BAB 4 PERANCANGAN

Bab ini menjelaskan mengenai formulasi permasalahan, siklus penyelesaian Algoritme Genetika dan algoritme *Extreme Learning Machine* (ELM), perhitungan manual, perancangan antarmuka serta perancangan skenario pengujian.

4.1 Formulasi Permasalahan

Permasalahan yang diangkat pada penelitian ini adalah peningkatan tingkat keakuratan penentuan kualitas air sungai. Solusi yang digunakan yaitu dengan mengaplikasikan metode Algoritme Genetika dengan *Extreme Learning Machine* (ELM). Masukkan dari pengguna adalah data 7 parameter kualitas air dalam format .xls dan beberapa parameter lain seperti ukuran populasi (*popsize*), ukuran generasi, cr (*crossover rate*), mr (*mutation rate*).

Pada penelitian ini, perhitungan yang terlebih dulu dilakukan adalah perhitungan dengan menggunakan Algoritme Genetika. Variabel yang dioptimasi adalah bobot *input*/bobot awal yang akan digunakan dalam penentuan kualitas air menggunakan metode ELM. Dalam proses perhitungan Algoritme Genetika terdapat beberapa tahap, antara lain yaitu representasi kromosom, inisialisasi, reproduksi, evaluasi dan seleksi. Pada tahap seleksi akan dihasilkan bobot yang optimal dengan membandingkan nilai *fitness* yang dihasilkan masing-masing individu.

Selanjutnya, proses perhitungan ELM dimulai dengan terlebih dahulu menginisialisasi data *training* dan data *testing*. Untuk meminimalisasi jarak persebaran antar data maka sebaiknya dilakukan normalisasi pada data *training* dan data *testing*. Inisialisasi bias dan bobot awal yang diambil dari bobot yang telah dioptimasi sebelumnya. Kemudian akan dilakukan proses *training* yang meliputi perhitungan keluaran *hidden layer* dan aktivasi *sigmoid*. Pada proses *training* akan diperoleh hasil bobot keluaran/ *output weight*. Tahap terakhir dalam ELM, yaitu *testing*, dimana pada tahap ini juga dilakukan perhitungan keluaran *hidden layer*, untuk memperoleh hasil prediksi. Berdasarkan hasil prediksi yang telah dilakukan, akan dihitung keakuratan prediksi dari sistem tersebut.

Adapun sampel data kualitas air yang digunakan pada penelitian ini dapat dilihat pada Tabel 4.1 (Lihat Lampiran A):

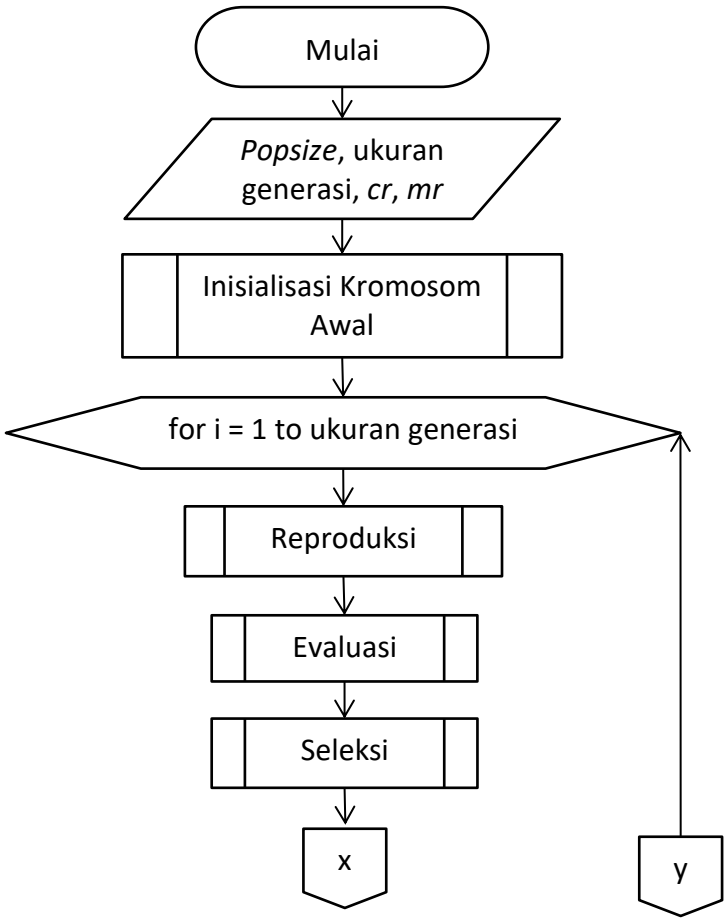
Tabel 4.1 Data parameter kualitas air sungai

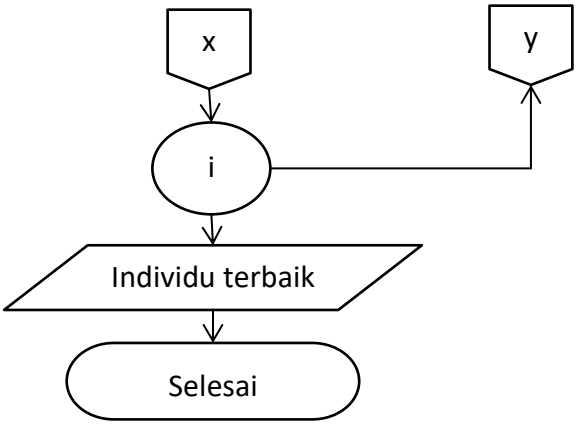
Tahun	Bulan	FISIKA	KIMIA anorganik				Organik	
		TSS	BOD	COD	DO	pH	Fenol	Minyak & Lemak
2005	Januari	0	15,35	56,8	11,2	7,5	0,0775	0,8
	November	17,75	5,05	20,6	3,5	6,9	0,0885	0,8
	Desember	25,35	4,1	16,85	6,75	7	0,151	3,5

Tahun	Bulan	FISIKA	KIMIA anorganik				Organic	
		TSS	BOD	COD	DO	pH	Fenol	Minyak & Lemak
2006	Mei	23,7	12,15	32,5	11,95	8,2	0,008	0
	Juni	17,8	3,4	10,25	6,8	7,45	0,176	0
2007	Januari	32,1	4,8	20,85	8,35	7,8	0,151	0,5
	Februari	27,6	7,15	20,85	10,2	7,65	0,0975	0,8
	Maret	54,3	14,4	27,3	11,2	7,55	0,1	3
	Agustus	13,65	4,8	9,3	6,3	7,3	0,046	0
2009	Oktober	42,3	3,75	9,7375	42,25	7,1	0	4,75

4.2 Siklus Metode Algoritme Genetika

Siklus metode Algoritme Genetika merupakan proses pengoptimalan variabel atau faktor yang berpengaruh dalam komputasi suatu metode/algoritme. Pada penelitian ini, variabel yang akan dioptimasi adalah bobot awal dalam penyelesaian masalah menggunakan metode ELM. Siklus metode Algoritme Genetika dapat dilihat pada Gambar 4.1.

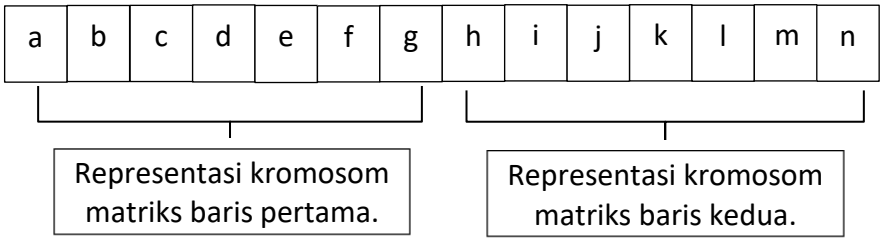




Gambar 4.1 *Flowchart* Algoritme Genetika

4.2.1 Proses Representasi Kromosom

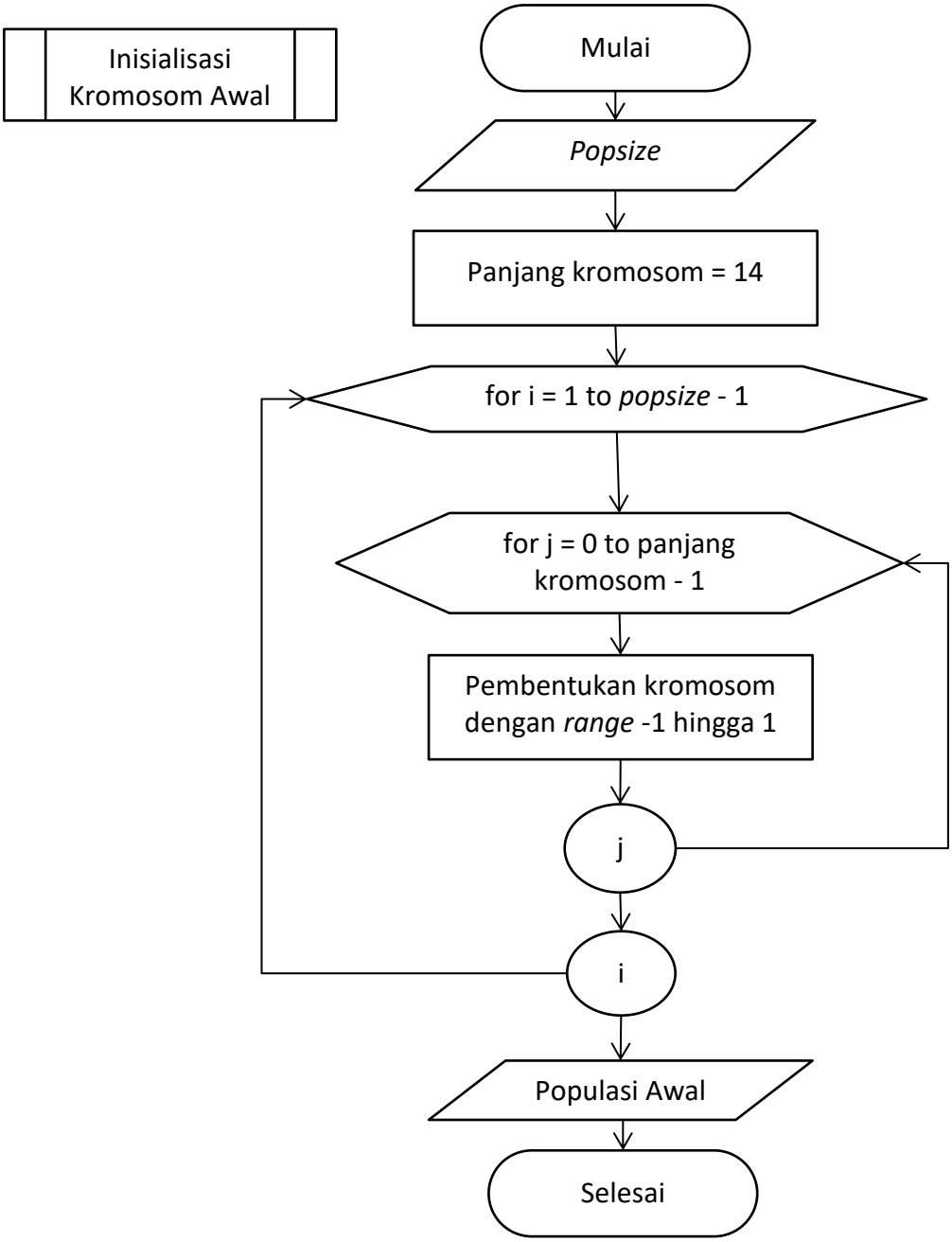
Representasi kromosom yang digunakan pada penelitian ini adalah representasi kromosom pengkodean *real* (*real code*), karena angka atau bilangan yang digunakan pada penelitian ini adalah bilangan *real* (bilangan bulat dan pecahan). Kromosom dari bobot awal akan direpresentasikan dan dibangkitkan secara *random*, dengan *range* bilangan -1 s.d. 1. Bobot awal pada ELM direpresentasikan dalam bentuk matriks. Ukuran matriks bobot awal adalah *j x k*, dimana j merupakan jumlah *hidden layer* dan k merupakan jumlah input layer yang akan digunakan. Jumlah *hidden layer* yang digunakan pada penelitian ini, sejumlah 7, sedangkan jumlah *input layer*-nya ada 7 yang merupakan representasi dari 7 parameter penentu kualitas air sungai. Dari uraian tersebut maka panjang kromosom yang akan direpresentasikan adalah 7 x 7, yaitu 49. Maka dalam penelitian ini kromosom yang dibangkitkan sepanjang 49. Contoh representasi kromosom pada baris pertama dan kedua ditunjukkan pada Gambar 4.2.



Gambar 4.2 Representasi kromosom *real-code*

4.2.2 Proses Inisialisasi

Proses inisialisasi kromosom awal dijelaskan pada Gambar 4.3.



Gambar 4.3 *Flowchart* proses inisialisasi kromosom

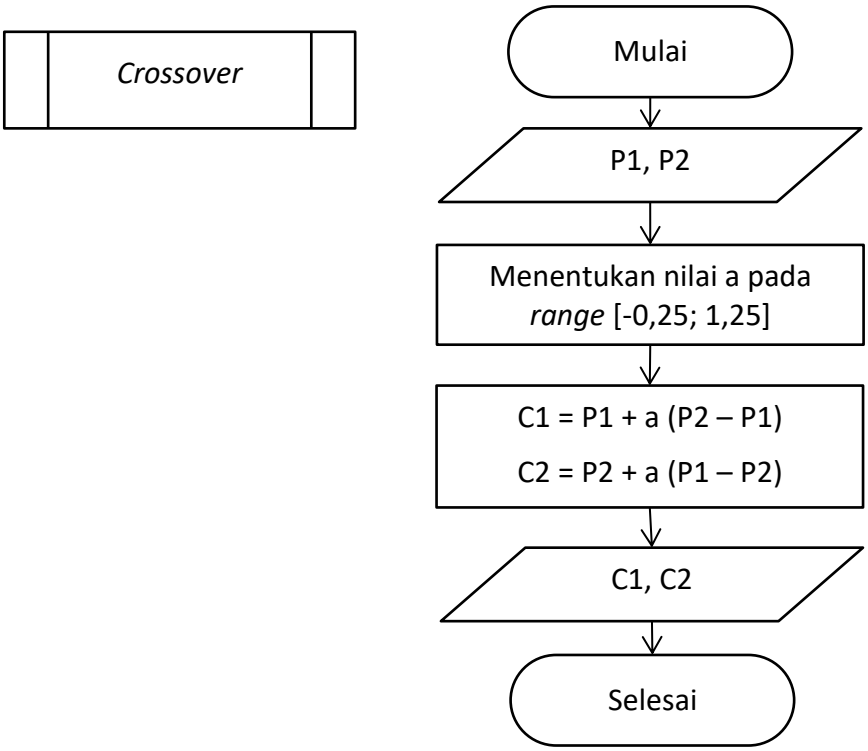
Berdasarkan Gambar 4.3, langkah inisialisasi kromosom adalah sebagai berikut:

1. Set ukuran populasi (*popsize*).
2. Inisialisasi panjang kromosom.
3. Pembangkitan kromosom sesuai dengan *popsize* dan panjang kromosom secara *random* dengan *range* -1 s.d. 1. Selama kromosom belum memenuhi panjang kromosom dan *popsize* yang diinisialisasi sebelumnya, akan terus dilakukan perulangan.

4. Setelah memenuhi kondisi berhenti pada perulangan, yaitu kromosom yang dibangkitkan sudah memenuhi panjang kromosom dan *popsize*, baru akan dihasilkan suatu populasi awal.

4.2.3 Proses Reproduksi

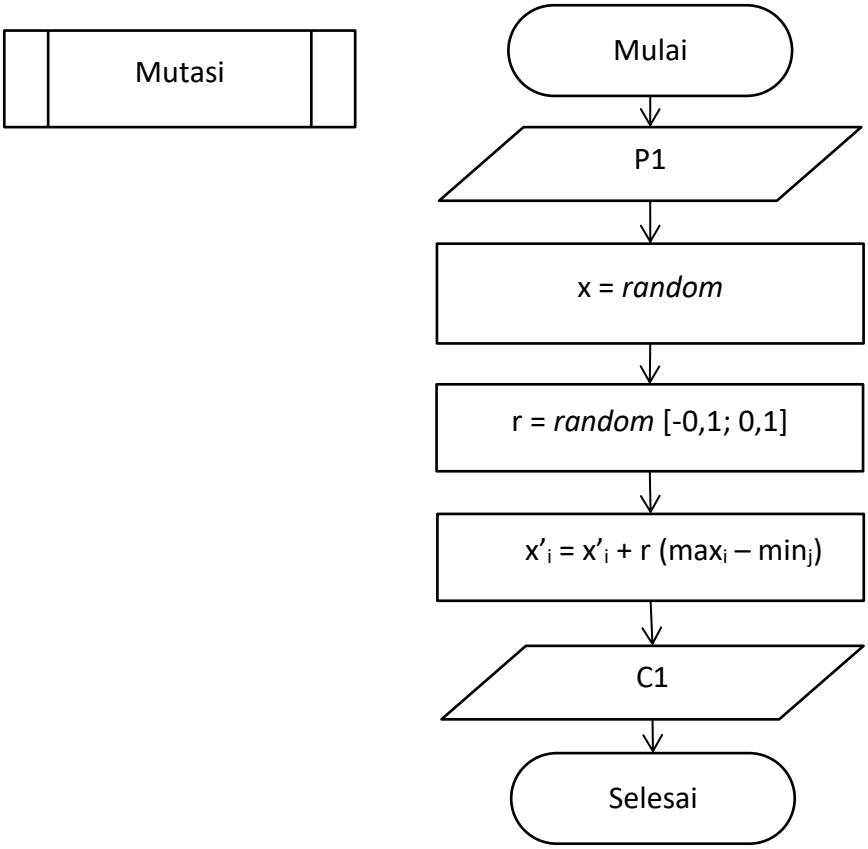
Dalam penelitian ini, proses reproduksi meliputi proses *crossover* dan mutasi untuk menghasilkan *offspring* (*child*). Jenis *crossover* yang digunakan adalah *extended intermediate crossover*, sedangkan jenis mutasi yang digunakan adalah *random mutation*. Selanjutnya tahap *crossover* akan dijelaskan pada Gambar 4.4 dan mutasi akan dijelaskan pada Gambar 4.5



Gambar 4.4 *Flowchart crossover*

Berdasarkan Gambar 4.4, langkah-langkah *crossover* dapat dijabarkan sebagai berikut:

1. Masukan individu *parent* 1 dan 2 (P1 dan P2).
2. Inisialisasi nilai a secara *random* dengan *range* [-0,25; 1,25].
3. Hitung nilai *offspring* menggunakan persamaan *extended intermediate crossover*. Banyaknya *offspring* yang akan dihasilkan pada proses *crossover* ditentukan oleh jumlah *cr* dan *popsize*.
4. Setelah dilakukan perhitungan nilai akan dihasilkan *offspring* baru, hasil pertukaran silang antar 2 individu lama.



Gambar 4.5 *Flowchart* mutasi

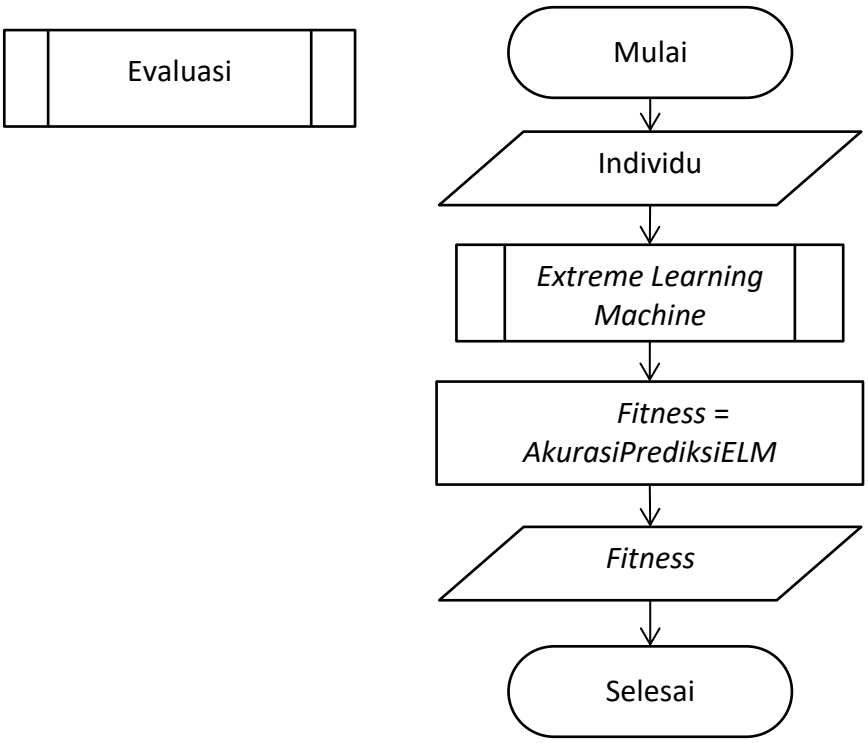
Berdasarkan gambar 4.5, langkah-langkah mutasi dapat dijabarkan sebagai berikut:

1. Masukan individu *parent* 1.
2. Tentukan posisi gen yang akan dimutasi secara *random*.
3. Inisialisasi nilai r secara *random* dengan *range* [-0,1; 0,1].
4. Hitung nilai *offspring* menggunakan persamaan *random mutation*. Banyaknya *offspring* yang akan dihasilkan pada proses mutasi ditentukan oleh jumlah mr dan *popsize*.

Setelah dilakukan perhitungan nilai akan dihasilkan *offspring* baru, hasil mutasi gen yang telah ditentukan sebelumnya.

4.2.4 Proses Evaluasi

Dalam proses evaluasi dilakukan perhitungan nilai *fitness* sehingga diperoleh solusi yang optimal. Tahapan evaluasi lebih lanjut dijabarkan pada Gambar 4.6.



Gambar 4.6 *Flowchart* proses evaluasi

Berdasarkan Gambar 4.6, langkah-langkah evaluasi dapat dijabarkan sebagai berikut:

1. Masukan individu yang akan dievaluasi.
2. Melakukan perhitungan menggunakan *Extreme Learning Machine*.
3. Memasukkan hasil akurasi prediksi sebagai nilai *fitness*.
4. Setelah dilakukan perhitungan akan dihasilkan nilai *fitness* untuk di proses ke tahap selanjutnya.

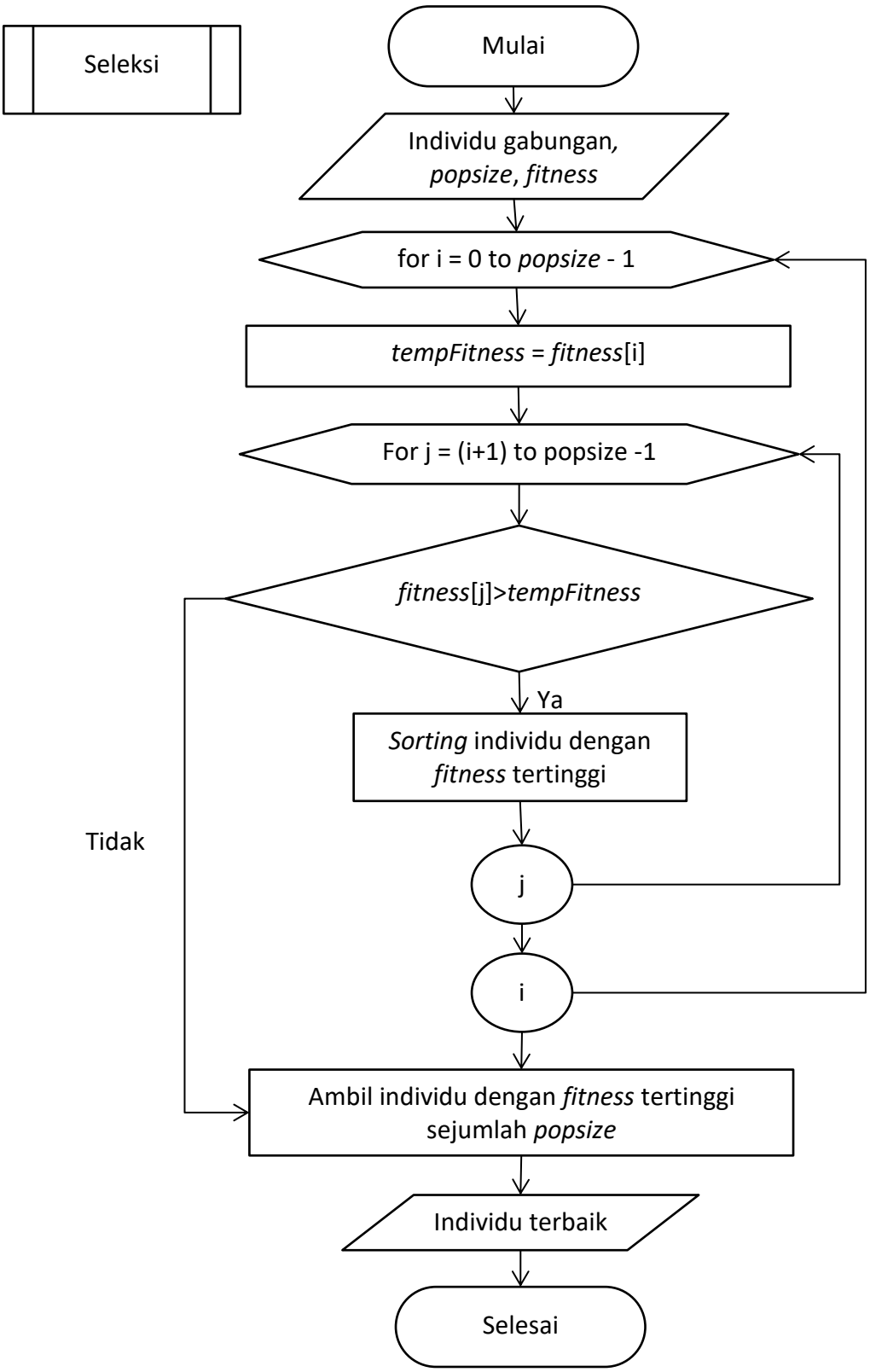
4.2.5 Proses Seleksi

Jika sebelumnya, pada tahap evaluasi sudah dihitung nilai *fitness* dari tiap individu, maka pada proses seleksi akan dilakukan *sorting*/pemilihan individu dengan nilai *fitness* terbaik. Seleksi ini biasa disebut dengan seleksi *elitisim*. Proses seleksi lebih lanjut akan dijelaskan pada Gambar 4.7.

Berdasarkan Gambar 4.7, langkah-langkah seleksi dapat dijabarkan sebagai berikut:

1. Masukan individu gabungan (*parent* dan *child*/*offspring*), *popsize* serta nilai *fitness*.
2. Melakukan *sorting* individu berdasarkan nilai *fitness*. *Sorting* dilakukan dengan mengurutkan nilai *fitness* tertinggi ke *fitness* terendah.
3. Ambil individu dengan nilai tertinggi sejumlah *popsize*.

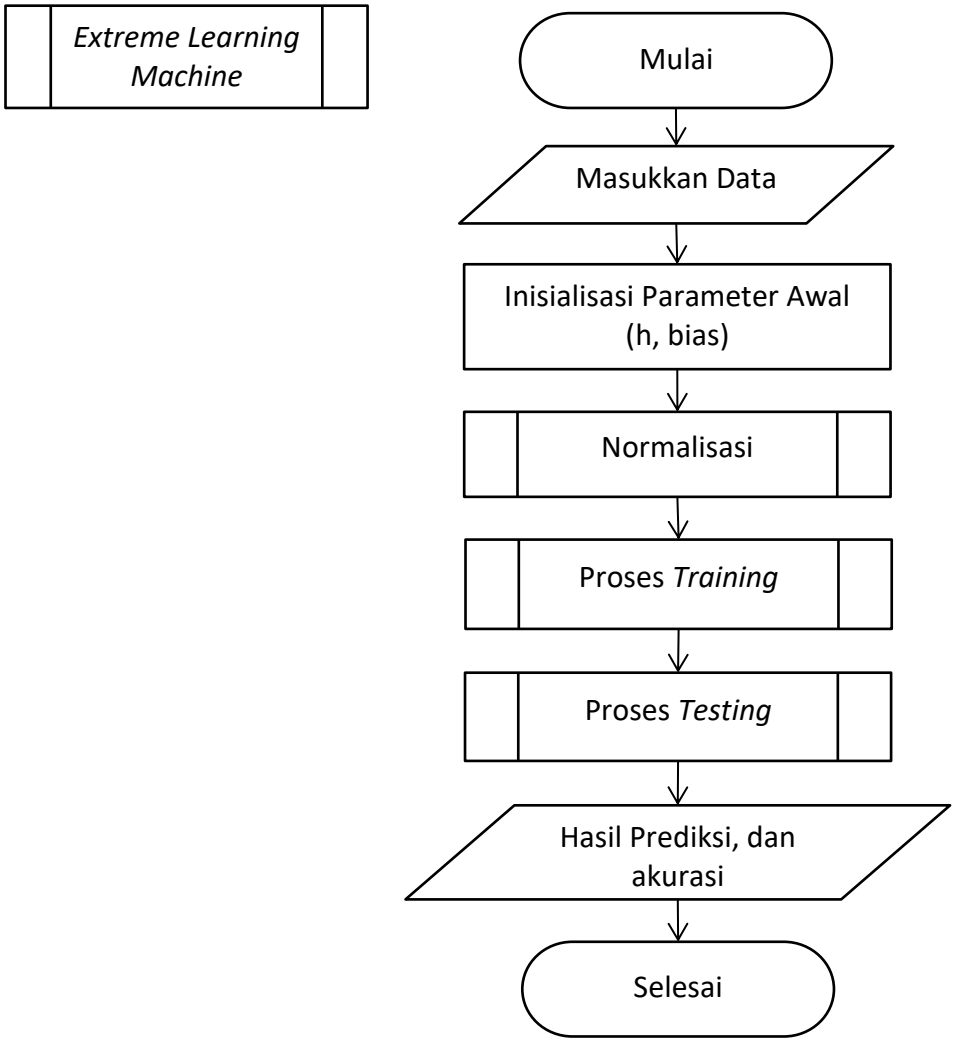
4. Dihasilkan individu terbaik yang lolos ke generasi berikutnya.



Gambar 4.7 *Flowchart* proses seleksi

4.3 Siklus Metode *Extreme Learning Machine* (ELM)

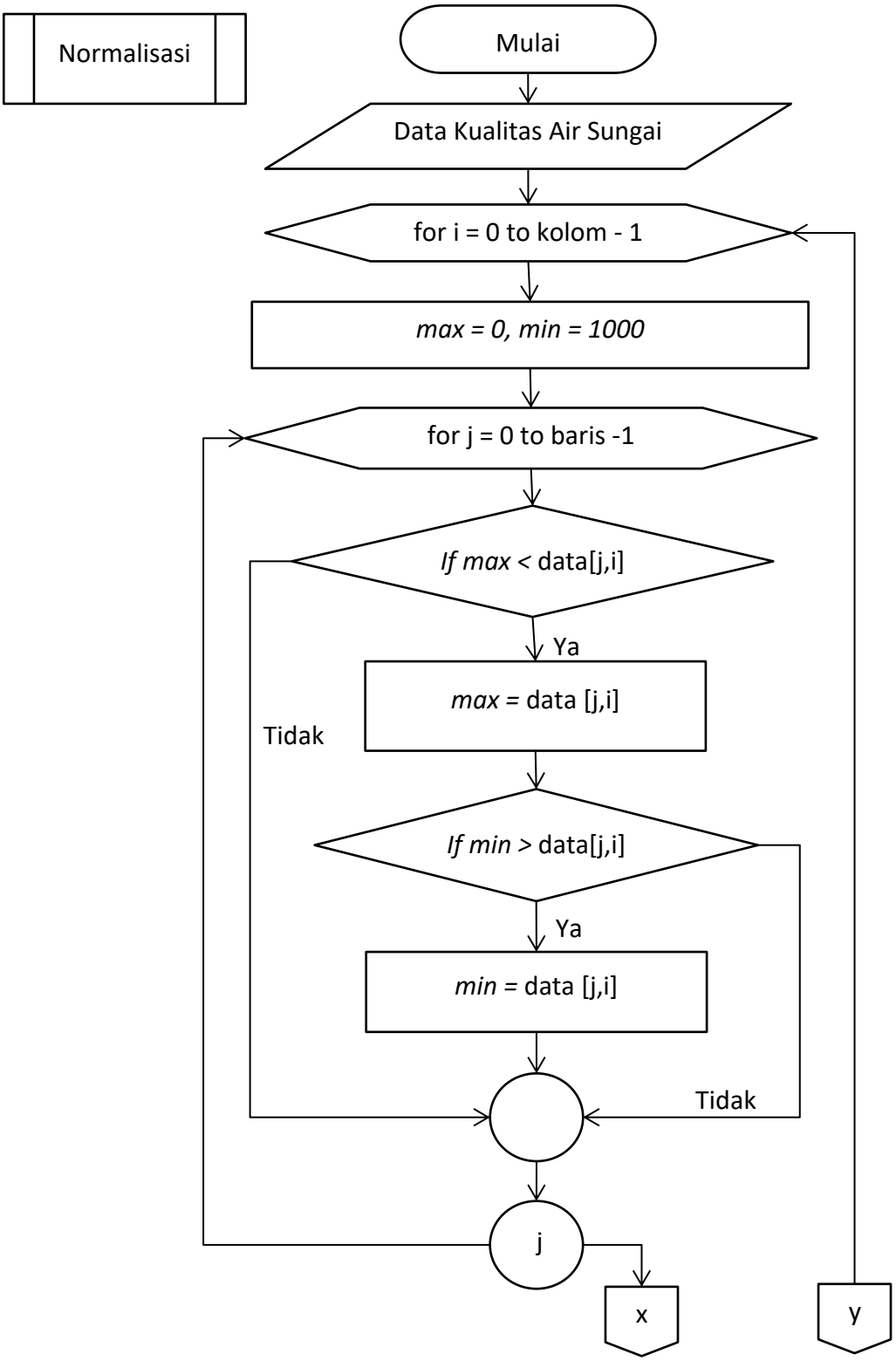
Siklus metode ELM berisi tentang proses dan tahapan penyelesaian masalah penentuan kualitas air sungai. Pada penelitian ini proses perhitungan ELM dilakukan dengan beberapa tahapan, diantaranya yaitu normalisasi data secara keseluruhan, baik data yang akan dijadikan data *training* maupun data *testing*, proses *training* dan proses *testing*. Proses komputasi lebih lanjut akan dijelaskan pada sub bab selanjutnya, yaitu sub bab perhitungan manual. Untuk aliran proses penyelesaian masalah menggunakan ELM yang diterapkan pada penelitian ini, dapat dilihat pada Gambar 4.8.

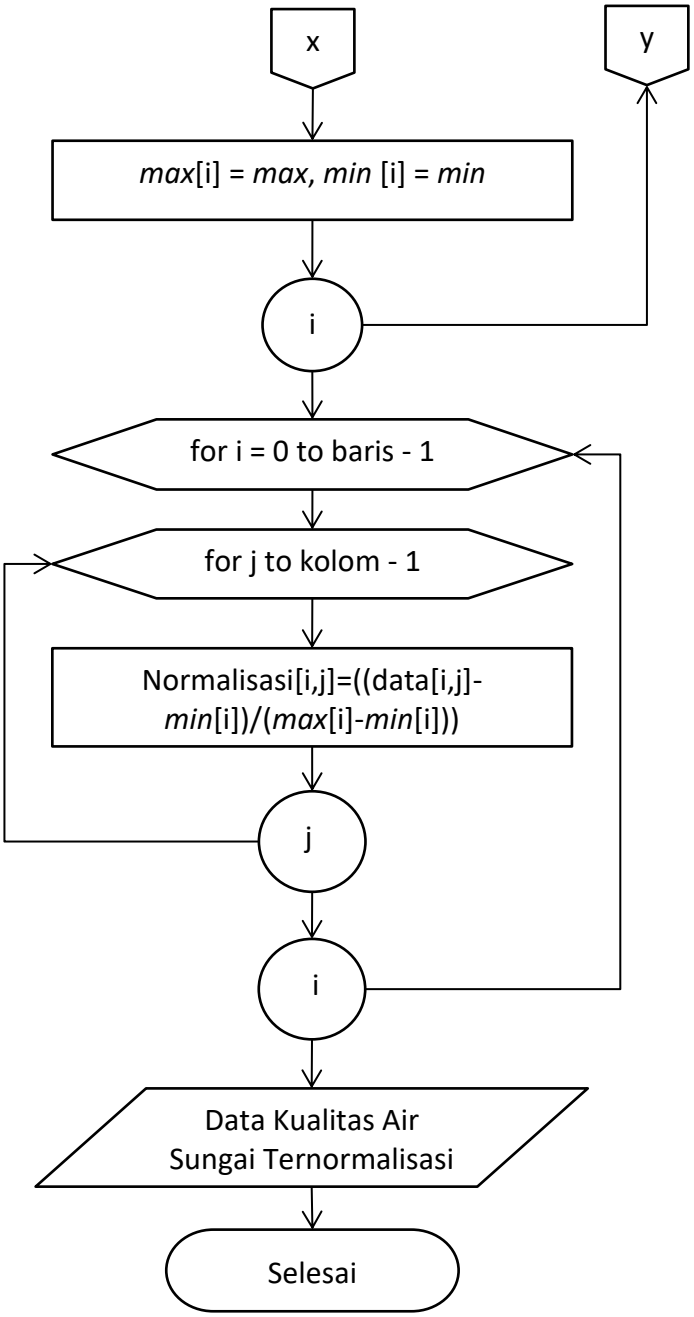


Gambar 4.8 *Flowchart Extreme Learning Machine*

4.3.1 Proses Normalisasi Data

Normalisasi data merupakan proses mengubah nilai dari suatu data yang bertujuan untuk mengurangi perbedaan *range* data. Metode yang digunakan untuk normalisasi data pada penilitian ini adalah metode *Min-Max*. Proses normalisasi data dapat dilihat pada Gambar 4.9.


Normalisasi
Mulai
Data Kualitas Air Sungai
for i = 0 to kolom - 1
max = 0, min = 1000
for j = 0 to baris -1
If max < data[j,i]
Ya
max = data [j,i]
Tidak
If min > data[j,i]
Ya
min = data [j,i]
Tidak
j
x
y



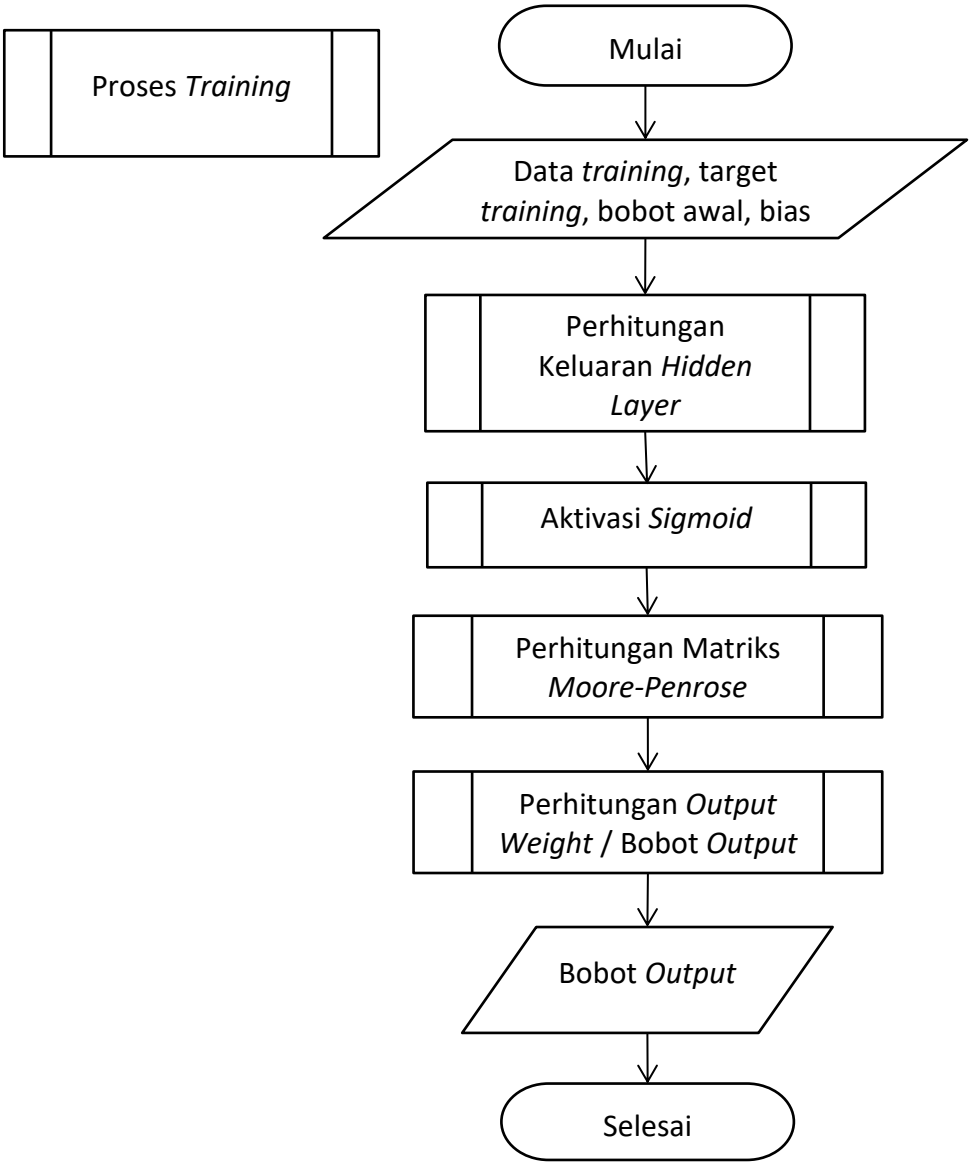
Gambar 4.9 *Flowchart* proses normalisasi

Berdasarkan Gambar 4.9, langkah-langkah proses normalisasi dapat dijabarkan sebagai berikut:

1. Masukan data kualitas air sungai yang memiliki ekstensi .xls.
2. Sistem mencari nilai minimal dan maksimal dari setiap parameter.
3. Sistem melakukan perhitungan normalisasi menggunakan persamaan normalisasi.
4. Dihasilkan data ternormalisasi dengan *range* antara 0-1.

4.3.2 Proses *Training*

Proses *training* merupakan proses mencari nilai *output weight* dengan beberapa proses meliputi perhitungan keluaran *hidden layer*, aktivasi, serta perhitungan matriks *Moore-Penrose*. Proses *training* dapat dilihat pada Gambar 4.10,



Gambar 4.10 *Flowchart* proses *training*

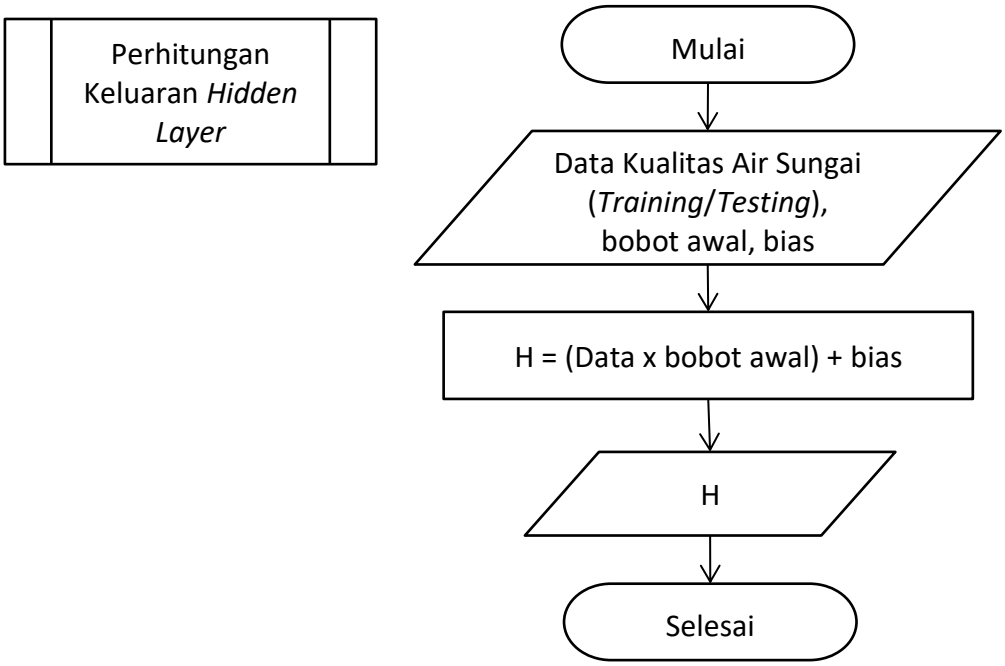
Berdasarkan Gambar 4.10, langkah-langkah proses *training* dapat dijabarkan sebagai berikut:

1. Masukan data kualitas air sungai yang dijadikan data latih/data *training*, target *training*, bobot awal dan bias dengan *file* berekstensi .xls.
2. Sistem melakukan perhitungan keluaran *hidden layer* kemudian dilakukan aktivasi menggunakan fungsi aktivasi *sigmoid biner* serta mencari matriks *Moore-Penrose*.

3. Dihasilkan bobot *output* yang akan digunakan pada proses *testing*.

4.3.3 Proses Perhitungan Keluaran *Hidden Layer*

Perhitungan keluaran *hidden layer* dilakukan pada proses *training* dan *testing*. Nilai keluaran *hidden layer* didapatkan dari hasil perkalian matriks data *training* maupun data *testing* yang kemudian akan dijumlahkan dengan biasnya. Proses perhitungan keluaran *hidden layer* dapat dilihat pada Gambar 4.11.



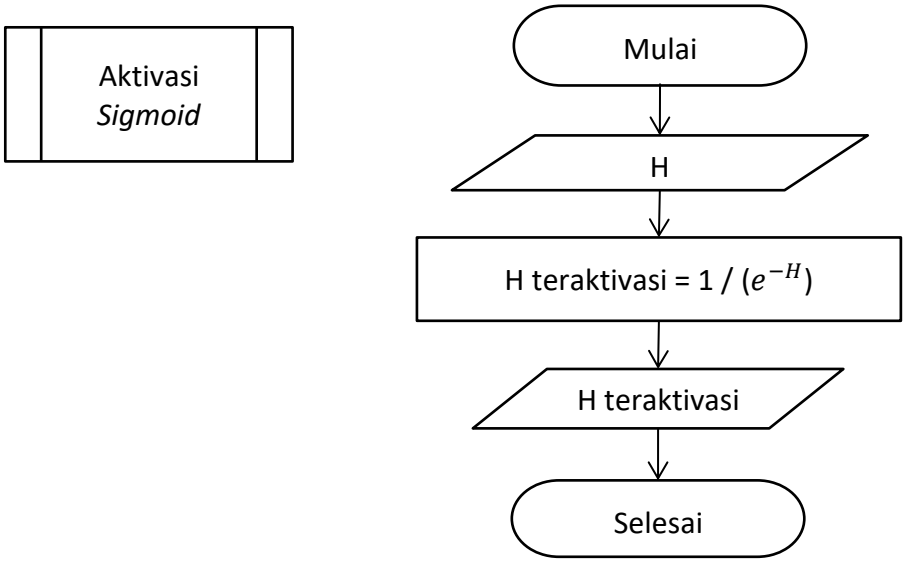
Gambar 4.11 *Flowchart* proses perhitungan keluaran *hidden layer*

Berdasarkan Gambar 4.11, langkah-langkah proses perhitungan keluaran *hidden layer* dapat dijabarkan sebagai berikut:

1. Masukan data kualitas air sungai data latih/data *training*, target *training*, bobot awal dan bias dengan *file* berekstensi .xls.
2. Sistem melakukan perhitungan keluaran *hidden layer* menggunakan Persamaan 2.2.
3. Dihasilkan keluaran *hidden layer*.

4.3.4 Proses Aktivasi *Sigmoid*

Aktivasi *sigmoid* dilakukan pada proses *training* dan *testing*. Proses aktivasi *sigmoid* dapat dilihat pada Gambar 4.12.



Gambar 4.12 *Flowchart* proses aktivasi *sigmoid*

Berdasarkan Gambar 4.12, langkah-langkah proses aktivasi *sigmoid* dapat dijabarkan sebagai berikut:

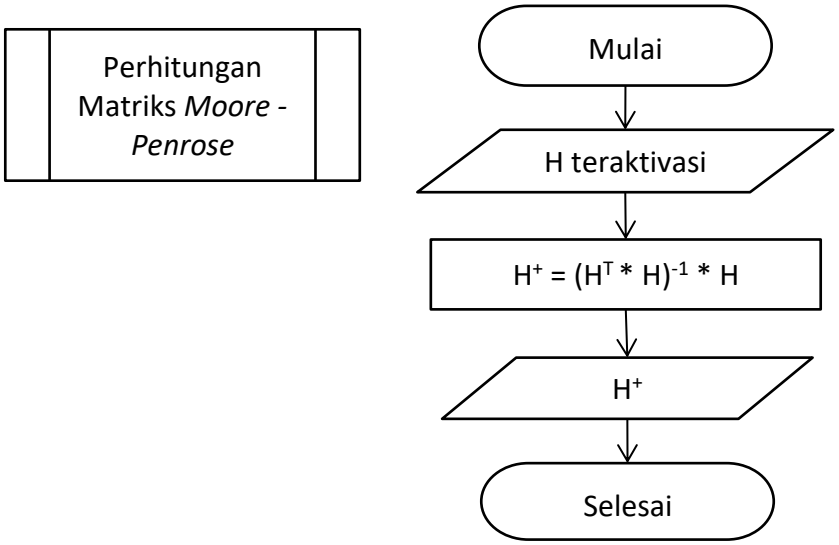
1. Masukan nilai H dari proses keluaran *hidden layer*.
2. Sistem melakukan perhitungan fungsi aktivasi menggunakan Persamaan 2.1.
3. Dihasilkan H teraktivasi.

4.3.5 Proses Perhitungan Matriks *Moore-Penrose*

Proses perhitungan matriks *Moore-Penrose* dilakukan hanya pada proses *training*, berbeda dengan proses keluaran *hidden layer* dan aktivasi *sigmoid* yang juga dilakukan pada proses *testing*. Pada proses perhitungan matriks ini melibatkan matriks keluaran *hidden layer* (H) yang telah diaktivasi dan beberapa operasi perhitungan pada matriks, diantaranya adalah perkalian, *transpose*, dan *inverse*. Proses perhitungan matriks *Moore-Penrose* dapat dilihat pada Gambar 4.13.

Berdasarkan Gambar 4.13, langkah-langkah proses aktivasi *sigmoid* dapat dijabarkan sebagai berikut:

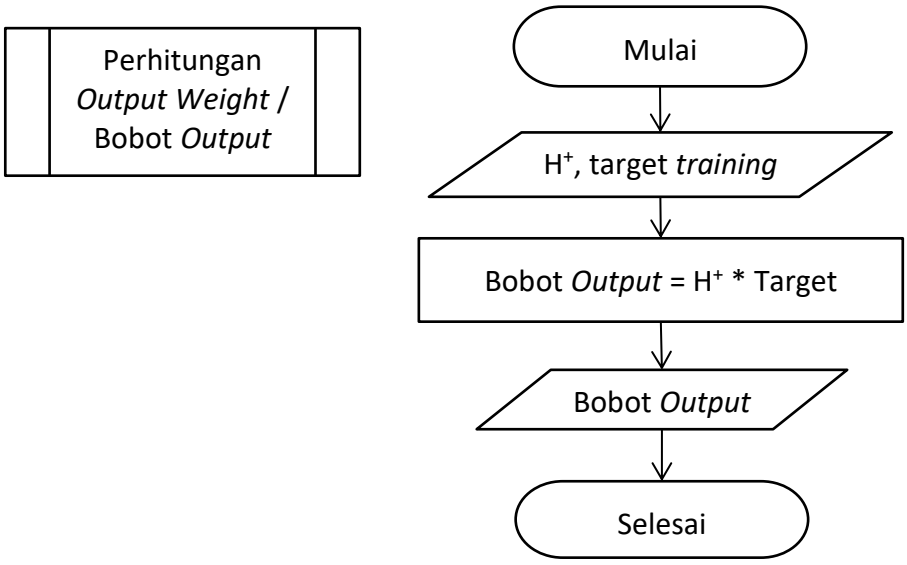
1. Masukan nilai H teraktivasi.
2. Sistem melakukan perhitungan fungsi matriks *Moore-Penrose* menggunakan Persamaan 2.3.
3. Dihasilkan H^{+}.



Gambar 4.13 *Flowchart* proses perhitungan matriks *Moore-Penrose*

4.3.6 Proses Perhitungan *Output Weight*

Perhitungan *output weight* hanya dilakukan pada proses *training*, akan tetapi hasil dari *output weight* akan digunakan dalam proses *testing*. Proses perhitungan *output weight* dapat dilihat pada Gambar 4.14.



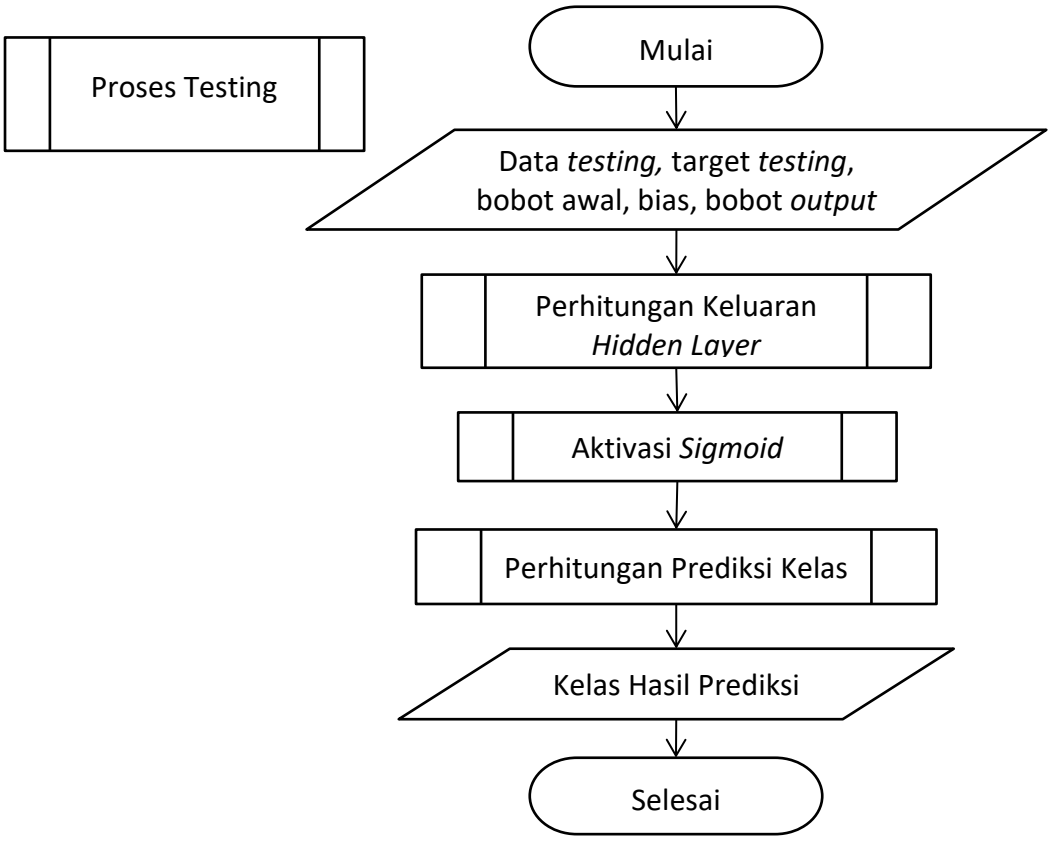
Gambar 4.14 *Flowchart* proses perhitungan *output weight*

Berdasarkan Gambar 4.14, langkah-langkah proses perhitungan *output weight* dapat dijabarkan sebagai berikut:

1. Masukan nilai H^+ dan target dari data *training*.
2. Sistem melakukan perhitungan *output weight* menggunakan Persamaan 2.4.
3. Dihasilkan bobot *output* / *output weight*.

4.3.7 Proses *Testing*

Proses *training* merupakan proses mencari hasil prediksi kelas kualitas air dengan beberapa proses meliputi perhitungan keluaran *hidden layer*, aktivasi, serta perhitungan prediksi kelas. Proses *testing* dapat dilihat pada Gambar 4.15.



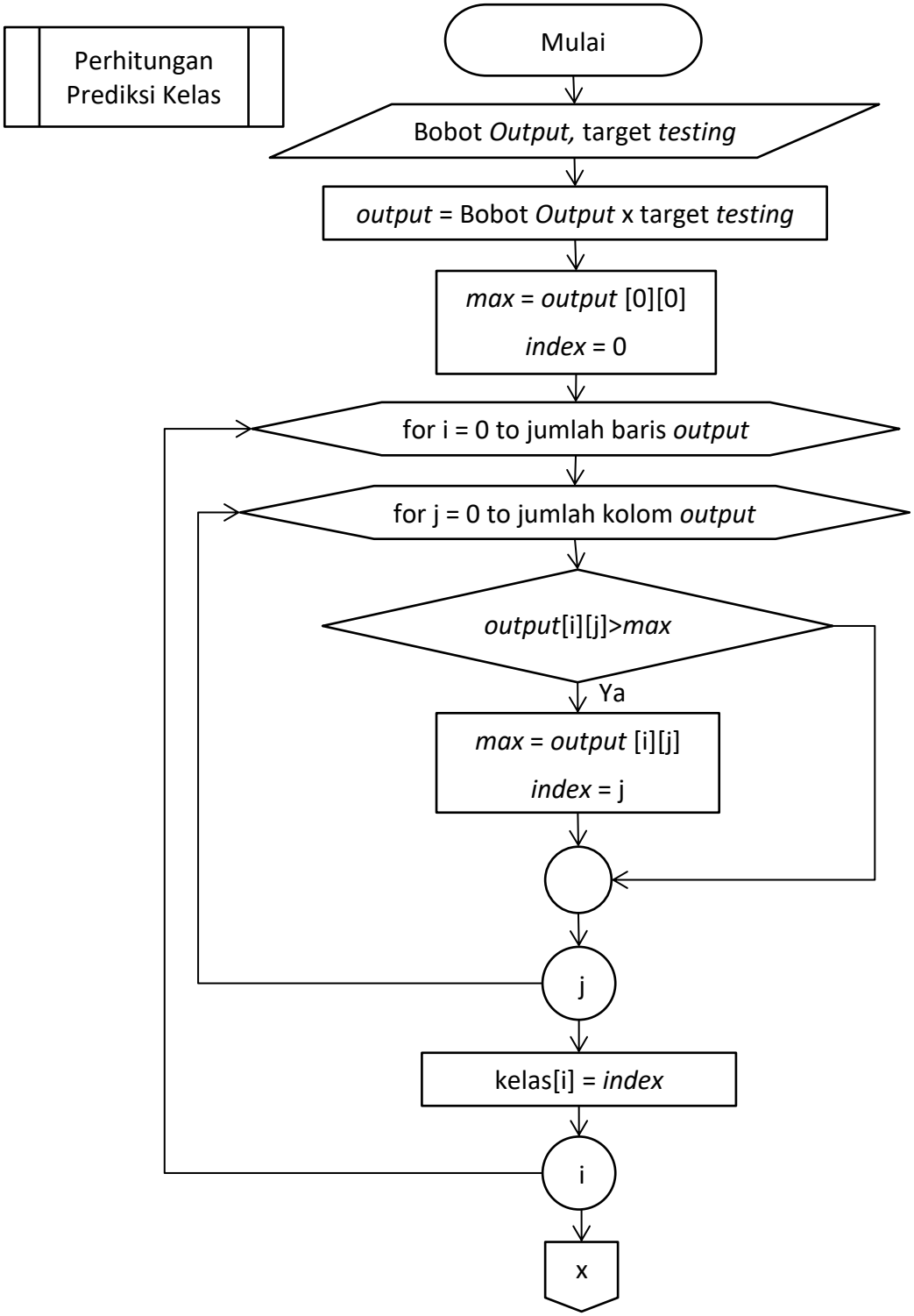
Gambar 4.15 *Flowchart* proses *testing*

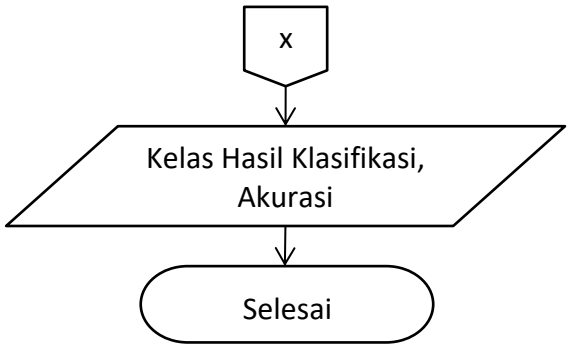
Berdasarkan Gambar 4.15, langkah-langkah proses *testing* dapat dijabarkan sebagai berikut:

1. Masukan data kualitas air sungai yang dijadikan data uji/data *testing*, target *testing*, bobot awal dan bias dengan *file* berekstensi .xls serta bobot *output* dari proses *training*.
2. Sistem melakukan perhitungan keluaran *hidden layer* kemudian dilakukan aktivasi menggunakan fungsi aktivasi *sigmoid biner* serta melakukan perhitungan prediksi kelas. Perhitungan keluaran *hidden layer* dan aktivasi *sigmoid* pada proses *testing* sama seperti perhitungan pada proses *training*.
3. Dihasilkan kelas hasil prediksi.

4.3.8 Proses Perhitungan Prediksi Kelas

Pada proses perhitungan prediksi kelas bertujuan untuk mendapatkan hasil klasifikasi. Proses perhitungan prediksi kelas dapat dilihat pada Gambar 4.16.


Perhitungan Prediksi Kelas
Mulai
Bobot *Output,* target *testing*
output = Bobot *Output* x target *testing*
max = *output* [0][0]
index = 0
for i = 0 to jumlah baris *output*
for j = 0 to jumlah kolom *output*
output[i][j]>*max*
Ya
max = *output* [i][j]
index = j
j
kelas[i] = *index*
i
x



Gambar 4.16 *Flowchart* proses perhitungan prediksi kelas

Berdasarkan Gambar 4.16, langkah-langkah proses perhitungan prediksi kelas dapat dijabarkan sebagai berikut:

1. Masukan bobot *output* dan target dari data *testing*.
2. Sistem melakukan perhitungan *output weight* menggunakan Persamaan 2.5.
3. Sistem melakukan pengecekkan nilai terbesar/maksimal. Nilai output terbesar akan diambil nilai indeksnya. Indeks dari nilai tersebut merupakan kelas hasil klasifikasinya.
4. Dihasilkan kelas hasil klasifikasi.

4.4 Perhitungan Manual

Pada penelitian ini, perhitungan manual dilakukan menggunakan data sebanyak 60 data kualitas air sungai, serta bobot awal yang akan dioptimasi dalam bentuk matriks dengan ordo 2 x 7 serta dengan klasifikasi ke dalam 3 kelas berbeda yaitu tercemar ringan, sedang dan berat, masing-masing kelas dikodekan dengan bilangan *integer* 1, 2 dan 3.

4.4.1 Inisialisasi Kromosom Awal

Kromosom awal dibangkitkan secara *random*, dengan *range* antara -1 s.d 1. Diketahui bahwa parameter Algoritme Genetika yang digunakan adalah sebagai berikut:

- Ukuran Populasi/*Popsize* : 4
- *Crossover rate* (cr) : 0,4
- *Mutation rate* (mr) : 0,2

Maka diperoleh 4 individu yang dibangkitkan secara *random* seperti yang terlihat pada Tabel 4.2.

Tabel 4.2 Inisialisasi kromosom awal

P1	0.55446	0.51253	0.52456	0.75643	0.51337	0.76815	0.57826	0.15947	0.25576	0.16099	0.62983	0.81140	0.91643	0.55148
P2	0.15162	0.94840	0.85896	0.67853	0.15761	0.50498	0.83315	0.33086	0.97367	0.34642	0.39449	0.32985	0.31198	0.74689
P3	0.82545	0.51893	0.58823	0.43719	0.13248	0.51837	0.45780	0.90255	0.67639	0.98517	0.20272	0.71358	0.82321	0.82302
P4	0.17582	0.78259	0.85220	0.41321	0.39399	0.55467	0.79016	0.71885	0.76107	0.66076	0.36333	0.97142	0.52096	0.67252

4.4.2 Reproduksi

Setelah dilakukan inisialisasi kromosom awal, langkah selanjutnya adalah proses reproduksi. Reproduksi pada penelitian ini terdapat 2 proses, yaitu *crossover* dan mutasi. Jenis *crossover* yang digunakan adalah *extended intermediate crossover*, sedangkan jenis mutasi yang digunakan adalah *random mutation*. Dalam perhitungan *extended intermediate crossover* diperlukan bantuan 1 variabel lagi, yaitu a atau *alpha*. Variabel a dibangkitkan secara *random* dengan *range* [-0,25; 1,25] (lihat Tabel 4.3). Sama halnya dengan *extended intermediate crossover*, *random mutation* juga memerlukan variabel tambahan, yaitu r, dimana r dibangkitkan secara *random* dengan *range* [-0,1; 0,1]. Berikut contoh perhitungan manualnya:

- **Pembangkitan variabel a dan r.**

Tabel 4.3 Pembangkitan nilai a

a	0,18	0,11	1.07	0,19	-0,06	1.24	0,39	0,44	0,73	-0,23	0,86	0,01	0,07	-0,16

r = -0,0584

- **Proses *crossover***

  Karena *crossover rate* diketahui sebesar 0,4; maka *offspring/child* yang akan dihasilkan dari proses *crossover* sebanyak cr x *popsize*, yaitu 0,4 x 4 (≈ 2). Perhitungan *crossover*:

  C1 = P1 + a (P2 – P1)

  = 0,55446 + 0,18(0,15162-0,55446)

  = 0,481951

  C2 = P2 + a (P1 – P2)

  = 0,15162 + 0,18(0,55446-0,15162)

  = 0,224134

- **Proses mutasi**

  Offspring yang dihasilkan dari proses mutasi sebanyak mr x *popsize* (0,2 x 4 ≈ 1). Misalkan ambil individu P4 untuk dilakukan mutasi. Untuk perhitungan *random mutation*, lihat contoh perhitungan berikut.

$x'_i = x'_i + r\ (max_i - min_j)$

$= 0{,}78259 + (-0{,}0584)(0{,}9484-0{,}51253)$

$= 0{,}75714$

Hasil reproduksi dapat dilihat pada Tabel 4.4.

Tabel 4.4 Hasil reproduksi (*offspring*)

C1	0.4820	0.5605	0.8824	0.7416	0.5347	0.4418	0.6777	0.2349	0.7798	0.1183	0.4274	0.8066	0.8741	0.5202
C2	0.2241	0.9005	0.5011	0.6933	0.1363	0.8313	0.7337	0.2554	0.4496	0.3891	0.5969	0.3347	0.3543	0.7781
C3	0.1758	0.7571	0.8522	0.4132	0.3940	0.5547	0.7902	0.7189	0.7611	0.6608	0.3633	0.9714	0.5210	0.6725

Offspring C1 dan C2 berasal dari hasil *crossover parent* P1 dan P2, sedangkan C3 hasil mutasi dari *parent* P4. Sehingga akan didapatkan individu gabungan seperti yang ditunjukkan pada Tabel 4.5.

Tabel 4.5 Individu Gabungan

P1	0.5545	0.5125	0.5246	0.7564	0.5134	0.7681	0.5783	0.1595	0.2558	0.1610	0.6298	0.8114	0.9164	0.5515
P2	0.1516	0.9484	0.8590	0.6785	0.1576	0.5050	0.8332	0.3309	0.9737	0.3464	0.3945	0.3299	0.3120	0.7469
P3	0.8255	0.5189	0.5882	0.4372	0.1325	0.5184	0.4578	0.9025	0.6764	0.9852	0.2027	0.7136	0.8232	0.8230
P4	0.1758	0.7826	0.8522	0.4132	0.3940	0.5547	0.7902	0.7189	0.7611	0.6608	0.3633	0.9714	0.5210	0.6725
C1	0.4820	0.5605	0.8824	0.7416	0.5347	0.4418	0.6777	0.2349	0.7798	0.1183	0.4274	0.8066	0.8741	0.5202
C2	0.2241	0.9005	0.5011	0.6933	0.1363	0.8313	0.7337	0.2554	0.4496	0.3891	0.5969	0.3347	0.3543	0.7781
C3	0.1758	0.7571	0.8522	0.4132	0.3940	0.5547	0.7902	0.7189	0.7611	0.6608	0.3633	0.9714	0.5210	0.6725

4.4.3 Proses Perhitungan ELM

4.4.3.1 Normalisasi Data

Setelah mendapatkan *offspring*/*child*, tahap selanjutnya adalah normalisasi data, tahap awal dari perhitungan metode ELM. Berikut contoh perhitungan manualisasinya, setelah diketahui nilai minimal dan maksimal dari keseluruhan data yang ditunjukkan pada Tabel 4.6.

Tabel 4.6 Nilai minimal dan maksimal setiap parameter

	TSS	BOD	COD	DO	pH	Fenol	Minyak & Lemak
Max	229,8	21,3	97,15	42,25	9	0,3825	6,85
Min	0	1,925	6,408	3,5	6,55	0	0

Misalkan data yang akan dinormalisasi adalah data kualitas air sungai pada bulan Oktober tahun 2009 (Tabel 4.1).

$$x = \frac{data_{awal} - data_{min}}{data_{max} - data_{min}}$$

$$TSS = \frac{42{,}3 - 0}{229{,}8 - 0} = 0{,}18407$$

$$BOD = \frac{3{,}75 - 1{,}925}{21{,}3 - 1{,}925} = 0{,}09419$$

$$COD = \frac{9{,}7375 - 6{,}408}{97{,}15 - 6{,}408} = 0{,}03669$$

$$DO = \frac{42{,}25 - 3{,}5}{42{,}25 - 3{,}5} = 1$$

$$pH = \frac{7{,}1 - 6{,}55}{9 - 6{,}55} = 0{,}22449$$

$$Fenol = \frac{0 - 0}{0{,}3825 - 0} = 0$$

$$Minyak\ dan\ Lemak\ = \frac{4{,}75 - 0}{6{,}85 - 0} = 0{,}69343$$

Maka akan diperoleh hasil seperti pada Tabel 4.7

Tabel 4.7 Hasil normalisasi

Bulan, Tahun	TSS	BOD	COD	DO	pH	Fenol	Minyak & Lemak
Oktober 2009	0,18407	0,09419	0,03669	1	0,22449	0	0,69343

4.4.3.2 Pembagian Data *Training* dan Data *Testing* Menggunakan K-*Fold Cross Validation*

Pada proses pembagian data *training* dan data *testing* menggunakan k-*fold cross validation*, perlu ditentukan jumlah k terlebih dahulu, sehingga bisa didapatkan *fold-fold* untuk dijadikan data *training* maupun *testing*. Berikut ini merupakan contoh perhitungan manualisasi pengambilan data dari masing-masing kelas untuk setiap *fold*:

Misalkan nilai k=3, dan diketahui jumlah data sebanyak 60 buah, dengan rincian kelas seperti yang digambarkan pada Tabel 4.8.

Tabel 4.8 Rincian jumlah data pada tiap kelas

Kelas 1	Kelas 2	Kelas 3	Total
3 data	44 data	13 data	60 data

Pengambilan data dilakukan dengan cara menghitung banyaknya data tiap kelas yang diambil untuk dimasukkan ke dalam *fold*, menggunakan persamaan sebagai berikut:

$$x = \frac{100}{k}\%$$

Pada sub bab 4.4.3.2 telah didefinisikan, k bernilai 3, maka x = 33,3%. Artinya *fold* dibentuk, dengan data yang diambil di tiap kelas sebesar 33% dari jumlah data di kelas tersebut. Contoh:

$Kelas\ 1 = 33\%\ x\ 3\ data \approx 1\ data$

$Kelas\ 2 = 33\%\ x\ 44\ data \approx 15\ data$

$Kelas\ 3 = 33\%\ x\ 13\ data \approx 4\ data$

Dari uraian perhitungan manual diatas nilai k berarti jumlah *fold* yang akan dibuat, dan pada masing-masing *fold* terdapat 1 data kelas 1, 15 data kelas 2, dan 4 data kelas 3, untuk *fold* terakhir boleh dimasukkan data sisa (lebih dari perhitungan pembagian kelas, misalkan memasukkan data kelas 2 sebanyak 17 data, padahal seharusnya cukup hanya dengan 15 data). Perlu dicatat bahwa tidak diperbolehkan ada kesamaan data antar *fold*.

Setelah terbentuk 3 *fold*, pilih salah satu *fold* untuk menjadi data *testing*, sedangkan *fold* yang lainnya akan digunakan sebagai data *training*. Contoh pengambilan data pada setiap *fold* dapat dilihat pada Tabel 4.9, 4.10 dan 4.11.

Tabel 4.9 Contoh pengambilan data pada *fold* 1

No	x1	x2	x3	x4	x5	x6	x7	Kelas
13	0,152306	0,029677	0,052809	0,139355	0,346939	0,491503	0,116788	1
1	0	0,692903	0,555333	0,19871	0,387755	0,202614	0,116788	2
5	0,043516	0,220645	0,082564	0,171613	0,408163	0,449673	0,218978	2
6	0,02698	0,483871	0,316193	0,346452	0,77551	0,224837	0,208029	2
7	0,105527	0,259355	0,101298	0,189677	0,306122	0,175163	0,116788	2
8	0,11423	0,308387	0,160808	0,212903	0,183673	0,512418	0	2
9	0,075718	0,256774	0,109012	0,114839	0,102041	0,334641	0,131387	2
10	0,092472	0,223226	0,077054	0,101935	0,163265	0,24183	0,116788	2
11	0,077241	0,16129	0,156399	0	0,142857	0,231373	0,116788	2
12	0,110313	0,112258	0,115074	0,083871	0,183673	0,394771	0,510949	2
14	1	0,099355	0,157501	0,095484	0,163265	0,296732	0	2
15	0,179721	0,56129	0,386172	0,16	0,571429	0,239216	0	2
17	0,103133	0,527742	0,28754	0,218065	0,673469	0,020915	0	2
18	0,077459	0,076129	0,04234	0,085161	0,367347	0,460131	0	2
19	0,030896	0,086452	0,037932	0,060645	0,326531	0,271895	0,189781	2
2	0	0,416774	0,312887	0,304516	0,612245	0,037908	0,656934	3
3	0,19517	0,550968	0,319499	0,388387	0,938776	0,309804	0,620438	3
4	0,31658	0,92	0,622005	0,179355	0,632653	0,449673	0	3
16	0,302872	1	1	0,250323	0,795918	0,311111	0	3

Tabel 4.10 Contoh pengambilan data pada *fold* 2

No	x1	x2	x3	x4	x5	x6	x7	Kelas
40	0,146214	0,029677	0,025809	0,068387	0,408163	0,101961	0,056934	1
20	0,06832	0,202581	0,145379	0,082581	0,265306	0,488889	0,423358	2
…	…	…	…	…	…	…	…	…
23	0,238903	0,298065	0,447334	0,131613	0,326531	0,271895	0,456204	3
…	…	…	…	…	…	…	…	…

Tabel 4.11 Contoh pengambilan data pada *fold* 3

No	x1	x2	x3	x4	x5	x6	x7	Kelas
55	0,087685	0	0	0,112258	0,530612	0	0,416058	1
39	0,367711	0,029677	0,020299	0,175484	0,693878	0,151634	0,729927	2
…	…	…	…	…	…	…	…	…
38	0,380113	0,091613	0,213705	0,100645	0,632653	0,320261	0,423358	3
…	…	…	…	…	…	…	…	…

4.4.3.3 Prediksi Kualitas Air Sungai Menggunakan ELM

Pada proses prediksi kualitas air sungai mengggunakan ELM, langkah awal yang harus dilakukan adalah menginisialisasi parameter ELM yang akan digunakan, diantaranya adalah h (*hidden layer*), i (*input layer*), bobot awal (yang akan dioptimasi menggunakan Algoritme Genetika), bias serta data kualitas air sungai yang akan digunakan.

Langkah 1: Inisialisasi Parameter

Misalkan diketahui: h = 2, i = 7

Bobot awal diinisialisasi dari salah satu individu gabungan (lihat Tabel 4.5). Semua individu gabungan akan dijadikan sebagai bobot awal secara bergantian. Karena telah diketahui nilai h = 2 dan i = 7, maka ukuran matriks bobot awal harus h x I atau 2 x 7. Pada individu gabungan P1, P2, P3 dan seterusnya memiliki ukuran matriks 1 x 14. Oleh karena itu ukuran matriks individu gabungan akan diubah menjadi ukuran matriks bobot awal. Misalkan, diambil contoh dari individu gabungan P4 sebagai bobot awal. Bobot awal yang telah dikonversi ke ukuran matriks yang sebenarnya dapat dilihat pada Tabel 4.12.

Tabel 4.12 Bobot awal

W	1	2	3	4	5	6	7
1	0,175821	0,782592	0,852197	0,413205	0,393991	0,554672	0,790159
2	0,718851	0,761067	0,660759	0,363328	0,971423	0,520959	0,672516

Bias dibangkitkan secara *random*, dengan *range* [0,1] serta dengan *ordo* h x 1. Inisialisasi bias dapat dilihat pada Tabel 4.13.

Tabel 4.13 Bias

B	1
1	0,352154
2	0,798536

Data yang digunakan dalam perhitungan manual ada 2 jenis data, yaitu data *training* dan data *testing*. Data *training* dibangkitkan dari *fold* 2 dan *fold* 3, sedangkan data *testing* dibangkitkan dari *fold* 1.

Langkah 2: Proses *Training*

Langkah 2.1: Hitung Keluaran *Hidden Layer*

Menghitung keluaran *hidden layer* menggunakan Persamaan 2.2, dimana data *input* yang digunakan merupakan data *training* dengan *ordo* 41 x 7, w^T merupakan bobot awal yang di-*transpose* sehingga memiliki *ordo* 7 x 2, contoh perhitungan manual dari perhitungan keluaran *hidden layer* pada baris 2 kolom 1:

$$H_{(2,1)} = [(0{,}087685 \times 0{,}175821) + (0 \times 0{,}782592) + (0 \times 0{,}852197) + (0{,}112258 \times 0{,}413205) + (0{,}530612 \times 0{,}393991) + (0 \times 0{,}554672) + (0{,}416058 \times 0{,}790159)] + 0{,}352154$$

$$= 0{,}599611$$

Hasil akhir keluaran *hidden layer* (memiliki *ordo* 41 x 2) dapat dilihat pada Tabel 4.14.

Tabel 4.14 Hasil keluaran *hidden layer*

H	1	2
1	0,713694	1,456035
2	0,951765	1,697609
3	1,39094	1,92502
…	…	…
40	1,350169	2,333247
41	1,539075	2,074534

Langkah 2.2: Hitung Fungsi Aktivasi *Sigmoid Biner*

Memetakan nilai keluaran *hidden layer* menggunakan aktivasi *sigmoid biner*. Persamaan yang digunakan adalah Persamaan 2.1, dimana x merupakan keluaran *hidden layer* atau H, contoh perhitungan manual aktivasi *sigmoid biner* pada baris 2 kolom 1:

$$H_{(2,1)} = \frac{1}{1+e^{-0{,}599611}} \quad = 0{,}72147$$

Hasil akhir fungsi aktivasi *sigmoid biner* (memiliki *ordo* 41 x 2) dapat dilihat pada Tabel 4.15.

Tabel 4.15 Hasil fungsi aktivasi *sigmoid biner*

H	1	2
1	0,713694	1,456035
2	0,951765	1,697609
3	1,39094	1,92502
…	…	…
40	1,350169	2,333247
41	1,539075	2,074534

Langkah 2.3: Hitung Bobot *Output*

Menghitung bobot *output* menggunakan matriks *Moore-Penrose* yang dikalikan dengan target matriks berupa kelas dari data *training*. Persamaan untuk menghitung matriks *Moore-Penrose* adalah Persamaan 2.3. sedangkan untuk mencari bobot *output* digunakan Persamaan 2.4. Berikut contoh perhitungan manual menghitung matriks *Moore-Penrose Pseudo Inverse*:

$$H^T H = \begin{bmatrix} 24{,}77563 & 27{,}79535 \\ 27{,}79535 & 31{,}20915 \end{bmatrix}$$

$$(H^T H)^{-1} = \frac{1}{\det A}\begin{bmatrix} 24{,}77563 & -27{,}79535 \\ -27{,}79535 & 31{,}20915 \end{bmatrix}$$

$$= \frac{1}{0{,}64487}\begin{bmatrix} 31{,}20915 & -27{,}79535 \\ -27{,}79535 & 24{,}77563 \end{bmatrix}$$

$$= \begin{bmatrix} 48{,}41237 & -43{,}1168 \\ -43{,}1168 & 38{,}43253 \end{bmatrix}$$

Contoh perhitungan manual menghitung matriks *Moore-Penrose Pseudo Inverse* baris ke-1 kolom ke-1.

$$H^{+}_{(1,1)} = [(48{,}41237 \times 0{,}671217) + (-43{,}1168 \times 0{,}810925)] = -2{,}46931$$

Pada Tabel 4.16 ditunjukkan matriks *Moore-Penrose Pseudo Inverse* yang memiliki *ordo* 2 x 41.

Tabel 4.16 Matriks *Moore-Penrose Pseudo Inverse*

H^+	1	2	3	…	40	41
1	-2,4693	-1,5152	1,13792	…	-0,858	1,55425
2	2,22519	1,37655	-0,9855	…	0,79332	-1,3558

Sebelum melakukan perhitungan bobot *output*, ubah matriks target seperti contoh pada Tabel 4.17.

Tabel 4.17 Matriks *Moore-Penrose Pseudo Inverse*

Target Awal	Target		
	t 1	t 2	t 3
1	1	0	0
1	1	0	0
2	0	1	0
2	0	1	0
2	0	1	0
...	...	...	...
3	0	0	1
3	0	0	1
...	...	...	...

Target awal ke-i akan bernilai 1 pada kolom t(i), sedangkan pada kolom lainnya akan bernilai 0, Kemudian dilakukan perhitungan bobot *output* dengan contoh perhitungan manual sebagai berikut ini.

$$\beta_{1,1} = [(-2{,}4693 \times 1) + (-1{,}5152 \times 1) + (1{,}13792 \times 0) + \cdots + (-0{,}858 \times 0) + (1{,}55425 \times 0)] = -3{,}98452$$

$$\beta_{2,1} = [(2{,}22519 \times 1) + (1{,}37655 \times 1) + (-0{,}9855 \times 0) + \cdots + (0{,}79332 \times 0) + (-1{,}3558 \times 0)] = 3{,}601737$$

Hasil perhitungan bobot output dapat dilihat pada Tabel 4.18.

Tabel 4.18 Hasil bobot output

β	1
1	-3,98452
2	3,601737

Langkah 3: Proses *Testing*

Langkah 3.1: Hitung Keluaran *Hidden Layer*

Sama seperti pada proses *training*, dalam proses *testing* juga dilakukan perhitungan keluaran *hidden layer* menggunakan persamaan $H = N * w^T + b$, perbedaannya N pada *testing* merupakan data *testing* dengan *ordo* 19 x 7, w^T merupakan bobot awal yang di-*transpose* sehingga memiliki *ordo* 7 x 2, contoh perhitungan manual dari perhitungan keluaran *hidden layer* pada baris 1 kolom 1:

$$H_{(1,1)} = [(0{,}152306 \times 0{,}175821) + (0{,}029677 \times 0{,}782592) + (0{,}052809 \times 0{,}852197) + (0{,}139355 \times 0{,}413205) + (0{,}346939 \times 0{,}393991) + (0{,}491503 \times 0{,}554672) + (0{,}116788 \times 0{,}790159)] + 0{,}352154$$

$$= 1{,}006339$$

Hasil akhir keluaran *hidden layer* (memiliki *ordo* 19 x 2) dapat dilihat pada Tabel 4.19.

Tabel 4.19 Hasil keluaran *hidden layer*

H	1	2
1	1,006339	1,687752
2	1,807213	2,325789
3	1,257014	1,892676
…	…	…
18	2,230662	3,051286
19	2,629777	3,46428

Langkah 3.2: Hitung Fungsi Aktivasi *Sigmoid Biner*

Contoh perhitungan manual aktivasi *sigmoid biner* pada baris 2 kolom 1:

$$H_{(2,1)} = \frac{1}{1+e^{-1,807213}}$$
$$= 0,859024681$$

Hasil akhir fungsi aktivasi *sigmoid biner* (memiliki *ordo* 19 x 2) dapat dilihat pada Tabel 4.20.

Tabel 4.20 Hasil fungsi aktivasi *sigmoid biner*

H	1	2
1	0,732303	0,843928
2	0,859025	0,91099
3	0,778512	0,86906
…	…	…
18	0,902969	0,954838
19	0,932754	0,969654

Langkah 3.3: Hitung *Output* dan Prediksi

Contoh perhitungan manual *output*:

$$Output = H\beta$$
$$Output_{1,1} = [(0,732303 \times -3,98452) + (0,843928 \times 3,601737)]$$
$$= 0,121735$$
$$Output_{1,2} = [(0,732303 \times -3,69131) + (0,843928 \times 4,119032)]$$
$$= 0,773011$$

$$Output_{1,3} = [(0{,}732303 \times 6{,}389059) + (0{,}843928 \times -5{,}42969)]$$
$$= 0{,}096457$$

Setelah didapatkan *output*-nya, untuk mengetahui hasil prediksi, harus dihitung nilai maksimal antar *output*. Nilai maksimal dengan kolom ke n merupakan kelas klasifikasinya. Pada perhitungan manual diatas, dapat diambil kesimpulan bahwa *output* $_{1,2}$ memiliki nilai tertinggi, maka ia diprediksikan di kelas 2. Dengan perhitungan data lengkap, akan dihasilkan *output* (dengan data *fold* 3 sebagai data uji) seperti pada Tabel. 4.21.

Tabel 4.21 *Output* dan hasil prediksi

No.	*Output* 1	*Output* 2	*Output* 3	Nilai Maksimal	Kelas Prediksi	Kelas Asli
1	0,121735	0,773011	0,096457	0,773011	2	1
2	-0,14165	0,581474	0,541962	0,581474	2	2
3	0,028135	0,705961	0,255227	0,705961	2	2
4	-0,05467	0,674042	0,407784	0,674042	2	2
5	0,08301	0,735795	0,15796	0,735795	2	2
6	0,007152	0,672578	0,282713	0,672578	2	2
7	0,01627	0,657464	0,257685	0,657464	2	2
8	0,070541	0,703323	0,168712	0,703323	2	2
9	0,064948	0,689968	0,17407	0,689968	2	2
10	-0,03283	0,640709	0,3491	0,640709	2	2
11	0,269689	0,942551	-0,12644	0,942551	2	2
12	-0,01419	0,700947	0,338197	0,700947	2	2
13	0,051864	0,752951	0,228225	0,752951	2	2
14	0,150259	0,786542	0,045023	0,786542	2	2
15	0,112796	0,746445	0,102712	0,746445	2	2
16	-0,11829	0,612821	0,50879	0,612821	2	3
17	-0,15477	0,608344	0,580237	0,608344	2	3
18	-0,15882	0,59987	0,584649	0,59987	2	3
19	-0,22413	0,550955	0,694495	0,694495	3	3

Langkah 3.4 : Hitung Akurasi

Perhitungan akurasi ini akan digunakan untuk proses evaluasi dan seleksi pada Algoritme Genetika. Nilai akurasi ini digunakan sebagai nilai *fitness* dari tiap individu. Perhitungan akurasi dihitung menggunakan Persamaan 2.8.

$$Akurasi = \frac{\sum data\ uji\ benar}{\sum total\ data\ uji} x100\%$$
$$= \frac{15}{19} x100\%$$
$$= 78{,}94737\ \%$$

4.4.4 Evaluasi dan Seleksi

Di dalam proses evaluasi dilakukan perhitungan nilai *fitness*, sedangkan di dalam proses seleksi dilakukan *sorting*/pengurutan nilai *fitness* agar memudahkan untuk memilih individu terbaik yang lolos ke generasi selanjutnya. Berikut merupakan contoh perhitungan manual nilai *fitness*.

$$f(x) = AkurasiPrediksiELM(x)$$

$$f(P4) = AkurasiPrediksiELM(P4)$$

$$= 78{,}94737$$

Setelah dilakukan perhitungan nilai *fitness* individu gabungan yang pertama *fold* pertama, lakukan langkah-langkah pada sub bab 4.4.4.3 dengan *fold* yang berbeda hingga *fold* ke-k. Kemudian lanjutkan dengan individu gabungan ke-2 hingga seterusnya dengan langkah yang sama. Contoh hasil evaluasi dapat dilihat pada Tabel 4.22.

Tabel 4.22 Hasil evaluasi

Individu	Fitness
…	…
P1_fold3	78,94737
P2_fold3	89,47368
P3_fold3	84,21053
P4_fold3	78,94737
C1_fold3	89,47368
C2_fold3	84,21053
C3_fold3	78,94737
…	…

Contoh hasil *sorting* individu yang lolos ke generasi berikutnya dapat dilihat pada Tabel 4.23 dan 4.24.

Tabel 4.23 Hasil seleksi

Individu	Fitness
P2_fold3	89,47368
C1_fold3	89,47368
P3_fold3	84,21053
C2_fold3	84,21053

Tabel 4.24 Hasil seleksi (lanjutan)

Individu	Fitness
P1_fold3	78,94737
P4_fold3	78,94737
C3_fold3	78,94737
…	…

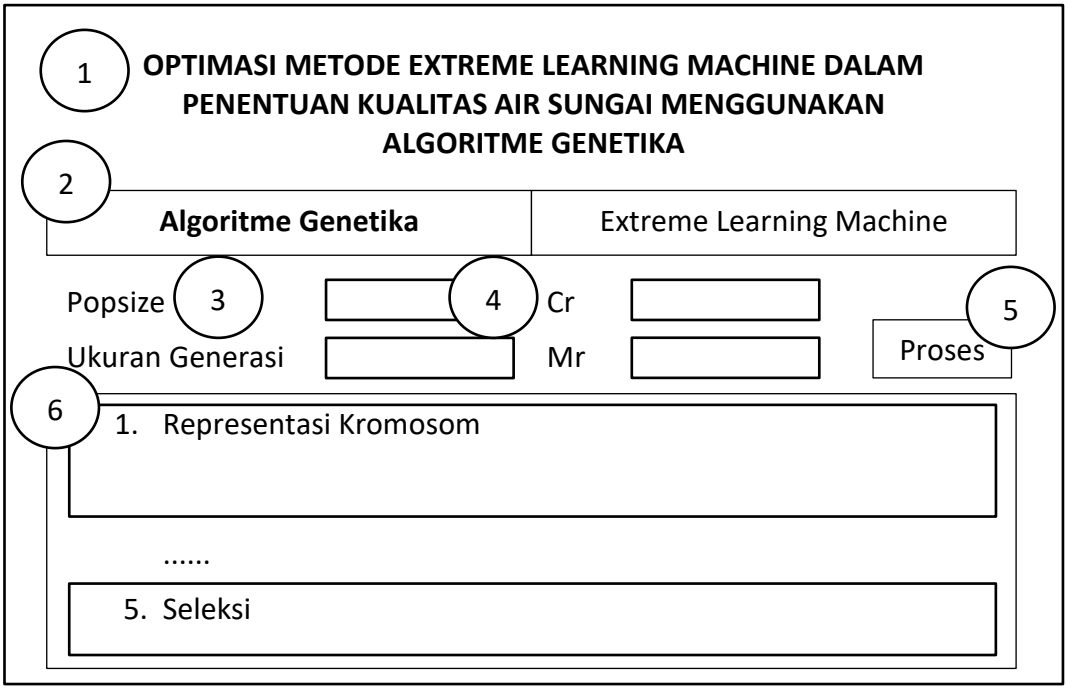
Pada proses seleksi akan diambil individu sejumlah *popsize* untuk dilakukan perhitungan kembali sebanyak jumlah generasi, hingga mendapatkan hasil yang paling optimal. Individu dengan nilai *fitness* tertinggi akan menjadi bobot awal/*input weight* yang terbaik.

4.5 Perancangan Antarmuka

Perancangan antarmuka dibuat untuk menggambarkan implementasi sistem yang akan dibuat pada penelitian ini. Implementasi terdiri dari 2 halaman utama yaitu Algoritme Genetika dan *Extreme Learning Machine*. Kemudian di dalam halaman *Extreme Learning Machine* terdapat 4 submenu, diantaranya adalah normalisasi data, data *fold*, *input weight* dan bias, evaluasi dan hasil prediksi.

4.5.1 Perancangan Antarmuka Algoritme Genetika

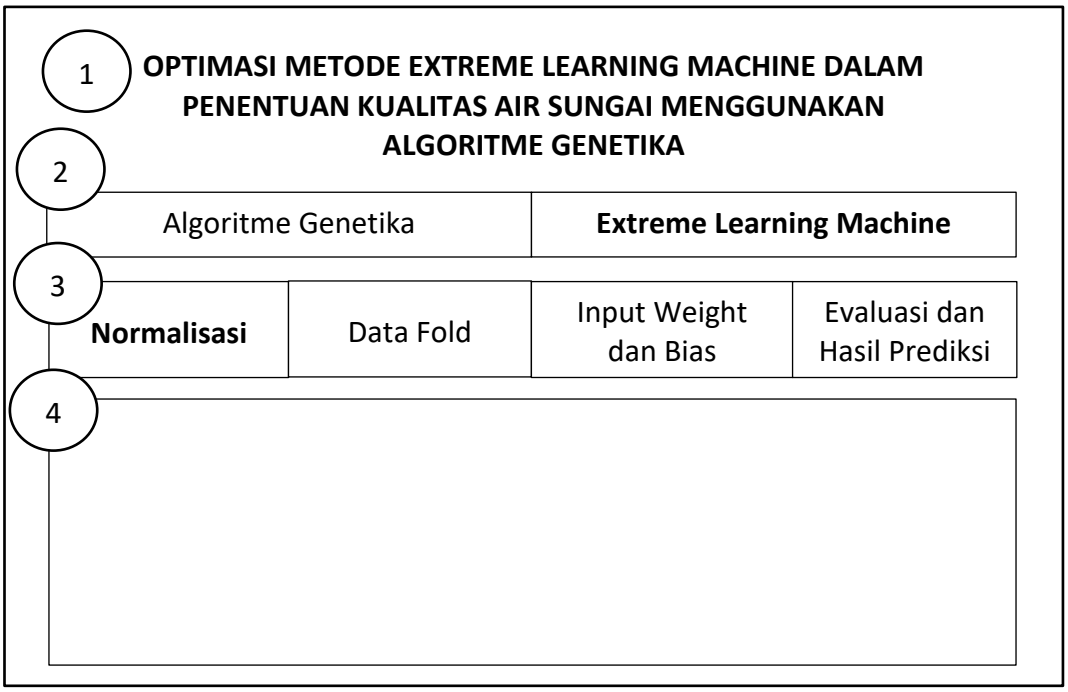
Antarmuka Algoritme Genetika digunakan untuk melakukan proses perhitungan optimasi bobot awal yang akan diterapkan pada ELM menggunakan metode Algoritme Genetika. Rancangan antarmuka halaman Algoritme Genetika dapat dilihat di Gambar 4.17.



Gambar 4.17 Perancangan antarmuka Algoritme Genetika

4.5.2 Perancangan Antarmuka Normalisasi

Antarmuka Normalisasi berfungsi untuk menampilkan data yang telah dinormalisasi. Rancangan antarmuka halaman Normalisasi dapat dilihat di Gambar 4.18.



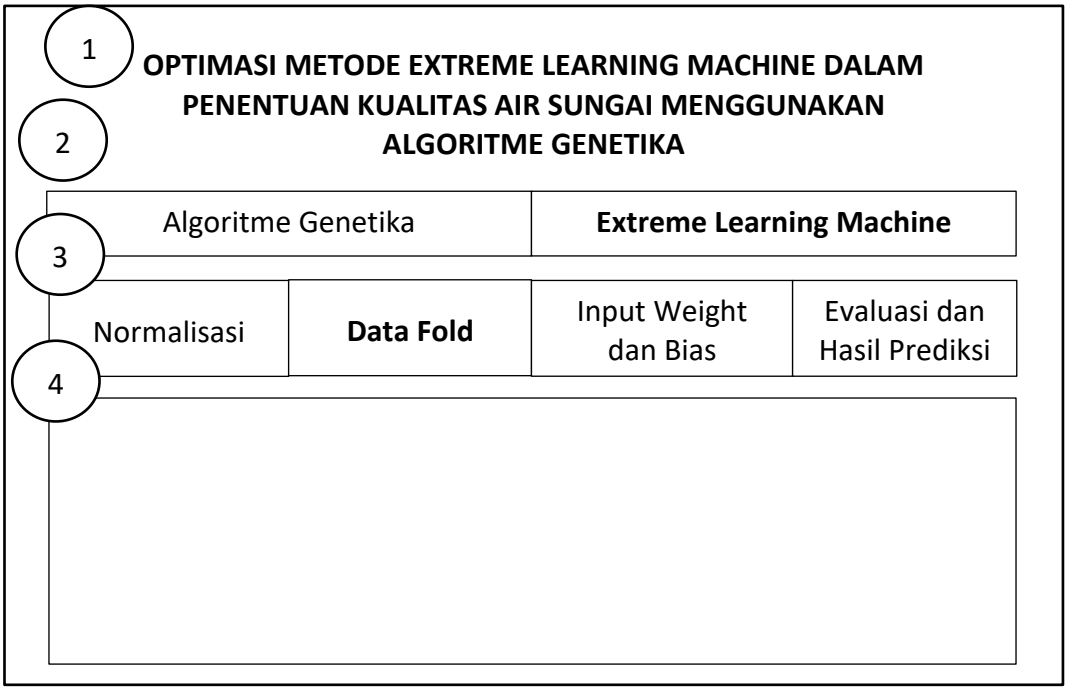
Gambar 4.18 Perancangan antarmuka normalisasi

Keterangan rancangan antarmuka halaman normalisasi data pada Gambar 4.14 adalah sebagai berikut:

1. *Header* program
2. *Tab* menu program, *tab* menu pada tulisan yang di-*bold* menandakan bahwa menu *Extreme Learning Machine* sedang aktif.
3. *Tab* menu program, *tab* menu pada tulisan yang di-*bold* menandakan bahwa menu Normalisasi sedang aktif.
4. Data *View* untuk menampilkan data yang sudah dinormalisasi

4.5.3 Perancangan Antarmuka Data *Fold*

Antarmuka Data *Fold* digunakan untuk menampilkan data yang telah terbagi ke dalam beberapa *fold,* yang akan digunakan untuk proses perhitungan. Rancangan antarmuka halaman Data *Fold* dapat dilihat di Gambar 4.19.



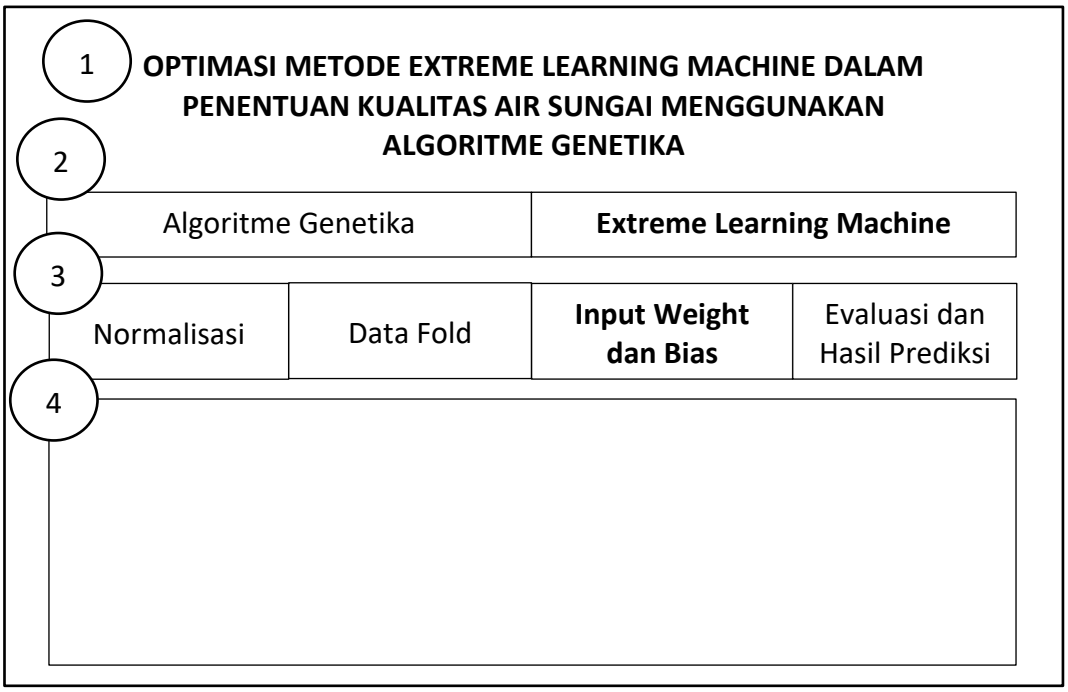
Gambar 4.19 Perancangan antarmuka data *fold*

Keterangan rancangan antarmuka halaman data *fold* data pada Gambar 4.19 adalah sebagai berikut:

1. *Header* program
2. *Tab* menu program, *tab* menu pada tulisan yang di-*bold* menandakan bahwa menu *Extreme Learning Machine* sedang aktif.
3. *Tab* menu program, *tab* menu pada tulisan yang di-*bold* menandakan bahwa menu Data *Fold* sedang aktif.
4. Data *View* untuk menampilkan data *fold*.

4.5.4 Perancangan Antarmuka *Input Weight* dan Bias

Antarmuka *Input Weight* dan Bias digunakan untuk menampilkan data *Input Weight* dan Bias yang akan digunakan untuk proses perhitungan. Rancangan antarmuka halaman *Input Weight* dan Bias dapat dilihat di Gambar 4.20,



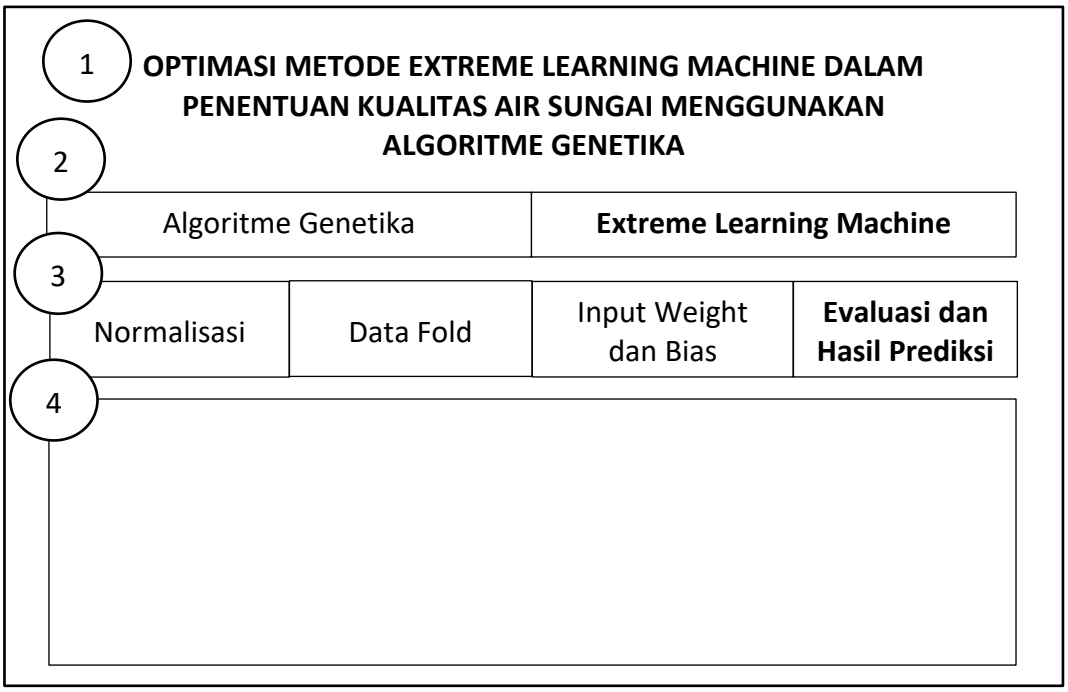
Gambar 4.20 Perancangan antarmuka *input weight* dan bias

Keterangan rancangan antarmuka halaman *input weight* dan bias data pada Gambar 4.20 adalah sebagai berikut:

1. *Header* program
2. *Tab* menu program, *tab* menu pada tulisan yang di-*bold* menandakan bahwa menu *Extreme Learning Machine* sedang aktif.
3. *Tab* menu program, *tab* menu pada tulisan yang di-*bold* menandakan bahwa menu *Input Weight* dan Bias sedang aktif.
4. Data *View* untuk menampilkan data *Input Weight* dan Bias.

4.5.5 Perancangan Antarmuka Evaluasi dan Hasil Prediksi

Antarmuka Evaluasi dan Hasil Prediksi digunakan untuk menampilkan perhitungan akurasi kinerja metode ELM-GA. Rancangan antarmuka halaman Evaluasi dan Hasil Prediksi dapat dilihat di Gambar 4.21.



Gambar 4.21 Perancangan antarmuka evaluasi dan hasil prediksi

Keterangan rancangan antarmuka halaman evaluasi dan hasil prediksi data pada Gambar 4.21 adalah sebagai berikut:

1. *Header* program
2. *Tab* menu program, *tab* menu pada tulisan yang di-*bold* menandakan bahwa menu *Extreme Learning Machine* sedang aktif.
3. *Tab* menu program, *tab* menu pada tulisan yang di-*bold* menandakan bahwa menu Evaluasi sedang aktif.
4. Data *View* untuk menampilkan hasil prediksi dan perhitungan akurasi predikis penentuan kualitas air sungai.

4.6 Skenario Pengujian Algoritme

Pengujian yang dilakukan pada penelitian ini bertujuan untuk mengetahui hasil optimasi dan pengaruh parameter yang digunakan pada Algoritme Genetika. Pengujian penelitian ini terdiri pengujian *popsize*, ukuran generasi dan cr, mr. Pengujian dilakukan dengan mengubah nilai parameter *popsize*, ukuran generasi dan cr, mr pada Algoritme Genetika dengan nilai tertentu. Selanjutnya, individu terbaik digunakan pada proses ELM. Evaluasi dilakukan dengan menghitung akurasi dari proses *testing* pada ELM

4.6.1 Pengujian *Popsize*

Pengujian *popsize* dilakukan dengan tujuan untuk mengetahui berapa *popsize* yang akan menghasilkan individu terbaik. Pengujian ini dilakukan dengan mengganti ukuran populasi setiap program dijalankan dengan jumlah generasi, cr dan mr tetap. Tabel 4.25 merupakan contoh skenario pengujian *popsize*.

Tabel 4.25 Skenario pengujian *popsize*

Popsize	Percobaan ke-							Rata-rata
	1	2	3	4	5	6	7	

4.6.2 Pengujian Ukuran Generasi

Pengujian ukuran atau jumlah generasi dilakukan dengan tujuan mengetahui berapa banyak generasi yang dibutuhkan untuk mencapai solusi yang optimal. Pengujian ini dilakukan dengan cara mengganti jumlah generasi yang akan di komputasi oleh program dengan *popsize*, cr dan mr tetap. Tabel 4.26 merupakan contoh skenario pengujian ukuran generasi.

Tabel 4.26 Skenario pengujian ukuran generasi

Ukuran Generasi	Percobaan ke-							Rata-rata
	1	2	3	4	5	6	7	

4.6.3 Pengujian Kombinasi Cr dan Mr

Pengujian cr dan mr dilakukan dengan tujuan mengetahui bagaimana kombinasi cr dan mr agar menghasilkan solusi yang optimal. Pengujian ini dilakukan dengan cara mengganti kombinasi cr dan mr pada setiap program dijalankan, dengan *popsize*, dan ukuran generasi tetap. Tabel 4.27 merupakan contoh skenario pengujian ukuran generasi.

Tabel 4.27 Skenario pengujian kombinasi cr dan mr

Kombinasi		Percobaan ke-							Rata-rata
Cr	Mr	1	2	3	4	5	6	7	