot
metode_id	char(1)		√	√	Metode id

Tabel 3.26 Struktur Tabel Prediksi

KOLOM	TIPE	PK	FK	NOT NULL	KETERANGAN
id	integer(2)	√		√	Id pengujian *Backpropagation*
tahun	integer(2)		√	√	Id pelatihan *Backpropagation*
target	integer(11)				Target
prediksi	integer(11)				Hasil prediksi *Backpropagation*
metode_id	char(1)		√	√	Metode id

3.8 Perancangan *Method*

Bagian ini menjelaskan *method-method* yang ada pada *class diagram.* diawali dari nama metode, deskpripsi dan algoritma dengan pendekatan *pseudocode.*

3.8.1 *Method Class* Data Master

Tabel 3.27 *Method Class Data Master*

Nama Metode	GetDataPrediksiById()
Deskripsi	Menampilkan data hasil prediksi

Algoritma	`GetDataPrediksiById(string metode_id)`
	`{`
	`  Call spGetDataPrediksiById(metode_id);`
	`}`

Nama Metode	ShowDataMaster()
Deskripsi	Menampilkan data master

Algoritma	`ShowDataMaster()`
	`{`
	`  Call spShowDataMaster();`
	`}`

Nama Metode	`InsertDataMaster()`
Deskripsi	`Entri data master baru`

Algoritma	`InsertDataMaster(int tahun, int jumlahPenduduk, int jumlahLulusan, int jumlahAngkatanKerja, int jumlahLowonganKerja, int jumlahPengangguran, int target)`
	`{`
	`Call sp_insert_data_master(tahun, jumlahPenduduk, jumlahLulusan, jumlahAngkatanKerja, jumlahLowonganKerja , jumlahPengangguran , target);`
	`}`

Nama Metode	`DeleteDataMasterById()`
Deskripsi	`Menghapus data master berdasarkan id`

Algoritma	`DeleteDataMasterById(int data_master_id)`
	`{`
	`  Call sp_delete_data_master_by_id(data_master_id);`
	`}`

Nama Metode	`DeletePrediksiById()`
Deskripsi	`Menghapus data prediksi berdasarkan id`

Algoritma	`DeletePrediksiById(int prediksi_id)`
	`{`
	`  Call spDeletePrediksiById(prediksi_id);`
	`}`

Nama Metode	`GetDataMasterById()`
Deskripsi	`Mengambil data master berdasarkan id data master`

Algoritma	`GetDataMasterById(int data_master_id)`
	`{`
	`  Call sp_data_master_by_id(data_master_id);`
	`}`

Nama Metode	UpdateDataMaster()
Deskripsi	Mengubah data master

Algortima	UpdateDataMaster(int id,int tahun, int jumlahPenduduk, int jumlahLulusan, int jumlahAngkatanKerja, int jumlahLowonganKerja, int jumlahPengangguran, int target)
	{
	Call sp_update_data_master (tahun, jumlahPenduduk, jumlahLulusan, jumlahAngkatanKerja, jumlahLowonganKerja , jumlahPengangguran , target);
	}

Nama Metode	GetDataBobot()
Deskripsi	Mengambil data bobot pelatihan

Algoritma	Doubel GetDataBobot(int metode_id)
	{
	call spGetDataBobot(metode_id);
	}

3.8.2 *Method Class Adaline*

Tabel 3.28 *Method Class Adaline*

Nama Metode	Normalisasi()
Deskripsi	Melakukan normalisasi data master

Algoritma	Double Normalisasi (double Xi, double max, int lowerbound, int upperbound)
	{
	double alpha,betha;
	alpha = (upperbound - lowerbound)/(max - min);
	betha = upperbound - (alpha * max);
	Xi = alpha * Xi + betha;
	return Xi;
	}

Nama Metode	RandomBobot()
Deskripsi	Melakukan inisialisasi bobot dan bias pelatihan secara random

Algoritma	Double RandomBobot(double low, double high)
	{
	double tmp=0.0;
	while (tmp == 0.0)
	{
	tmp = _rand.NextDouble();
	}
	return (tmp * (high - low) + low);
	}

Nama Metode	FungsiAktivasi()
Deskripsi	Mengenakan fungsi aktifasi pada hasil keluaran jaringan

Algoritma	Double FungsiAktivasi(double net)
	{
	double neNet = 0;
	if (net > 0)

	neNet = 1;
	else
	neNet = 0;
	return neNet;
	}

Nama Metode	Denormalisasi
Deskripsi	Melakukan denormalisasi data untuk mengembalikan data yang diskala (normalisasi) menjadi data aslinya kemudian ditampilkan ke layar komputer
Algoritma	

3.8.3 *Method Class Backpropagation*

Tabel 3.29 *Method Class Backpropagation*

Nama Metode	Normalisasi()
Deskripsi	Melakukan normalisasi data master

Algoritma	Double Normalisasi (double Xi, double max, int lowerbound, int upperbound)
	{
	double alpha,betha;
	alpha = (upperbound - lowerbound)/(max - min);
	betha = upperbound - (alpha * max);
	Xi = alpha * Xi + betha;
	return Xi;
	}

Nama Metode	RandomBobot()
Deskripsi	Melakukan inisialisasi bobot dan bias pelatihan secara random

Algoritma	Double RandomBobot(double low, double high)
	{
	double tmp=0.0;
	while (tmp == 0.0)
	{
	tmp = _rand.NextDouble();
	}
	return (tmp * (high - low) + low);
	}

Nama Metode	FungsiAktivasi()
Deskripsi	Mengenakan fungsi aktifasi pada hasil keluaran jaringan

Algoritma	Double FungsiAktivasi(double net)
	{
	double newNet;
	newNet =(1 / (1 + Exp(-net)));
	return newNet;
	}

Nama Metode	DFungsiAktivasi();

Deskripsi	Fungsi Diferensial aktifasi *Backpropagation*

Algoritma	`Double DFungsiAktivasi(double net)`
	`{`
	`double newNet = 0;`
	`double tmp = 0;`
	`tmp = FungsiAktivasi (net);`
	`newNet = tmp*(1 - tmp);`
	`return newNet;`
	`}`

Nama Metode	Denormalisasi
Deskripsi	Melakukan denormalisasi data untuk mengembalikan data yang diskala (normalisasi) menjadi data aslinya kemudian ditampilkan ke layar komputer
Algoritma	

3.8.4 *Method Class DBConnect*

Tabel 3.30 *Method Class DBConnect*

Nama Metode	`DBConnect ()`
Deskripsi	Konstruktor

Algoritma	`DBConnect()`
	`{`
	`string _server = "localhost";`
	`string _database = "aback";`
	`string _uid = "root";`
	`tring _password = "";`
	`string connectionString;`
	`connectionString = "SERVER=" +_server + ";" + "DATABASE=" +`
	`_database + ";" + "UID=" +_uid + ";" + "PASSWORD=" +_password + ";";`
	`_connect = new MySqlConnection(connectionString);`
	`}`

Nama Metode	`MySqlConnection Connect`
Deskripsi	Propertis Connect

Algoritma	`MySqlConnection Connect`
	`{`
	`get { return_connect;}`
	`set {_connect = value;}`
	`}`

Nama Metode	`CloseConnection()`
Deskripsi	Menutup koneksi *database*

Algoritma	`bool CloseConnection()`
	`{`
	`_connect.Close();`
	`return true;`
	`}`

Nama Metode	`OpenConnection()`
Deskripsi	Membuka koneksi databse

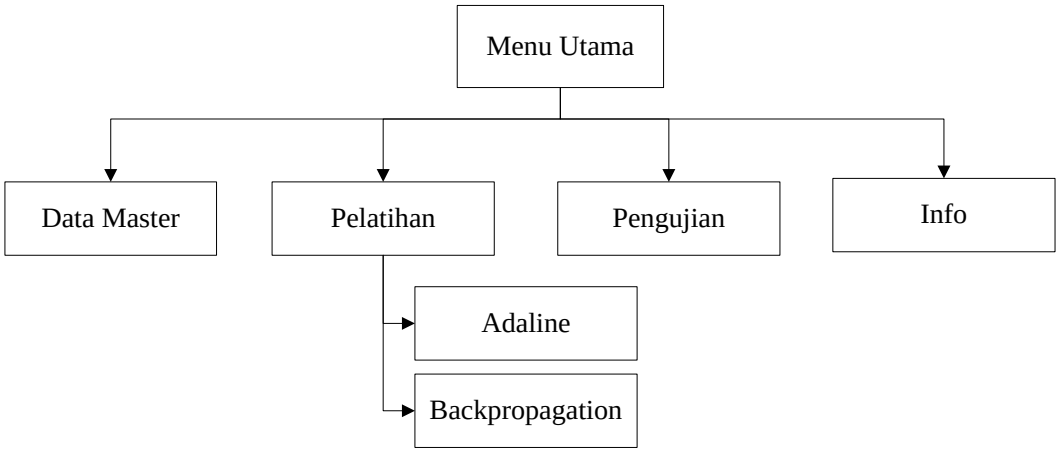
Algoritma	`bool OpenConnection()`
	`{`
	`  this._connect.Open();`
	`  return true;`
	`}`

3.9 Perancangan Antarmuka

Antarmuka berfungsi sebagai *interface* antara user dengan sistem. Perancangan antarmuka pada sub bab ini dibagi menjadi dua, yaitu struktur menu dan GUI.

3.9.1 Perancangan Struktur Menu

Struktur menu aplikasi perbandingan metode *Adaline* dan *Backpropagation* diperlihatkan di Gambar 3.27 sedangkan penjelasan dari tiap bagian menu diperlihatkan di tabel 3.31.



Gambar 3.28 Struktur Menu Aplikasi

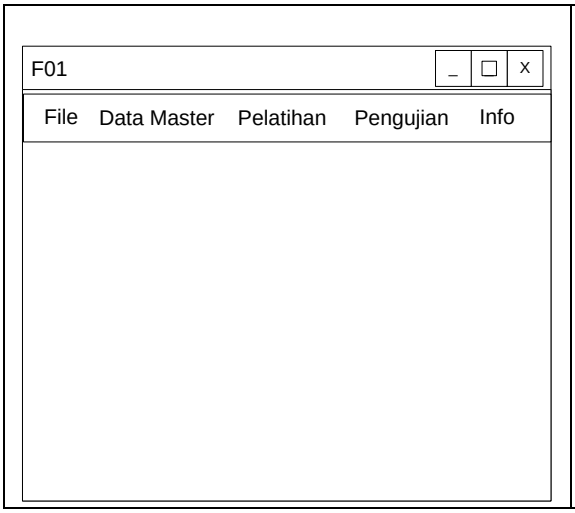
Tabel 3.31 Penjelasan Menu

Nama Menu	Keterangan
Menu Utama	Ditampilkan saat pertama kali user membuka aplikasi
File	Untuk keluar aplikasi
Data Master	Form untuk mengelola data master yang terdiri dari Tambah

Nama Menu	Keterangan
	Data, Ubah Data, Hapus Data, dan menampilkan data dalam bentuk angka dan grafik.
Palatihan	Form yang berisi pilihan metode pelatihan yang teridiri dari metode *Adaline* dan *Backpropagataion*
Adaline	Form untuk menginputkan parameter pelatihan metode *Adaline* disertai dengan grafik *error* pelatihan dan grafik hasil pelatihan disertai dengan grafik *error* pelatihan dan grafik hasil pelatihan.
Backpropagation	Form untuk menginputkan parameter pelatihan metode *Backpropagation*
Pengujian	Form yang berisi pilihan metode pangujian yang teridiri dari metode *Adaline* dan *Backpropagataion.* Disertai dengan hasil prediksi dalam bentuk angka dan grafik.
Info	Berisi informasi pembuat sistem

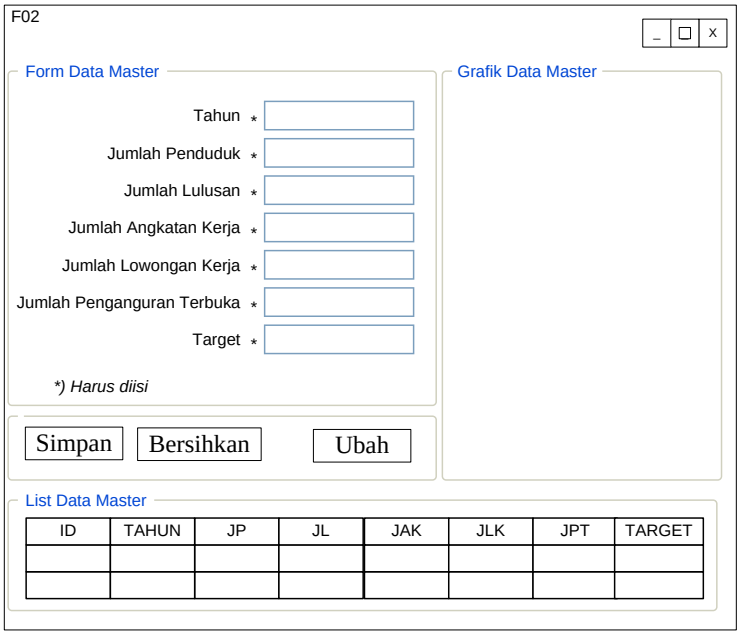
3.9.2 Perancangan *Graphical User Interface* (GUI)

GUI adalah *interface* yang menghubungkan pengguna dengan aplikasi atau sistem yang dibuat. GUI dapat berupa form untuk mengisi data atau menampilkan data kepada pengguna. Pada bagian ini akan menjelaskan perancangan GUI aplikasi perbandingan metode *Adaline* dan *Backpropagation* untuk peramalan jumlah pencari kerja adalah sebagai berikut:



F01 adalah menu utama aplikasi
a) Klik menu File, akan tampil sub menu exit
b) Klik menu Data Master, akan tampil F02
c) Klik menu Pelatihan, akan tampil sub menu Adaline dan Backpropagation
d) Klik menu Pengujian, akan tampil F05
e) Klik menu Info, akan tampil F06

Gambar 3.29 Perancangan GUI Menu Utama



F02 adalah form mangamen data master yang merupakan form untuk tambah, ubah, dan hapus, menampilkan grafik dan list data master.

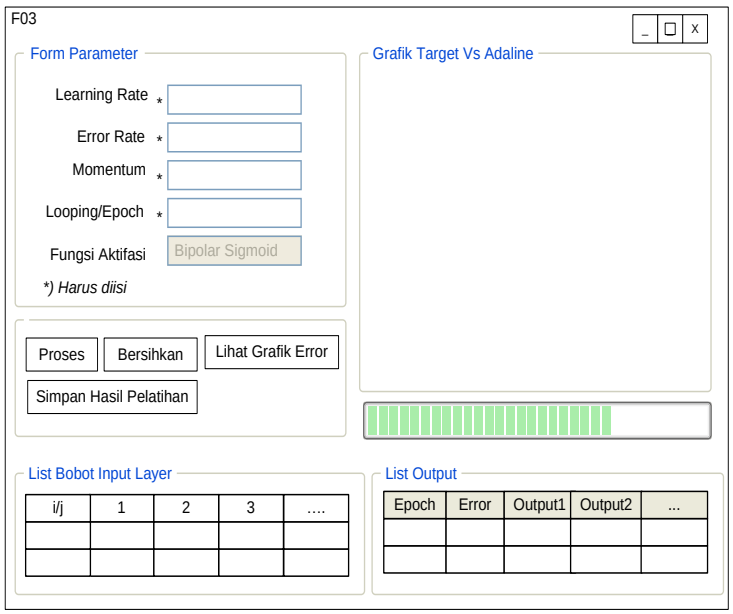
Untuk menambah data:
a) Isi semua data master
b) Klik tombol **Simpan**.

Untuk membersihkan form:
a) Klik tombol **Bersihkan**

Untuk menghapus data:
a) Klik dua kali pada data yang ingin dihapus di list data master.
b) Klik tombol **Yes** pada konfirmasi penghapusan data

Untuh mengubah data:
a) Klik satu kali pada data yang ingin diubah di list data.
b) Ubah dengan data yang sesuai
c) Klik tombol **Ubah**

Gambar 3.30 Perancangan GUI Data Master



F03 adalah form pelatihan dan pengujan data master dengan metode *Adaline*

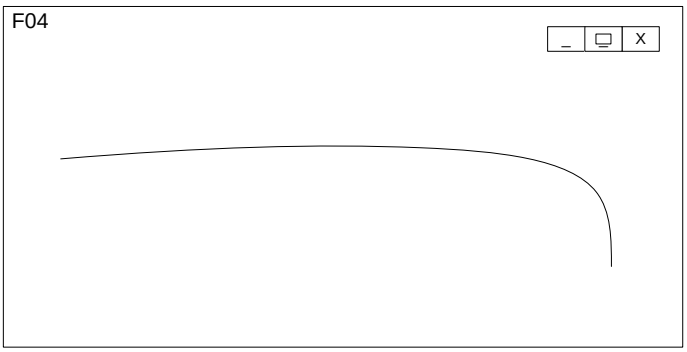
Untuk pelatihan dan pengujian data master:
a) Isi data parameter
b) Klik tombol **Proses**

Untuk membersihkan form:
a) Klik tombol **Bersihkan**

Untuk penyimpan hasil pelatihan:
a) Klik tombol Hasil Pelatihan

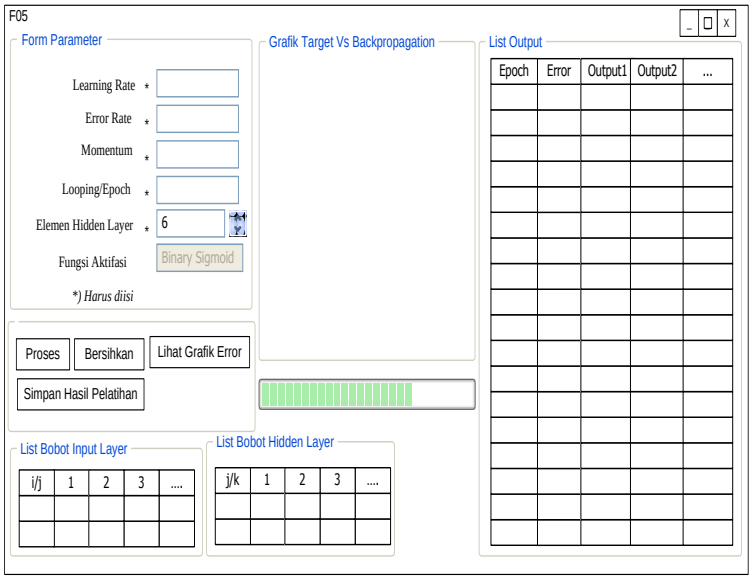
Untuk melihat grafik error pelatihan:
a) Klik tombol **Lihat Grafik Error**

Gambar 3.31 Perancangan GUI Pelatihan *Adaline*



F04 adalah GUI grafik *error* pelatihan *Adaline*

Gambar 3.32 Perancangan GUI Grafik *Error* Pelatihan *Adaline*



F05 adalah form pelatihan dan pengujan data master dengan metode *Backpropagation*

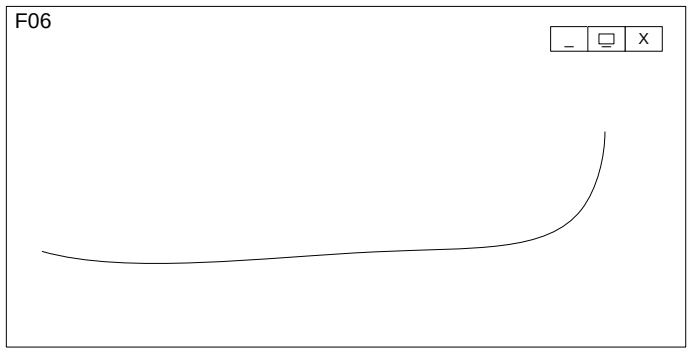
Untuk pelatihan dan pengujian data master:
a) Isi data parameter
b) Klik tombol **Proses**

Untuk membersihkan form:
a) Klik tombol **Bersihkan**

Untuk penyimpan hasil pelatihan:
a) Klik tombol Hasil Pelatihan

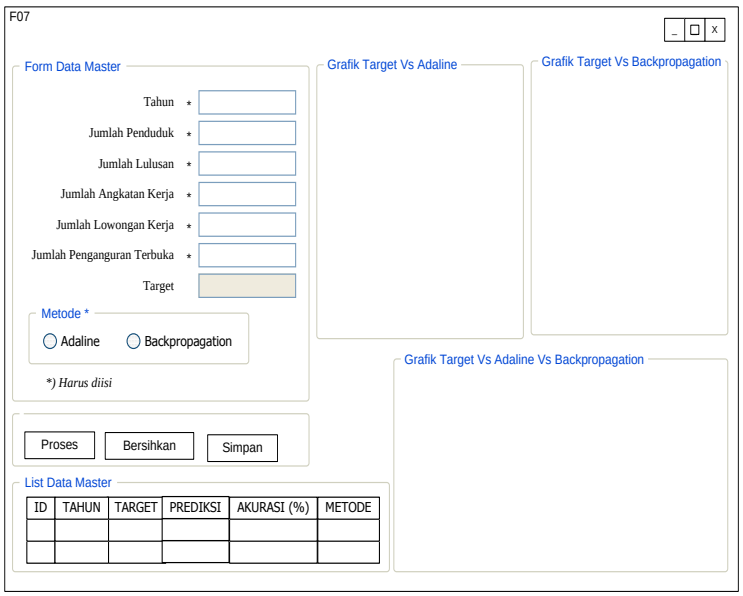
Untuk melihat grafik error pelatihan:
a) Klik tombol **Lihat Grafik Error**

Gambar 3.33 GUI Pelatihan *Backpropagation*



F06 adalah GUI grafik *error* pelatihan *Backpropagation*

Gambar 3.34 Perancangan GUI Grafik *Error* Pelatihan *Backpropagation*



F07 adalah form pengujian *Adaline* atau *Backpropagation.*

a) Isi semua Form Data Master
b) Pilih metode pengujian
c) Klik tombol **Proses**

Untuk membersihkan form:
a) Klik tombol **Bersihkan**

Untuk menyimpan hasil pengujian:
a) Klik tombol **Simpan**

Gambar 3.35 Perancangan GUI Pengujian



F08 adalah GUI yang berisi informasi nama aplikasi, programmer, versi aplikasi, dan instansi tempat programmer menimba ilmu

Gambar 3.36 Perancangan GUI Info

3.9.3 Perancangan Pesan

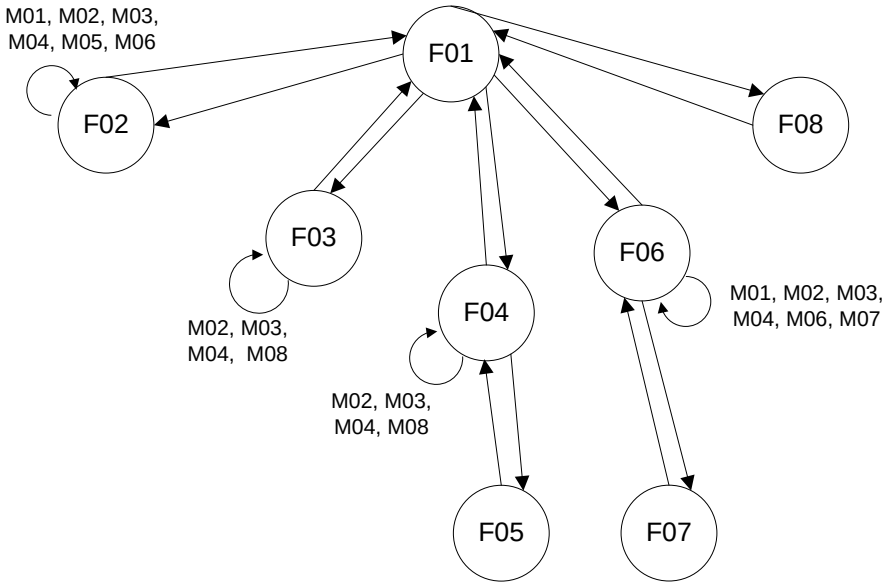
Perancangan pesan memberikan informasi kepada pengguna terkait aksi yang dilakukan oleh pengguna dalam menjalankan aplikasi. Berikut ini adalah perancangan pesan yang ada pada aplikasi sistem perbandingan metode *Adaline* dan *Backpropagtion*:



Gambar 3.37 Perancangan Pesan

3.9.4 Jaringan Semantik

Jaringan semantik dalam dokumentasi ini menggambarkan interaksi GUI aplikasi secara global. Jaringan semantik pada aplikasi perbandingan metode *Adaline* dan *Backpropagation* adalah sebagai berikut:



Gambar 3.38 Jaringan Semantik