

**OPTIMASI VEKTOR BOBOT PADA *LEARNING VECTOR*  
QUANTIZATION DENGAN ALGORITMA GENETIKA UNTUK  
KLASIFIKASI TINGKAT RESIKO PENYAKIT STROKE**

**SKRIPSI**

Untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Komputer

Disusun oleh:  
MUHAMMAD RIFQI  
NIM: 115060807111072



PROGRAM STUDI TEKNIK INFORMATIKA  
JURUSAN TEKNIK INFORMATIKA  
FAKULTAS ILMU KOMPUTER  
UNIVERSITAS BRAWIJAYA  
MALANG  
2016

## PENGESAHAN

OPTIMASI VEKTOR BOBOT PADA *LEARNING VECTOR QUANTIZATION* DENGAN  
ALGORITMA GENETIKA UNTUK KLASIFIKASI TINGKAT RESIKO PENYAKIT STROKE

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Komputer

Disusun Oleh :  
Muhammad Rifqi  
NIM: 115060807111072

Skripsi ini telah diuji dan dinyatakan lulus pada  
15 Agustus 2016

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Dosen Pembimbing II

Imam Cholissodin, S.Si, M.Kom  
NIK: 201201 850719 1 001

Edy Santoso, S.Si, M.Kom  
NIP: 19740414 200312 1 004

Mengetahui  
Ketua Jurusan Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D  
NIP: 19710518 200312 1 001

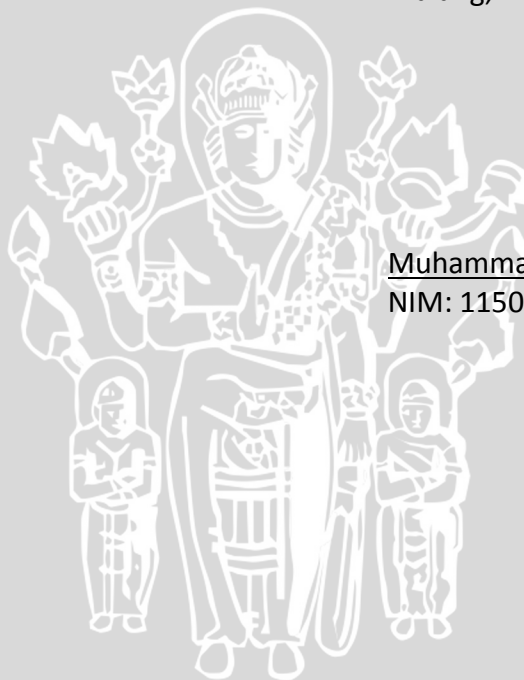
## PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 2 Agustus 2016

Muhammad Rifqi  
NIM: 115060807111072



## KATA PENGANTAR

Puji dan syukur Penulis panjatkan kehadirat Allah SWT, karena hanya dengan rahmat dan karunia-Nya Penulis dapat menyelesaikan skripsi dengan judul “Optimasi Vektor Bobot Pada *Learning Vector Quantization* Dengan Algoritma Genetika Untuk Klasifikasi Tingkat Resiko Penyakit Stroke” dengan baik. Melalui kesempatan ini, Penulis ingin menyampaikan rasa hormat dan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan bantuan dan dukungan selama pengerjaan skripsi, diantaranya:

1. Imam Cholissodin, S.Si, M.Kom., selaku dosen pembimbing I yang telah banyak memberikan ilmu, bimbingan, arahan, motivasi, serta meluangkan waktunya selama penyusunan skripsi ini.
2. Edy Santoso, S.Si, M.Kom., selaku dosen pembimbing II yang telah banyak memberikan ilmu, bimbingan, arahan, nasihat, serta meluangkan waktunya selama penyusunan skripsi ini.
3. Wayan Firdaus Mahmudy, S.Si, Ph.D., Ir. Heru Nurwarsito, M.Kom., Drs. Mardji, M.T., dan Edy Santoso, S.Si, M.Kom., selaku Ketua, Wakil Ketua I, Wakil Ketua II, dan Wakil Ketua III Fakultas Ilmu Komputer Universitas Brawijaya.
4. Agus Wahyu Widodo, S.T, M.Cs., selaku Ketua Program Studi Informatika/Illmu Komputer Universitas Brawijaya.
5. Kedua orang tua Salim B. dan Sultana yang telah memberi motivasi, kasih sayang, semangat serta dukungan moril dan materil.
6. Ketiga adik penulis Farid, Faiq dan Usama yang selalu memberi semangat serta membantu penulis.
7. Segenap bapak dan ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada Penulis selama menempuh pendidikan di Fakultas Ilmu Komputer Universitas Brawijaya.
8. Staf administrasi Program Studi Informatika/Illmu Komputer, Fakultas Ilmu Komputer.
9. Keluarga Besar Mahasiswa Informatika/Illmu Komputer khususnya angkatan 2011, dan teman-teman dari semester 1 sampai akhir perjuangan ini terima kasih atas segala bantuan dan dukungannya selama ini.
10. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya skripsi ini.

Semoga jasa dan amal baik mendapatkan balasan dari Allah SWT. Dengan segala kerendahan hati, penulis menyadari sepenuhnya bahwa skripsi ini masih jauh dari sempurna karena keterbatasan materi dan pengetahuan yang dimiliki

penulis. Akhirnya, semoga skripsi ini dapat bermanfaat dan berguna bagi pembaca terutama mahasiswa Fakultas Ilmu Komputer Universitas Brawijaya

Malang, 2 Agustus 2016

Penulis  
rifqibrd@gmail.com



## ABSTRAK

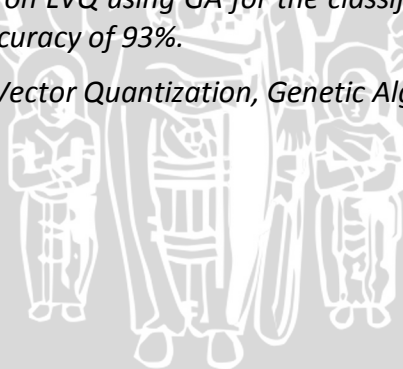
Otak manusia merupakan pusat pengaturan yang terdiri atas jutaan sel saraf atau neuron. Otak mengatur dan mengkoordinasi sebagian besar, gerakan, perilaku dan fungsi tubuh seperti detak jantung, tekanan darah, keseimbangan cairan tubuh dan suhu tubuh, maka jika terjadi gangguan akan memberikan dampak yang mematikan. Stroke atau gangguan peredaran darah otak merupakan sebuah penyakit yang menyebabkan gangguan yang terjadi pada otak. Stroke dapat ditandai pada saat hilangnya fungsi bagian tubuh tertentu (kelumpuhan), hal tersebut disebabkan terjadinya gangguan pada aliran darah otak. Stroke merupakan penyakit yang perlu ditangani secara cepat dan tepat. Penyakit stroke pada prinsipnya bisa dicegah, akan tetapi pada tahap deteksi dini masih sulit dan membutuhkan waktu yang lama. Oleh karena itu, dibutuhkan sistem cerdas yang mampu melakukan klasifikasi secara tepat serta mempercepat dan memudahkan pendeteksian awal penyakit stroke. *Learning Vector Quantization* (LVQ) merupakan metode klasifikasi yang memiliki struktur yang simpel dan pembelajaran yang cepat serta dapat diandalkan untuk klasifikasi tingkat resiko penyakit stroke, namun jika vektor bobot tidak tepat maka hasil klasifikasi yang didapatkan bisa jadi tidak optimal karena terjebak pada optimum lokal. Algoritma Genetika (GA) dapat diterapkan secara luas dengan metode kecerdasan buatan lain. Pada penelitian ini membahas vektor bobot pada LVQ yang digunakan untuk klasifikasi tingkat resiko penyakit stroke yang akan dioptimasi dengan menggunakan GA agar mendapatkan vektor bobot yang paling optimal dalam melakukan klasifikasi. Penelitian ini dilakukan menggunakan data pasien sebanyak 200 dengan 5 fitur dan 3 kategori atau kelas yang menunjukkan metode yang diusulkan mampu melakukan klasifikasi dengan hasil pengujian akurasi yang baik. Hasil pengujian akurasi pada sistem optimasi vektor bobot pada LVQ dengan menggunakan GA untuk klasifikasi tingkat resiko penyakit stroke didapatkan akurasi terbaik sebesar 93%.

Kata Kunci: Stroke, *Learning Vector Quantization*, Algoritma Genetika, Klasifikasi.

## ABSTRACT

*The human brain is the control center of millions of nerve cells or neurons. Brain control and coordinate most movement, behavior and bodily functions such as heart rate, blood pressure, body fluid balance and body temperature, so if there is disruption will have an impact deadly. Stroke or circulatory disorders of the brain is a disease that causes disruption of the brain. Stroke can be marked at the time of the loss of the function of certain body parts (paralysis), that caused disruptions in brain blood flow. Stroke is a disease that needs to be handle quickly and appropriately. Stroke in principle could be prevented, but at this stage of early detection is difficult and takes a long time. Therefore, it takes an intelligent system that is capable to accelerate classification precisely and facilitate early detection of stroke. Learning Vector Quantization (LVQ) is a classification method that has a simple structure and fast learning also reliable for the classification the risk level of stroke, but if the weight vector is not appropriate, the classification results may not be optimal because it stuck in a local optimum. Genetic algorithm (GA) can be applied widely to other methods of artificial intelligence. In this research study on LVQ weight vector used for the classification the risk level of stroke that will be optimized by using GA to obtain the most optimal weight vector within the classification. This study was conducted using the data of 200 patients with five features and three categories or classes that show the proposed method is able to classify the test results good accuracy. Results of testing the accuracy of the weight vector optimization system on LVQ using GA for the classification the risk level of stroke obtained the best accuracy of 93%.*

**Keyword:** Stroke, Learning Vector Quantization, Genetic Algorithm, Classification.



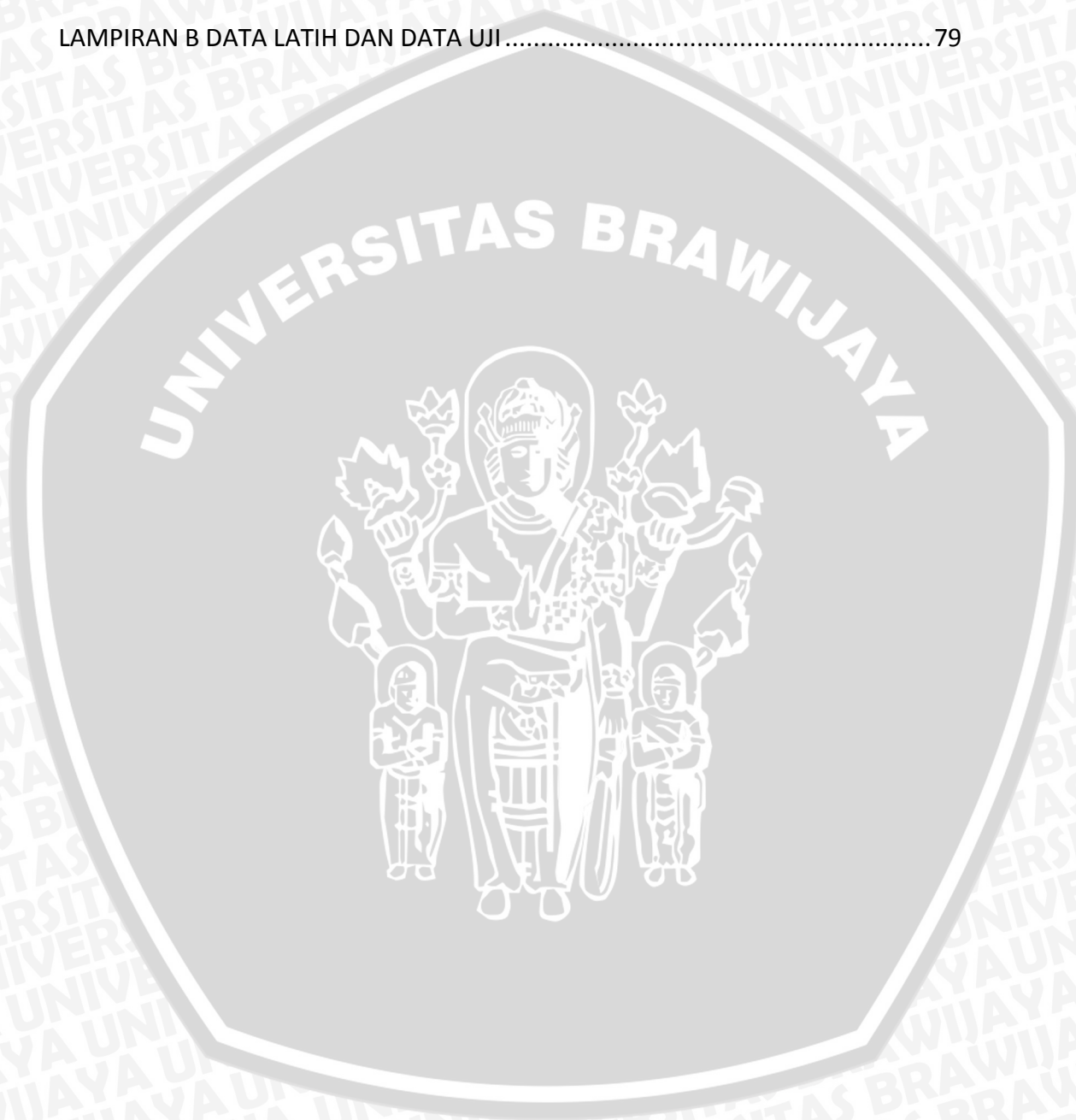
## DAFTAR ISI

|                                                           |      |
|-----------------------------------------------------------|------|
| PENGESAHAN .....                                          | ii   |
| PERNYATAAN ORISINALITAS.....                              | iii  |
| KATA PENGANTAR .....                                      | iv   |
| ABSTRAK .....                                             | vi   |
| ABSTRACT .....                                            | vii  |
| DAFTAR ISI .....                                          | viii |
| DAFTAR TABEL .....                                        | xii  |
| DAFTAR GAMBAR .....                                       | xiii |
| DAFTAR LAMPIRAN.....                                      | xv   |
| BAB 1 PENDAHULUAN.....                                    | 1    |
| 1.1 Latar Belakang.....                                   | 1    |
| 1.2 Rumusan Masalah .....                                 | 2    |
| 1.3 Tujuan.....                                           | 2    |
| 1.4 Manfaat .....                                         | 2    |
| 1.5 Batasan Masalah .....                                 | 3    |
| 1.6 Sistematika Pembahasan.....                           | 3    |
| BAB 2 LANDASAN KEPUSTAKAAN.....                           | 4    |
| 2.1 Kajian Pustaka .....                                  | 4    |
| 2.2 Klasifikasi .....                                     | 8    |
| 2.3 Stroke .....                                          | 8    |
| 2.3.1 Faktor Stroke .....                                 | 9    |
| 2.4 Jaringan Saraf Tiruan .....                           | 10   |
| 2.4.1 Konsep Dasar Jaringan Saraf Tiruan .....            | 11   |
| 2.4.2 Arsitektur Jaringan.....                            | 11   |
| 2.5 <i>Learning Vector Quantization</i> (LVQ) .....       | 13   |
| 2.5.1 Algoritma <i>Learning Vector Quantization</i> ..... | 14   |
| 2.6 Algoritma Genetika .....                              | 14   |
| 2.6.1 Struktur Umum Algoritma Genetika.....               | 15   |
| 2.6.2 <i>Real coded Genetic Algorithm</i> .....           | 16   |
| 2.6.3 Siklus RCGA.....                                    | 17   |

|                                                                  |           |
|------------------------------------------------------------------|-----------|
| 2.7 Optimasi Vektor Bobot LVQ dengan Algoritma Genetika .....    | 17        |
| <b>BAB 3 METODOLOGI .....</b>                                    | <b>19</b> |
| 3.1 Tahapan Penelitian.....                                      | 19        |
| 3.2 Studi Literatur .....                                        | 20        |
| 3.3 Pengumpulan dan Analisis Data .....                          | 20        |
| 3.4 Analisa dan Kebutuhan Sistem .....                           | 20        |
| 3.5 Perancangan Sistem .....                                     | 20        |
| 3.6 Implementasi dan Pembahasan.....                             | 21        |
| 3.7 Pengujian dan Analisis.....                                  | 21        |
| 3.8 Pengambilan Kesimpulan dan Saran .....                       | 22        |
| <b>BAB 4 PERANCANGAN.....</b>                                    | <b>23</b> |
| 4.1 Formulasi Perancangan .....                                  | 23        |
| 4.2 Perancangan Algoritma .....                                  | 23        |
| 4.2.1 Bangkitkan Populasi .....                                  | 24        |
| 4.2.2 Pelatihan LVQ .....                                        | 25        |
| 4.2.3 Hitung Jarak <i>Euclidean</i> .....                        | 27        |
| 4.2.4 <i>Update</i> Bobot .....                                  | 28        |
| 4.2.5 Algoritma Genetika Operator .....                          | 29        |
| 4.2.6 Hitung <i>Crossover</i> .....                              | 30        |
| 4.2.7 <i>Shuffle Parent Crossover</i> .....                      | 32        |
| 4.2.8 Hitung <i>Mutation</i> .....                               | 33        |
| 4.2.9 Evaluasi Akurasi .....                                     | 34        |
| 4.2.10 Seleksi.....                                              | 35        |
| 4.3 Perhitungan Manual.....                                      | 37        |
| 4.3.1 Bangkitkan Populasi .....                                  | 37        |
| 4.3.2 Pelatihan LVQ .....                                        | 37        |
| 4.3.3 Algoritma Genetika Operator .....                          | 40        |
| 4.3.4 Evaluasi Hasil dan Seleksi .....                           | 42        |
| 4.4 Perancangan Antarmuka .....                                  | 43        |
| 4.4.1 Rancangan Antarmuka Parameter dan Data .....               | 44        |
| 4.4.2 Rancangan Antarmuka Hasil .....                            | 44        |
| 4.4.3 Rancangan Antarmuka Akurasi dan Hasil <i>Sorting</i> ..... | 45        |

|                                                        |    |
|--------------------------------------------------------|----|
| 4.4.4 Rancangan Antarmuka Grafik Akurasi .....         | 46 |
| 4.5 Perancangan Uji Coba dan Evaluasi .....            | 46 |
| 4.5.1 Uji Coba Jumlah Iterasi Algoritma Genetika ..... | 47 |
| 4.5.2 Uji Coba Iterasi Laju Pembelajaran LVQ.....      | 47 |
| 4.5.3 Uji Coba Laju Pembelajaran .....                 | 48 |
| 4.5.4 Uji Coba Laju Pembelajaran Minimal.....          | 48 |
| 4.5.5 Uji Coba Laju Pengurang Pembelajaran.....        | 49 |
| 4.5.6 Uji Coba Nilai <i>Cr</i> dan <i>Mr</i> .....     | 49 |
| 4.5.7 Uji Coba Jumlah Individu ( <i>Popsiz</i> e)..... | 50 |
| BAB 5 IMPLEMENTASI .....                               | 51 |
| 5.1 Implementasi Algoritma .....                       | 51 |
| 5.1.1 Algoritma Membangkitkan Populasi .....           | 51 |
| 5.1.2 Algoritma Pelatihan LVQ.....                     | 52 |
| 5.1.3 Algoritma <i>Crossover</i> .....                 | 54 |
| 5.1.4 Algoritma <i>Mutation</i> .....                  | 54 |
| 5.1.5 Algoritma Uji Akurasi .....                      | 55 |
| 5.1.6 Algoritma <i>Sorting</i> .....                   | 56 |
| 5.1.7 Algoritma Menyimpan Individu.....                | 56 |
| 5.2 Implementasi Antarmuka .....                       | 57 |
| 5.2.1 Antarmuka Parameter dan Data .....               | 57 |
| 5.2.2 Antarmuka Hasil .....                            | 58 |
| 5.2.3 Antarmuka Akurasi dan <i>Sorting</i> .....       | 59 |
| 5.2.4 Antarmuka Grafik Akurasi .....                   | 59 |
| BAB 6 PENGUJIAN DAN ANALISIS .....                     | 60 |
| 6.1 Pengujian Jumlah Iterasi Algoritma Genetika .....  | 60 |
| 6.2 Pengujian Iterasi Laju Pembelajaran LVQ.....       | 61 |
| 6.3 Pengujian Laju Pembelajaran .....                  | 63 |
| 6.4 Pengujian Laju Pembelajaran Minimal.....           | 64 |
| 6.5 Pengujian Laju Pengurang Pembelajaran.....         | 65 |
| 6.6 Pengujian Nilai <i>Cr</i> dan <i>Mr</i> .....      | 67 |
| 6.7 Pengujian Jumlah Individu ( <i>Popsiz</i> e).....  | 68 |
| BAB 7 PENUTUP .....                                    | 70 |

|                                          |    |
|------------------------------------------|----|
| 7.1 Kesimpulan .....                     | 70 |
| 7.2 Saran .....                          | 70 |
| DAFTAR PUSTAKA .....                     | 71 |
| LAMPIRAN A DATA .....                    | 73 |
| LAMPIRAN B DATA LATIH DAN DATA UJI ..... | 79 |



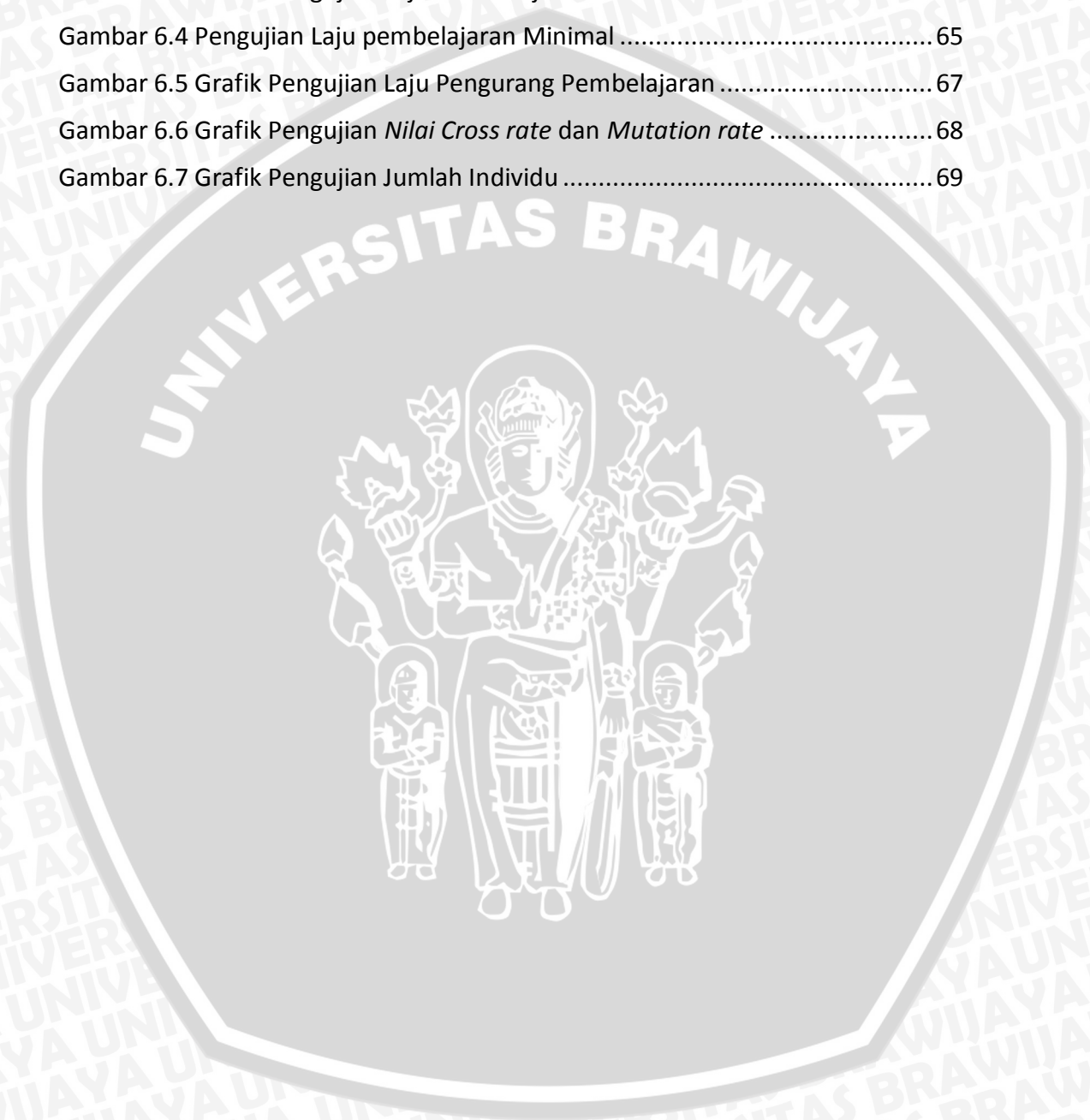
## DAFTAR TABEL

|                                                                   |    |
|-------------------------------------------------------------------|----|
| Tabel 2.1 Perbandingan Penelitian Sebelumnya .....                | 4  |
| Tabel 2.2 Umur .....                                              | 9  |
| Tabel 2.3 Total Kolesterol.....                                   | 9  |
| Tabel 2.4 HDL.....                                                | 10 |
| Tabel 2.5 LDL .....                                               | 10 |
| Tabel 2.6 Trigliserida .....                                      | 10 |
| Tabel 4.1 Data Individu .....                                     | 37 |
| Tabel 4.2 Data Latih .....                                        | 37 |
| Tabel 4.3 Bobot Baru Individu Pertama .....                       | 39 |
| Tabel 4.4 Bobot Akhir Individu Pertama .....                      | 39 |
| Tabel 4.5 Data Individu <i>Update</i> .....                       | 39 |
| Tabel 4.6 <i>Child</i> Pertama .....                              | 41 |
| Tabel 4.7 <i>Child</i> Kedua.....                                 | 41 |
| Tabel 4.8 <i>Child Mutation</i> .....                             | 42 |
| Tabel 4.9 Rancangan Uji Coba Jumlah Iterasi .....                 | 47 |
| Tabel 4.10 Rancangan Uji Coba Laju Pembelajaran LVQ .....         | 47 |
| Tabel 4.11 Rancangan Uji Coba Laju Pembelajaran .....             | 48 |
| Tabel 4.12 Rancangan Uji Coba Laju Pembelajaran Minimal.....      | 48 |
| Tabel 4.13 Rancangan Uji Coba Laju Pengurang Pembelajaran.....    | 49 |
| Tabel 4.14 Rancangan Uji Coba Nilai <i>Cr</i> Dan <i>Mr</i> ..... | 49 |
| Tabel 4.15 Rancangan Uji Coba Jumlah Individu.....                | 50 |
| Tabel 6.1 Hasil Pengujian Jumlah Iterasi.....                     | 60 |
| Tabel 6.2 Hasil Pengujian Iterasi Laju Pembelajaran .....         | 62 |
| Tabel 6.3 Hasil Laju Pembelajaran.....                            | 63 |
| Tabel 6.4 Hasil Laju Pembelajaran Minimal .....                   | 65 |
| Tabel 6.5 Hasil Pengujian Laju Pengurang Pembelajaran .....       | 66 |
| Tabel 6.6 Pengujian Nilai <i>Cr</i> dan <i>Mr</i> .....           | 67 |
| Tabel 6.7 Hasil Pengujian Jumlah Individu .....                   | 69 |

## DAFTAR GAMBAR

|                                                                 |    |
|-----------------------------------------------------------------|----|
| Gambar 2.1 Stroke .....                                         | 8  |
| Gambar 2.2 Jaringan Lapis Tunggal .....                         | 12 |
| Gambar 2.3 Jaringan Saraf Tiruan <i>Feedforward</i> .....       | 13 |
| Gambar 3.1 Tahapan Penelitian .....                             | 19 |
| Gambar 4.1 Diagram Alir Sistem .....                            | 24 |
| Gambar 4.2 Bangkitkan Populasi .....                            | 25 |
| Gambar 4.3 Pelatihan LVQ .....                                  | 26 |
| Gambar 4.4 Hitung Jarak <i>Euclidean</i> .....                  | 28 |
| Gambar 4.5 Update Bobot .....                                   | 29 |
| Gambar 4.6 Algoritma Genetika Operator .....                    | 30 |
| Gambar 4.7 Hitung <i>Crossover</i> .....                        | 31 |
| Gambar 4.8 <i>Shuffle Parent Crossover</i> .....                | 32 |
| Gambar 4.9 Hitung <i>Mutation</i> .....                         | 33 |
| Gambar 4.10 Evaluasi Akurasi .....                              | 34 |
| Gambar 4.11 Seleksi .....                                       | 36 |
| Gambar 4.12 Halaman Antarmuka Parameter dan Data .....          | 44 |
| Gambar 4.13 Halaman Antarmuka Hasil .....                       | 45 |
| Gambar 4.14 Halaman Antarmuka Akurasi .....                     | 45 |
| Gambar 4.15 Halaman Antarmuka Grafik .....                      | 46 |
| Gambar 5.1 Implementasi Membangkitkan Populasi .....            | 52 |
| Gambar 5.2 Implementasi Menghitung Jarak <i>Euclidean</i> ..... | 53 |
| Gambar 5.3 Implementasi <i>Update</i> Vektor Bobot .....        | 54 |
| Gambar 5.4 Implementasi <i>Crossover</i> .....                  | 54 |
| Gambar 5.5 Implementasi <i>Mutation</i> .....                   | 55 |
| Gambar 5.6 Implementasi Uji Akurasi .....                       | 55 |
| Gambar 5.7 Implementasi <i>Sorting</i> .....                    | 56 |
| Gambar 5.8 Implementasi Menyimpan Individu .....                | 57 |
| Gambar 5.9 Antarmuka Parameter dan Data .....                   | 58 |
| Gambar 5.10 Antarmuka Hasil .....                               | 58 |
| Gambar 5.11 Antarmuka Akurasi dan <i>Sorting</i> .....          | 59 |

|                                                                                    |    |
|------------------------------------------------------------------------------------|----|
| Gambar 5.12 Antarmuka Grafik Akurasi .....                                         | 59 |
| Gambar 6.1 Grafik Pengujian Iterasi Algoritma Genetika.....                        | 61 |
| Gambar 6.2 Grafik Pengujian Jumlah Iterasi Laju Pembelajaran LVQ.....              | 62 |
| Gambar 6.3 Grafik Pengujian Laju Pembelajaran.....                                 | 64 |
| Gambar 6.4 Pengujian Laju pembelajaran Minimal .....                               | 65 |
| Gambar 6.5 Grafik Pengujian Laju Pengurang Pembelajaran .....                      | 67 |
| Gambar 6.6 Grafik Pengujian <i>Nilai Cross rate</i> dan <i>Mutation rate</i> ..... | 68 |
| Gambar 6.7 Grafik Pengujian Jumlah Individu .....                                  | 69 |



## DAFTAR LAMPIRAN

LAMPIRAN A DATA.....Error! Bookmark not defined.

LAMPIRAN B DATA LATIH DAN DATA UJI .....Error! Bookmark not defined.



## BAB 1 PENDAHULUAN

### 1.1 Latar Belakang

Manusia memiliki sel saraf hingga 100 miliar dan memiliki triliunan sambungan saraf pada otaknya. Meskipun otak hanya memiliki 2 persen dari total berat tubuh yang dimiliki manusia, akan tetapi 70 persen kebutuhan oksigen dan juga kebutuhan gizi lainnya sangat diperlukan tubuh manusia agar otak dapat berkerja dengan baik. Tidak seperti otot, otak tidak mampu menyimpan cadangan zat gizi, sehingga diperlukan aliran darah agar otak bisa bekerja secara maksimal (Holistic Health Solution, 2011).

Stroke merupakan sebuah penyakit yang menyebabkan gangguan yang terjadi pada otak. Stroke dapat di tandai pada saat hilangnya fungsi bagian tubuh tertentu (kelumpuhan), hal tersebut disebabkan terjadinya gangguan pada aliran darah otak yang berguna untuk mengelola bagian tubuh yang kehilangan fungsinya. Menurut Statistik, penyakit stroke dalam satu tahun dapat mencapai 0.2% dari jumlah penduduk dan sekitar 1% lebih bisa dijumpai pada orang yang telah berusia lebih dari 65 tahun (Cahyono, 2008). Penyakit stroke umumnya menjadi faktor kematian peringkat ke-3 pada usia lanjut, setelah faktor tertinggi yaitu penyakit jantung dan kanker. Penyakit stroke banyak menyebabkan kelumpuhan pada usia lebih dari 45 tahun (Anies, 2006). Menurut WHO, di Indonesia stroke menduduki peringkat ke-97 dunia dengan jumlah penderita terbanyak, dengan jumlah kematian yang mencapai sekitar 138.268 orang atau 9,70% dari total kematian tahun 2011 (Anon., n.d.).

Berdasarkan hasil laporan yang telah dilakukan oleh Menteri Kesehatan RI, dr. Nafsiah Mboi, Sp.A, MPH, menurutnya ada beberapa hal utama yang menjadi permasalahan pada kesehatan otak dan juga saraf, salah satunya yaitu: Penyakit otak dan saraf menimbulkan angka kecacatan dan angka kematian yang tinggi, yang kedua peningkatan pada usia harapan hidup (UHH) berdampak pada organ tubuh yang mengalami proses penuan termasuk otak dan jaringan saraf (Anon., 2014). Beberapa upaya yang telah dilakukan oleh pemerintah adalah membangun rumah sakit pusat otak nasional. Namun, pada tahun 2014 baru diresmikan satu rumah sakit yang berfokus pada penyakit stroke. Selain itu ketika pasien memeriksakan diri ke lab tidak dapat diketahui penyakit yang diderita melainkan masih perlu dilakukan pemeriksaan hasil lab ke dokter, serta tim medis juga tidak bisa menentukan selain dokter. Dengan demikian, dibutuhkan sebuah sistem cerdas dalam yang mampu digunakan dalam identifikasi secara dini untuk tingkat resiko seseorang menderita stroke khususnya untuk tim medis dan pasien.

Pada penelitian yang telah dilakukan oleh Maharani Dessy Wuryandari dan Irawan Afrianto (2012) dengan membandingkan metode *backpropagation* dan metode *Learning Vector Quantization* (LVQ) pada pengenalan wajah, tingkat kesamaan hasil pengenalan bergantung pada kondisi parameter yang ditentukan saat proses pembelajarannya. Metode LVQ memiliki hasil akurasi yang lebih baik jika dibandingkan dengan *backpropagation*, hasil akurasi yang dimiliki LVQ

mencapai 37,63% dengan waktu pengenalan 32 *milisecond*, jika dibandingkan dengan dengan *backpropagation* akurasi hanya mencapai 37,33% akan tetapi waktu pengenalan yang dibutuhkan 130 *milisecond* (Wuryandari & Afianto, 2012). Penelitian yang dilakukan Ning Chen, et al. (2010), menggabungkan algoritma genetika dan LVQ. LVQ digunakan karena metode tersebut sangat cocok dalam klasifikasi untuk menyelesaikan permasalahan dalam memprediksi kebangkrutan. Algoritma genetika digunakan untuk mengoptimalkan konfigurasi dari LVQ dan mendapatkan hasil yang terbaik (Chen, et al., 2010). Untuk meningkatkan akurasi yang dihasilkan LVQ maka diperlukan optimasi menggunakan algoritma genetika.

Berdasarkan fakta dan permasalahan yang ada maka pada skripsi ini digunakan judul “Optimasi Vektor Bobot pada *Learning Vector Quantization* dengan Algoritma Genetika untuk klasifikasi tingkat resiko penyakit stroke”. Metode tersebut digunakan karena memiliki hasil yang terbaik dibandingkan dengan metode lainnya. Diharapkan dengan metode tersebut bisa mendapatkan hasil akurasi yang lebih baik.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan sebelumnya maka didapatkan persamaan masalah sebagai berikut:

1. Bagaimana menerapkan Algoritma Genetika untuk optimasi vektor bobot pada LVQ dalam mengetahui tingkat resiko penyakit stroke.
2. Bagaimana tingkat akurasi yang dihasilkan dari Algoritma Genetika untuk optimasi vektor bobot pada LVQ dalam mengetahui tingkat resiko penyakit stroke.

## 1.3 Tujuan

Tujuan yang ingin dicapai dari penelitian ini sebagai berikut :

1. Menerapkan Algoritma Genetika untuk optimasi vektor bobot pada LVQ dalam mengetahui tingkat resiko penyakit stroke.
2. Mengetahui tingkat akurasi yang dihasilkan dari Algoritma Genetika untuk optimasi vektor bobot pada LVQ dalam mengetahui tingkat resiko penyakit stroke.

## 1.4 Manfaat

Manfaat yang didapatkan dari penelitian ini sebagai berikut :

1. Membantu dalam mengklasifikasikan tingkat resiko penyakit stroke secara efisien dan efektif.
2. Membantu memberikan informasi pada pasien mengenai tingkat resiko penyakit stroke secara lebih dini.
3. Meningkatkan kesadaran masyarakat terhadap tingkat resiko penyakit stroke.
4. Mempermudah pasien atau tim medis dalam menentukan tingkat penyakit stroke yang diderita.

## 1.5 Batasan Masalah

Penelitian ini dibatasi oleh hal-hal sebagai berikut:

1. LVQ hanya digunakan untuk klasifikasi serta membarui vektor bobot sehingga didapat akurasi yang lebih baik.
2. Nilai fitness pada Algoritma Genetika didapatkan dari nilai akurasi yang dihasilkan oleh klasifikasi LVQ.
3. Klasifikasi terbatas pada 3 kelas status resiko stroke yaitu normal, rentan, dan mengkhawatirkan.
4. Data yang digunakan adalah data rekam medik yang diperoleh dari Laboratorium klinik sejahtera sebanyak 200 buah.
5. Parameter LVQ yang dioptimasi adalah vektor bobot yang digunakan untuk klasifikasi pada LVQ.

## 1.6 Sistematika Pembahasan

Untuk mencapai tujuan yang diharapkan, maka sistematika penulisan yang disusun dalam penelitian ini adalah sebagai berikut:

### BAB I PENDAHULUAN

Bab ini memuat tentang latar belakang, persamaan masalah, batasan masalah, tujuan, manfaat, dan sistematika penulisan dari penelitian.

### BAB II LANDASAN KEPUSTAKAAN

Bab ini membahas kajian dari penelitian yang telah dilakukan sebelumnya serta membahas teori-teori yang mendukung penyelesaian masalah dalam penelitian ini.

### BAB III METODOLOGI

Bab ini membahas tentang metode yang digunakan dalam penulisan yang terdiri dari studi literatur, perancangan perangkat lunak, implementasi perangkat lunak, pengujian dan analisis.

### BAB IV PERANCANGAN

Bab ini akan membahas tentang hasil perancangan dari analisis kebutuhan dan implementasi metode Algoritma Genetika dalam mengoptimasi LVQ untuk sistem klasifikasi tingkat resiko penyakit stroke

### BAB V IMPLEMENTASI

Bab ini memuat tentang implementasi algoritma dan interface sistem yang dikembangkan.

### BAB VI PENGUJIAN DAN ANALISIS

Bab ini diuraikan tentang pengujian sistem yang telah dikembangkan dan analisis hasil penelitian.

### BAB VII PENUTUP

Bab ini diuraikan tentang kesimpulan yang didapatkan dari pengembangan dan pengujian sistem dalam penelitian ini serta saran-saran untuk pengembangan lebih lanjut.

## BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini dipaparkan tentang kajian pustaka dan dasar teori yang mendukung pengembangan metode *Learning Vector Quantization* yang dioptimasi menggunakan Algoritma Genetika untuk klasifikasi tingkat resiko penyakit stroke.

### 2.1 Kajian Pustaka

Kajian pustaka dalam bagian ini akan menjelaskan tentang penelitian sebelumnya yang akan dianalisa untuk keperluan acuan pustaka. Tiap pustaka dianalisa untuk memperoleh bagian terpenting yaitu masukkan yang diberikan, proses dari penyelesaian masalah, dan *output* yang dijelaskan pada Tabel 2.1.

**Tabel 2.1 Perbandingan Penelitian Sebelumnya**

| Judul                                                                                                       | Objek                                                                                    | Metode                                                                                                                                                       | Output                                                                                                                                                |
|-------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                                             | Input dan Parameter                                                                      | Proses                                                                                                                                                       | Hasil penelitian                                                                                                                                      |
| Hybrid Genetic Algorithm and Learning Vector Quantization Modeling for Cost-Sensitive Bankruptcy Prediction | Prediksi Kebangkrutan                                                                    | LVQ dioptimasi dengan Algoritma Genetika                                                                                                                     | Hasil Klasifikasi yang dievaluasi dari error rate yang dihasilkan.                                                                                    |
|                                                                                                             | -Nilai Individual<br>-Vektor Prototype                                                   | - Inisialisasi Populasi<br>- Klasifikasi dengan LVQ<br>- Evaluasi Fitness<br>- Genetic Operator                                                              | Klasifikasi LVQ dengan dioptimasi menggunakan Algoritma Genetika menghasilkan performa yang terbaik dan efektif untuk menghindari nilai lokal minimum |
| Classifier Ensemble Optimization for Gender Classification using Genetic Algorithm                          | Klasifikasi Jenis Kelamin                                                                | Kombinasi Weight Majority Voting dan GA                                                                                                                      | Akurasi Klasifikasi individu, kombinasi klasifikasi dan akurasi optimasi                                                                              |
|                                                                                                             | - Nilai Data Set<br>- Nilai Klasifikasi Individual<br>- Kombinasi                        | - Training Data Set<br>- Inisialisasi Populasi<br>- Kombinasi<br>- Evaluasi Fintess                                                                          | Kombinasi Klasifikasi memiliki akurasi yang lebih tinggi dibanding individual dan meningkat saat di optimasi.                                         |
| Research on color recognition of urine test paper based on Learning vector quantization (LVQ)               | Pengenalan warna dari test urin                                                          | Learning Vector Quantization                                                                                                                                 | Prediksi hasil untuk urobilinogen test                                                                                                                |
|                                                                                                             | - nilai input sample vector<br>- jumlah saraf<br>- variasi input samples<br>-data sample | - Inisialisasi partikel<br>- simulasi dan evaluasi urin dengan nilai RGB<br>- Dihasilkan 5 level output<br>- Training data sample ke LVQ<br>- Prediksi Hasil | Hasil akurasi prediksi untuk Urobilinogen test hanya mencapai 57,1% yang mana sangat perlu dilakukan normalisasi data menggunakan LVQ                 |
| Perbandingan Metode jaringan Syaraf Tiruan                                                                  | Pengenalan wajah untuk berbagai                                                          | Perbandingan antara backpropagation dan Learning Vector Quantization                                                                                         | Akurasi wajah yang dikenali dan benar.                                                                                                                |

|                                                                        |                                                                        |                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                   |
|------------------------------------------------------------------------|------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Backpropagation dan Learning Vector Quantization pada Pengenalan Wajah | <p>macam kebutuhan seperti absensi</p> <p>- Gambar Wajah seseorang</p> | <p>- Input Gambar</p> <p>- Scaling &amp; Grayscale Tiap Gambar</p> <p>- Proses edgedetection</p> <p>- Simpan ke database</p> <p>- Proses pembelajaran backpropagation</p> <p>- Proses pembelajaran LVQ</p> <p>- Nilai bobot hasil pembelajaran disimpan ke database.</p> <p>-Pengenalan nilai bobot</p> <p>-Hasilkan pengenalan yang cocok</p> | <p>Akurasi yang dihasilkan backpropagation mencapai 37,33% yang berhasil dikenali, sedangkan LVQ memiliki akurasi 37,63% untuk hasil yg berhasil dikenali. Rata-rata waktu pengelanaan 130 detik untuk backpropagation dan 32 detik untuk LVQ</p> |
|------------------------------------------------------------------------|------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Beberapa hasil penelitian yang digunakan sebagai kajian pustaka yang dapat mendukung penelitian ini antara lain : pada penelitian yang dilakukan oleh Ning Chen (2010) yang berjudul *"Hybrid Genetic Algorithm and Learning Vector Quantization Modeling for Cost-Sensitive Bankruptcy Prediction"* mengimplementasikan algoritma Hybrid GA-LVQ sebagai solusi dari masalah prediksi kebangkrutan dan untuk meminimalisir biaya terjadinya kesalahan klasifikasi. LVQ digunakan sebagai model prediksi dan GA digunakan untuk mencari solusi terbaik. Tahapan pertama yang dilakukan adalah dengan membangkitkan sejumlah individu yang berisi vektor prototipe (vektor bobot). Kemudian setiap individu di proses ke dalam LVQ, dan nilai *fitness* dihitung dengan rumus kriteria EMC menggunakan data latih. Kemudian dilakukan proses genetika operator yaitu: seleksi, kawin silang dan mutasi untuk memberikan solusi baru yang mampu menghasilkan populasi berikutnya. Solusi yang paling optimal menjadi keluaran ketika setiap kriteria telah terpenuhi. Kemudian solusi optimal digunakan untuk memprediksi kebangkrutan dengan menggunakan data latih (Chen, et al., 2010).

Pada penelitian lain yang dilakukan oleh Yasir Mehmood (2010) dengan judul *"Classifier Ensemble Optimization for Gender Classification using Genetic Algorithm"*. Penelitian ini menyelesaikan masalah klasifikasi jenis kelamin dengan menggunakan algoritma genetika. Ada tiga tahap yang dilakukan. Pada klasifikasi, 5 metode klasifikasi digunakan untuk melatih dan mengevaluasi dataset gambar wajah. Metode klasifikasi yang digunakan adalah *Back Propagation Neural Network Classifier* (BPNN), *k Nearest Neighbour Classifier*(kNNC), *Linear Discriminate Analysis Classifier* (LDAC), *Fisher Linear Discriminate Classifier* (FLDC), dan *Support Vector Machine Classifier* (SVMC). Kesalahan pada tiap pengklasifikasi menggunakan proses ensemble dengan aturan bobot suara terbanyak. Dimana GA digunakan untuk mengoptimalkan Bobot tersebut.

Tahap berikutnya adalah bobot suara terbanyak. Pada seluruh metode klasifikasi secara umum dilatih dengan data latih yang berbeda dan bobot

digunakan untuk mengarahkan pada rasio pengelanaan yang lebih tinggi. Biasanya pengklasifikasi dasar yang bukan dari akurasi yang sama dan keputusan dari pengklasifikasi dengan akurasi yang lebih rendah harus mempertimbangkan dengan keputusan pengklasifikasi dengan akurasi yang lebih tinggi. Untuk mencapai tujuan ini keputusan masing-masing pengklasifikasi ditimbang sedemikian rupa bahwa kontribusi dari setiap pengklasifikasi pada keputusan akhir sebanding dengan akurasi. Tahap akhir yaitu optimasi dengan menggunakan GA, bobot dari bobot suara terbanyak dioptimasi melalui GA agar sistem dapat memprediksi jenis kelamin dengan benar. GA digunakan untuk masalah optimasi karena tidak memerlukan pengetahuan seara spesifik dari domain masalah, GA mencari kualitas dari beberapa wilayah secara bersamaan. GA memulai dengan memilih sollusi secara acak sejumlah  $N$  dari domain masalah. Bobot dari semua pengklasifikasi dinormalisasikan antara  $[0-1]$  dan panjang dari kromosom  $m$  sama dengan jumlah dari pengklasifikasi  $I$ . Dengan cara ini inisialisasi secara acak dari kromosom dievaluasi menurut fungsi fitness dimana tingkat kesalahan dari pengklasifikasi. Seleksi turnamen digunakan untuk membangkitkan populasi baru. Pada tiap generasi baru aturan elitism di gunakan dimana sejumlah  $e$  dari kromosom terbaik digunakan untuk generasi berikutnya tanpa kompetisi, ini akan membantu mempertahankan tingkat kesalahan yang dicapai melalui proses evolusi. Kemudian operasi *crossover*  $cr$  dan *mutation*  $mu$  di gunakan untuk evolusi bobot dari pengklasifikasi. GA akan dihentikan jika generasi mecapa  $G_{max}$  atau populasi konvergen hingga solusi yang memuaskan (Mehmood, et al., 2010).

Penelitian Berikutnya dilakukan oleh Wang Chunhong (2012) yang berjudul "*Research on color recognition of urine test paper based on Learning vector quantization (LVQ)*". Penelitian ini mengangkat permasalahan pengenalan warna dari kertas tes urin dengan menggunakan LVQ. Tahapan yang dilakukan adalah pembentukan jaringan saraf LVQ. Perumusan fungsi LVQ dari desain fungsi jaringan LVQ `newlvq()` adalah `net=newlvq (minmax(P), n, [a, b, c, d, e])`, disini  $P$  adalah vektor sampel masukan,  $n$  adalah jumlah saraf pada lapisan kompetitif,  $a, b, c, d$ , masing-masing mewakili proporsi sebagai sampel masukan. Jumlah neuron dilapisan *input* relatif terhadap jumlah indeks evaluasi. Mensimulasikan dan mengevaluasi urin analisis biokimia dengan menggunakan nilai warna RGB(red, green, blue). Dengan cara ini, jumlah neuron pada lapisan input 3 menurut data yang tersedia, masing-masin mode memiliki lima sampel antara sampel pelatihan 25. Oleh karena itu, nilai  $a, b, c, d, e$  semua 0,2. Laju pembelajaran mendapat nilai *default* 0,01. Langkah pelatihan(*epoch*) diatur 100. Dengan fungsi `ind2vec`, katerogir vektor dapat diubah menjadi PNN yang dapat digunakan atau mengubah hasil klasifikasi menjadi vektor kategori mudah diidentifikasi oleh fungsi `ven2ind`. Keluaran dari jaringan memiliki 5 tingkat yaitu --, --+, +, ++, ++++. Sehingga jumlah neuron *output* 5. Tahap Berikutnya adalah pelatihan dan prediksi dari jaringan saraf LVQ. Sampel pelatihan adalah nilai warna RGB setelah interasi dipstick urin dan cairan ambang batas standar. Ada 5 ambang setiap tes, dan masing-masing tes memiliki 5 eksperimen kelompok. Sehingga setiap tes berisi 25 sampel pelatihan sekaligus. Catat nilai warna RGB disptick urin setelah reaksi cairan urin diuji. Masukan data sampel validasi yang telah dinormalisasi ke jaringan saraf LVQ

yang telah menyelesaikan pelatihan kemudian prediksi:  $Y = \text{sim}(\text{net}, X)$ . disini net adalah saraf LVQ yang telah dilatih, X adalah vektor sampel validasi, Y adalah hasil prediksi. Hasil prediksi kedua adalah hasil uji ketika pelatihan dan sampel validasi tidak di normalisasi. Hasil simulasi prediksi dengan normalisasi data benar-benar konsisten dengan nilai output yang diharapkan, yang menunjukkan bahwa jaringan memiliki akurasi prediksi yang relatif baik. Namun tingkat yang benar hasil simulasi prediksi dengan data mentah hanya 57,1% yang menunjukkan bahwa itu benar-benar memerlukan normalisasi data dengan jaringan LVQ dalam pengolahan klasifikasi (Chunhong, et al., 2012).

Penelitian berikutnya dengan judul “Perbandingan Metode Jaringan Syaraf Tiruan *Backpropagation* dan *Learning Vector Quantization* pada Pengenalan Wajah” yang dilakukan oleh Maharani dan Irawan (2012). Penelitian ini mencoba membandingkan metode jaringan saraf tiruan *Backpropagation* dan LVQ untuk pengenalan wajah. Tahapan proses yang dilakukan adalah : masukan berupa gambar sebanyak 5 buah melalui proses *scaling* untuk diubah ukurannya sehingga semua ukuran sama. Setelah melalui proses *scaling*, gambar masukan melalui proses *grayscale* untuk mengubah warna gambar menjadi abu-abu. Setelah gambar masukan berwarna abu-abu kemudian melalui proses *edgedetection* (deteksi tepi), untuk mendapatkan tepi dari setiap gambar. Kemudian gambar masukan disimpan kedalam database. Proses berikutnya yaitu *thresholding* dilakukan ketika gambar masukan yang telah melalui *edgedetection* akan digunakan untuk proses pembelajaran dengan *Backpropagation* dan LVQ dan pengenalan. Proses ini mengubah gambar hasil menjadi matriks representasi gambar tersebut dengan isi nilai biner 1 atau 0.

Proses berikutnya merupakan proses pembelajaran menggunakan *backpropagation*. Setiap matrik gambar *learning* akan dipelajari dan menghasilkan 3 jenis nilai bobot berupa matriks dan array kemudian bobot-bobot tersebut disimpan kedalam database. Kemudian dilakukan proses pembelajaran dengan menggunakan LVQ. Setiap matriks gambar akan dipelajari dan menghasilkan 1 matriks nilai bobot yang kemudian disimpan ke database. Proses pengenalan gambar menggunakan metode *backpropagation*, gambar tes hasil proses *thresholding* menjadi data masukan kedalam proses ini. Mengambil nilai bobot hasil pembelajaran *backpropagation* yang telah disimpan di dalam database untuk proses pengenalan *backpropagation*. Hasil pengenalan berupa nilai kelas gambar yang cocok kemudian hasilnya disimpan ke dalam database. Sedangkan proses pengenalan gambar dengan LVQ. Gambar tes hasil proses *thresholding* menjadi data masukan ke dalam proses ini. Mengambil nilai bobot hasil pembelajaran LVQ yang telah disimpan kedalam database untuk melakukan pengenalan LVQ. Hasil pengenalan berupa nilai kelas gambar yang cocok kemudian hasil tersebut disimpan kedalam database. Data-data masukan, hasil pembelajaran dan hasil pengenalan ditampilkan. Hasil dari penelitian ini adalah tingkat kecocokan hasil pengenalan tergantung pada kombinasi nilai parameter yang digunakan dalam proses pembelajarannya. Dari hasil pengujian LVQ lebih direkomendasikan karena lebih baik dari segi akurasi dan waktu yaitu 37,63% dibandingkan dengan *backpropagation* yang hanya 37,33% (Wuryandari & Afianto, 2012).

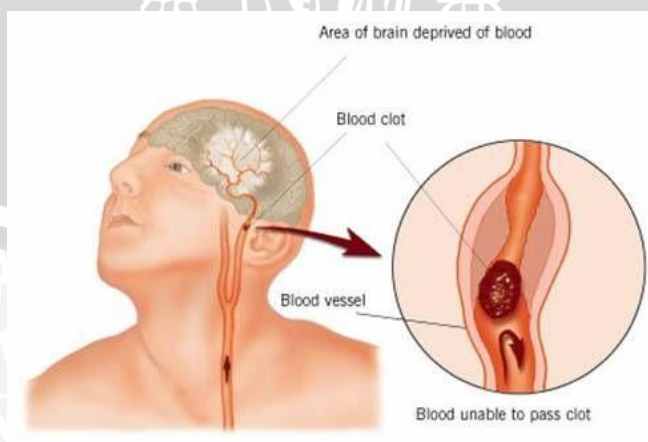
Berdasarkan beberapa penelitian yang telah dilakukan sebelumnya, maka pada penelitian ini untuk menyelesaikan masalah klasifikasi tingkat resiko penyakit stroke dengan mengoptimasi vektor bobot pada LVQ dengan menggunakan algoritma genetika. Adapun kombinasi individu yang dibangkitkan merupakan kombinasi dari dataset pada masing-masing kelas. Serta memodifikasi fungsi fitness menjadi nilai akurasi yang dihasilkan ketika pelatihan LVQ. Keluaran yang diharapkan merupakan vektor bobot yang mampu melakukan klasifikasi tingkat resiko penyakit stroke dengan akurasi yang paling baik.

## 2.2 Klasifikasi

Klasifikasi adalah proses pengelompokan yang sistematis dari sekumpulan objek, buku, gagasan, dan benda-benda lain ke dalam suatu kelas atau golongan tertentu berdasarkan ciri-ciri yang sama. Dalam proses klasifikasi dilakukan pengelompokan benda yang sama serta memisahkan benda yang tidak sama. Secara umum pengertian klasifikasi merupakan susunan pengetahuan atau objek ke dalam tata urutan yang sesuai. Pada kehidupan sehari-hari klasifikasi telah banyak dilakukan oleh manusia. Seperti contoh di pasar, di supermarket, di toko buku, setiap pedagang mengelompokkan barang dagangannya yang memiliki jenis yang sama ke dalam satu kelompok yang sama. Hal ini dimaksudkan untuk mempermudah pembeli untuk memilih kebutuhan yang diperlukan (Hamakonda & Tairas, 2008).

## 2.3 Stroke

Stroke atau gangguan peredaran darah otak (GPDO) merupakan penyakit saraf yang sering ditemui dan perlu ditangani secara tepat dan cepat. Stroke terjadi karena kelainan fungsi otak yang timbul mendadak yang disebabkan terjadinya gangguan peredaran darah otak yang bisa menjangkiti setiap orang setiap waktu. Stroke merupakan penyakit yang sering mengakibatkan kecacatan berupa kelumpuhan gangguan bicara, gangguan gerak, kemampuan mengingat, proses berpikir, dan juga kecacatan yang lain sebagai akibat gangguan fungsi otak (Muttaqin, 2008).



Gambar 2.1 Stroke

Stroke dapat juga diartikan sebagai gangguan fungsi sistem saraf yang terjadi mendadak dan disebabkan oleh gangguan yang terjadi pada pembuluh darah diotak. Gangguan peredaran darah otak adalah seperti tersumbatnya pembuluh darah otak yang terjadi seperti pada Gambar 2.1 atau pecahnya pembuluh darah di otak. Otak yang seharusnya mendapat pasukan oksigen dan zat makanan menjadi terganggu. Otak yang mengalami kekurangan pasokan oksigen dapat menimbulkan kematian sel saraf (neuron). Gangguan fungsi otak tersebut juga dapat menimbulkan gejala stroke (Pinzon & Asanti, 2010).

### 2.3.1 Faktor Stroke

Menurut Sejumlah peneliti kemungkinan terjadinya Stroke masih dapat dicegah. Oleh karena itu, perlu diwaspadai faktor risiko yang memicu munculnya pembunuh nomer tiga di Indonesia ini (Utami, 2009). Setiap orang memiliki peluang adanya beberapa faktor resiko terkena penyakit Stroke.

Ada dua macam golongan penyebab dan faktor resiko yang bisa menyebabkan seseorang terkena stroke yaitu, faktor yang dapat tidak dapat diubah seperti: keturunan, jenis kelamin, umur dan ras. Sedangkan, faktor yang dapat diubah adalah obesitas, diabetes melitus, penyakit jantung dan hipertensi, hiperkolesterol serta faktor gaya hidup (Sutrisno, 2007). Beberapa faktor yang memiliki pengaruh yang biasanya diperiksa dalam laboratorium untuk deteksi stroke (Iman, 2012):

#### 1. Umur

Kejadian stroke bisa terjadi pada semua umur namun sebagian besar penderita berumur di atas 55 tahun, dan setiap kali bertambah 10 tahun setelah umur 55 tahun resiko stroke mengalami peningkatan sebesar dua kali lipat. Skala pengelompokan dapat dilihat pada Tabel 2.2.

**Tabel 2.2 Umur**

| Range   | Keterangan |
|---------|------------|
| < 35    | Muda       |
| 35 – 55 | Paruh baya |
| > 55    | Tua        |

#### 2. Total Kolesterol

Total kolesterol adalah kadar keseluruhan kolesterol yang beredar dalam tubuh. Peredaran kolesterol juga melewati pembuluh darah. Skala pengelompokan dapat dilihat pada Tabel 2.3.

**Tabel 2.3 Total Kolesterol**

| Range           | Keterangan    |
|-----------------|---------------|
| < 200 mg/dl     | Normal        |
| 200 – 239 mg/dl | Tinggi        |
| > 239 mg/dl     | Sangat tinggi |

### 3. HDL (High Density Lipoprotein)

HDL adalah transportasi yang membawa kolesterol kembali ke liver dari berbagai sel-sel jaringan tubuh. Disebut sebagai kolesterol baik karena semakin tinggi HDL akan menyebabkan penurunan dari penumpukan kolesterol, sehingga semakin baik bagi tubuh. Skala pengelompokan dapat dilihat pada Tabel 2.4.

**Tabel 2.4 HDL**

| Range         | Keterangan           |
|---------------|----------------------|
| < 35 mg/dl    | Terlalu rendah       |
| 35 – 60 mg/dl | Menguntungkan        |
| > 60 mg/dl    | Sangat menguntungkan |

### 4. LDL (Low Density Lipoprotein)

LDL adalah transportasi yang membawa kolesterol dari liver ke banyak jaringan. Disebut sebagai kolesterol jahat karena memiliki peran untuk membawa kolesterol ke banyak jaringan tubuh yang dapat menghasilkan peluang terjadinya penumpukan kolesterol di berbagai jaringan tubuh termasuk pembuluh darah. Skala pengelompokan dapat dilihat pada Tabel 2.5.

**Tabel 2.5 LDL**

| Range       | Keterangan    |
|-------------|---------------|
| < 100 mg/dl | Normal        |
| 100 – 199   | Batas Tinggi  |
| > 199       | Sangat tinggi |

### 5. Trigliserida

Trigliserida adalah sejenis lemak pada darah yang bermanfaat sebagai sumber energi. Jika konsumsi makanan lebih dari yang diperlukam maka kalori yang berlebih akan disimpan sebagai trigliserida dalam jaringan lemak untuk simpanan penggunaan dikemudian waktu jika diperlukan. Skala pengelompokan dilihat pada Tabel 2.6.

**Tabel 2.6 Trigliserida**

| Range           | Keterangan    |
|-----------------|---------------|
| < 150 mg/dl     | Normal        |
| 150 – 199 mg/dl | Batas tinggi  |
| 200 – 499 mg/dl | Tinggi        |
| > 500 mg/dl     | Sangat tinggi |

## 2.4 Jaringan Saraf Tiruan

Jaringan Saraf Tiruan merupakan metode pengolah informasi yang mengadopsi cara kerja menyerupai sistem saraf secara biologis, seperti otak. Elemen kunci dari jaringan saraf tiruan yaitu sejumlah elemen saraf pemrosesan yang saling terhubung (*neuron*) yang sangat besar, bekerja bersama untuk menyelesaikan sebuah permasalahan (Yani, 2005).

Jaringan Saraf Tiruan dapat memodelkan permasalahan *non-linier* yang kompleks yang susah diselesaikan dengan merumuskan persamaan matematis biasa. Struktur jaringan saraf tiruan terdiri dari 3 lapisan yaitu lapisan *input*,

lapisan tersembunyi, dan lapisan *output*. Tiap-tiap node antar lapisan akan dihubungkan dengan bobot dan dipengaruhi oleh bias (Jin & Shu, 2013).

#### 2.4.1 Konsep Dasar Jaringan Saraf Tiruan

Setiap bentuk informasi yang ada pada *input* dan *output* akan digunakan pada JST untuk diproses dalam *neuron*. *Neuron-neuron* tersebut terdapat pada lapisan-lapisan yang juga disebut dengan *neuron layers*. Lapisan-lapisan penyusun JST tersebut terbagi menjadi 3, yaitu :

1. Lapisan *Input*

Unit-unit yang ada di dalam lapisan *input* disebut juga unit-unit *input*. Unit-unit *input* kemudian mendapatkan pola *inputan* data dari luar yang dapat menggambarkan sebuah permasalahan.

2. Lapisan Tersembunyi

Unit-unit yang ada di dalam lapisan tersembunyi disebut juga unit-unit tersembunyi. Sehingga outputnya tidak dapat diamati secara langsung.

3. Lapisan *Output*

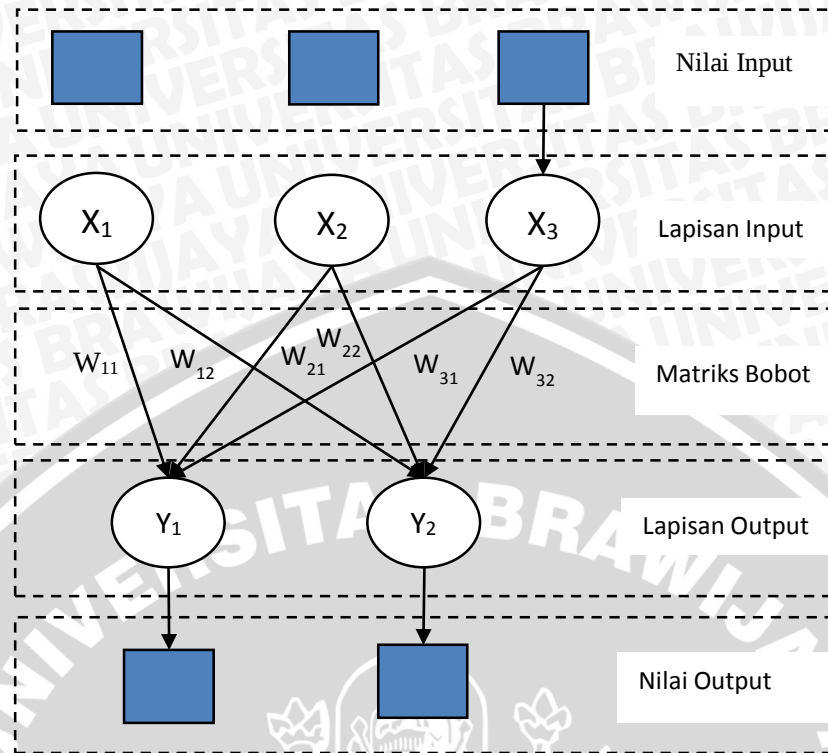
Unit-unit yang ada di dalam lapisan *output* disebut juga unit-unit *output*. Hasil *output* pada lapisan ini yang akan menjadi solusi dari JST pada sebuah permasalahan.

#### 2.4.2 Arsitektur Jaringan

Jaringan Saraf Tiruan mempunyai beberapa macam arsitektur jaringan yang sering dimanfaatkan untuk berbagai macam kebutuhan aplikasi. Arsitektur JST tersebut, yaitu (Kusumadewi, 2003):

##### Jaringan Lapisan Tunggal (Single Layer Network)

Jaringan dengan lapis tunggal ini hanya memiliki 1 lapisan *input* dan 1 lapisan *output*. Setiap unit yang ada pada lapisan *input* selalu terhubung pada setiap unit yang ada pada lapisan *output*. Jaringan ini akan menerima *input* kemudian akan langsung memprosesnya menjadi *output* tanpa memerlukan lapisan tersembunyi. Seperti yang ada pada Gambar 2.2.



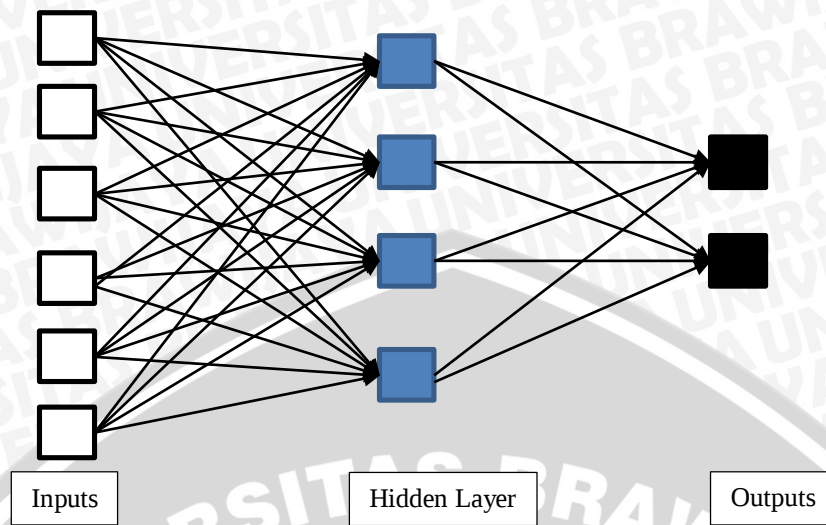
**Gambar 2.2 Jaringan Lapis Tunggal**

Berdasarkan dari arsitekturnya, Jaringan Saraf Tiruan memiliki kategori :

#### 2.4.2.1 Struktur *Feedforward*

Sebuah jaringan yang sederhana mempunyai struktur *feedforward* dimana sebuah sinyal akan bergerak melalui lapisan *input* kemudian akan melalui lapisan tersembunyi hingga sampai pada unit *output*.

Tipe jaringan ini memiliki sel saraf yang terdiri dari berbagai lapisan. Lapisan *input* tidak termasuk sel saraf. Lapisan ini hanya berfungsi untuk memperkenalkan sebuah nilai pada suatu variabel. Lapisan *output* dan lapisan tersembunyi pada sel saraf saling terhubung antara satu sama lain dengan lapisan sebelumnya. Peluang yang muncul adalah terjadinya hubungan dengan beberapa unit dari lapisan yang terhubung sebelumnya atau terhubung semuanya (lebih baik) (Yani, 2005). Seperti yang ada pada Gambar 2.3.



**Gambar 2.3 Jaringan Saraf Tiruan *Feedforward***

Yang merupakan bagian dari struktur *feedforward* :

- *Single-layer perceptron*
- *Multilayer perceptron*
- *Radial-basis function networks*
- *Polynomial learning networks*
- *Higher-order networks*

## 2.5 Learning Vector Quantization (LVQ)

*Learning Vector Quantization* (LVQ) adalah salah satu algoritma yang ada di dalam jaringan saraf tiruan (JST) dengan tipe arsitektur jaringan lapis-tunggal umpan-maju (*Single Layer Feedforward*) yang terdiri dari unit *input* dan unit *output*. LVQ melakukan pembelajaran di lapisan kompetitif yang terawasi. Pada metode ini maka pola pengklasifikasian dilakukan dengan setiap unit *output* akan mewakili satu kelas atau kategori tertentu.

Metode LVQ menggunakan vector acuan (*vector reference*) dimana sebuah unit *output* akan menjadi acuan pada satu kelas atau kategori yang akan diwakilkan oleh *output* tersebut. Maka pendekatan yang dilakukan adalah dengan mengelompokkan vektor *input* yang diukur kedekatan jaraknya terhadap vektor bobot (menggunakan metode kuadrat jarak *euclidean* minimum). *Learning Vector Quantization* adalah metode jaringan saraf tiruan yang menggunakan basis kompetisi dengan mekanisme *squared euclidean distance* dalam memilih vektor perwakilan pemenang untuk menentukan kategori vektor *input*. Proses pembelajaran LVQ merupakan pembelajaran *supervised* atau bisa dibilang menggunakan pengarah, yang bertujuan untuk mendapatkan vektor-vektor perwakilan yang akan melakukan kuantisasi terhadap vektor *input* (Kusumadewi, 2003).

### 2.5.1 Algoritma *Learning Vector Quantization*

Algoritma pada jaringan LVQ memiliki tujuan untuk mendapatkan unit *output* yang paling mendekati dengan vektor *input*. Untuk mencapai tujuan itu, jika  $x$  dan  $w_c$  memiliki kelas yang sama, maka vektor bobot akan disesuaikan agar mendekati dengan vektor *input*. Jika  $x$  dan  $w_c$  memiliki kelas yang berbeda, maka vektor bobot akan disesuaikan agar menjauhi vektor masukan. Tata nama yang dipakai untuk algoritma ini adalah sebagai berikut (Fausett, 1994):

|             |                                                                                                    |
|-------------|----------------------------------------------------------------------------------------------------|
| $x$         | Vektor pelatihan ( $x_1, \dots, x_i, \dots, x_n$ ).                                                |
| $T$         | Kelas atau kategori untuk vektor pelatihan.                                                        |
| $w_j$       | Vektor bobot terhadap kelas ke- $j$ .                                                              |
| $C_j$       | Kategori atau kelas ke- $j$                                                                        |
| $  x-w_j  $ | Jarak <i>Euclidean</i> antara vektor <i>input</i> dan (vektor bobot untuk) <i>output</i> ke- $j$ . |

Langkah-langkah algoritma pembelajaran metode LVQ dapat dijelaskan sebagai berikut (Fausett, 1994):

**Langkah 0** Menginisialisasi vektor referensi dan menginisialisasi laju pembelajaran ( $\alpha$ ).

**Langkah 1** Selama kondisi berhenti adalah salah, lakukan Langkah 2 sampai dengan Langkah 6.

**Langkah 2** Untuk setiap vektor pelatihan  $x$ , lakukan Langkah 3 sampai dengan Langkah 4.

**Langkah 3** Tentukan  $j$  (sebut dengan  $C_j$ ), dimana  $||x-w_j||$  adalah minimum.

**Langkah 4** Update  $w_j$  sebagai berikut:

Jika  $T=C_j$ , maka

$$w_j(\text{baru}) = w_j(\text{lama}) + \alpha(x(t) - w_j(\text{lama})) \quad (2.1)$$

Jika  $T \neq C_j$ , maka

$$w_j(\text{baru}) = w_j(\text{lama}) - \alpha(x(t) - w_j(\text{lama})) \quad (2.2)$$

**Langkah 5** Kurangi laju pembelajaran.

$$\alpha = \alpha * \text{pengurang} \quad (2.3)$$

**Langkah 6** Uji kondisi berhenti:

Kondisi berhenti dapat ditentukan dengan menetapkan jumlah iterasi atau laju pembelajaran mencapai nilai yang cukup kecil yaitu tidak kurang dari laju pembelajaran minimum.

### 2.6 Algoritma Genetika

Ada empat metode utama pada algoritma evolusi yang digunakan saat ini, tiga di antaranya secara independen dikembangkan lebih dari 30 tahun yang lalu. algoritma itu adalah algoritma genetika (GA) yang dibuat oleh John Holland (1975) dan terkenal dengan David Goldberg (1989), pemrograman evolusioner (EP) yang dibuat oleh Lawrence Fogel (1963) dan dikembangkan lebih lanjut oleh anaknya David Fogel (1992) dan evolusi strategi (ES) yang dibuat oleh Ingo Rechenberg (1973) dan hari ini sangat dipromosikan oleh Thomas Back(1996).

keempat algoritma evolusi utama adalah variasi yang lebih baru dan sangat populer dari algoritma genetika oleh John Koza (1992), yang dikenal sebagai pemrograman genetika.

Algoritma Genetika adalah algoritma yang berusaha menerapkan pemahaman mengenai evolusi alamiah pada tugas-tugas pemecahan masalah (*problem solving*). Pendekatan yang diambil oleh algoritma ini adalah dengan menggabungkan secara acak berbagai pilihan solusi terbaik di dalam suatu kumpulan untuk mendapatkan generasi solusi terbaik berikutnya yaitu pada suatu kondisi yang memaksimalkan kecocokannya atau lazim disebut *fitness*. Generasi ini akan merepresentasikan perbaikan-perbaikan pada populasi awalnya. Dengan melakukan proses ini secara berulang, algoritma ini diharapkan dapat mensimulasikan proses evolusioner.

### 2.6.1 Struktur Umum Algoritma Genetika

Algoritma Genetika memberikan suatu pilihan bagi penentuan nilai parameter dengan meniru cara reproduksi genetika, pembentukan kromosom baru serta seleksi alami seperti yang terjadi pada makhluk hidup.

Menurut Goldberg (1989) mengemukakan bahwa Algoritma Genetika mempunyai karakteristik-karakteristik yang perlu diketahui sehingga dapat terbedakan dari prosedur pencarian atau optimasi yang lain, yaitu:

- Algoritma Genetika dengan pengkodean dari himpunan solusi permasalahan berdasarkan parameter yang telah ditetapkan dan bukan parameter itu sendiri.
- Algoritma Genetika pencarian pada sebuah solusi dari sejumlah individu-individu yang merupakan solusi permasalahan bukan hanya dari sebuah individu.
- Algoritma Genetika informasi fungsi objektif (*fitness*), sebagai cara untuk mengevaluasi individu yang mempunyai solusiterbaik, bukan turunan dari suatu fungsi.
- Algoritma Genetika menggunakan aturan-aturan transisi peluang, bukan aturan-aturan deterministik.

Variabel dan parameter yang digunakan pada Algoritma Genetika adalah:

- Fungsi *fitness* (fungsi tujuan) yang dimiliki oleh masing-masing individu untuk menentukan tingkat kesesuaian individu tersebut dengan kriteria yang ingin dicapai.
- Populasi jumlah individu yang dilibatkan pada setiap generasi.
- Probabilitas terjadinya *crossover* (persilangan) pada suatu generasi.
- Probabilitas terjadinya mutasi pada setiap individu.
- Jumlah generasi yang akan dibentuk yang menentukan lama penerapan Algoritma Genetika.

Secara umum struktur dari suatu Algoritma Genetika dapat mendefinisikan dengan langkah-langkah sebagai berikut:

1. Membangkitkan populasi awal

Populasi awal ini dibangkitkan secara random sehingga didapatkan solusi awal. Populasi itu sendiri terdiri atas sejumlah kromosom yang merepresentasikan solusi yang diinginkan.

## 2. Reproduksi

Untuk membentuk generasi baru, digunakan operator reproduksi atau seleksi, *crossover* dan *mutation*. Proses ini dilakukan berulang-ulang sehingga didapatkan jumlah kromosom yang cukup untuk membentuk generasi baru dimana generasi baru ini merupakan representasi dari solusi baru. Generasi baru ini dikenal dengan istilah anak (*offspring*).

## 3. Evaluasi solusi

Pada tiap generasi, kromosom akan melalui proses evaluasi dengan menggunakan alat ukur yang dinamakan *fitness*. Nilai *fitness* suatu kromosom menggambarkan kualitas kromosom dalam populasi tersebut. Proses ini akan mengevaluasi setiap populasi dengan menghitung nilai *fitness* setiap kromosom dan mengevaluasinya sampai terpenuhi kriteria berhenti. Bila kriteria berhenti belum terpenuhi maka akan dibentuk lagi generasi baru dengan mengulangi langkah b. Beberapa kriteria berhenti yang sering digunakan antara lain: berhenti pada generasi tertentu, berhenti setelah dalam beberapa generasi berturut-turut didapatkan nilai *fitness* tertinggi tidak berubah, berhenti dalam ngenerasi tidak didapatkan nilai *fitness* yang lebih tinggi.

Fungsi *fitness* tersebut adalah sebagai berikut:

$$Fitness = \frac{1}{1 + Penalty} \quad (2.4)$$

Dimana:

$$Penalty = \sum Bp \sum Np \quad (2.5)$$

Dari persamaan 2.4 diatas nilai *fitness* ditentukan oleh nilai *penalty*. *Penalty* tersebut pada persamaan 2.5 menunjukkan jumlah pelanggaran kendala pada suatu kromosom. Semakin tinggi nilai *fitness* akan semakin besar kemungkinan kromosom tersebut terpilih kegenerasi berikutnya. Jadi nilai *penalty* berbanding terbalik dengan nilai *fitness*, semakin kecil nilai *penalty*(jumlah pelanggaran) semakin besar nilai fitnessnya. Jadi fungsi fitness adalah sebagai berikut:

$$Fitness = \frac{1}{1 + \sum Bp + \sum Np} \quad (2.5)$$

Keterangan :

Bp : Bobot Pelanggaran

Np : IndikatorPelanggaran

### 2.6.2 Real coded Genetic Algorithm

Algoritma genetika yang menggunakan pengkodean biner memiliki kelemahan jika digunakan untuk optimasi fungsi karena tidak mampu menjangkau

beberapa titik solusi jika solusi tersebut berada pada daerah kontinyu. Sehingga optimasi fungsi yang lebih kompleks dan membutuhkan banyak generasi, operasi transformasi dari biner ke desimal ataupun sebaliknya memerlukan waktu yang lebih lama. Oleh karena itu, pengkodean *real* (*real-coded genetic algorithm/RCGA*) mampu menjadi solusi untuk permasalahan tersebut (Mahmudy, 2013).

### 2.6.3 Siklus RCGA

1. Representasi *chromosome*  
Pada bagian ini variabel keputusan akan langsung menjadi gen *string chromosome* sehingga panjang *chromosome* akan sesuai dengan variabel yang ada.
2. Inisialisasi Populasi  
Populasi dibangkitkan secara acak sesuai dengan jumlah *popsiz*e yang telah ditentukan terlebih dahulu.
3. Reproduksi
  - Metode *crossover* menggunakan *extended intermediate crossover* dimana *crossover* dilakukan dengan memilih dua induk (*parent/P*) secara acak dari populasi, kemudian akan menghasilkan *offspring* dari kombinasi *parent* tersebut. Jika P1 dan P2 adalah dua kromosom yang akan digunakan untuk *crossover* maka *offspring* yang dihasilkan yaitu *Child* pertama dan kedua (C1 & C2) akan sesuai dengan rumusan berikut (Mahmudy, 2013):
 
$$C_1 = P_1 + \alpha(P_2 - P_1) \quad (2.6)$$

$$C_2 = P_2 + \alpha(P_1 - P_2) \quad (2.7)$$
 Nilai  $\alpha$  dipilih secara acak antara  $[-0,25;1,25]$
  - Metode *Mutation* menggunakan *random mutation* dimana mutasi dilakukan dengan memilih satu induk, kemudian nilai gen yang terpilih dengan bilangan *random* yang kecil akan ditambah atau dikurangi. Jika domain variabel  $x_j$  adalah  $[\min_j, \max_j]$  dan *offspring* yang dihasilkan adalah  $C=[x'_1, x'_n]$ , maka *offspring* yang dihasilkan akan sesuai dengan rumusan berikut (Mahmudy, 2013):
 
$$x'_i = x'_i + r(\max_j - \min_j) \quad (2.8)$$
  - Seleksi *Elitism* merupakan sebuah metode seleksi yang bekerja dengan mengumpulkan seluruh individu (*parent*) pada populasi dan *offspring* kedalam satu variabel (tempat) yang sama. Individu terbaik pada variabel tersebut akan dipilih dan masuk ke generasi selanjutnya. Metode ini akan menjamin individu yang terbaik yang akan terpilih (Mahmudy, 2013).

### 2.7 Optimasi Vektor Bobot LVQ dengan Algoritma Genetika

Pada tahap ini LVQ akan digunakan untuk klasifikasi dari vektor bobot yang telah *dirandom* sebelumnya. Kemudian GA akan mencari solusi terbaik dari bobot yang telah diupdate pada LVQ serta akan dipilih sebagai model yang paling optimal (Chen, et al., 2010). Berikut tahapan optimasi vektor bobot pada LVQ dengan algoritma genetika:

1. Inisialisasi parameter
2. Bangkitkan populasi yang berisi vektor bobot
3. Kemudian setiap individu dimasukan kedalam LVQ
4. Setiap vektor bobot dihitung dengan vektor pelatihan pada LVQ
5. Bobot yang telah *diupdate* akan dihitung menggunakan GA untuk menghasilkan generasi baru
6. Hitung *crossover* serta *mutation* agar menghasilkan solusi baru untuk generasi berikutnya
7. Hitung akurasi setiap *parent* serta *child* yang dihasilkan
8. Nilai akurasi akan dijadikan sebagai nilai *fitness* untuk diseleksi
9. Jika bobot yang dihasilkan tidak sesuai syarat iterasi atau nilai optimum maka kembali ke langkah ke-4
10. Vektor bobot yang terbaik nantinya dapat digunakan sebagai acuan untuk klasifikasi tingkat resiko penyakit stroke.

Solusi yang paling optimal ditentukan dari batasan iterasi maksimum dan diambil sejumlah *popsiz*e. Kedua syarat tersebut dapat digunakan sebagai kondisi berhenti dan dianggap sebagai solusi yang paling optimal.

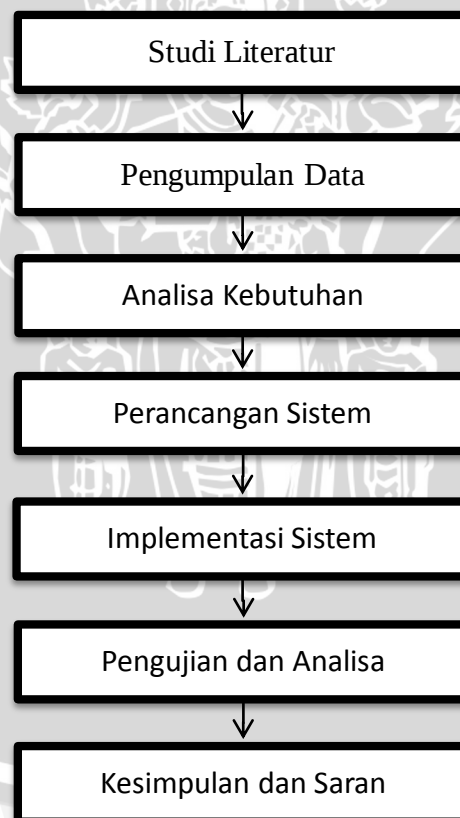


## BAB 3 METODOLOGI

Bab ini menjelaskan mengenai metode penelitian yang dilakukan dalam penelitian optimasi optimasi vektor bobot pada LVQ dengan Algoritma Genetika untuk klasifikasi tingkat resiko penyakit stroke. Beberapa hal dalam metodologi penelitian yaitu tahapan penelitian, studi literatur, pengumpulan dan analisis data, analisa dan kebutuhan sistem, perancangan sistem, implementasi dan pembahasan, pengujian dan analisis, pengambilan kesimpulan dan saran.

### 3.1 Tahapan Penelitian

Tahapan penelitian membahas mengenai langkah-langkah yang akan dilakukan dalam pembuatan sistem Optimasi Vektor Bobot pada *Learning Vector Quantization* dengan Algoritma Genetika untuk klasifikasi tingkat resiko penyakit stroke, yaitu perancangan, implementasi dan pengujian dari aplikasi perangkat lunak yang akan dibuat. Secara umum, langkah-langkah penelitian yang dilakukan dapat digambarkan dalam bentuk diagram seperti yang ditunjukkan pada Gambar 3.1.



**Gambar 3.1 Tahapan Penelitian**

Berdasarkan Gambar 3.1 penjabaran langkah-langkah yang dilakukan dalam penelitian ini adalah :

### 3.2 Studi Literatur

Studi Literatur dilakukan dengan cara mengumpulkan dan mempelajari literatur-literatur yang berkaitan dengan sistem, penyakit stroke, metode algoritma genetika, metode *Learning Vector Quantization*, serta proses pengujian sistem. Sumber literatur dapat berupa buku teks, *paper*, jurnal, karya ilmiah, dan juga penelitian-penelitian sebelumnya.

### 3.3 Pengumpulan dan Analisis Data

Pengumpulan data dilakukan dengan menggunakan arsip data yang telah dilakukan pada penelitian sebelumnya. Adapun yang didapatkan dari pengambilan data adalah data yang telah dikelompokkan oleh dokter yang dijadikan acuan untuk pemberian nilai dari kriteria-kriteria tingkat resiko penyakit stroke. Selanjutnya sebagai acuan kedepan untuk diagnosa dini penyakit stroke dilakukan dengan menghitung *input* dari user ke kelas terdekat dari *vector* bobot yang terbaik dari sistem untuk menghasilkan tingkat resiko yang diderita.

### 3.4 Analisa dan Kebutuhan Sistem

Analisis kebutuhan bertujuan untuk mengetahui secara keseluruhan kebutuhan yang diperlukan dalam membangun sistem. Secara keseluruhan kebutuhan yang digunakan dalam implementasi penelitian ini adalah sebagai berikut :

- Kebutuhan *hardware*, meliputi:
  - Laptop dengan *memory* 4 GB
- Kebutuhan *software*, meliputi:
  - Microsoft Windows 7 sebagai sistem operasi
  - Microsoft Visual Studio 2013 sebagai *deskstop framework*
- Kebutuhan data, meliputi:
  - Data hasil pemeriksaan pasien laboratorium.

Data hasil pemeriksaan pasien sebanyak 200 data di Laboratorium Klinik Sejahtera Kabupaten Probolinggo.

### 3.5 Perancangan Sistem

Perancangan sistem merupakan tahapan penelitian yang menjelaskan tentang desain dari sistem secara menyeluruh, baik dari segi model ataupun arsitektur yang akan digunakan dalam pembuatan sistem. Perancangan sistem dibuat berdasarkan hasil yang telah diperoleh dalam tahap pengumpulan data dan analisa kebutuhan dari *user* dan narasumber. Perancangan sistem dilakukan agar proses implementasi sistem menjadi lebih mudah.

- Diagram Blok Sistem

Diagram blok sistem adalah diagram untuk memetakan proses kinerja sistem yang berbentuk blok-blok yang menggambarkan aliran proses dari komponen-komponen sistem mulai dari masukan hingga keluaran hasil dari sistem.

- Masukan

Sistem akan parameter-parameter yang akan menjadi inisialisasi untuk proses perhitungan optimasi vektor bobot.

- Proses

Proses perhitungan pada sistem ini menggunakan metode algoritma genetika dan LVQ. Metode algoritma genetika digunakan untuk mengoptimasi bobot yang dihasilkan dari metode LVQ, pada metode LVQ bobot digunakan untuk mengklasifikasi data hasil pemeriksaan.

- Keluaran

Hasil keluaran adalah hasil individu (vektor bobot) yang telah mengalami perbaikan sehingga dapat digunakan untuk klasifikasi tingkat resiko penyakit stroke.

### 3.6 Implementasi dan Pembahasan

Implementasi dalam penelitian ini dilakukan dengan mengacu kepada perancangan sistem. Implementasi sistem dilakukan dengan menggunakan bahasa pemrograman dekstop yaitu C#, serta *tools* pendukung lain seperti Visual Studio dan lain sebagainya. Masukan sistem adalah parameter yang digunakan untuk optimasi vektor bobot. Dari data tersebut kemudian diproses didalam perhitungan algoritma genetika dan LVQ. Sedangkan untuk keluaran dari penelitian ini adalah vektor bobot yang dapat digunakan untuk klasifikasi tingkat resiko penyakit stroke. Tahapan-tahapan yang ada dalam implementasi antara lain:

1. Pembuatan antarmuka sistem.
2. Perhitungan metode algoritma genetika dan LVQ.
3. Perhitungan akurasi yang dihasilkan oleh sistem.

Keluaran berupa tingkat resiko penyakit stroke yang diderita oleh pasien atau seseorang.

### 3.7 Pengujian dan Analisis

Pada penelitian ini pengujian dilakukan agar sistem yang dibangun mampu bekerja sesuai dengan spesifikasi dari kebutuhan yang telah ditentukan oleh pengembang. Selain itu perlu dilakukan pengujian mana yang akan menghasilkan tingkat akurasi paling baik.

### 3.8 Pengambilan Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahapan perancangan, implementasi, dan pengujian sistem telah selesai dilakukan yang didasarkan pada teori dan praktik. Kesimpulan diambil berdasarkan hasil dari pengujian sistem dan analisa dari penggunaan metode algoritma genetika dan LVQ dengan tujuan untuk menjawab rumusan masalah yang telah ditentukan sebelumnya. Tahap terakhir dari penulisan adalah saran untuk memperbaiki kesalahan-kesalahan yang terjadi pada sistem saat menggunakan metode algoritma genetika dan LVQ sehingga dapat diperbaiki pada penelitian selanjutnya.



## BAB 4 PERANCANGAN

Pada bab ini akan dibahas beberapa hal, yaitu formulasi perancangan, perancangan algoritma, perhitungan manual, perancangan antarmuka pengguna, dan yang terakhir adalah perancangan uji coba dan evaluasi.

### 4.1 Formulasi Perancangan

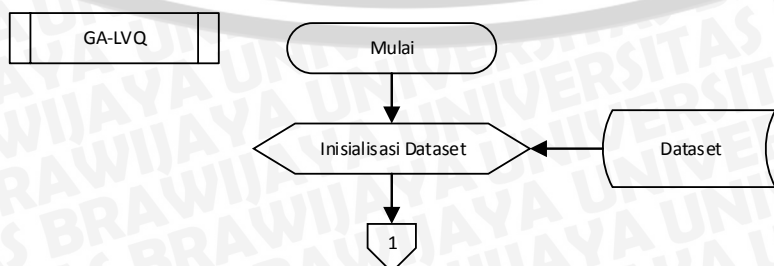
Permasalahan yang akan diselesaikan adalah klasifikasi data menggunakan vektor bobot yang terus diperbaiki dengan menggunakan data latih. Hasil klasifikasi tersebut akan dievaluasi dengan menghitung nilai akurasi yang dihasilkan pada setiap vektor bobot.

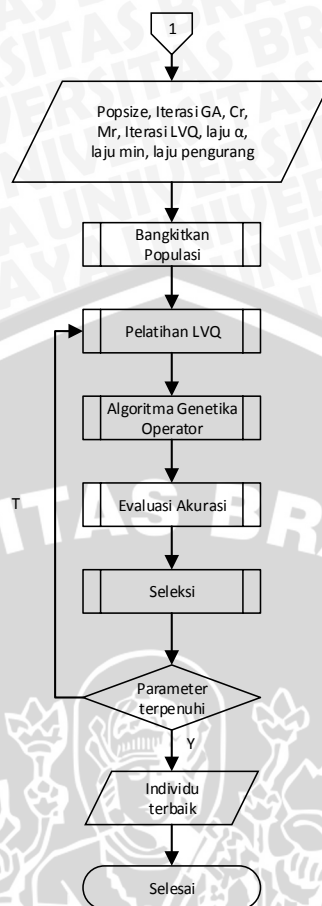
Solusi penyelesaian hasil optimasi adalah vektor bobot pada generasi berikutnya. Vektor bobot yang dihasilkan akan dievaluasi dengan vektor bobot *parent* kemudian hasil akurasi akan diseleksi berdasarkan akurasi yang terbaik. Semakin tinggi nilai akurasi yang didapatkan maka semakin akurat kecocokan klasifikasi yang dihasilkan. Proses perancangan algoritma sistem akan dijelaskan pada sub bab berikutnya.

### 4.2 Perancangan Algoritma

Perancangan Algoritma akan membahas proses perancangan sistem untuk menghasilkan solusi yang terbaik. Pada tahap awal ini adalah dengan inisialisasi parameter. Masukan dari sistem berupa data latih yang diambil dari database dan parameter meliputi : jumlah *popsiz*e, jumlah iterasi algoritma genetika, nilai *cross rate*, nilai *mutation rate*, jumlah iterasi LVQ, laju pembelajaran alpha, laju pembelajaran minimal dan laju pengurangan pembelajaran. Selanjutnya masuk ke proses perhitungan LVQ dengan menghitung jarak *euclidean* yang akan menghasilkan jarak terdekat dan dibandingkan dengan target sebagai tolak ukur untuk perbaikan bobot hingga mencapai batas maksimal iterasi atau laju pembelajaran kurang dari laju pembelajaran minimal.

Tahap berikutnya adalah perhitungan dengan operator algoritma genetika yaitu : *crossover*, *mutation* dan *selection* untuk menghasilkan generasi berikutnya. Individu yang dihasilkan akan dievaluasi akurasinya pada klasifikasi LVQ. Hasil akurasi akan diseleksi untuk dilakukan proses perbaikan dan iterasi berikutnya jika parameter tidak terpenuhi. Jika parameter telah terpenuhi maka akan diambil solusi yang paling optimal. Langkah yang telah dijelaskan sebelumnya dapat dilihat pada Gambar 4.1.

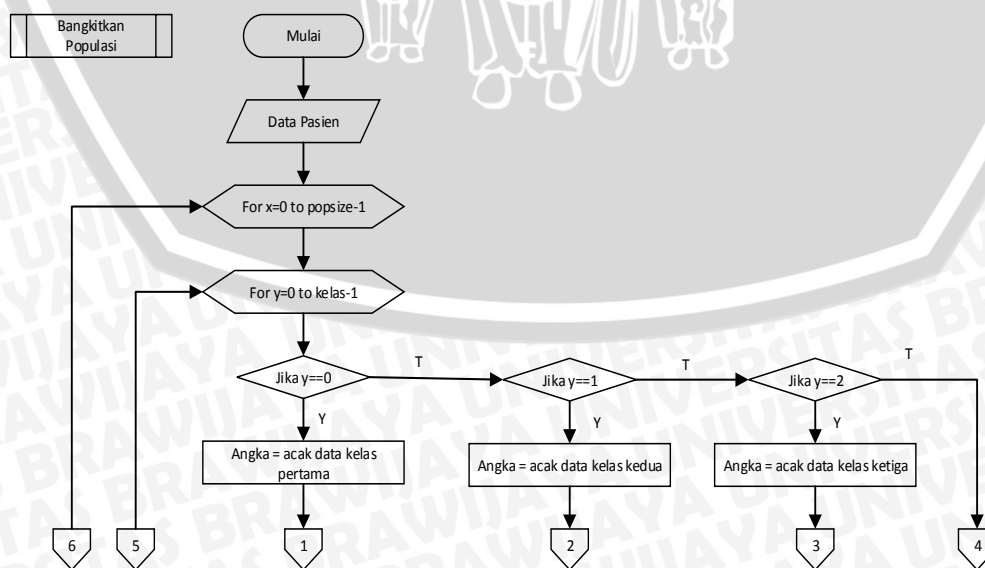


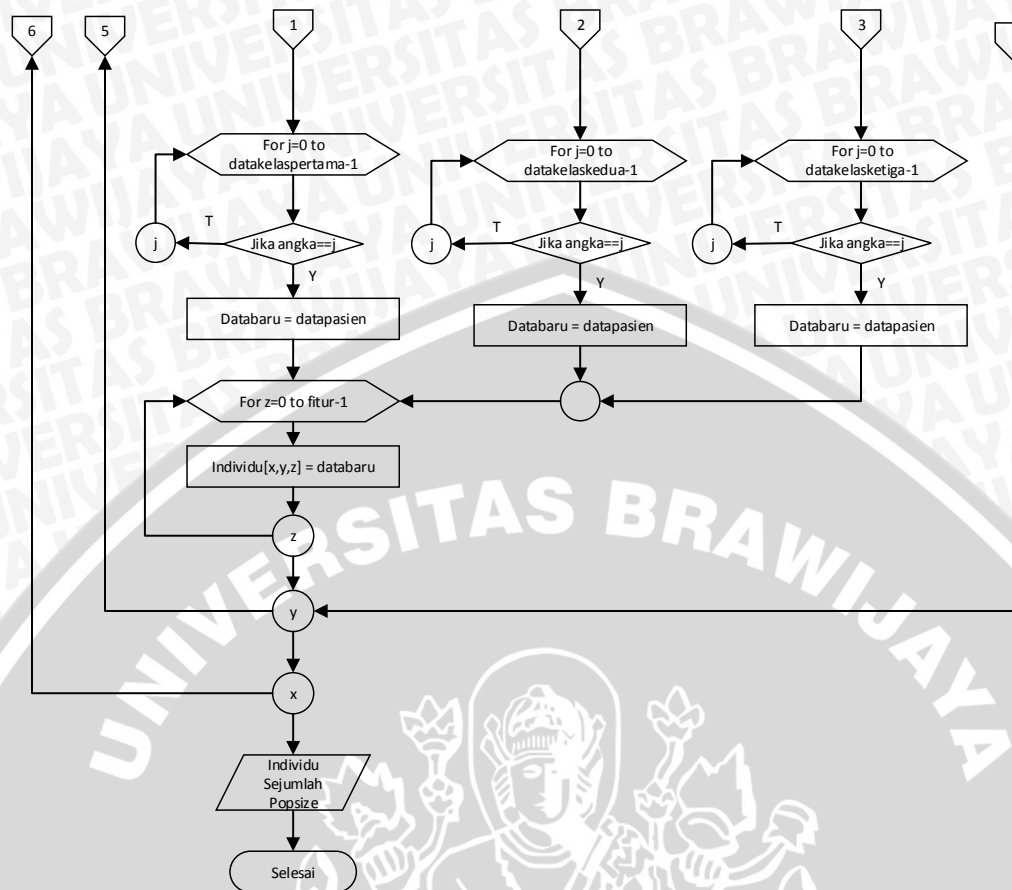


**Gambar 4.1 Diagram Alir Sistem**

#### 4.2.1 Bangkitkan Populasi

Bangkitkan populasi merupakan proses pada algoritma genetika untuk menghasilkan individu awal. Proses yang dilakukan pada tahap ini adalah menghasilkan kombinasi individu dari data pasien secara acak. Diagram alir proses pembangkitan individu ditunjukkan pada Gambar 4.2.





**Gambar 4.2 Bangkitkan Populasi**

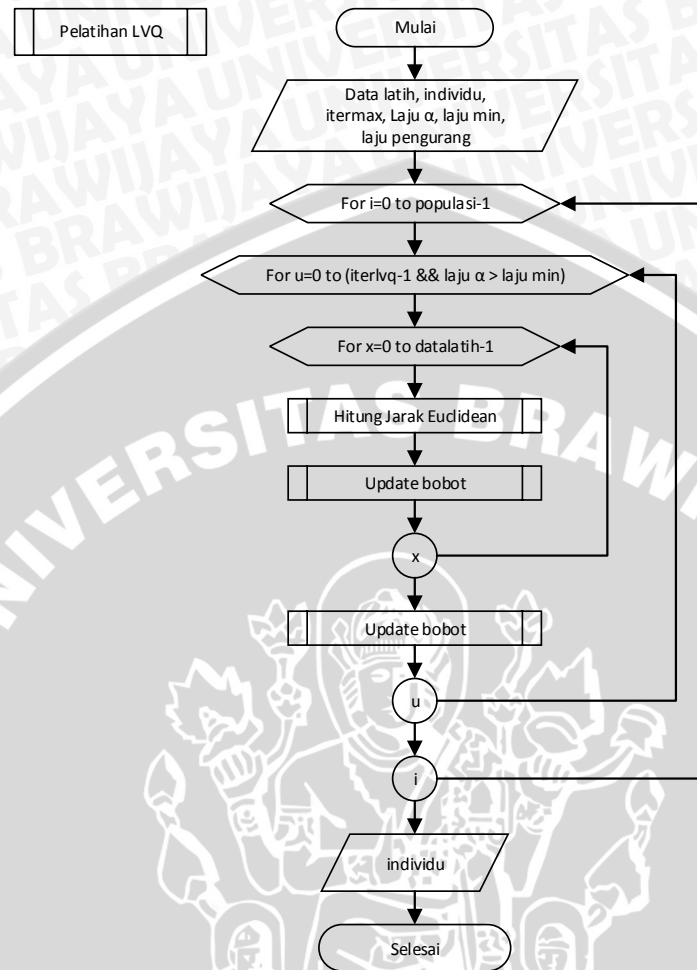
Langkah-langkah pelatihan menggunakan LVQ berdasarkan Gambar 4.2 adalah sebagai berikut :

1. Data latih yang telah disimpan akan dimasukkan untuk membangkitkan populasi.
2. Lakukan perulangan(x) untuk tiap individu hingga individu terakhir.
3. Lakukan perulangan untuk masing-masing kelas bobot.
4. Jika  $y = 0$  maka lakukan *random* data pasien pada kelas pertama, lakukan perulangan(j) untuk menyimpan data.
5. Jika  $y = 1$  maka lakukan *random* data pasien pada kelas kedua, lakukan perulangan(j) untuk menyimpan data.
6. Jika  $y = 2$  maka lakukan *random* data pasien pada kelas ketiga, lakukan perulangan(j) untuk menyimpan data.
7. Lakukan perulangan pada fitur data (z) dan Simpan data pasien pada masing-masing kelas pada variabel individu.
8. Hasil akhir akan di dapatkan individu sejumlah *popsiz e* dari individu yang masing-masing individu memiliki data pada masing-masing kelas yang telah *dirandom*.

#### 4.2.2 Pelatihan LVQ

Pelatihan LVQ merupakan proses pelatihan menggunakan metode LVQ untuk mendapatkan bobot yang optimal. Proses yang perlu dilakukan adalah

vektor bobot akan dihitung dengan vektor data latih. Diagram alir proses pelatihan LVQ ditunjukkan pada Gambar 4.3.



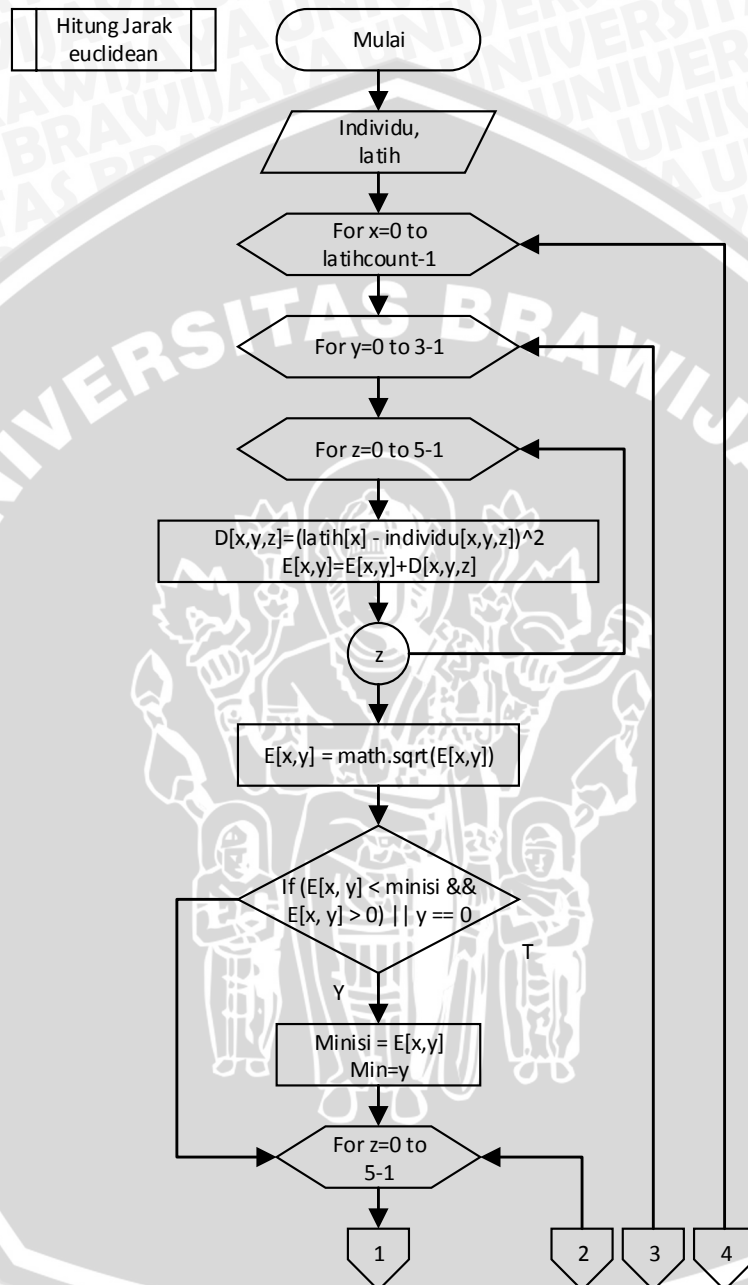
**Gambar 4.3 Pelatihan LVQ**

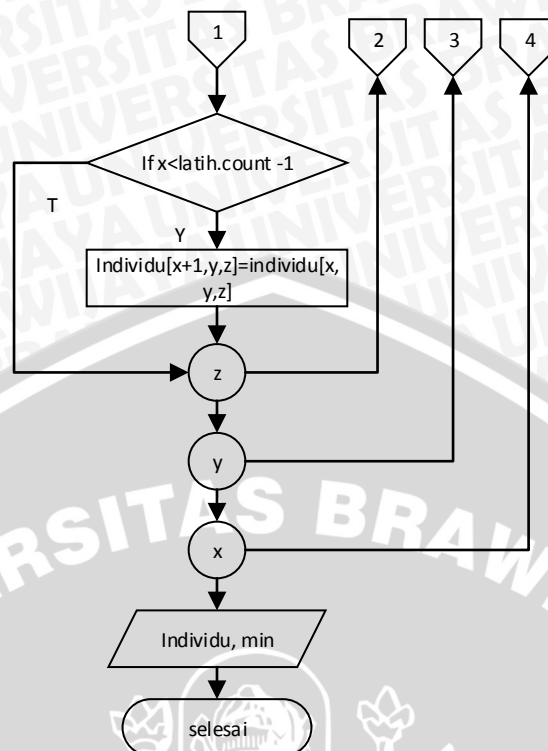
Langkah-langkah pelatihan menggunakan LVQ berdasarkan Gambar 4.3 adalah sebagai berikut :

1. Sistem menerima masukan berupa data latih, individu(vektor bobot), jumlah iterasi (*itermax*), laju  $\alpha$ , laju minimal, dan laju pengurang.
2. Lakukan perulangan(*i*) untuk perulangan hingga jumlah *popsiz*e terpenuhi.
3. Lakukan perulangan(*u*) hingga *iterasilvq* dan laju *alpha* lebih besar dari laju minimal terpenuhi.
4. Lakukan perulangan(*x*) untuk tiap data latih.
5. Kemudian hitung jarak euclidean masing individu dengan data latih.
6. Lakukan update bobot sesuai dengan hasil *euclidean* minimum(*C*) yang di cocokkan dengan hasil data latih(*T*).
7. Bobot yang di *update* dengan data latih terakhir akan disimpan pada untuk iterasi berikutnya.
8. Hingga dihasilkan individu yang telah di perbaiki bobotnya.

### 4.2.3 Hitung Jarak *Euclidean*

Pada proses ini dilakukan proses hitung jarak *euclidean* untuk mengukur jarak minimal data latih dengan tiap individu sebagai penentu untuk proses *update* bobot setelah dihasilkan nilai atau jarak minimal.





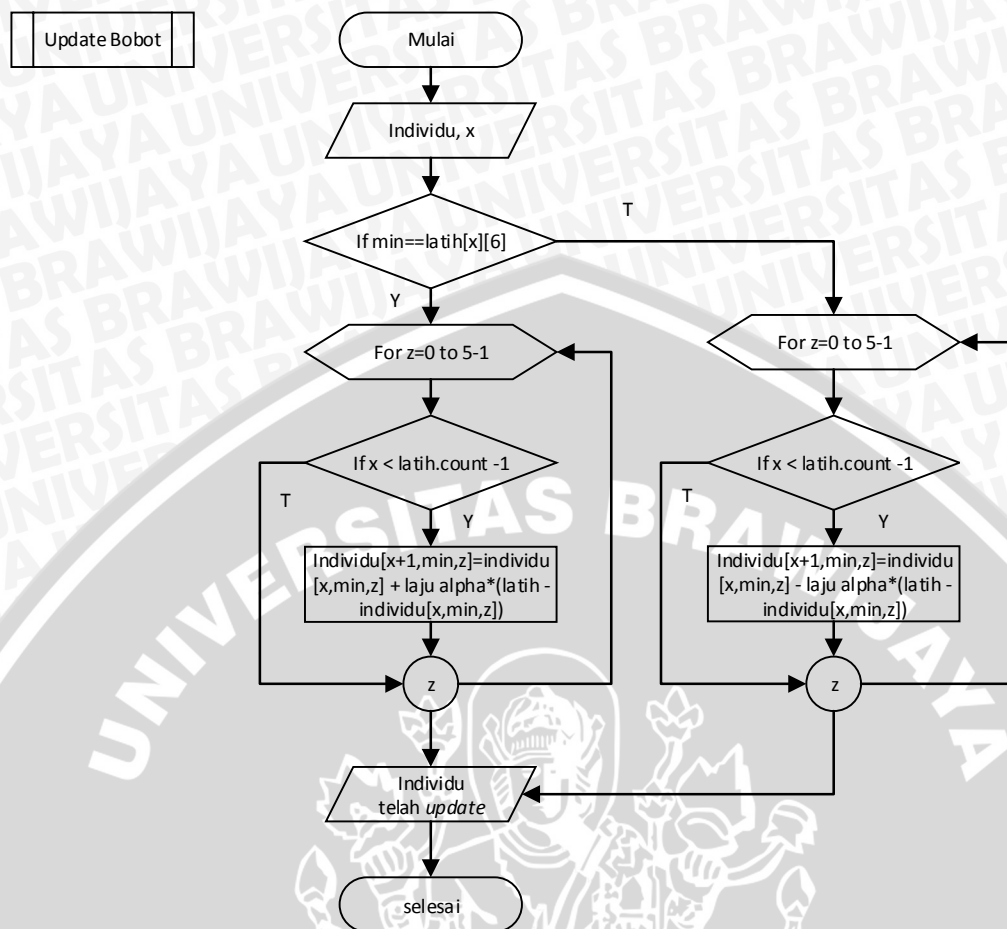
**Gambar 4.4 Hitung Jarak Euclidean**

Langkah-langkah hitung jarak euclidiean berdasarkan Gambar 4.4 adalah sebagai berikut :

1. Sistem menerima masukkan berupa individu, dan data latih.
2. Lakukan perulangan(x) hingga mencapai seluruh data latih.
3. Lakukan perulangan(y) hingga mencapai seluruh kelas.
4. Lakukan perulangan(z) hingga mencapai seluruh fitur data.
5. Hitung dengan persamaan *euclidean*.
6. Jarak *euclidean* minimum akan disimpan dan dibandingkan pada tiap kelas.
7. Individu pada iterasi sebelumnya akan disimpan untuk iterasi berikutnya.
8. Kemudian akan dilakukan proses *update* bobot.
9. Hingga dihasilkan individu yang telah *update* bobotnya.

#### 4.2.4 Update Bobot

Proses *update* bobot adalah proses untuk memperbaiki bobot sehingga bobot mampu menghasilkan akurasi yang semakin baik dengan mendekatkan jarak dari bobot dengan data latih. Jika hasil hitung *euclidean* kelas dengan target data latih maka bobot akan *update* mendekati, sedangkan jika hasil tidak sama maka bobot akan diperbarui menjauhi.



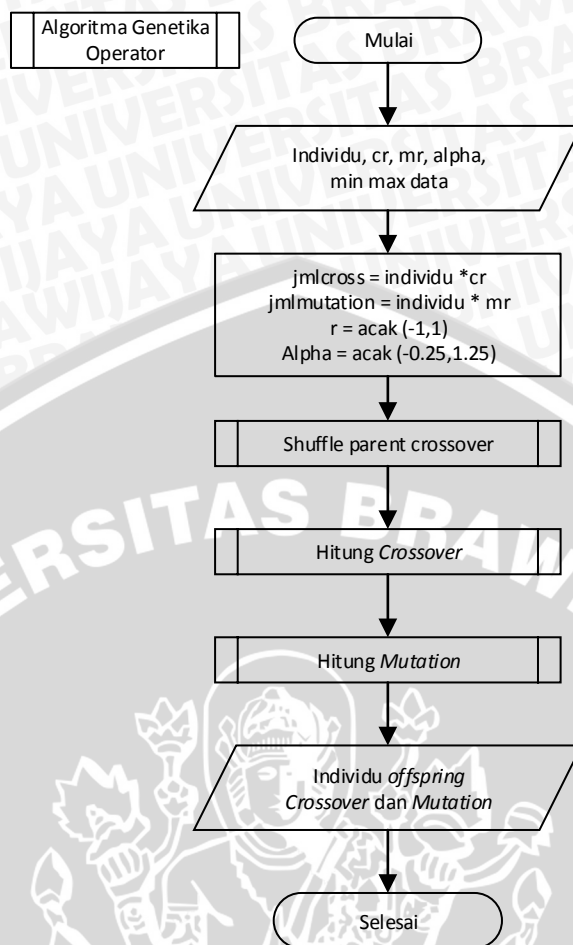
**Gambar 4.5 Update Bobot**

Langkah-langkah *update* bobot berdasarkan Gambar 4.5 adalah sebagai berikut :

1. Sistem menerima masukan berupa individu serta data latih ke-x.
2. Jika *euclidean* minimal(*min*) sama dengan data latih maka:
3. Lakukan perulangan untuk tiap fitur.
4. Kemudian bobot akan diperbaiki sesuai dengan persamaan 2.1.
5. Jika hasilnya tidak sama maka:
6. Bobot akan diperbaiki sesuai dengan persamaan 2.2.
7. Hingga dihasilkan individu yang telah diperbaiki bobotnya.

#### 4.2.5 Algoritma Genetika Operator

Pada Proses Algoritma Genetika operator adalah proses optimasi individu yang bertujuan mempercepat dan mengurangi iterasi yang diperlukan LVQ serta akan memperbaiki hasil keluaran. Hasil keluaran dari optimasi ini berupa individu yang mampu melakukan klasifikasi dengan akurat. Untuk menghasilkan individu yang maksimal akan dilakukan perulangan dan di *update* kembali pada klasifikasi LVQ hingga mencapai batas iterasi sehingga didapatkan individu yang terbaik. Diagram alir algoritma genetika operator ditunjukkan pada Gambar 4.6.



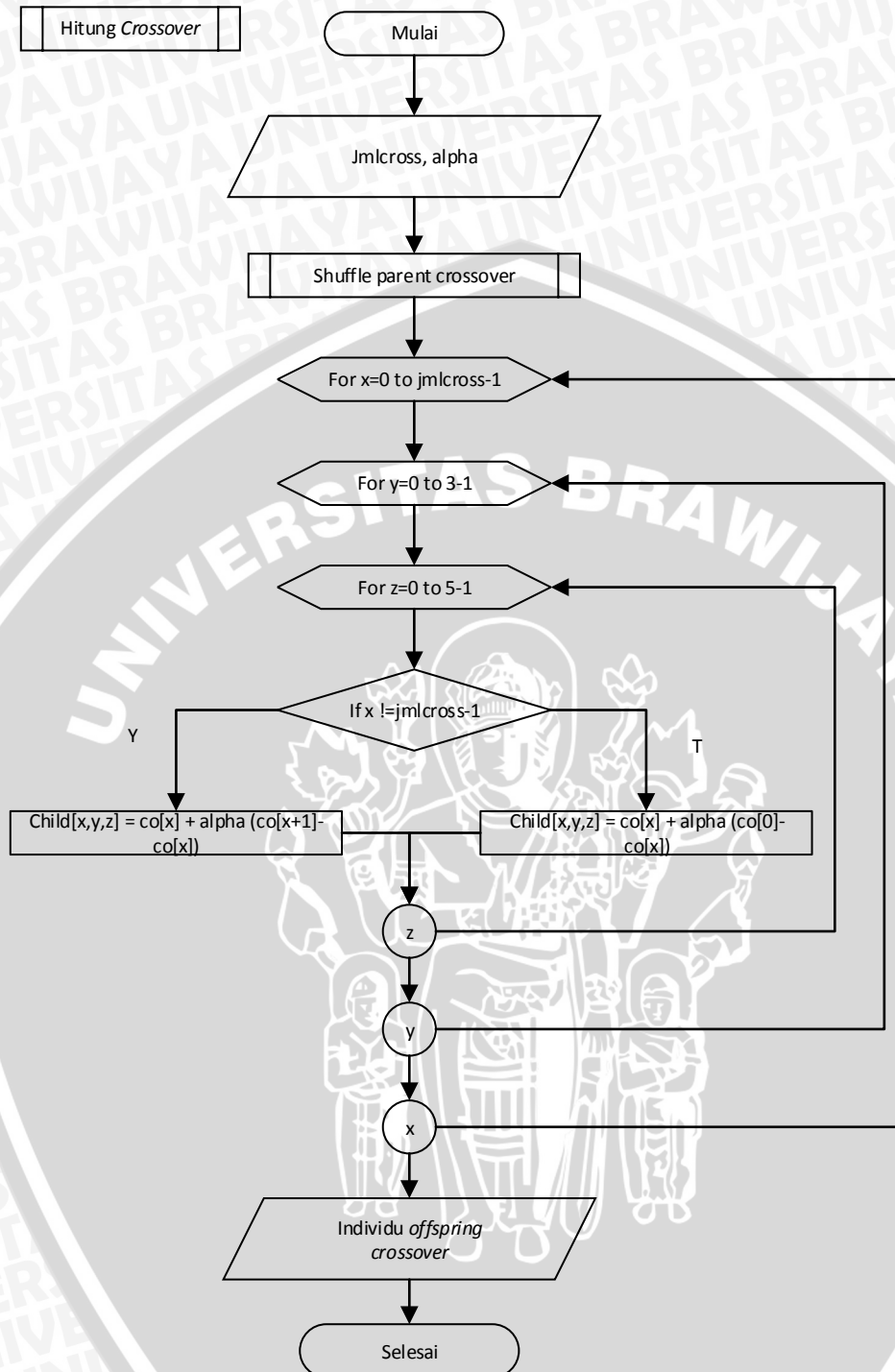
**Gambar 4.6 Algoritma Genetika Operator**

Langkah-langkah optimasi menggunakan algoritma genetika operator berdasarkan Gambar 4.6 adalah sebagai berikut :

1. Masukan berupa individu, *cross rate* dan *mutation rate*, nilai alpha dan domain(*min-max*) tiap data yang telah tersimpan dalam sistem.
2. Hitung *offspring* yang akan dihasilkan *crossover* dan *mutation* dan lakukan *random* untuk nilai *r*.
3. Kemudian lakukan proses *shuffle parent* untuk *crossover*.
4. Kemudian lakukan proses hitung *crossover*.
5. Dan lakukan proses hitung *mutation*.
6. Hingga menghasilkan bobot baru yaitu *offspring(child)* dari *crossover* dan *mutation*.

#### 4.2.6 Hitung Crossover

Pada proses hitung *crossover* merupakan proses kawin silang antara 2 *parent* yang telah dipilih, proses ini akan menghasilkan *offspring* yang telah mengalami kawin silang hingga menghasilkan individu baru. Diagram alir proses perhitungan *offspring* dari *crossover* ditunjukkan pada Gambar 4.8.



**Gambar 4.7 Hitung Crossover**

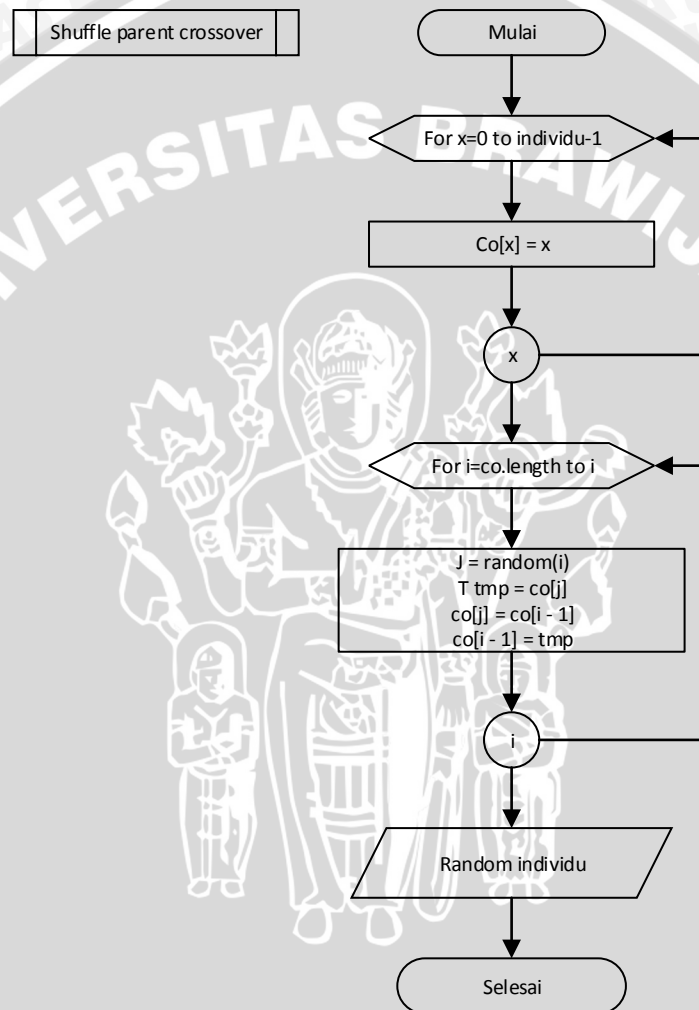
Langkah-langkah menghitung *crossover* berdasarkan Gambar 4.8 adalah sebagai berikut :

1. Masukan berupa data jumlah *crossover*, *alpha*, serta *array parent(co)* yang telah dipilih sebelumnya.
2. Perulangan(x) untuk jumlah *offspring crossover* yang dihasilkan.
3. Perulangan(y) untuk kelas bobot.

4. Perulangan(z) untuk masing-masing fitur bobot.
5. Jika nilai x tidak sama dengan jumlah *crossover* – 1.
6. Maka akan dihasilkan *offspring* baru berdasarkan *parent* yang telah dipilih.
7. Akan dihasilkan *offspring* baru dari *crossover*.

#### 4.2.7 Shuffle Parent Crossover

Pada proses ini dilakukan acak untuk individu yang akan dijadikan *parent* pada proses hitung *crossover*. Pada proses ini sehingga individu yang terpilih hasilnya acak dan tidak akan terpilih lagi.



**Gambar 4.8 Shuffle Parent Crossover**

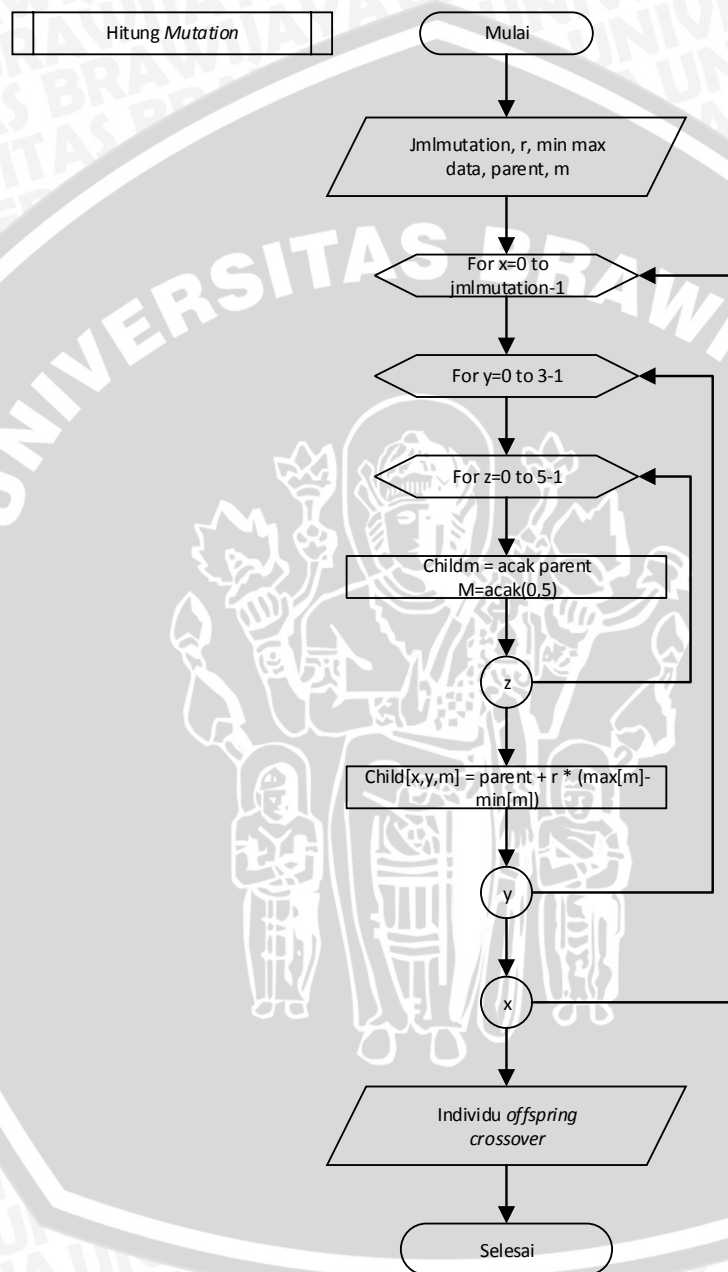
Langkah-langkah *shuffle parent crossover* berdasarkan Gambar 4.7 adalah sebagai berikut :

1. Lakukan perulangan hingga seluruh untuk meyimpan angka sejumlah *popsze*.
2. Simpan nilai x kedalam *crossover parent* (co).
3. Lakukan perulangan(i) dari panjang *array co* hingga lebih dari satu, setiap perulangan panjang array akan dikurangi.
4. Kemudian *random array* yang akan ditukar.
5. Lakukan penukaran nilai pada *array co*.

6. Hingga menghasilkan nilai(*id*) individu yang telah diacak.

#### 4.2.8 Hitung *Mutation*

Pada proses hitung *mutation* akan dijabarkan proses perhitungan untuk menghasilkan *offspring* dari *mutation*. Diagram alir proses perhitungan *offspring* dari *crossover* ditunjukkan pada Gambar 4.9.



**Gambar 4.9 Hitung *Mutation***

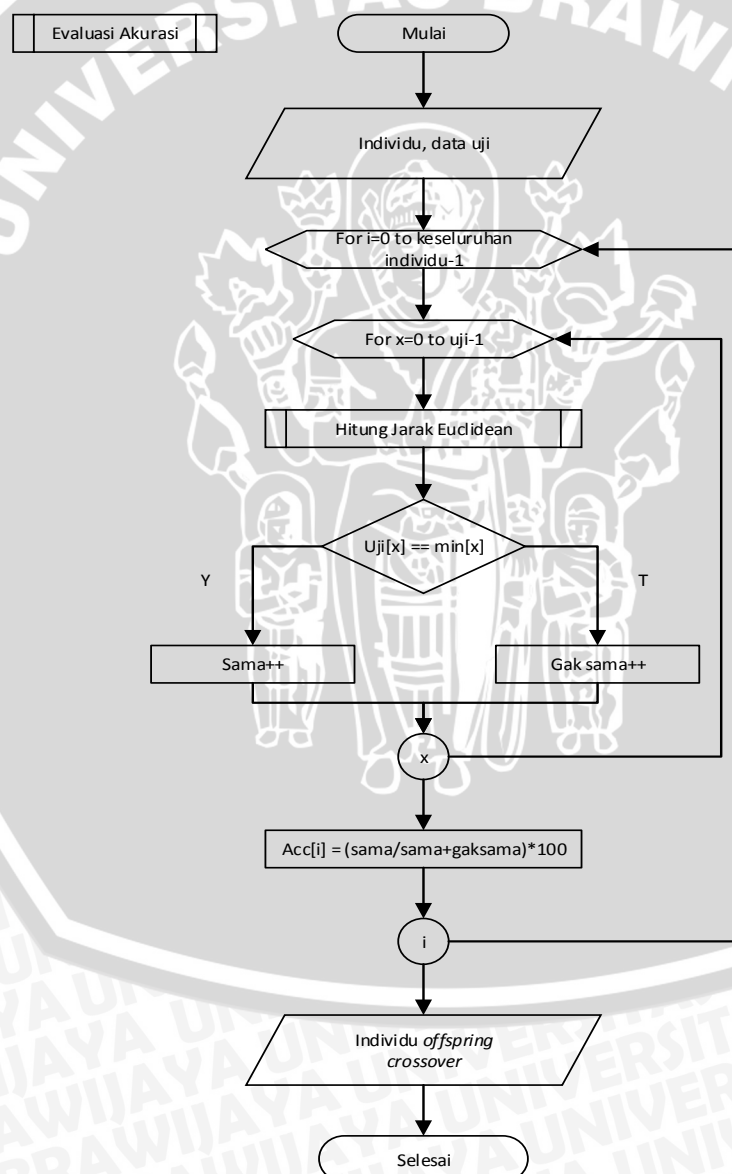
Langkah-langkah menghitung *mutation* berdasarkan Gambar 4.9 adalah sebagai berikut :

1. Masukan berupa data jumlah *mutation*, nilai *r*, domain(*min-max*) fitur data, individu *parent*, dan sel(*m*) yang akan dimutasi.

2. Lakukan perulangan(x) untuk mencapai batas jumlah *mutation*.
3. Lakukan perulangan(y) untuk mencapai batas kelas.
4. Lakukan perulangan(z) untuk mencapai batas fitur.
5. Simpan individu *parent* yang telah di acak kedalam *childm*, dan acak sel yang akan dimutasi.
6. Kemudian lakukan mutasi pada sel yang telah dipilih.
7. Hingga dihasilkan individu(*offspring*) yang telah mengalami mutasi.

#### 4.2.9 Evaluasi Akurasi

Pada proses Evaluasi akurasi akan dijabarkan proses perhitungan akurasi yang digunakan sebagai pengganti nilai *fitness* dimana hasilnya akan digunakan untuk melakukan evaluasi dan *sorting* individu yang terbaik. Diagram alir proses perhitungan evaluasi akurasi ditunjukkan pada Gambar 4.10.



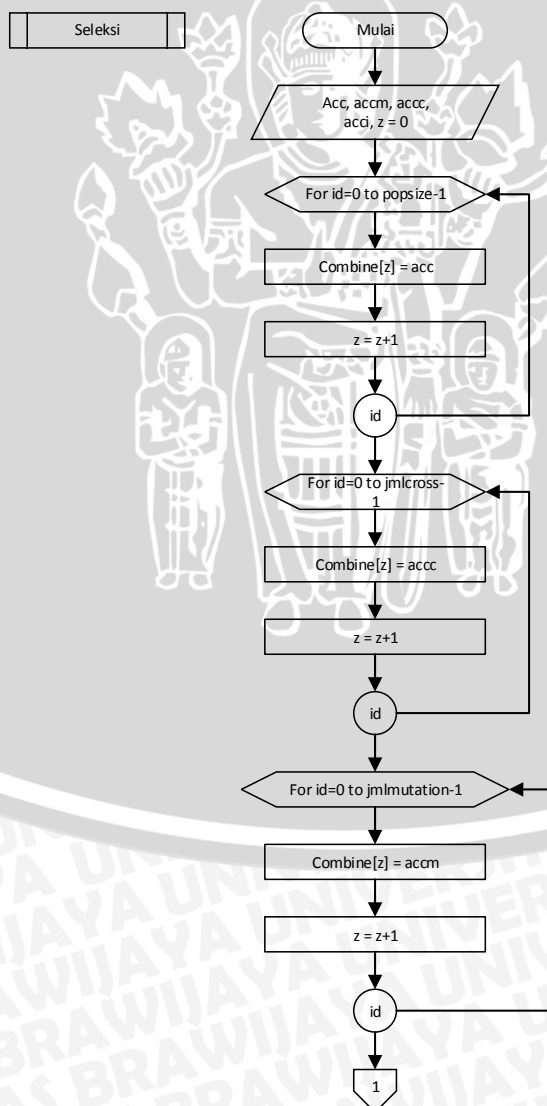
Gambar 4.10 Evaluasi Akurasi

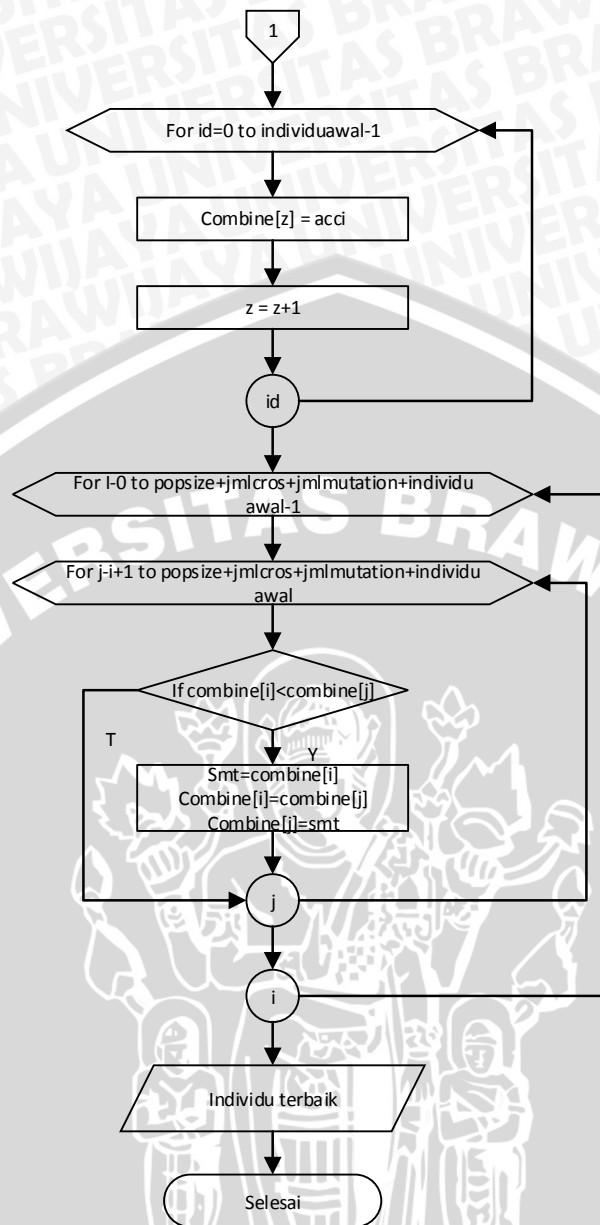
Langkah-langkah evaluasi akurasi berdasarkan Gambar 4.10 adalah sebagai berikut :

1. Masukkan individu dan data uji.
2. Perulangan untuk individu dari awal hingga akhir.
3. Perulangan untuk tiap data uji.
4. Hitung jarak euclidean pada masing-masing bobot dengan data uji.
5. Jika benar maka sama akan ditambah dan jika salah gaksama akan ditambah.
6. Kemudian jika seluruh data uji telah dihitung maka akan dihitung akurasi yang dihasilkan
7. Lakukan seluruh perhitungan hingga seluruh individu telah dihitung.

#### 4.2.10 Seleksi

Pada proses ini akan dilakukan proses pengurutan dan memilih individu yang terbaik sejumlah *popsiz* yang akan digunakan pada iterasi berikutnya. Setiap akurasi pada individu yang telah diupdate maupun individu awal dan *offspring* dari *crossover* dan *mutation* akan disimpan menjadi 1 *array*. Hingga memudahkan proses penukaran.





**Gambar 4.11 Seleksi**

Langkah-langkah seleksi berdasarkan Gambar 4.11 adalah sebagai berikut :

1. Masukan individu dan data uji.
2. Lakukan perulangan untuk masing-masing individu.
3. Kemudian simpan akurasi pada tiap-tiap individu.
4. Lakukan perulangan(*i*) untuk menukar nilai *array* pertama hingga seluruh individu-1.
5. Lakukan perulangan(*j*) untuk menukar nilai yang ada para *array* berikutnya.
6. Jika *array* pertama lebih kecil dari *array* berikutnya maka:
7. Tukar nilai pada *array* pertama dengan *array* berikutnya.
8. Lakukan proses penukaran hingga dihasilkan individu terbaik sejumlah popsize.

### 4.3 Perhitungan Manual

Pada bagian ini dijelaskan proses perhitungan manual mulai dari inisialisasi parameter yang diperlukan seperti : data latih, jumlah iterasi, laju  $\alpha$ , laju minimal, dan laju pengurang dan lain sebagainya. Individu merupakan vektor bobot yang nantinya akan digunakan untuk klasifikasi hingga didapatkan vektor bobot yang paling optimal. Langkah-langkah proses manualisasi akan dijelaskan pada sub bab berikutnya.

#### 4.3.1 Bangkitkan Populasi

Pada tahap ini proses yang dilakukan adalah membangkitkan populasi sejumlah 5 individu. Dimana setiap individu adalah sebuah vektor bobot yang akan digunakan pada Pelatihan LVQ. Setiap vektor bobot memiliki 3 kelas. Individu dibangkitkan dari *random* data latih yang ada untuk tiap kelas. Individu yang dibangkitkan dapat dilihat pada Tabel 4.1.

**Tabel 4.1 Data Individu**

| Individu 1 |    |     |      |       |     |
|------------|----|-----|------|-------|-----|
| $w_1$      | 54 | 198 | 38.7 | 142.1 | 86  |
| $w_2$      | 54 | 186 | 36.6 | 109.8 | 198 |
| $w_3$      | 57 | 212 | 36.9 | 102.9 | 361 |
| Individu 2 |    |     |      |       |     |
| $w_1$      | 62 | 168 | 35.1 | 110.1 | 114 |
| $w_2$      | 50 | 67  | 42.1 | 112.1 | 164 |
| $w_3$      | 81 | 233 | 40.7 | 159.5 | 164 |
| Individu 3 |    |     |      |       |     |
| $w_1$      | 34 | 140 | 38.8 | 137.8 | 128 |
| $w_2$      | 41 | 209 | 37.1 | 138.3 | 168 |
| $w_3$      | 64 | 192 | 36.1 | 101.9 | 260 |
| Individu 4 |    |     |      |       |     |
| $w_1$      | 62 | 147 | 35.1 | 94.5  | 87  |
| $w_2$      | 54 | 186 | 36.6 | 109.8 | 198 |
| $w_3$      | 44 | 291 | 46.7 | 180.5 | 319 |
| Individu 5 |    |     |      |       |     |
| $w_1$      | 63 | 249 | 41.5 | 183.7 | 119 |
| $w_2$      | 28 | 185 | 42.3 | 111.5 | 156 |
| $w_3$      | 53 | 246 | 44.9 | 153.5 | 238 |

#### 4.3.2 Pelatihan LVQ

Proses pelatihan dengan algoritma LVQ dalam perhitungan manual ini diperlukan data latih sebanyak 6 data. 6 data tersebut terdiri dari 2 data dengan tingkat resiko normal, 2 data dengan tingkat resiko rentan serta 2 data dengan tingkat resiko mengkhawatirkan. Tabel 4.2 merepresentasikan data latih untuk perhitungan manual dalam klasifikasi LVQ.

**Tabel 4.2 Data Latih**

| No | Umur | Cholestrol Total | HDL  | LDL   | Trigliserida | Kelompok (T) | Status Risiko |
|----|------|------------------|------|-------|--------------|--------------|---------------|
| 1  | 48   | 197              | 30.5 | 120.5 | 79           | 1            | Normal        |
| 2  | 61   | 205              | 45.2 | 140.6 | 96           | 1            | Normal        |

| No | Umur | Cholestrol Total | HDL  | LDL   | Trigliserida | Kelompok (T) | Status Risiko   |
|----|------|------------------|------|-------|--------------|--------------|-----------------|
| 3  | 52   | 319              | 57.2 | 228   | 169          | 2            | Rentan          |
| 4  | 32   | 237              | 46.4 | 159   | 158          | 2            | Rentan          |
| 5  | 44   | 291              | 46.7 | 180.5 | 319          | 3            | Mengkhawatirkan |
| 6  | 67   | 276              | 51.4 | 188.2 | 182          | 3            | Mengkhawatirkan |

Pelatihan dengan algoritma LVQ dilakukan sesuai dengan teori algoritma jaringan LVQ yang telah dibahas pada bab 2. Dari data latih diatas maka dapat dilakukan pelatihan sebagai berikut :

Langkah 1 : Menginisialisasi parameter yang diperlukan

- Menginisialisasi vektor bobot sesuai dengan Tabel 4.1.
- Menginisialisasi laju pembelajaran  $\alpha = 0,1$
- Menginisialisasi laju pembelajaran minimum = 0,0001
- Menginisialisasi iterasi = 0
- Menginisialisasi iterasi maksimum = 2

Langkah 2 : jika iterasi < iterasi maksimum dan laju pembelajaran  $\alpha >$  laju pembelajaran minimum lakukan langkah berikutnya.

Langkah 3 : Untuk vektor pelatihan x pertama dan bobot individu pertama, tentukan kelas  $C_j$  dimana  $||x-w_j||$  dengan persamaan jarak Euclidean adalah minimum.

- Jarak ke  $w_1$  dihitung dengan cara

$$D_1 = \sqrt{\sum_{i=1}^5 (x_i - w_{i1})^2}$$

$$= \sqrt{(48 - 54)^2 + (197 - 198)^2 + (30,5 - 38,7)^2 + (120,5 - 142,1)^2 + (79 - 86)^2}$$

$$= 24.89578278$$

- Jarak ke  $w_2$  dihitung dengan cara

$$D_2 = \sqrt{\sum_{i=1}^5 (x_i - w_{i2})^2}$$

$$= \sqrt{(48 - 54)^2 + (197 - 186)^2 + (30,5 - 36,6)^2 + (120,5 - 109,8)^2 + (79 - 198)^2}$$

$$= 120.29007$$

- Jarak ke  $w_3$  dihitung dengan cara

$$D_3 = \sqrt{\sum_{i=1}^5 (x_i - w_{i3})^2}$$

$$= \sqrt{(48 - 57)^2 + (197 - 212)^2 + (30,5 - 36,9)^2 + (120,5 - 102,9)^2 + (79 - 361)^2}$$

$$= 283.162$$

- Maka diperoleh jarak terdekat yaitu jarak ke  $w_1$ , sehingga  $C_1 = 1$ .

Langkah 4 : Jika  $T=1$  dan  $C=1$ , maka perbarui bobot sesuai dengan persamaan (2.1)

$$= \begin{pmatrix} 54 \\ 198 \\ 38,7 \\ 142,1 \\ 86 \end{pmatrix} + 0,1 \left[ \begin{pmatrix} 48 \\ 197 \\ 30,5 \\ 120,5 \\ 79 \end{pmatrix} - \begin{pmatrix} 54 \\ 198 \\ 38,7 \\ 142,1 \\ 86 \end{pmatrix} \right]$$

$$= \begin{pmatrix} 53,4 \\ 197,9 \\ 37,88 \\ 139,94 \\ 85,3 \end{pmatrix}$$

- Sehingga diperoleh bobot baru untuk individu pertama sebagai berikut

**Tabel 4.3 Bobot Baru Individu Pertama**

|       |      |       |       |        |      |
|-------|------|-------|-------|--------|------|
| $w_1$ | 53.4 | 197.9 | 37.88 | 139.94 | 85.3 |
| $w_2$ | 54   | 186   | 36.6  | 109.8  | 198  |
| $w_3$ | 57   | 212   | 36.9  | 102.9  | 361  |

Langkah 5 : Lakukan langkah ke 3-4 dengan melanjutkan vektor pelatihan berikutnya dan tetap pada vektor bobot pertama hingga mencapai batas maksimal iterasi atau laju pembelajaran minimal hingga akan dihasilkan vektor bobot individu pertama sebagai berikut

**Tabel 4.4 Bobot Akhir Individu Pertama**

|       |          |          |          |          |          |
|-------|----------|----------|----------|----------|----------|
| $w_1$ | 54.38918 | 187.4274 | 36.82464 | 131.1768 | 78.66978 |
| $w_2$ | 49.2651  | 185.577  | 36.68667 | 110.158  | 193.6612 |
| $w_3$ | 55.349   | 222.033  | 38.1446  | 112.7552 | 355.666  |

Langkah 6 : Lanjutkan ke vektor bobot individu ke 2. Kemudian lakukan kembali langkah 3-5 untuk semua individu hingga akan dihasilkan vektor bobot untuk masing2 individu sebagai berikut.

**Tabel 4.5 Data Individu Update**

| Individu 1 |          |          |          |          |          |
|------------|----------|----------|----------|----------|----------|
| $w_1$      | 54.38918 | 187.4274 | 36.82464 | 131.1768 | 78.66978 |
| $w_2$      | 49.2651  | 185.577  | 36.68667 | 110.158  | 193.6612 |
| $w_3$      | 55.349   | 222.033  | 38.1446  | 112.7552 | 355.666  |
| Individu 2 |          |          |          |          |          |
| $w_1$      | 60.28298 | 175.891  | 35.82992 | 115.0681 | 107.784  |
| $w_2$      | 50       | 67       | 42.1     | 112.1    | 164      |
| $w_3$      | 83.54918 | 234.8952 | 40.23372 | 157.6942 | 184.8366 |
| Individu 3 |          |          |          |          |          |
| $w_1$      | 38.94336 | 154.5514 | 38.64979 | 136.179  | 118.4539 |
| $w_2$      | 37.84782 | 218.1978 | 39.09486 | 146.1081 | 164.9741 |
| $w_3$      | 61.46    | 204.573  | 37.4462  | 111.8822 | 267.493  |

| Individu 4 |          |          |          |          |          |
|------------|----------|----------|----------|----------|----------|
| $w_1$      | 60.28298 | 159.8863 | 35.82992 | 103.1789 | 87.20656 |
| $w_2$      | 49.21355 | 181.489  | 36.05718 | 106.5425 | 194.4465 |
| $w_3$      | 43.04785 | 287.6675 | 45.4503  | 174.8466 | 336.8529 |
| Individu 5 |          |          |          |          |          |
| $w_1$      | 67.36101 | 234.9828 | 39.42295 | 173.832  | 104.1135 |
| $w_2$      | 26       | 183.8    | 43.48    | 110.6    | 163.7    |
| $w_3$      | 53.75811 | 252.9813 | 45.58712 | 158.9416 | 242.2626 |

### 4.3.3 Algoritma Genetika Operator

Pada tahap ini individu yang telah diperbaharui bobotnya kemudian di *crossover*, *mutation* dan *selection*. Untuk menghasilkan individu generasi berikutnya. Sebelum di seleksi akan dievaluasi terlebih dahulu hingga menghasilkan akurasi, akurasi terbaik akan dipilih sejumlah *popsiz*e yaitu 5 individu tertinggi.

Langkah 1: Inisialisasi dan hitung nilai *crossover offspring* dan *mutation offspring*.

- $cr = 0,4 \times 5 = 2 \text{ offspring}$
- $mr = 0,2 \times 5 = 1 \text{ offspring}$
- Nilai  $r = 0.030687$
- Nilai  $\alpha$

|          |          |          |          |          |
|----------|----------|----------|----------|----------|
| 0.323251 | -0.12945 | 1.055757 | 0.491571 | 0.904976 |
|----------|----------|----------|----------|----------|

- Nilai *min* data dan *max* data

|    |     |      |       |      |
|----|-----|------|-------|------|
| 23 | 67  | 30.2 | 38.3  | 33.8 |
| 95 | 514 | 57.2 | 338.1 | 813  |

Langkah 2 : Menghitung *Child* dari *crossover*. Tentukan 2 *Parent* yang akan digunakan. Sebagai contoh disini *parent* yang akan digunakan adalah individu 5 dan individu 3 pada Tabel 4.5. Maka *offspring* yang dihasilkan adalah sebagai berikut :

- Hitung *Child* pertama dengan persamaan :

$$C_1 = P_1 + \alpha(P_2 - P_1)$$

$$C_1 w_1 = \begin{pmatrix} 67,36 \\ 234,98 \\ 39,42 \\ 173,83 \\ 104,11 \end{pmatrix} + \begin{pmatrix} 0,32 \\ -0,12 \\ 1,05 \\ 0,49 \\ 0,9 \end{pmatrix} \left[ \begin{pmatrix} 38,94 \\ 154,55 \\ 38,64 \\ 136,17 \\ 118,45 \end{pmatrix} - \begin{pmatrix} 67,36 \\ 234,98 \\ 39,42 \\ 173,83 \\ 104,11 \end{pmatrix} \right]$$

$$C_1 w_1 = \begin{pmatrix} 58,17 \\ 245,39 \\ 38,6 \\ 155,32 \\ 117,09 \end{pmatrix}$$

- Lanjutkan untuk  $w$  ke-2

$$C_1w_2 = \begin{pmatrix} 26 \\ 183,8 \\ 43,48 \\ 110,6 \\ 163,7 \end{pmatrix} + \begin{pmatrix} 0,32 \\ -0,12 \\ 1,05 \\ 0,49 \\ 0,9 \end{pmatrix} \left[ \begin{pmatrix} 37,84 \\ 218,19 \\ 39,09 \\ 146,1 \\ 164,97 \end{pmatrix} - \begin{pmatrix} 26 \\ 183,8 \\ 43,48 \\ 110,6 \\ 163,7 \end{pmatrix} \right]$$

$$C_1w_2 = \begin{pmatrix} 29,82 \\ 179,34 \\ 38,85 \\ 128,05 \\ 164,85 \end{pmatrix}$$

- Lanjutkan untuk  $w$  ke-3

$$C_1w_3 = \begin{pmatrix} 53,75 \\ 252,98 \\ 45,58 \\ 158,94 \\ 242,26 \end{pmatrix} + \begin{pmatrix} 0,32 \\ -0,12 \\ 1,05 \\ 0,49 \\ 0,9 \end{pmatrix} \left[ \begin{pmatrix} 61,46 \\ 204,57 \\ 37,44 \\ 111,88 \\ 267,49 \end{pmatrix} - \begin{pmatrix} 53,75 \\ 252,98 \\ 45,58 \\ 158,94 \\ 242,26 \end{pmatrix} \right]$$

$$C_1w_3 = \begin{pmatrix} 56,24 \\ 259,24 \\ 36,99 \\ 135,80 \\ 265,09 \end{pmatrix}$$

- Sehingga akan dihasilkan *Child* ke-1 sebagai berikut

**Tabel 4.6 Child Pertama**

|       |          |          |          |          |          |
|-------|----------|----------|----------|----------|----------|
| $w_1$ | 58.17498 | 245.395  | 38.60668 | 155.3229 | 117.0912 |
| $w_2$ | 29.82982 | 179.3471 | 38.85036 | 128.0548 | 164.853  |
| $w_3$ | 56.24775 | 259.2479 | 36.99229 | 135.8086 | 265.0955 |

- Ulangi Langkah 2 dengan merubah *parent*. Untuk *child* ke 2 *parent* yang digunakan adalah individu 3 dan individu 5 pada Tabel 4.5 hitung hingga menghasilkan *child* sebagai berikut :

**Tabel 4.7 Child Kedua**

|       |          |          |          |          |          |
|-------|----------|----------|----------|----------|----------|
| $w_1$ | 48.1294  | 144.1392 | 39.46606 | 154.6881 | 105.4762 |
| $w_2$ | 34.018   | 222.6507 | 43.7245  | 128.6534 | 163.8211 |
| $w_3$ | 58.97036 | 198.3064 | 46.04103 | 135.0153 | 244.6601 |

Langkah 3 : Menghitung *Child* dari *mutation*. Tentukan 1 *Gen* yang akan di ubah. Sebagai contoh disini *gen* yang akan dipilih adalah *gen* ke 4. Individu yang digunakan adalah individu ke 2 pada Tabel 4.5. Maka *offspring* yang dihasilkan adalah sebagai berikut :

- Hitung *mutation* pada *gen* ke 4 dengan persamaan sebagai berikut:

$$x_i = x_{ii} + r(\max_i - \min_i)$$

$$x_2w_1 = 115,06 + (0.030687(338,1 - 38,3))$$

$$x_2w_1 = 124,26$$

$$x_2w_2 = 112,1 + (0.030687(338,1 - 38,3))$$

$$x_2w_2 = 121,3$$

$$x_2w_3 = 157,69 + (0.030687(338,1 - 38,3))$$

$$x_2w_3 = 166,89$$

- Sehingga akan dihasilkan bobot baru yaitu

**Tabel 4.8 Child Mutation**

|       |          |          |          |          |          |
|-------|----------|----------|----------|----------|----------|
| $w_1$ | 60.28298 | 175.891  | 35.82992 | 124.268  | 107.784  |
| $w_2$ | 50       | 67       | 42.1     | 121.3    | 164      |
| $w_3$ | 83.54918 | 234.8952 | 40.23372 | 166.8941 | 184.8366 |

#### 4.3.4 Evaluasi Hasil dan Seleksi

Pada tahap ini proses yang dilakukan untuk mengukur *fitness* akan digantikan dengan menggunakan akurasi sebagai evaluasi hasil yang akan digunakan untuk seleksi. Untuk itu langkah yang perlu dilakukan adalah sebagai berikut :

Langkah 1 : Lakukan langkah 1 hingga 3 pada tahap pelatihan LVQ. Hanya saja iterasi yang dilakukan hanya 1 kali iterasi. Hitung pada semua individu pada Tabel 4.5 dan juga pada seluruh *child* yang telah dihasilkan. Contoh:

- Hitung seluruh vektor pelatihan pada individu pertama dan akan menghasilkan kelas terdekat (C) pada masing-masing vektor pelatihan seperti berikut :

| Vektor Pelatihan ke | T | C |
|---------------------|---|---|
| 1                   | 1 | 1 |
| 2                   | 1 | 1 |
| 3                   | 2 | 2 |
| 4                   | 2 | 2 |
| 5                   | 3 | 3 |
| 6                   | 3 | 2 |

- Hitung akurasi yang dihasilkan pada pelatihan LVQ untuk individu pertama dengan persamaan sebagai berikut

$$Akurasi = \frac{\text{jumlah data relevan}}{\text{jumlah seluruh data}} \times 100\%$$

Maka hasilnya adalah

$$Akurasi = \frac{5}{6} \times 100\%$$

$$Akurasi = 83,33\%$$

- Ulangi langkah ini hingga seluruh individu serta *child* dihitung dan hasilnya adalah sebagai berikut :

| Individu ke-   | Akurasi |
|----------------|---------|
| Individu 1     | 83,33%  |
| Individu 2     | 66,67%  |
| Individu 3     | 83,33%  |
| Individu 4     | 66,67%  |
| Individu 5     | 66,67%  |
| <i>Child</i> 1 | 50%     |
| <i>Child</i> 2 | 83,33%  |
| <i>Child</i> 3 | 66,67%  |

- Seleksi individu yang akan digunakan untuk iterasi berikutnya sejumlah *popsiz*

Maka hasil seleksi yang didapatkan adalah :

| Individu ke- | Individu lama yang dipilih |
|--------------|----------------------------|
| Individu 1   | Individu 1                 |
| Individu 2   | Individu 3                 |
| Individu 3   | <i>Child</i> 2             |
| Individu 4   | Individu 4                 |
| Individu 5   | Individu 5                 |

Hasilnya adalah untuk yang menjadi individu pertama pada iterasi berikutnya adalah individu 1 sesuai dengan Tabel 4.5. begitu pula untuk individu yang lainnya. Setelah dilakukan seleksi kemudian kembali ke proses pelatihan LVQ hingga tahap ini hingga mencapai batas iterasi yang ditentukan. Bobot terbaik yang akan digunakan untuk proses klasifikasi nantinya.

#### 4.4 Perancangan Antarmuka

Antarmuka sistem ini terdiri dari empat halaman tab. Halaman tab pertama adalah halaman masukan untuk inialisasi parameter LVQ dan algoritma genetika, dan juga digunakan *generate* individu baru yang digunakan dalam sistem. Halaman tab kedua adalah halaman tampilan hasil perhitungan dari klasifikasi LVQ, hasil optimasi algoritma genetika. Halaman tab ketiga adalah akurasi yang dihasilkan, akurasi yang telah diseleksi serta bobot yang didapat setelah proses *sorting*. Halaman tab keempat berisi akurasi tertinggi pada tiap individu serta grafik yang dihasilkan.

#### 4.4.1 Rancangan Antarmuka Parameter dan Data

Rancangan halaman parameter dan data ini berisi *field* masukan untuk memberikan parameter terhadap perhitungan yang akan digunakan. Beserta tampilan seluruh data latih yang ditampilkan pada Gambar 4.4.

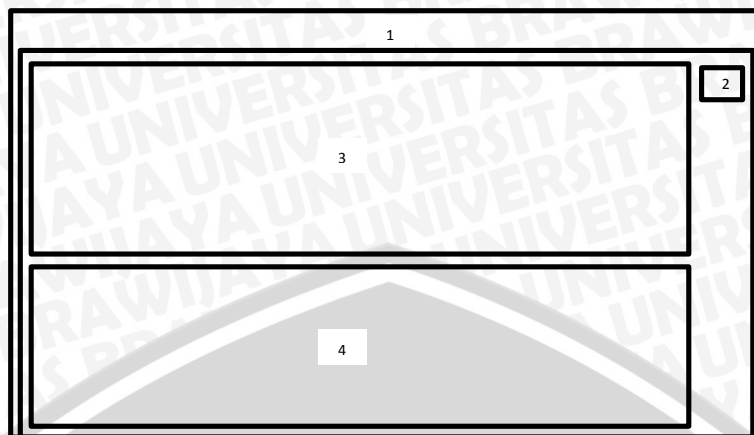
**Gambar 4.12 Halaman Antarmuka Parameter dan Data**

Keterangan :

1. Nama aplikasi.
2. *Textbox* jumlah *popsize*.
3. *Textbox* iterasi algoritma genetika.
4. *Textbox* *cross rate*.
5. *Textbox* *mutation rate*.
6. *Textbox* iterasi LVQ.
7. *Textbox* laju pembelajaran  $\alpha$ .
8. *Textbox* laju pembelajaran minimal.
9. *Textbox* laju pengurang pembelajaran.
10. *Checkbox* *load data*.
11. Tombol hitung.
12. Tombol *generate* individu.
13. Tabel data.

#### 4.4.2 Rancangan Antarmuka Hasil

Rancangan halaman hasil adalah tab yang berisi hasil perhitungan dari bobot yang telah diupdate dan juga hasil dari perhitungan bobot yang telah dioptimasi dan menghasilkan *child*. Serta *combobox* untuk memilih bobot yang ditampilkan dapat dilihat pada Gambar 4.5.



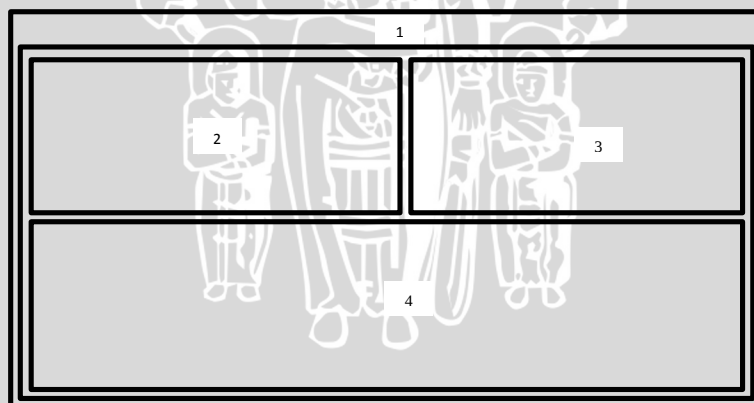
**Gambar 4.13 Halaman Antarmuka Hasil**

Keterangan :

1. Nama aplikasi dan logo.
2. *Dropdown* berisi pilihan bobot yang akan ditampilkan.
3. Tabel hasil perhitungan LVQ.
4. Tabel hasil bobot *child* yang telah dioptimasi algoritma genetika.

#### 4.4.3 Rancangan Antarmuka Akurasi dan Hasil *Sorting*

Rancangan halaman akurasi dan hasil *sorting* adalah tab yang berisi akurasi hasil dari individu yang telah di update bobotnya serta yang telah dioptimasi dengan algoritma genetika. Hasil *sorting* akan ditampilkan berupa individu yang tertinggi sampai sejumlah *popsi*. Tampilan halaman sesuai dengan Gambar 4.6.



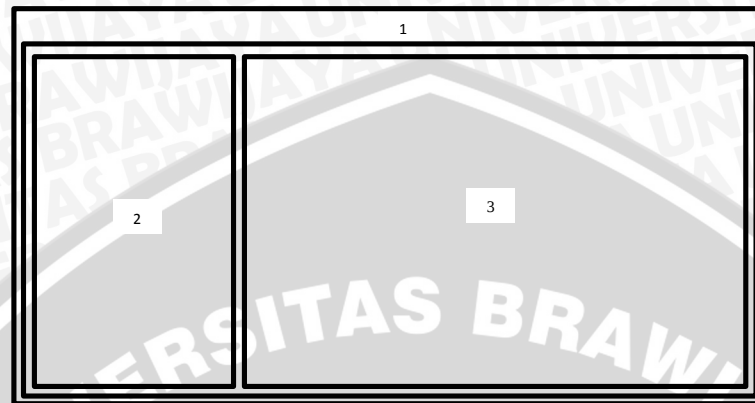
**Gambar 4.14 Halaman Antarmuka Akurasi**

Keterangan :

1. Nama aplikasi.
2. Tabel untuk menampilkan akurasi yang dihasilkan oleh masing-masing bobot.
3. Tabel untuk menampilkan akurasi yang telah disort sejumlah *popsi*.
4. Tabel untuk menampilkan bobot yang telah disort sejumlah *popsi*.

#### 4.4.4 Rancangan Antarmuka Grafik Akurasi

Rancangan halaman grafik akurasi berupa akurasi tertinggi pada tiap iterasi dan juga ditampilkan kedalam grafik. Tampilan halaman sesuai dengan Gambar 4.7.



**Gambar 4.15 Halaman Antarmuka Grafik**

Keterangan :

1. Nama aplikasi.
2. Tabel untuk menampilkan akurasi tertinggi pada tiap iterasi.
3. Grafik untuk menampilkan akurasi tertinggi pada tiap iterasi.

#### 4.5 Perancangan Uji Coba dan Evaluasi

Program yang optimal didapatkan dengan cara mengevaluasi parameter terbaik dari metode algoritma genetika dan LVQ sehingga perlu dilakukan pengujian program dengan uji coba agar mendapatkan dan mengetahui parameter yang paling optimal. Uji coba tersebut antara lain :

1. Uji coba untuk menentukan jumlah iterasi algoritma genetika yang optimal.
2. Uji coba untuk menentukan jumlah iterasi pembelajaran yang optimal.
3. Uji coba untuk mengetahui laju pembelajaran yang optimal.
4. Uji coba untuk laju pembelajaran minimal yang optimal.
5. Uji coba untuk mengetahui laju pengurang pembelajaran yang optimal.
6. Uji coba untuk menentukan nilai *cross rate*(*cr*) dan *mutation rate*(*mr*) yang optimal.
7. Uji coba untuk jumlah individu yang optimal.

Uji coba dilakukan dengan pengujian tingkat akurasi dengan membandingkan data aktual dengan data prediksi dari algoritma genetika dan LVQ. Pada penelitian ini dilakukan pencarian nilai akurasi rata-rata dari setiap pengujian parameter. Rata-rata akurasi didapatkan dari 5 percobaan untuk masing-masing proses pengujian.

#### 4.5.1 Uji Coba Jumlah Iterasi Algoritma Genetika

Uji coba banyaknya jumlah iterasi algoritma genetika adalah uji coba pada sistem untuk mengetahui jumlah iterasi yang digunakan agar dapat menghasilkan vektor bobot (individu) serta akurasi yang optimal. Banyaknya variasi jumlah iterasi yang digunakan adalah 10. Rancangan uji coba jumlah iterasi algoritma genetika dapat dilihat pada Tabel 4.10.

**Tabel 4.9 Rancangan Uji Coba Jumlah Iterasi**

| Jumlah iterasi | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|----------------|--------------------------------|---|---|---|---|---------|
|                | 1                              | 2 | 3 | 4 | 5 |         |
| 2              |                                |   |   |   |   |         |
| 4              |                                |   |   |   |   |         |
| 6              |                                |   |   |   |   |         |
| 8              |                                |   |   |   |   |         |
| 10             |                                |   |   |   |   |         |
| 12             |                                |   |   |   |   |         |
| 14             |                                |   |   |   |   |         |
| 16             |                                |   |   |   |   |         |
| 18             |                                |   |   |   |   |         |
| 20             |                                |   |   |   |   |         |

#### 4.5.2 Uji Coba Iterasi Laju Pembelajaran LVQ

Uji coba banyaknya jumlah iterasi laju pembelajaran adalah uji coba pada sistem untuk mengetahui jumlah iterasi yang digunakan agar dapat menghasilkan individu serta akurasi yang optimal. Banyaknya variasi iterasi laju pembelajaran yang digunakan adalah 10. Rancangan uji coba jumlah iterasi laju pembelajaran dapat dilihat pada Tabel 4.12.

**Tabel 4.10 Rancangan Uji Coba Laju Pembelajaran LVQ**

| Jumlah iterasi | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|----------------|--------------------------------|---|---|---|---|---------|
|                | 1                              | 2 | 3 | 4 | 5 |         |
| 2              |                                |   |   |   |   |         |
| 4              |                                |   |   |   |   |         |
| 6              |                                |   |   |   |   |         |
| 8              |                                |   |   |   |   |         |
| 10             |                                |   |   |   |   |         |
| 12             |                                |   |   |   |   |         |
| 14             |                                |   |   |   |   |         |
| 16             |                                |   |   |   |   |         |
| 18             |                                |   |   |   |   |         |

| Jumlah iterasi | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|----------------|--------------------------------|---|---|---|---|---------|
|                | 1                              | 2 | 3 | 4 | 5 |         |
| 20             |                                |   |   |   |   |         |

#### 4.5.3 Uji Coba Laju Pembelajaran

Uji coba nilai laju pembelajaran ( $\alpha$ ) adalah uji coba pada sistem untuk mengetahui pengaruh nilai laju pembelajaran yang digunakan agar dapat menghasilkan akurasi yang optimal. Banyaknya variasi laju pembelajaran yang digunakan adalah 10. Rancangan uji coba laju pembelajaran dapat dilihat pada Tabel 4.14.

**Tabel 4.11 Rancangan Uji Coba Laju Pembelajaran**

| Laju $\alpha$ | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|---------------|--------------------------------|---|---|---|---|---------|
|               | 1                              | 2 | 3 | 4 | 5 |         |
| 0,1           |                                |   |   |   |   |         |
| 0,2           |                                |   |   |   |   |         |
| 0,3           |                                |   |   |   |   |         |
| 0,4           |                                |   |   |   |   |         |
| 0,5           |                                |   |   |   |   |         |
| 0,6           |                                |   |   |   |   |         |
| 0,7           |                                |   |   |   |   |         |
| 0,8           |                                |   |   |   |   |         |
| 0,9           |                                |   |   |   |   |         |
| 1             |                                |   |   |   |   |         |

#### 4.5.4 Uji Coba Laju Pembelajaran Minimal

Uji coba laju pembelajaran minimal adalah uji coba pada sistem untuk mengetahui pengaruh nilai laju pembelajaran minimal yang digunakan agar dapat menghasilkan akurasi yang optimal. Banyaknya variasi nilai laju pembelajaran minimal yang digunakan adalah 10. Rancangan uji coba laju pembelajaran minimal dapat dilihat pada Tabel 4.13.

**Tabel 4.12 Rancangan Uji Coba Laju Pembelajaran Minimal**

| Laju minimal | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|--------------|--------------------------------|---|---|---|---|---------|
|              | 1                              | 2 | 3 | 4 | 5 |         |
| $10^{-2}$    |                                |   |   |   |   |         |
| $10^{-4}$    |                                |   |   |   |   |         |
| $10^{-6}$    |                                |   |   |   |   |         |
| $10^{-8}$    |                                |   |   |   |   |         |
| $10^{-10}$   |                                |   |   |   |   |         |
| $10^{-11}$   |                                |   |   |   |   |         |

| Laju minimal | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|--------------|--------------------------------|---|---|---|---|---------|
|              | 1                              | 2 | 3 | 4 | 5 |         |
| $10^{-12}$   |                                |   |   |   |   |         |
| $10^{-13}$   |                                |   |   |   |   |         |
| $10^{-14}$   |                                |   |   |   |   |         |
| $10^{-15}$   |                                |   |   |   |   |         |

#### 4.5.5 Uji Coba Laju Pengurang Pembelajaran

Uji coba nilai laju pengurang pembelajaran adalah uji coba pada sistem untuk mengetahui pengaruh nilai laju pengurang pembelajaran terhadap akurasi yang dihasilkan. Banyaknya variasi laju pengurang yang digunakan adalah 10. Rancangan uji coba laju pembelajaran dapat dilihat pada Tabel 4.15.

**Tabel 4.13 Rancangan Uji Coba Laju Pengurang Pembelajaran**

| Laju pengurang | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|----------------|--------------------------------|---|---|---|---|---------|
|                | 1                              | 2 | 3 | 4 | 5 |         |
| 0,1            |                                |   |   |   |   |         |
| 0,2            |                                |   |   |   |   |         |
| 0,3            |                                |   |   |   |   |         |
| 0,4            |                                |   |   |   |   |         |
| 0,5            |                                |   |   |   |   |         |
| 0,6            |                                |   |   |   |   |         |
| 0,7            |                                |   |   |   |   |         |
| 0,8            |                                |   |   |   |   |         |
| 0,9            |                                |   |   |   |   |         |
| 1              |                                |   |   |   |   |         |

#### 4.5.6 Uji Coba Nilai *Cr* dan *Mr*

Uji coba nilai *cross rate*(*cr*) dan *mutation rate*(*mr*) adalah uji coba pada sistem untuk mengetahui kombinasi nilai *cr* dan *mr* yang paling optimal dalam menghasilkan individu baru yang memiliki akurasi yang lebih baik. Banyaknya variasi nilai *cr* dan *mr* yang digunakan adalah 5. Rancangan uji coba nilai *cr* dan *mr* dapat dilihat pada Tabel 4.11.

**Tabel 4.14 Rancangan Uji Coba Nilai *Cr* Dan *Mr***

| Nilai <i>cr</i> | Nilai <i>mr</i> | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|-----------------|-----------------|--------------------------------|---|---|---|---|---------|
|                 |                 | 1                              | 2 | 3 | 4 | 5 |         |
| 1               | 0               |                                |   |   |   |   |         |
| 0,9             | 0,1             |                                |   |   |   |   |         |
| 0,8             | 0,2             |                                |   |   |   |   |         |

| Nilai cr | Nilai mr | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|----------|----------|--------------------------------|---|---|---|---|---------|
|          |          | 1                              | 2 | 3 | 4 | 5 |         |
| 0,7      | 0,3      |                                |   |   |   |   |         |
| 0,6      | 0,4      |                                |   |   |   |   |         |
| 0,5      | 0,5      |                                |   |   |   |   |         |
| 0,4      | 0,6      |                                |   |   |   |   |         |
| 0,3      | 0,7      |                                |   |   |   |   |         |
| 0,2      | 0,8      |                                |   |   |   |   |         |
| 0,1      | 0,9      |                                |   |   |   |   |         |
| 0        | 1        |                                |   |   |   |   |         |

#### 4.5.7 Uji Coba Jumlah Individu (*Popsi*)

Uji coba jumlah individu atau *popsi* adalah uji coba pada sistem untuk mengetahui kombinasi individu yang berisi vektor bobot sehingga bisa diketahui individu yang dihasilkan merupakan individu yang paling optimal. Banyaknya variasi jumlah individu yang akan digunakan adalah 10. Rancangan uji coba kombinasi individu dapat di lihat pada Tabel 4.9.

**Tabel 4.15 Rancangan Uji Coba Jumlah Individu**

| Jumlah individu | Akurasi Percobaan Ke- <i>i</i> |   |   |   |   | Akurasi |
|-----------------|--------------------------------|---|---|---|---|---------|
|                 | 1                              | 2 | 3 | 4 | 5 |         |
| 5               |                                |   |   |   |   |         |
| 8               |                                |   |   |   |   |         |
| 11              |                                |   |   |   |   |         |
| 14              |                                |   |   |   |   |         |
| 17              |                                |   |   |   |   |         |
| 20              |                                |   |   |   |   |         |
| 23              |                                |   |   |   |   |         |
| 25              |                                |   |   |   |   |         |
| 27              |                                |   |   |   |   |         |
| 30              |                                |   |   |   |   |         |

## BAB 5 IMPLEMENTASI

Bab ini akan membahas tentang implementasi perangkat lunak yang telah di analisis sesuai dengan kebutuhan dan percangan yang telah dibuat sebelumnya. Implementasi yang di bahas meliputi implementasi algoritma dan implementasi antarmuka.

### 5.1 Implementasi Algoritma

Optimasi Vektor Bobot pada *Learning Vector Quantization* dengan Algoritma Genetika untuk klasifikasi tingkat resiko penyakit stroke mempunyai beberapa proses utama yang akan dikerjakan seperti yang dijelaskan sebelumnya pada pohon implementasi. Berikut akan dijelaskan implementasi masing-masing proses yang dikerjakan.

#### 5.1.1 Algoritma Membangkitkan Populasi

Pada tahap ini merupakan proses awal setelah melakukan inisialisasi parameter yang diperlukan. Populasi akan dibangkitkan secara acak dimana setiap populasi merupakan vektor bobot yang terdiri dari 3 kelas. Setiap kelas dipilih secara acak dari 200 yang ada sesuai dengan kelas masing-masing hingga terbentuk satu individu. Individu yang telah terbentuk akan disimpan pada sebuah variabel yaitu individu3 dimana data pada variabel tersebut tidak akan berubah sehingga invidu hanya akan berubah ketika dibangkitkan kembali. Gambar 5.1 menjelaskan proses membangkitkan populasi.

```

1  for (x = 0; x < popsize; x++)
2  {
3      for (y = 0; y < 3; y++)
4      {
5          if (y == 0)
6          {
7              angka = Acak.Next(1, 118);
8              for (j = 0; j < 118; j++)
9              {
10                 if (angka == Data[j][0])
11                 {
12                     databaru[0] = Data[j];
13                 }
14             }
15             for (z = 0; z < 5; z++)
16             {
17                 individu3[0, 0, x, y, z] = databaru[0][z + 1];
18             }
19         }
20     }
21     if (y == 1)
22     {
23         angka = Acak.Next(119, 155);
24         for (j = 0; j < 155; j++)
25         {
26             if (angka == Data[j][0])
27             {
28                 databaru[0] = Data[j];

```

```

29         }
30     }
31     for (z = 0; z < 5; z++)
32     {
33         individu3[0, 0, x, y, z] = databaru[0][z + 1];
34     }
35 }
36
37 if (y == 2)
38 {
39     angka = Acak.Next(156, 200);
40     for (j = 0; j < 200; j++)
41     {
42         if (angka == Data[j][0])
43         {
44             databaru[0] = Data[j];
45         }
46     }
47     for (z = 0; z < 5; z++)
48     {
49         individu3[0, 0, x, y, z] = databaru[0][z + 1];
50     }
51 }
52 }
53 }

```

**Gambar 5.1 Implementasi Membangkitkan Populasi**

1. Pada baris ke 7-14 proses untuk mengacak sebuah data pada kelas pertama kemudian akan disimpan untuk dipindahkan ke variabel utama.
2. Pada baris ke 15-18 proses untuk menyimpan data acak sebelumnya ke dalam variabel global untuk digunakan proses berikutnya.

### 5.1.2 Algoritma Pelatihan LVQ

Pada proses pelatihan LVQ vektor bobot akan diukur kedekatannya dengan data latih dimana vektor bobot merupakan individu. Kemudian setiap inidividu akan diperbaiki hingga mencapai batas iterasi atau laju pengurang minimum. Penjelasan pelatihan LVQ ini akan dibagi dua yaitu : proses menghitung dengan menggunakan jarak *euclidean* dan proses *update* bobot

#### 5.1.2.1 Menghitung Jarak *Euclidean*

Pada proses ini untuk mencari jarak terdekat data latih dengan vektor bobot untuk menentukan kelas yang diperbarui sehingga kelas pada bobot bisa mendekati atau menjauhi kelas target. Implementasi proses menghitung jarak *euclidean* dapat dilihat pada Gambar 5.2.

```

1  for (y = 0; y < 3; y++)
2  {
3      for (z = 0; z < 5; z++)
4      {
5          D[x, y, z] = Math.Pow((latih1[x][z + 1] - individu[u,
6              x, id, y, z]), 2);
7          E[x, y] = E[x, y] + D[x, y, z];
8      }
9      E[x, y] = Math.Sqrt(E[x, y]);

```

```

10
11     if ((E[x, y] < minisi && E[x, y] > 0) || y == 0)
12     {
13         minisi = E[x, y];
14         min = y;
15     }
16
17     for (z = 0; z < 5; z++)
18     {
19         if (x < latih1.Count-1)
20         {
21             individu[u, x + 1, id, y, z] = individu[u, x,
22             id, y, z];
23         }
24     }
25 }

```

**Gambar 5.2 Implementasi Menghitung Jarak Euclidean**

1. Pada baris ke 3-9 menghitung akar kuadrat jarak *euclidean* data latih dengan vektor bobot.
2. Pada baris ke 11-15 menentukan jarak terdekat dengan menentukan nilai yang paling minimal(paling rendah).
3. Pada baris 17-24 menyimpan bobot sebelumnya untuk dihitung jaraknya dengan data latih berikutnya

#### 5.1.2.2 Update vektor bobot

Pada proses ini. Vektor bobot sebelumnya yang telah disimpan akan di perbaiki terlebih dahulu sesuai dengan nilai minimal jarak *euclidean* sebelumnya. Bobot yang telah diperbaiki akan dihitung dengan data latih berikutnya dan diperbarui hingga data latih terakhir. Vektor bobot yang di perbarui akan diperbarui mendekati data latih atau menjauhi data latih sesuai dengan jarak minimal yang dihasilkan pada perhitungan jarak *euclidean*. Implementasi proses *update* vektor bobot dapat dilihat pada Gambar 5.3.

```

1  if (min == latih1[x][6])
2  {
3      min = min - 1;
4      for (int z = 0; z < 5; z++)
5      {
6          if (x < latih1.Count - 1)
7          {
8              individu[u, x + 1, id, min, z] = individu[u,
9              x, id, min, z] + lajualpha[u] * (latih1[x][z +
10             1] - individu[u, x, id, min, z]);
11          }
12      }
13  }
14  else
15  {
16      min = min - 1;
17      for (int z = 0; z < 5; z++)
18      {
19          if (x < latih1.Count - 1)
20          {
21

```

|    |                                                  |
|----|--------------------------------------------------|
| 22 | individu[u, x + 1, id, min, z] = individu[u,     |
| 23 | x, id, min, z] - lajua[alpha[u] * (latih1[x][z + |
| 24 | 1] - individu[u, x, id, min, z]);                |
| 25 | }                                                |
| 26 | }                                                |
| 27 | }                                                |
| 28 |                                                  |

**Gambar 5.3 Implementasi Update Vektor Bobot**

1. Pada baris ke 1-13 jika kelas minimal sesuai dengan kelas target maka bobot akan di perbaiki mendekati data latih sesuai dengan langkah 4 pada algoritma LVQ yang telah dijelaskan di Bab 2.
2. Pada baris ke 14-28 hanya dilakukan jika kelas minimal tidak sesuai dengan kelas target sehingga bobot akan di perbaiki menjauhi data latih.

### 5.1.3 Algoritma Crossover

Pada bagian ini membahas proses algoritma genetika operator dengan menggunakan metode *extended intermediate crossover* seperti yang telah dijelaskan pada Bab 2. Pada proses ini akan dipilih 2 individu sebagai *parent* yang akan di *crossover*. Implementasi algoritma ini dapat dilihat pada Gambar 5.4.

|    |                                               |
|----|-----------------------------------------------|
| 1  | for (x = 0; x < jmlcross; x++)                |
| 2  | {                                             |
| 3  | for (y = 0; y < 3; y++)                       |
| 4  | {                                             |
| 5  | for (z = 0; z < 5; z++)                       |
| 6  | {                                             |
| 7  | if (x != jmlcross - 1)                        |
| 8  | {                                             |
| 9  | childc[0, 0, x, y, z] =                       |
| 10 | individu[iterlvq, 0, co[x], y, z] +           |
| 11 | alpha[z] * (individu[iterlvq, 0, co[x +       |
| 12 | 1], y, z] - individu[iterlvq, 0, co[x],       |
| 13 | y, z]);                                       |
| 14 | }                                             |
| 15 | else                                          |
| 16 | childc[0, 0, x, y, z] = individu[iterlvq, 0,  |
| 17 | co[x], y, z] + alpha[z] * (individu[iterlvq,  |
| 18 | 0, co[0], y, z] - individu[iterlvq, 0, co[x], |
| 19 | y, z]);                                       |
| 20 | }                                             |
| 21 | }                                             |
| 22 | }                                             |

**Gambar 5.4 Implementasi Crossover**

### 5.1.4 Algoritma Mutation

Pada proses ini menggunakan metode *random mutation* dimana dipilih satu individu *parent* secara acak kemudian akan diupdate pada salah 1 gennya. Pada tahap awal individu yang akan dimutasi akan disimpan kedalam variabel mutasi kemudian gen yang telah dipilih akan dimutasi. Proses implementasi *mutation* dapat dilihat pada Gambar 5.5.

```

1  for (y = 0; y < 3; y++)
2  {
3      for (z = 0; z < 5; z++)
4      {
5          childm[0, 0, x, y, z] = individu[iterlvq, 0, i, y,
6          z];
7      }
8      childm[0, 0, x, y, m] = individu[iterlvq, 0, i, y, m] + r *
9      (maxdata[m] - mindata[m]);
10 }

```

**Gambar 5.5 Implementasi Mutation**

### 5.1.5 Algoritma Uji Akurasi

Pada tahap ini setiap individu akan di uji dengan data uji untuk mendapatkan akurasi. Proses yang dilakukan hampir sama dengan pelatihan LVQ, hanya saja pada proses ini tidak dilakukan perbaikan vektor bobot. Implementasi uji akurasi dapat dilihat pada Gambar 5.6.

```

1  for (x = 0; x < uji.Count; x++)
2  {
3      for (y = 0; y < 3; y++)
4      {
5          for (z = 0; z < 5; z++)
6          {
7              D[x, y, z] = Math.Pow((uji[x][z + 1] -
8              individu[iterlvq, 0, id, y, z]), 2);
9              ea[x, y] = ea[x, y] + D[x, y, z];
10         }
11         ea[x, y] = Math.Sqrt(ea[x, y]);
12
13         if ((ea[x, y] < minisi && ea[x, y] > 0) || y == 0)
14         {
15             minisi = ea[x, y];
16             mina[x] = y;
17         }
18     }
19     if (uji[x][6] == (mina[x] + 1))
20     {
21         sama++;
22     }
23     else
24     {
25         gaksama++;
26     }
27 }
28 acc[0, id] = Math.Round((sama / (sama + gaksama) * 100), 9);

```

**Gambar 5.6 Implementasi Uji Akurasi**

1. Pada baris ke 5-11 menghitung jarak terdekat dengan menggunakan jarak *euclidean* dengan mengukur jarak data uji dengan masing-masing vektor bobot.
2. Pada baris ke 13-17 menyimpan kelas minimal pada tiap data uji.
3. Pada baris ke 19-28 menyimpan nilai kelas yang sama atau tidak, kemudian akan dihitung tingkat kecocokan yang dihasilkan.

### 5.1.6 Algoritma *Sorting*

Pada tahap ini setiap individu pada *popsize* maupun *offspring* yang di hasilkan akan disimpan kedalam sebuah variabel *combine* untuk dilakukan *sorting* berdasarkan nilai akurasi tertinggi hingga terendah. Berikut implementasi *sorting* dapat di lihat pada Gambar 5.7.

```

1  for (i = 0; i < popsize + jmlcross + jmlmutation - 1; i++)
2  {
3      for (j = i + 1; j < popsize + jmlcross + jmlmutation; j++)
4      {
5          if (combine[0, i, 0] < combine[0, j, 0])
6          {
7              smt[0] = combine[0, i, 0];
7              smt[1] = combine[0, i, 1];
7              smt[2] = combine[0, i, 2];
10             combine[0, i, 0] = combine[0, j, 0];
11             combine[0, i, 1] = combine[0, j, 1];
12             combine[0, i, 2] = combine[0, j, 2];
13             combine[0, j, 0] = smt[0];
14             combine[0, j, 1] = smt[1];
15             combine[0, j, 2] = smt[2];
16         }
17     }
18 }

```

Gambar 5.7 Implementasi *Sorting*

### 5.1.7 Algoritma Menyimpan Individu

Pada tahap ini setiap individu yang telah di *sorting* akan dipilih sejumlah *popsize* untuk melakukan generasi berikutnya. Proses ini akan menggantikan nilai individu diawal jika iterasi algoritma genetika lebih dari satu karena proses menggantikan tersebut hanya dilakukan jika individu yang dihasilkan akan diperbarui kembali. Implementasi untuk menyimpan individu dapat dilihat pada Gambar 5.8.

```

1  for (i = 0; i < popsize; i++)
2  {
3      if (combine[0, i, 1] == 0)
4      {
5          id = Convert.ToInt32(combine[0, i, 2]);
6          for (y = 0; y < 3; y++)
7          {
8              for (z = 0; z < 5; z++)
9              {
10                 individu[0, 0, i, y, z] =
11                 individu[iterlvq, 0, id, y, z];
12             }
13         }
14     }
15     else if (combine[0, i, 1] == 1)
16     {
17         id = Convert.ToInt32(combine[0, i, 2]);
18         for (y = 0; y < 3; y++)
19         {
20             for (z = 0; z < 5; z++)
21             {
22

```

```

23         individu[0, 0, i, y, z] = childc[0, 0,
24         id, y, z];
25     }
26 }
27 }
28 else
29 {
30     id = Convert.ToInt32(combine[0, i, 2]);
31     for (y = 0; y < 3; y++)
32     {
33         for (z = 0; z < 5; z++)
34         {
35             individu[0, 0, i, y, z] = childm[0, 0,
36             id, y, z];
37         }
38     }
39 }
40 }
```

**Gambar 5.8 Implementasi Menyimpan Individu**

1. Pada baris ke 3 merupakan kondisi untuk menentukan bobot tersebut milik *parent*, *offspring crossover* atau *mutation*.
2. Pada baris ke 8-12 meengganti bobot individu(*parent*) sebelumnya untuk menjadi bobot individu untuk generasi berikutnya.
3. Pada baris ke 15-27 hanya dilakukan jika bobot tersebut milik *offspring crossover*.
4. Pada baris ke 28-39 hanya dilakukan jika bobot tersebut milik *offspring mutation*.

## 5.2 Implementasi Antarmuka

Antarmuka Optimasi Vektor Bobot pada *Learning Vector Quantization* dengan Algoritma Genetika untuk klasifikasi tingkat resiko penyakit stroke digunakan oleh pengguna sistem untuk dapat berinteraksi secara langsung dengan sistem. Antarmuka dari sistem ini terdiri dari empat bagian yaitu antarmuka parameter dan data, antarmuka hasil, halaman hasil akurasi dan *sorting* dan antarmuka grafik akurasi.

### 5.2.1 Antarmuka Parameter dan Data

Gambar 5.9 ini merupakan tampilan dari halaman parameter dan data. Halaman berisi form untuk memberi masukkan parameter pada parameter GA-LVQ. Parameter tersebut antara lain yaitu jumlah *popsi*, jumlah iterasi, nilai *cross rate* dan *mutation rate*, jumlah iterasi LVQ, laju pembelajaran  $\alpha$ , laju pembelajaran minimal dan juga laju pengurang pembelajaran. Setiap parameter akan menjadi batasan untuk mendapatkan bobot yang terbaik.

| No | Umur | KolesterolTotal | HDL  | LDL   | Trigliserida |
|----|------|-----------------|------|-------|--------------|
| 1  | 44   | 174             | 36,5 | 123,3 | 71           |
| 2  | 60   | 169             | 36,9 | 110,1 | 110          |
| 3  | 23   | 196             | 40,3 | 137,1 | 93           |
| 4  | 41   | 213             | 47,4 | 143,2 | 112          |
| 5  | 40   | 253             | 46,5 | 179,3 | 136          |
| 6  | 48   | 197             | 30,5 | 120,5 | 79           |
| 7  | 49   | 171             | 40,1 | 104,1 | 134          |
| 8  | 40   | 198             | 40,5 | 139,5 | 90           |
| 9  | 49   | 184             | 38,4 | 119,2 | 133          |

Gambar 5.9 Antarmuka Parameter dan Data

### 5.2.2 Antarmuka Hasil

Gambar 5.10 merupakan halaman hasil, dimana *datagriedview* yang atas berisi individu hasil pelatihan LVQ sedangkan yang bawah berisi *offspring* dari *crossover* serta *mutation*.

| Column1           | Column2           | Column3           | Column4           | Column5           |
|-------------------|-------------------|-------------------|-------------------|-------------------|
| 20,31768678931... | 196,7875631564... | 48,17003554503... | 154,9284752643... | 158,7475138676... |
| 50,03045652232... | 232,7395587301... | 40,15767404268... | 129,1300200932... | 280,6467162188... |
| 53,80147171485... | 204,6094771477... | 40,21858925580... | 140,7900638838... | 104,8020506385... |
| 20,31768678931... | 196,7875631564... | 48,17003554503... | 154,9284752643... | 158,7475138676... |
| 50,03044992740... | 232,7393747969... | 40,15763207357... | 129,1294387811... | 280,6478704798... |
| Bobot             | Ke                | 1                 |                   |                   |
| 53,80147171485... | 204,6094771477... | 40,21858925580... | 140,7900638838... | 104,8020506385... |
| 20,31768678931... | 196,7875631564... | 48,17003554503... | 154,9284752643... | 158,7475138676... |
| 50,03048813252... | 232,7395492649... | 40,15764514472... | 129,1297054727... | 280,6475783335... |

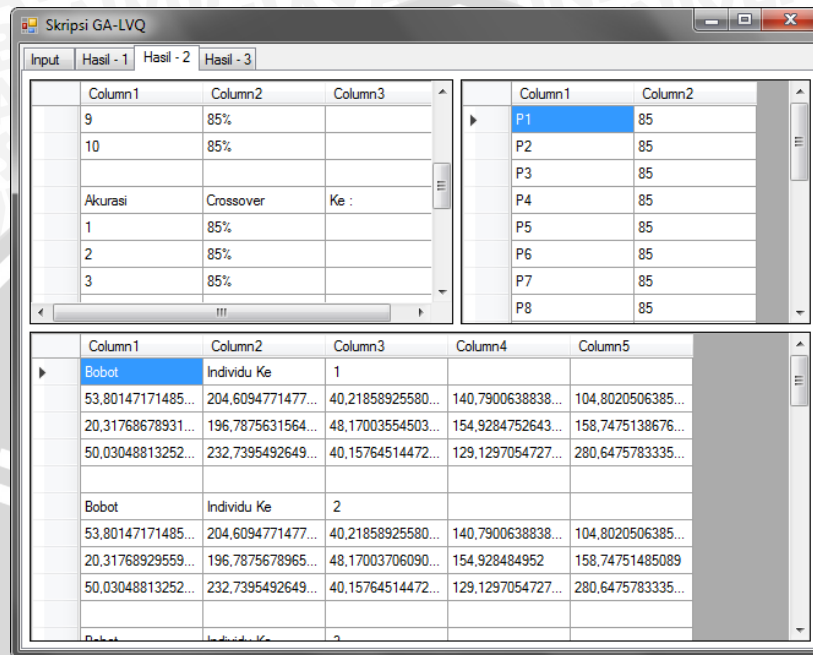
  

| Column1           | Column2            | Column3            | Column4           | Column5           |
|-------------------|--------------------|--------------------|-------------------|-------------------|
| Crossover         | Ke                 | 1                  |                   |                   |
| 53,80147171485... | 204,6094771477...  | 40,21858925580...  | 140,7900638838... | 104,8020506385... |
| 20,31768926847... | 196,78756428722... | 48,170034544344... | 154,9284857254... | 158,7475132606... |
| 50,03048813252... | 232,7395492649...  | 40,15764514472...  | 129,1297054727... | 280,6475783335... |
| Crossover         | Ke                 | 2                  |                   |                   |
| 53,80147171485... | 204,6094771477...  | 40,21858925580...  | 140,7900638838... | 104,8020506385... |
| 20,31768829373... | 196,7875627029...  | 48,170032762075... | 154,9284763127... | 158,7475420892... |
| 50,03048813252... | 232,7395492649...  | 40,15764514472...  | 129,1297054727... | 280,6475783335... |

Gambar 5.10 Antarmuka Hasil

### 5.2.3 Antarmuka Akurasi dan Sorting

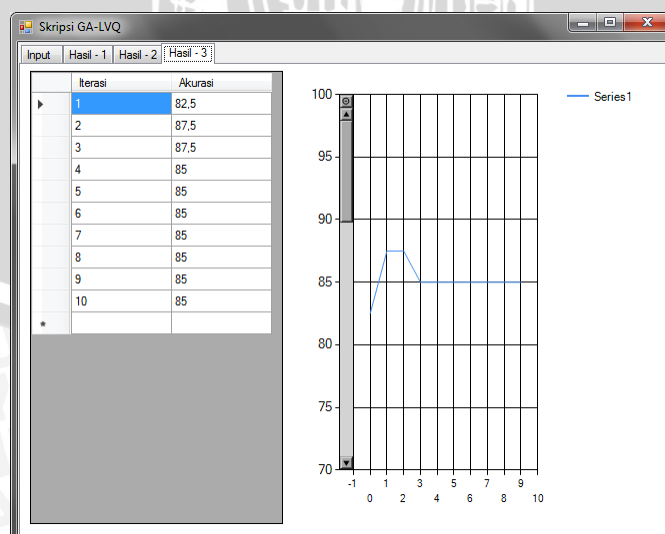
Gambar 5.11 merupakan halaman hasil akurasi dan hasil *sorting*. Dimana pada halaman ini berisi tiga *datagridview*, pada *griedview* kiri atas akan menampilkan seluruh akurasi yang dihasilkan, pada *griedview* kanan atas akan menampilkan akurasi yang telah diseleksi, sedangkan pada *griedview* yang bawah akan menampilkan individu baru yang telah diseleksi.



Gambar 5.11 Antarmuka Akurasi dan Sorting

### 5.2.4 Antarmuka Grafik Akurasi

Gambar 5.12 merupakan halaman yang berisi grafik serta nilai akurasi tertinggi pada tiap iterasi. Hasil akurasi tertinggi tersebut ditampilkan kedalam grafik akurasi. Tampilan halaman grafik akurasi dapat dilihat seperti berikut.



Gambar 5.12 Antarmuka Grafik Akurasi

## BAB 6 PENGUJIAN DAN ANALISIS

Pada bab ini dibahas mengenai proses pengujian metode GA-LVQ. Proses pengujian dilakukan menggunakan pengujian nilai akurasi. Pengujian nilai akurasi digunakan untuk mengukur tingkat akurasi dari sistem dengan membandingkan hasil dari keluaran sistem dan hasil dari Laboratorium.

### 6.1 Pengujian Jumlah Iterasi Algoritma Genetika

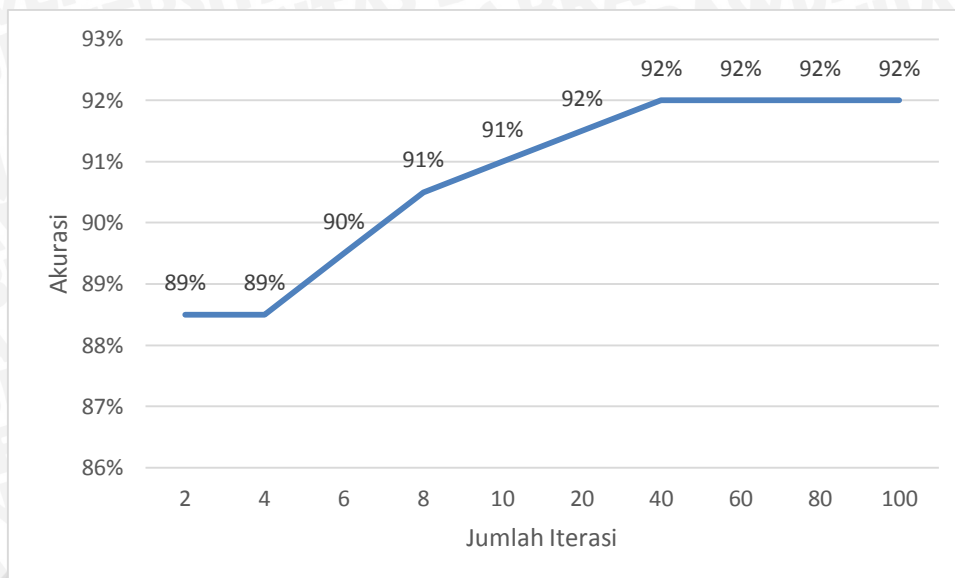
Pengujian jumlah iterasi algoritma genetika akan digunakan untuk mencari jumlah iterasi terbaik untuk menghasilkan individu yang terbaik untuk melakukan klasifikasi. Jumlah iterasi yang akan dilakukan adalah 2, 4, 6, 8, 10, 20, 40, 60, 80, 100. Pada tiap percobaan digunakan data latih sebanyak 160 data dan data uji sebanyak 40 data seperti pada Tabel 6.1. Dengan parameter yang lain telah ditentukan sebagai berikut :

1. Jumlah *popsize* : 10
2. *cr* : 0,1
3. *mr* : 0,9
4. Jumlah iterasi LVQ : 7
5. Laju pembelajaran  $\alpha$  : 0,1
6. Laju pembelajaran min : 0,0000000000000001
7. Laju pengurang pemb : 0,2

**Tabel 6.1 Hasil Pengujian Jumlah Iterasi**

| Jumlah Iterasi | Akurasi Percobaan Ke- <i>i</i> |       |       |       |       | Akurasi |
|----------------|--------------------------------|-------|-------|-------|-------|---------|
|                | 1                              | 2     | 3     | 4     | 5     |         |
| 2              | 100.0%                         | 87.5% | 85.0% | 82.5% | 87.5% | 88.5%   |
| 4              | 100.0%                         | 87.5% | 85.0% | 82.5% | 87.5% | 88.5%   |
| 6              | 100.0%                         | 87.5% | 87.5% | 85.0% | 87.5% | 89.5%   |
| 8              | 100.0%                         | 90.0% | 87.5% | 87.5% | 87.5% | 90.5%   |
| 10             | 100.0%                         | 92.5% | 87.5% | 87.5% | 87.5% | 91.0%   |
| 20             | 100.0%                         | 92.5% | 87.5% | 87.5% | 90.0% | 91.5%   |
| 40             | 100.0%                         | 92.5% | 90.0% | 87.5% | 90.0% | 92.0%   |
| 60             | 100.0%                         | 92.5% | 90.0% | 87.5% | 90.0% | 92.0%   |
| 80             | 100.0%                         | 92.5% | 90.0% | 87.5% | 90.0% | 92.0%   |
| 100            | 100.0%                         | 92.5% | 90.0% | 87.5% | 90.0% | 92.0%   |

Pada Gambar 6.1 grafik menunjukkan akurasi mengalami peningkatan ketika iterasi semakin banyak. Dimana perbaikan akurasi terus mengalami peningkatan hingga mencapai iterasi 40 yang kemudian nilainya mulai menjadi konstan.



**Gambar 6.1 Grafik Pengujian Iterasi Algoritma Genetika**

Pada Gambar 6.1 menunjukkan bahwa pada iterasi 2 hingga 4 tidak terjadi perubahan akurasi, perubahan akurasi baru mulai mengalami peningkatan saat mencapai iterasi 6 dan terus mengalami kenaikan hingga akhirnya nilainya mulai konvergen saat mencapai iterasi 40. Iterasi algoritma genetika berpengaruh untuk menghasilkan individu baru saat algoritma genetika operator dilakukan. Pengujian ini menunjukkan semakin besar iterasi yang dilakukan maka akan dihasilkan akurasi yang semakin baik pula.

## 6.2 Pengujian Iterasi Laju Pembelajaran LVQ

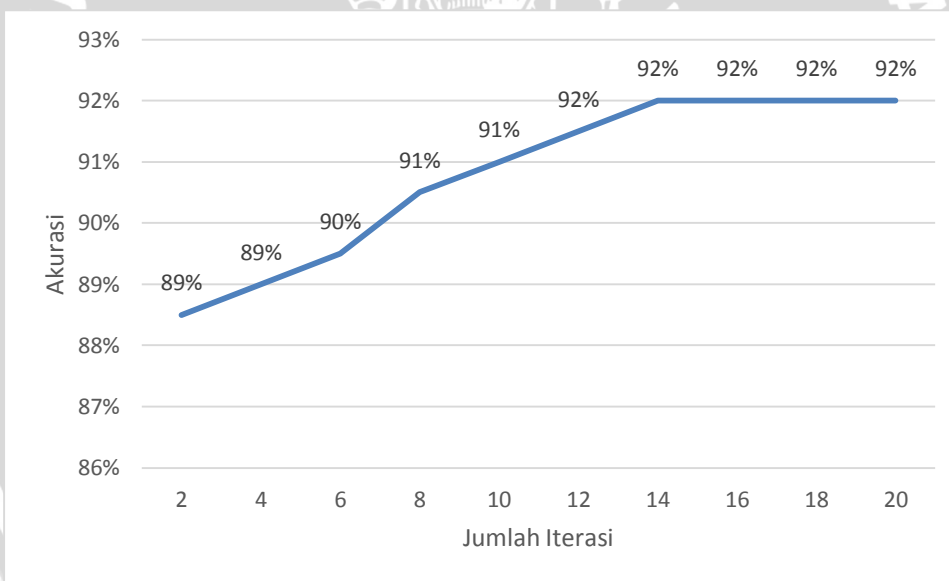
Pengujian jumlah iterasi laju pembelajaran akan digunakan untuk mencari jumlah iterasi terbaik untuk menghasilkan individu yang terbaik setelah mengalami proses pelatihan pada LVQ. Jumlah pengujian iterasi yang akan dilakukan adalah 2, 4, 6, 8, 10, 12, 14, 16, 18, 20. Pada tiap percobaan digunakan data latih sebanyak 160 data dan data uji sebanyak 40 data seperti pada Tabel 6.2. Dengan parameter yang lain telah ditentukan sebagai berikut :

1. Jumlah *popsize* : 10
2. *cr* : 0,1
3. *mr* : 0,9
4. Jumlah iterasi GA : 40
5. Laju pembelajaran  $\alpha$  : 0,1
6. Laju pembelajaran min : 0,0000000000000001
7. Laju pengurang pemb : 0,2

**Tabel 6.2 Hasil Pengujian Iterasi Laju Pembelajaran**

| Jumlah Iterasi | Akurasi Percobaan Ke- <i>i</i> |       |       |       |       | Akurasi |
|----------------|--------------------------------|-------|-------|-------|-------|---------|
|                | 1                              | 2     | 3     | 4     | 5     |         |
| 2              | 92.5%                          | 90.0% | 85.0% | 85.0% | 90.0% | 88.5%   |
| 4              | 95.0%                          | 90.0% | 85.0% | 85.0% | 90.0% | 89.0%   |
| 6              | 95.0%                          | 90.0% | 85.0% | 87.5% | 90.0% | 89.5%   |
| 8              | 95.0%                          | 92.5% | 87.5% | 87.5% | 90.0% | 90.5%   |
| 10             | 95.0%                          | 92.5% | 87.5% | 87.5% | 92.5% | 91.0%   |
| 12             | 97.5%                          | 92.5% | 87.5% | 87.5% | 92.5% | 91.5%   |
| 14             | 97.5%                          | 92.5% | 90.0% | 87.5% | 92.5% | 92.0%   |
| 16             | 97.5%                          | 92.5% | 90.0% | 87.5% | 92.5% | 92.0%   |
| 18             | 97.5%                          | 92.5% | 90.0% | 87.5% | 92.5% | 92.0%   |
| 20             | 97.5%                          | 92.5% | 90.0% | 87.5% | 92.5% | 92.0%   |

Pada Gambar 6.2 grafik menunjukkan akurasi mengalami peningkatan ketika iterasi semakin banyak. Dimana perbaikan akurasi terus mengalami peningkatan hingga mencapai iterasi 14 yang kemudian nilainya mulai menjadi konstan.



**Gambar 6.2 Grafik Pengujian Jumlah Iterasi Laju Pembelajaran LVQ**

Pada Gambar 6.2 sejak iterasi 2 menuju 4 sudah mengalami kenaikan meskipun peningkatan yang terjadi tidak terlalu signifikan. Walau jarak jumlah iterasi tidak terlalu besar akan tetapi selalu terjadi peningkatan hingga mencapai iterasi 14. Dimana hal tersebut menunjukkan bahwa iterasi pelatihan pada LVQ, cukup berpengaruh pada saat proses pelatihan. Hal ini dibuktikan dengan jumlah iterasi yang tidak terlalu besar akan tetapi mampu menghasilkan akurasi yang cukup baik.

### 6.3 Pengujian Laju Pembelajaran

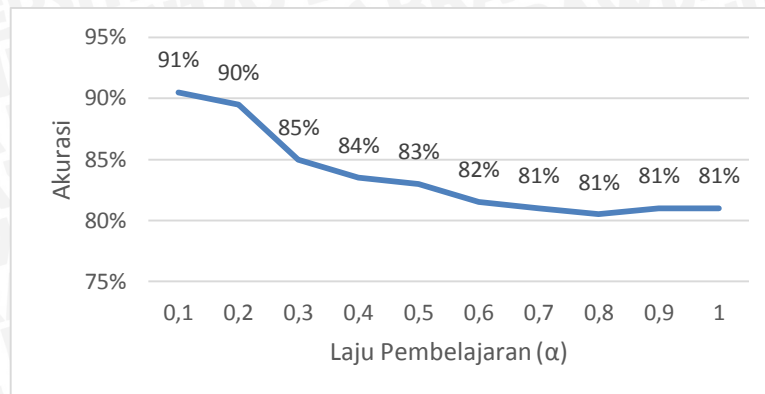
Pengujian laju pembelajaran digunakan untuk mencari nilai laju pembelajaran yang terbaik agar mampu memperbaiki individu sehingga mampu menghasilkan hasil klasifikasi yang lebih baik. Pengujian laju pembelajaran yang akan dilakukan adalah 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. Pada tiap percobaan digunakan data latih sebanyak 160 data dan data uji sebanyak 40 data seperti pada Tabel 6.3. Dengan parameter yang lain telah ditentukan sebagai berikut :

1. Jumlah *popsize* : 10
2. *cr* : 0,1
3. *mr* : 0,9
4. Jumlah iterasi LVQ : 14
5. Jumlah iterasi GA : 40
6. Laju pembelajaran min : 0,0000000000000001
7. Laju pengurang pemb : 0,2

**Tabel 6.3 Hasil Laju Pembelajaran**

| Laju $\alpha$ | Akurasi Percobaan Ke- <i>i</i> |       |       |       |       | Akurasi |
|---------------|--------------------------------|-------|-------|-------|-------|---------|
|               | 1                              | 2     | 3     | 4     | 5     |         |
| 0,1           | 97.5%                          | 92.5% | 85.0% | 85.0% | 92.5% | 90.5%   |
| 0,2           | 100.0%                         | 90.0% | 90.0% | 80.0% | 87.5% | 89.5%   |
| 0,3           | 90.0%                          | 80.0% | 85.0% | 80.0% | 90.0% | 85.0%   |
| 0,4           | 85.0%                          | 80.0% | 82.5% | 80.0% | 90.0% | 83.5%   |
| 0,5           | 82.5%                          | 80.0% | 85.0% | 80.0% | 87.5% | 83.0%   |
| 0,6           | 82.5%                          | 77.5% | 85.0% | 80.0% | 82.5% | 81.5%   |
| 0,7           | 82.5%                          | 77.5% | 82.5% | 80.0% | 82.5% | 81.0%   |
| 0,8           | 80.0%                          | 77.5% | 82.5% | 80.0% | 82.5% | 80.5%   |
| 0,9           | 82.5%                          | 77.5% | 82.5% | 80.0% | 82.5% | 81.0%   |
| 1             | 82.5%                          | 77.5% | 82.5% | 80.0% | 82.5% | 81.0%   |

Pada Gambar 6.3 grafik menunjukkan penurunan akurasi ketika laju pembelajaran semakin besar dimana laju pembelajaran digunakan untuk memperbaiki bobot(individu) pada saat pelatihan LVQ. Hanya laju pembelajaran yang paling rendah yang mendapatkan akurasi paling tinggi sekitar 91% yaitu pada nilai 0,1.



**Gambar 6.3 Grafik Pengujian Laju Pembelajaran**

Pada Gambar 6.3 menunjukkan nilai akurasi yang mengalami penurunan. Dari hasil tersebut diketahui bahwa semakin besar nilai laju pembelajaran mengakibatkan semakin rendah tingkat akurasinya. Pada LVQ, laju pembelajaran akan berpengaruh pada saat perbaikan nilai bobot sesuai dengan persamaan yang digunakan untuk memperbarui nilai bobot dalam algoritma pelatihan LVQ. Jadi, semakin besar nilai laju pembelajaran maka semakin besar langkah pembelajaran yang dilakukan sehingga akan sulit untuk mencapai konvergensi pada perubahan bobot yang dihasilkan.

#### 6.4 Pengujian Laju Pembelajaran Minimal

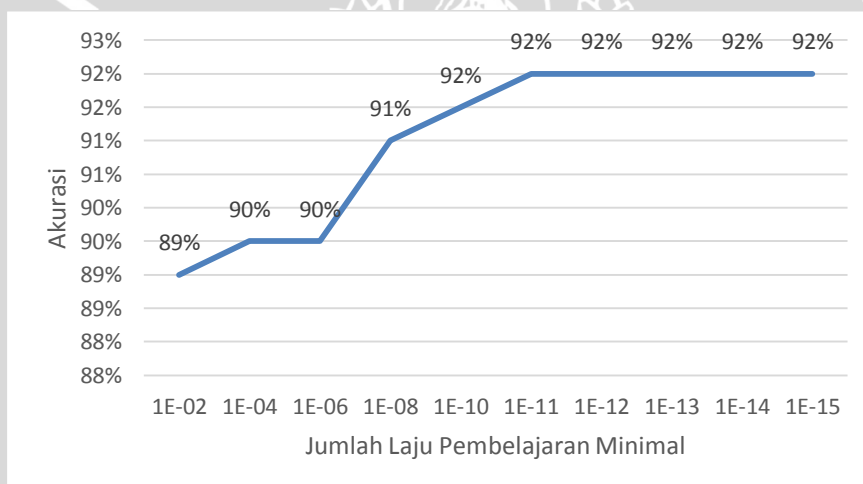
Pengujian laju pembelajaran minimal digunakan untuk mencari nilai laju pembelajaran minimal yang tedekat dimana laju pembelajaran minimal akan berfungsi seperti iterasi laju pembelajaran agar mampu memperbaiki individu sehingga mampu menghasilkan hasil klasifikasi yang lebih baik. Pengujian laju pembelajaran minimal yg digunakan adalah  $10^{-2}$ ,  $10^{-4}$ ,  $10^{-6}$ ,  $10^{-8}$ ,  $10^{-10}$ ,  $10^{-11}$ ,  $10^{-12}$ ,  $10^{-13}$ ,  $10^{-14}$ ,  $10^{-15}$ . Pada tiap percobaan digunakan data latih sebanyak 160 data dan data uji sebanyak 40 data seperti pada Tabel 6.4. Dengan parameter yang lain telah ditentukan sebagai berikut :

1. Jumlah *popsize* : 10
2. *cr* : 0,1
3. *mr* : 0,9
4. Jumlah iterasi LVQ : 7
5. Laju pembelajaran  $\alpha$  : 0,1
6. Jumlah iterasi GA : 40
7. Laju pengurang pemb : 0,2

**Tabel 6.4 Hasil Laju Pembelajaran Minimal**

| Laju pemb minimal | Akurasi Percobaan Ke- <i>i</i> |       |       |       |       | Akurasi 2 |
|-------------------|--------------------------------|-------|-------|-------|-------|-----------|
|                   | 1                              | 2     | 1     | 2     | 1     |           |
| $10^{-2}$         | 95.0%                          | 90.0% | 85.0% | 85.0% | 90.0% | 89.0%     |
| $10^{-4}$         | 97.5%                          | 90.0% | 85.0% | 85.0% | 90.0% | 89.5%     |
| $10^{-6}$         | 97.5%                          | 90.0% | 85.0% | 85.0% | 90.0% | 89.5%     |
| $10^{-8}$         | 97.5%                          | 92.5% | 87.5% | 85.0% | 92.5% | 91.0%     |
| $10^{-10}$        | 97.5%                          | 92.5% | 87.5% | 87.5% | 92.5% | 91.5%     |
| $10^{-11}$        | 97.5%                          | 92.5% | 87.5% | 87.5% | 95.0% | 92.0%     |
| $10^{-12}$        | 97.5%                          | 92.5% | 87.5% | 87.5% | 95.0% | 92.0%     |
| $10^{-13}$        | 97.5%                          | 92.5% | 87.5% | 87.5% | 95.0% | 92.0%     |
| $10^{-14}$        | 97.5%                          | 92.5% | 87.5% | 87.5% | 95.0% | 92.0%     |
| $10^{-15}$        | 97.5%                          | 92.5% | 87.5% | 87.5% | 95.0% | 92.0%     |

Pada Gambar 6.4 grafik menunjukkan terjadinya peningkatan akurasi saat laju pembelajaran minimum mencapai jumlah tertentu. Jumlah pengintkatan mulai mengalami konvergen pada saat mencapai laju pembelajaran dengan nilai  $10^{-11}$ .



**Gambar 6.4 Pengujian Laju pembelajaran Minimal**

Akurasi pada Gambar 6.4 mengalami kenaikan ketika jumlah laju pembelajaran minimal semakin kecil. Laju pembelajaran minimal juga merupakan salah satu kondisi berhenti ketika pelatihan pada LVQ. Semakin banyak iterasi maka semakin kecil pula nilai laju pembelajaran, laju pembelajaran minimum yang makin kecil menunjukkan bahwa iterasi semakin banyak dan akurasi juga semakin baik.

## 6.5 Pengujian Laju Pengurang Pembelajaran

Pengujian laju pengurang pembelajaran digunakan untuk mencari nilai laju pengurang pembelajaran yang paling optimal sehingga dapat menghasilkan akurasi yang lebih baik serta akan menentukan batasan iterasi pula jika mencapai nilai tertentu. Laju pengurang pembelajaran yang diuji yaitu 0.1, 0.2, 0.3, 0.4, 0.5,

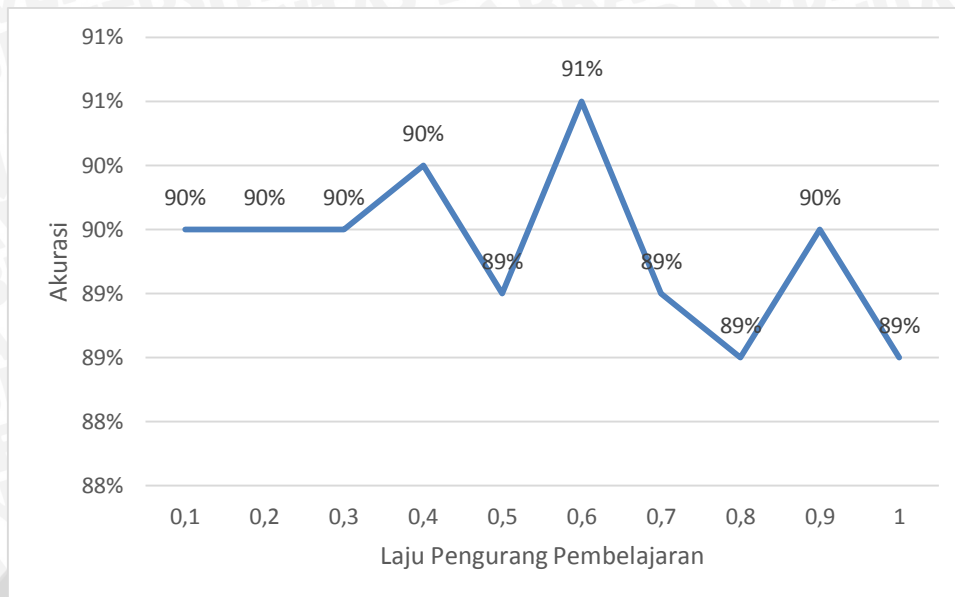
0.6, 0.7, 0.8, 0.9, 1. Pada tiap percobaan digunakan data latih sebanyak 160 data dan data uji sebanyak 40 data seperti pada Tabel 6.5. Dengan parameter yang lain telah ditentukan sebagai berikut :

1. Jumlah *popsize* : 10
2. *cr* : 0,1
3. *mr* : 0,9
4. Jumlah iterasi LVQ : 14
5. Laju pembelajaran  $\alpha$  : 0,1
6. Laju pembelajaran min : 0,000000000001
7. Jumlah Iterasi GA : 40

**Tabel 6.5 Hasil Pengujian Laju Pengurang Pembelajaran**

| Laju pengurang | Akurasi Percobaan Ke- <i>i</i> |       |       |       |       | Akurasi |
|----------------|--------------------------------|-------|-------|-------|-------|---------|
|                | 1                              | 2     | 3     | 4     | 5     |         |
| 0,1            | 95.0%                          | 90.0% | 85.0% | 85.0% | 92.5% | 89.5%   |
| 0,2            | 95.0%                          | 92.5% | 85.0% | 85.0% | 90.0% | 89.5%   |
| 0,3            | 95.0%                          | 90.0% | 85.0% | 85.0% | 92.5% | 89.5%   |
| 0,4            | 92.5%                          | 90.0% | 90.0% | 87.5% | 90.0% | 90.0%   |
| 0,5            | 90.0%                          | 90.0% | 85.0% | 87.5% | 92.5% | 89.0%   |
| 0,6            | 90.0%                          | 90.0% | 90.0% | 90.0% | 92.5% | 90.5%   |
| 0,7            | 90.0%                          | 87.5% | 87.5% | 90.0% | 90.0% | 89.0%   |
| 0,8            | 90.0%                          | 87.5% | 85.0% | 90.0% | 90.0% | 88.5%   |
| 0,9            | 90.0%                          | 87.5% | 90.0% | 87.5% | 92.5% | 89.5%   |
| 1              | 95.0%                          | 90.0% | 80.0% | 85.0% | 92.5% | 88.5%   |

Pada Gambar 6.5 grafik menunjukkan akurasi yang tidak stabil ketika laju pengurang pembelajaran semakin besar dimana laju pengurang pembelajaran digunakan untuk memperbaiki bobot(individu) dan mempengaruhi nilai laju pembelajaran pada tiap iterasi pada saat pelatihan LVQ. Akurasi terbaik didapatkan ketika laju pengurang pembelajaran pada nilai 0,6.



**Gambar 6.5 Grafik Pengujian Laju Pengurang Pembelajaran**

Grafik pada Gambar 6.5 mengalami nilai yang stabil saat nilai pengurang pembelajaran dari 0,1-0,3 akan tetapi mulai tidak stabil ketika nilai laju pengurang semakin besar. Laju pengurang mempengaruhi besar kecilnya pergeseran nilai laju pembelajaran saat pelatihan LVQ. Grafik yang tidak stabil bisa dipengaruhi oleh perbedaan data uji yang digunakan pada tiap percobaan.

## 6.6 Pengujian Nilai $Cr$ dan $Mr$

Pengujian nilai  $cr$  dan  $mr$  digunakan untuk mencari kombinasi nilai  $cr$  dan  $mr$  yang terbaik agar mampu menghasilkan *offspring* yang lebih baik. Pada tiap percobaan digunakan data latih sebanyak 160 data dan data uji sebanyak 40 data. Dengan parameter yang lain telah ditentukan sebagai berikut :

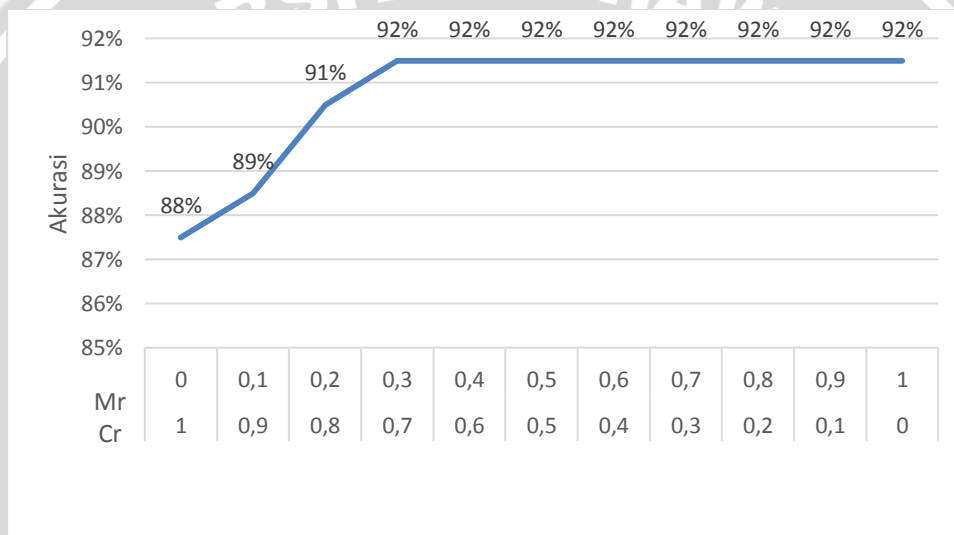
1. Jumlah *popsize* : 10
2. Jumlah iterasi GA : 40
3. Jumlah iterasi LVQ : 14
4. Laju pembelajaran  $\alpha$  : 0,1
5. Laju pembelajaran min : 0,000000000001
6. Laju pengurang pemb : 0,6

**Tabel 6.6 Pengujian Nilai  $Cr$  dan  $Mr$**

| Nilai $C_r$ | Nilai $M_r$ | Akurasi Percobaan Ke- $i$ |       |       |       |       | Akurasi |
|-------------|-------------|---------------------------|-------|-------|-------|-------|---------|
|             |             | 1                         | 2     | 3     | 4     | 5     |         |
| 1           | 0           | 90.0%                     | 90.0% | 85.0% | 85.0% | 87.5% | 87.5%   |
| 0,9         | 0,1         | 90.0%                     | 90.0% | 85.0% | 87.5% | 90.0% | 88.5%   |
| 0,8         | 0,2         | 90.0%                     | 90.0% | 92.5% | 90.0% | 90.0% | 90.5%   |
| 0,7         | 0,3         | 90.0%                     | 92.5% | 92.5% | 90.0% | 92.5% | 91.5%   |
| 0,6         | 0,4         | 90.0%                     | 92.5% | 92.5% | 90.0% | 92.5% | 91.5%   |

| Nilai $C_r$ | Nilai $M_r$ | Akurasi Percobaan Ke- $i$ |       |       |       |       | Akurasi |
|-------------|-------------|---------------------------|-------|-------|-------|-------|---------|
|             |             | 1                         | 2     | 3     | 4     | 5     |         |
| 0,5         | 0,5         | 90.0%                     | 92.5% | 92.5% | 90.0% | 92.5% | 91.5%   |
| 0,4         | 0,6         | 90.0%                     | 92.5% | 92.5% | 90.0% | 92.5% | 91.5%   |
| 0,3         | 0,7         | 90.0%                     | 92.5% | 92.5% | 90.0% | 92.5% | 91.5%   |
| 0,2         | 0,8         | 90.0%                     | 92.5% | 92.5% | 90.0% | 92.5% | 91.5%   |
| 0,1         | 0,9         | 90.0%                     | 92.5% | 92.5% | 90.0% | 92.5% | 91.5%   |
| 0           | 1           | 90.0%                     | 92.5% | 92.5% | 90.0% | 92.5% | 91.5%   |

Pada Gambar 6.6 grafik menunjukkan peningkatan akurasi ketika nilai *cross rate* semakin rendah dan *mutation rate* semakin tinggi. Dimana semakin banyak *offspring mutation* yang dihasilkan maka akurasi juga semakin baik. Dapat dilihat akurasi mulai konvergen saat *cross rate* dan *mutation rate* pada nilai 0,7 dan 0,3.



**Gambar 6.6 Grafik Pengujian Nilai *Cross rate* dan *Mutation rate***

Grafik pada Gambar 6.6 menunjukkan bahwa *mr* lebih berpengaruh ketika jumlah *mr* semakin besar dan *cr* semakin kecil. Hal tersebut ditunjukkan saat *mr* 0 hingga 0,3 kemudian nilainya mulai konvergen. Proses mutasi terjadi hanya pada salah satu sel, akan tetapi memberikan hasil yang lebih baik dibandingkan *crossover*.

## 6.7 Pengujian Jumlah Individu (*Popsi*ze)

Pengujian jumlah individu digunakan untuk mencari jumlah serta kombinasi individu yang dapat menghasilkan akurasi yang terbaik. Setiap individu dirandom sesuai dengan *popsi*ze yang telah ditentukan. Pada pengujian ini *popsi*ze yang digunakan adalah 5, 10, 15, 20, 25, 30, 35, 40, 50, 100. Pada tiap percobaan digunakan data latih sebanyak 160 data dan data uji sebanyak 40 data seperti pada Tabel 6.7. Dengan parameter yang lain telah ditentukan sebagai berikut :

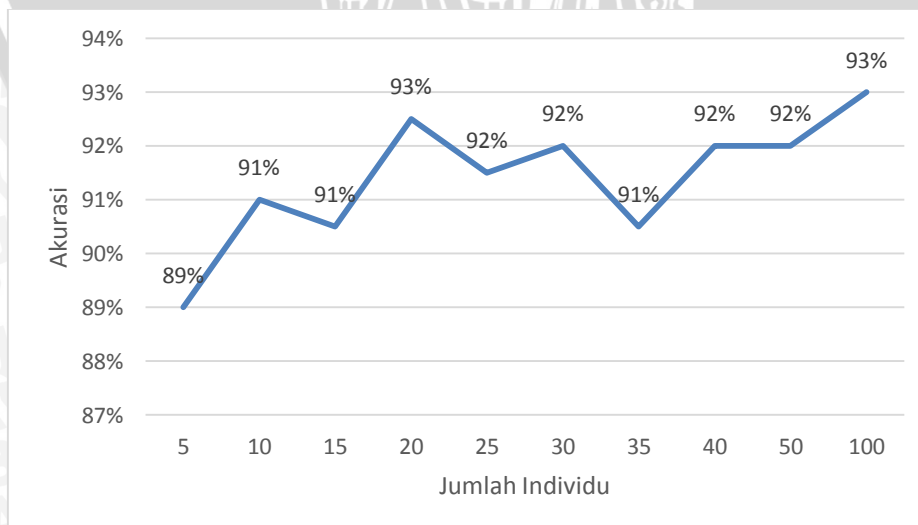
1. Jumlah iterasi GA : 40

2.  $cr$  : 0,3
3.  $mr$  : 0,7
4. Jumlah iterasi LVQ : 14
5. Laju pembelajaran  $\alpha$  : 0,1
6. Laju pembelajaran min : 0,000000000001
7. Laju pengurang pemb : 0,6

**Tabel 6.7 Hasil Pengujian Jumlah Individu**

| <i>Popsi</i> | Akurasi Percobaan Ke- <i>i</i> |       |       |       |       | Akurasi |
|--------------|--------------------------------|-------|-------|-------|-------|---------|
|              | 1                              | 2     | 3     | 4     | 5     |         |
| 5            | 90.0%                          | 87.5% | 90.0% | 87.5% | 90.0% | 89.0%   |
| 10           | 90.0%                          | 90.0% | 92.5% | 90.0% | 92.5% | 91.0%   |
| 15           | 90.0%                          | 90.0% | 92.5% | 90.0% | 90.0% | 90.5%   |
| 20           | 97.5%                          | 90.0% | 92.5% | 90.0% | 92.5% | 92.5%   |
| 25           | 90.0%                          | 92.5% | 92.5% | 90.0% | 92.5% | 91.5%   |
| 30           | 92.5%                          | 92.5% | 92.5% | 90.0% | 92.5% | 92.0%   |
| 35           | 95.0%                          | 87.5% | 90.0% | 87.5% | 92.5% | 90.5%   |
| 40           | 92.5%                          | 92.5% | 92.5% | 90.0% | 92.5% | 92.0%   |
| 50           | 95.0%                          | 92.5% | 92.5% | 90.0% | 90.0% | 92.0%   |
| 100          | 97.5%                          | 92.5% | 92.5% | 90.0% | 92.5% | 93.0%   |

Pada Gambar 6.7 grafik menunjukkan akurasi mengalami naik turun yang tidak stabil yang disebabkan oleh *random* individu yang tidak menentu akurasinya. Hasil *random* setiap individu yang dihasilkan akan sangat berpengaruh dengan nilai akurasinya. Semakin banyak *popsi* maka semakin banyak kombinasi individu yang dihasilkan sehingga akurasi akan semakin baik. Akurasi terbaik didapatkan saat *popsi* berjumlah 100.



**Gambar 6.7 Grafik Pengujian Jumlah Individu**

## BAB 7 PENUTUP

Bagian ini memuat kesimpulan dan saran dari penelitian optimasi vektor bobot pada LVQ dengan menggunakan algoritma genetika. Dengan penjelasan sebagai berikut:

### 7.1 Kesimpulan

Berdasarkan pada hasil perancangan, implementasi, dan pengujian yang telah dilakukan maka kesimpulan yang didapat adalah didapatkan kesimpulan sebagai berikut.

1. Pada penelitian ini implementasi vektor bobot pada *Learning Vector Quantization* dengan algoritma genetika untuk klasifikasi tingkat resiko penyakit stroke dapat dilakukan dengan menentukan inisialisasi parameter yang diperlukan seperti : data latih, jumlah iterasi, laju  $\alpha$ , laju minimal, laju pengurang, nilai  $cr$  dan  $mr$ , dan individu. Individu merupakan vektor bobot yang nantinya akan digunakan untuk klasifikasi hingga didapatkan vektor bobot yang paling optimal. Langkah awal adalah dengan membangkitkan populasi sejumlah *popsiz*, setiap individu tersebut akan dilatih dengan data latih menggunakan metode LVQ yang akan menghasilkan individu baru yang telah diperbaiki. Individu yang telah mengalami perbaikan tersebut kemudian akan dioptimasi dengan menggunakan algoritma genetika operator yaitu *crossover* dan *mutation* hingga dihasilkan *offspring* sesuai dengan *cross rate* dan *mutation rate*. Kemudian setiap individu awal(sebelum diperbaiki pada LVQ), individu baru(hasil perbaikan pada LVQ), dan *offspring* yang dihasilkan dari *crossover* dan *mutation* akan di uji akurasi dengan data uji untuk melakukan klasifikasi dengan menggunakan metode LVQ tanpa melakukan proses perbaikan bobot. Dari tiap akurasi yang dihasilkan akan menjadi nilai fitness yang kemudian akan diseleksi untuk iterasi berikutnya hingga mencapai batas berhenti.
2. Berdasarkan hasil pengujian parameter LVQ dan algoritma genetika dapat disimpulkan bahwa akurasi terbaik sebesar 93% dengan detail parameter yang optimal yaitu jumlah iterasi algoritma genetika = 40, nilai *cross rate* = 0.3, nilai *mutation rate* = 0.7, jumlah iterasi LVQ = 14, laju pembelajaran  $\alpha$  = 0,1, laju pembelajaran minimal = 0.000000000001, laju pengurang pembelajaran = 0,6 dan jumlah individu 100.

### 7.2 Saran

1. Saran yang diberikan adalah masih dibutuhkan data yang lebih banyak untuk melakukan pelatihan agar mampu menghasilkan tingkat akurasi yang lebih maksimal.
2. Pengembangan sistem untuk penelitian selanjutnya yaitu dengan menggunakan *cost-sensitive learning* untuk mengukur serta meminimalisir misklasifikasi yang terjadi.

## DAFTAR PUSTAKA

- Anies, 2006. Waspada Ancaman Penyakit Tidak Menular Solusi Pencegahan dari Aspek Perilaku dan Lingkungan. Jakarta: PT. Elex Media Komputindo.
- Anon., 2014. Presiden Resmikan RS Pusat Otak Nasional. [Online] Available at: <http://www.depkes.go.id/article/print/201407200001/presiden-resmikan-rs-pusat-otak-nasional.html> [Accessed 17 April 2015].
- Anon., n.d. Epidemiologi Stroke. [Online] Available at: [http://www.academia.edu/8957119/epidemiologi\\_stroke](http://www.academia.edu/8957119/epidemiologi_stroke) [Accessed 17 April 2015].
- Cahyono, J. S. B., 2008. Gaya Hidup & Penyakit Modern. Jakarta: Kanisius.
- Chen, N. et al., 2010. Hybrid Genetic Algorithm and Learning Vector Quantization Modeling for Cost-Sensitive Bankruptcy Prediction.
- Chunhong, W., Hongqiang, Z. & Changxing, Y., 2012. Research on Color Recognition of Urine Test Paper Based on Learning Vector Quantization (LVQ).
- Fausett, L., 1994. Fundamentals of Neural Network: Architectures, Algorithms, and Applications. New Jersey: Prentice-Hall, Inc.
- Goldberg, D. E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Canada: Addison-Wesley Publishing.
- Hamakonda, T. P. & Tairas, J., 2008. Pengantar Klasifikasi Persepuluhan Dewey. Jakarta: PT. BPK Gunung Mulia.
- Holistic Health Solution, 2011. Stroke di Usia Muda. Jakarta: PT. Gramedia Widiasarana Indonesia.
- Iman, S., 2012. Serangan Jantung dan Stroke, Hubungannya Dengan Lemak dan Kolesterol. Edisi Kedua penyunt. Jakarta: PT. Gramedia Pustaka.
- Jin, F. & Shu, G., 2013. Back Propagation Neural Network Based on Artificial Bee Colony Algorithm. pp. 1-4.
- Kusumadewi, S., 2003. Artificial Intelligence (Teknik dan Aplikasinya). Yogyakarta: Graha Ilmu.
- Mahmudy, W. F., 2013. Algoritma Evolusi. Universitas Brawijaya: Program Teknologi Informasi dan Ilmu Komputer.
- Mehmood, Y., Ishtiaq, M., Tariq, M. & Jaffar, M. A., 2010. Classifier Ensemble Optimization for Gender Classification using Genetic Algorithm.
- Muttaqin, A., 2008. Buku Ajar Asuhan Keperawatan Klien dengan Gangguan Sistem Persarafan. Jakarta: Salemba Medika.

Pinzon, R. & Asanti, L., 2010. Awas Stroke! Pengertian, Gejala, Tindakan, Perawatan, dan Pencegahan. Yogyakarta: CV. Andi Offset.

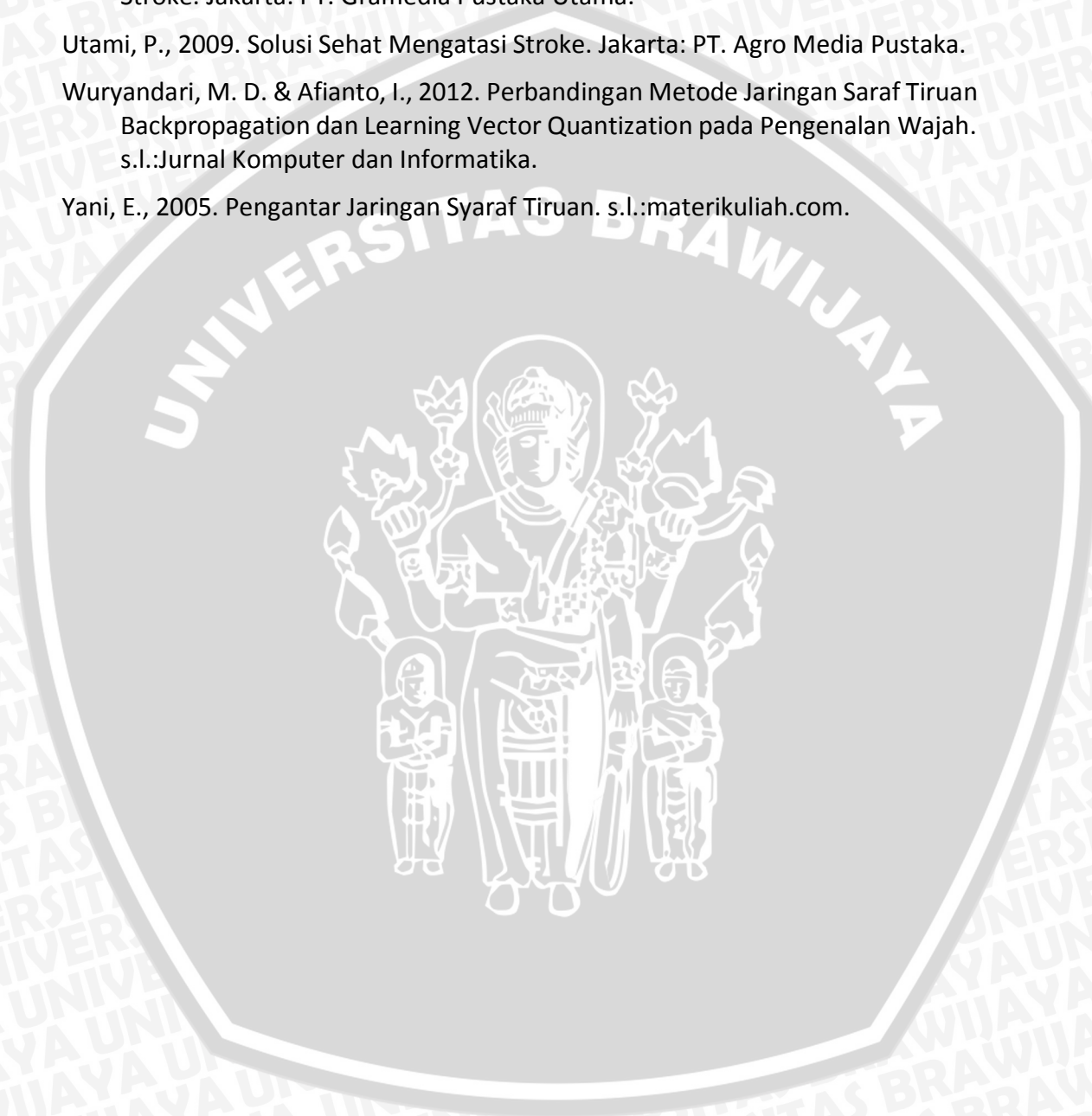
Susanto, I., 2009. STROKE = CEREBRO VASCULAR ACCIDENT = CVA = GANGGUAN PEMBULUH DARAH OTAK = GPDO. Surabaya: s.n.

Sutrisno, A., 2007. Stroke??? Sebaiknya Anda Tahu Sebelum Anda Terserang Stroke. Jakarta: PT. Gramedia Pustaka Utama.

Utami, P., 2009. Solusi Sehat Mengatasi Stroke. Jakarta: PT. Agro Media Pustaka.

Wuryandari, M. D. & Afianto, I., 2012. Perbandingan Metode Jaringan Saraf Tiruan Backpropagation dan Learning Vector Quantization pada Pengenalan Wajah. s.l.:Jurnal Komputer dan Informatika.

Yani, E., 2005. Pengantar Jaringan Syaraf Tiruan. s.l.:materikulia.com.



## LAMPIRAN A DATA

| No | Umur | Cholestrol<br>Total | HDL  | LDL   | Trigliserida | Kelompok |
|----|------|---------------------|------|-------|--------------|----------|
| 1  | 44   | 174                 | 36,5 | 123,3 | 71           | 1        |
| 2  | 60   | 169                 | 36,9 | 110,1 | 110          | 1        |
| 3  | 23   | 196                 | 40,3 | 137,1 | 93           | 1        |
| 4  | 41   | 213                 | 47,4 | 143,2 | 112          | 1        |
| 5  | 40   | 253                 | 46,5 | 179,3 | 136          | 1        |
| 6  | 48   | 197                 | 30,5 | 120,5 | 79           | 1        |
| 7  | 49   | 171                 | 40,1 | 104,1 | 134          | 1        |
| 8  | 40   | 198                 | 40,5 | 139,5 | 90           | 1        |
| 9  | 49   | 184                 | 38,4 | 119,2 | 133          | 1        |
| 10 | 59   | 280                 | 30,5 | 202,8 | 128          | 1        |
| 11 | 60   | 201                 | 41,6 | 136,4 | 115          | 1        |
| 12 | 29   | 145                 | 30,5 | 94,7  | 99           | 1        |
| 13 | 46   | 181                 | 37,7 | 118,3 | 125          | 1        |
| 14 | 66   | 278                 | 57,2 | 201,2 | 98           | 1        |
| 15 | 48   | 186                 | 36,2 | 121,8 | 140          | 1        |
| 16 | 66   | 186                 | 38,1 | 132,1 | 79           | 1        |
| 17 | 61   | 205                 | 45,2 | 140,6 | 96           | 1        |
| 18 | 28   | 182                 | 37,7 | 116,9 | 137          | 1        |
| 19 | 34   | 140                 | 38,8 | 137,8 | 128          | 1        |
| 20 | 75   | 170                 | 37,4 | 110,2 | 112          | 1        |
| 21 | 59   | 280                 | 47,6 | 203,2 | 146          | 1        |
| 22 | 45   | 197                 | 37,4 | 144,4 | 76           | 1        |
| 23 | 49   | 186                 | 36,1 | 127,5 | 112          | 1        |
| 24 | 37   | 188                 | 36,4 | 126,4 | 126          | 1        |
| 25 | 39   | 159                 | 36,7 | 149,9 | 142          | 1        |
| 26 | 53   | 277                 | 51,7 | 208,1 | 86           | 1        |
| 27 | 72   | 253                 | 35,1 | 188,1 | 149          | 1        |
| 28 | 54   | 232                 | 47,9 | 164,9 | 96           | 1        |
| 29 | 71   | 150                 | 34,5 | 134,6 | 138          | 1        |
| 30 | 32   | 229                 | 43,5 | 166,3 | 96           | 1        |
| 31 | 54   | 171                 | 38,1 | 115,9 | 85           | 1        |
| 32 | 52   | 166                 | 37,8 | 112,6 | 33,8         | 1        |
| 33 | 52   | 203                 | 38,6 | 135,6 | 144          | 1        |
| 34 | 58   | 194                 | 40,1 | 65,9  | 129          | 1        |
| 35 | 62   | 168                 | 35,1 | 110,1 | 114          | 1        |
| 36 | 71   | 174                 | 35,7 | 122,1 | 81           | 1        |
| 37 | 34   | 251                 | 47,1 | 182,9 | 105          | 1        |

| No | Umur | Cholestrol<br>Total | HDL  | LDL   | Trigliserida | Kelompok |
|----|------|---------------------|------|-------|--------------|----------|
| 38 | 45   | 207                 | 46,3 | 138,9 | 109          | 1        |
| 39 | 60   | 180                 | 42,1 | 121,1 | 84           | 1        |
| 40 | 48   | 278                 | 30,2 | 218,8 | 145          | 1        |
| 41 | 47   | 284                 | 52,8 | 201,4 | 149          | 1        |
| 42 | 50   | 184                 | 33,6 | 111,8 | 93           | 1        |
| 43 | 53   | 170                 | 39,7 | 103,5 | 134          | 1        |
| 44 | 43   | 211                 | 35,9 | 152,3 | 114          | 1        |
| 45 | 35   | 169                 | 37,1 | 117,1 | 74           | 1        |
| 46 | 70   | 170                 | 47,1 | 150,7 | 109          | 1        |
| 47 | 49   | 236                 | 42,8 | 171,4 | 109          | 1        |
| 48 | 49   | 178                 | 42,1 | 118,9 | 85           | 1        |
| 49 | 62   | 179                 | 40,1 | 120,9 | 90           | 1        |
| 50 | 35   | 207                 | 40,2 | 145,4 | 107          | 1        |
| 51 | 36   | 209                 | 44,1 | 145,3 | 98           | 1        |
| 52 | 59   | 191                 | 36,1 | 126,2 | 142          | 1        |
| 53 | 52   | 211                 | 42,7 | 139,1 | 146          | 1        |
| 54 | 62   | 169                 | 35,5 | 110   | 104          | 1        |
| 55 | 59   | 189                 | 38,2 | 134,6 | 81           | 1        |
| 56 | 46   | 232                 | 41,6 | 163,8 | 133          | 1        |
| 57 | 32   | 170                 | 36,5 | 113,9 | 98           | 1        |
| 58 | 61   | 193                 | 36,7 | 138,7 | 88           | 1        |
| 59 | 57   | 206                 | 37,6 | 140   | 142          | 1        |
| 60 | 38   | 300                 | 48,1 | 225,3 | 133          | 1        |
| 61 | 56   | 178                 | 38,4 | 110,8 | 144          | 1        |
| 62 | 28   | 276                 | 54,6 | 202,6 | 94           | 1        |
| 63 | 39   | 276                 | 34,5 | 293,6 | 115          | 1        |
| 64 | 49   | 182                 | 38,6 | 122,6 | 104          | 1        |
| 65 | 70   | 216                 | 42,5 | 144,1 | 147          | 1        |
| 66 | 73   | 200                 | 47,1 | 133,1 | 99           | 1        |
| 67 | 50   | 200                 | 40   | 141   | 95           | 1        |
| 68 | 68   | 219                 | 41,6 | 148,4 | 145          | 1        |
| 69 | 76   | 197                 | 37,6 | 137,4 | 110          | 1        |
| 70 | 56   | 261                 | 40,5 | 193,9 | 133          | 1        |
| 71 | 30   | 199                 | 40,2 | 141,6 | 86           | 1        |
| 72 | 23   | 223                 | 44,6 | 158,8 | 98           | 1        |
| 73 | 63   | 249                 | 41,5 | 183,7 | 119          | 1        |
| 74 | 57   | 229                 | 40,8 | 159,8 | 142          | 1        |
| 75 | 66   | 224                 | 37,2 | 157,4 | 147          | 1        |
| 76 | 30   | 180                 | 38,4 | 122,4 | 96           | 1        |

| No  | Umur | Cholestrol<br>Total | HDL  | LDL   | Trigliserida | Kelompok |
|-----|------|---------------------|------|-------|--------------|----------|
| 77  | 73   | 161                 | 37,8 | 105,6 | 88           | 1        |
| 78  | 62   | 160                 | 37,2 | 107   | 79           | 1        |
| 79  | 57   | 240                 | 44,2 | 171,6 | 121          | 1        |
| 80  | 50   | 184                 | 36,7 | 125,9 | 107          | 1        |
| 81  | 56   | 148                 | 32,8 | 102,8 | 67           | 1        |
| 82  | 57   | 179                 | 39,5 | 120,7 | 94           | 1        |
| 83  | 49   | 258                 | 40,5 | 192,7 | 124          | 1        |
| 84  | 60   | 182                 | 37,9 | 125,5 | 93           | 1        |
| 85  | 38   | 246                 | 45,1 | 141,1 | 134          | 1        |
| 86  | 41   | 238                 | 44,2 | 168   | 129          | 1        |
| 87  | 56   | 240                 | 45,1 | 176,3 | 133          | 1        |
| 88  | 56   | 274                 | 39,5 | 206,1 | 142          | 1        |
| 89  | 58   | 164                 | 39,5 | 107,1 | 87           | 1        |
| 90  | 60   | 192                 | 37,2 | 126,8 | 140          | 1        |
| 91  | 70   | 173                 | 36,2 | 111   | 132          | 1        |
| 92  | 52   | 197                 | 40,6 | 136,6 | 94           | 1        |
| 93  | 60   | 169                 | 37,8 | 102,6 | 143          | 1        |
| 94  | 53   | 163                 | 40,1 | 105,8 | 83           | 1        |
| 95  | 54   | 221                 | 39,6 | 156,3 | 126          | 1        |
| 96  | 57   | 170                 | 37,6 | 109,4 | 115          | 1        |
| 97  | 43   | 175                 | 38,6 | 117   | 97           | 1        |
| 98  | 78   | 199                 | 36,6 | 145,2 | 86           | 1        |
| 99  | 42   | 207                 | 45,3 | 133,3 | 142          | 1        |
| 100 | 58   | 225                 | 38,4 | 159,2 | 137          | 1        |
| 101 | 39   | 179                 | 38,4 | 116,2 | 122          | 1        |
| 102 | 27   | 165                 | 39,2 | 106,8 | 95           | 1        |
| 103 | 40   | 211                 | 45,6 | 167,8 | 104          | 1        |
| 104 | 30   | 176                 | 36,7 | 121,6 | 86           | 1        |
| 105 | 60   | 281                 | 48,7 | 206,5 | 129          | 1        |
| 106 | 48   | 210                 | 34,8 | 213,6 | 116          | 1        |
| 107 | 43   | 206                 | 33,4 | 107,8 | 91           | 1        |
| 108 | 62   | 147                 | 35,1 | 94,5  | 87           | 1        |
| 109 | 95   | 192                 | 39,2 | 123,8 | 145          | 1        |
| 110 | 52   | 241                 | 40,8 | 177,4 | 114          | 1        |
| 111 | 43   | 181                 | 39,8 | 124,8 | 82           | 1        |
| 112 | 46   | 255                 | 53,7 | 110,3 | 90           | 1        |
| 113 | 33   | 227                 | 40,1 | 164,5 | 112          | 1        |
| 114 | 56   | 162                 | 35,4 | 112   | 73           | 1        |
| 115 | 70   | 200                 | 36,5 | 136,1 | 137          | 1        |

| No  | Umur | Cholestrol<br>Total | HDL  | LDL   | Trigliserida | Kelompok |
|-----|------|---------------------|------|-------|--------------|----------|
| 116 | 45   | 219                 | 41,5 | 152,1 | 127          | 1        |
| 117 | 54   | 198                 | 38,7 | 142,1 | 86           | 1        |
| 118 | 50   | 172                 | 38,7 | 112,5 | 106          | 1        |
| 119 | 55   | 255                 | 37,8 | 184,6 | 163          | 2        |
| 120 | 42   | 191                 | 40,1 | 117,1 | 169          | 2        |
| 121 | 42   | 242                 | 42,5 | 162,5 | 185          | 2        |
| 122 | 25   | 210                 | 40,1 | 134,1 | 179          | 2        |
| 123 | 35   | 197                 | 39,6 | 119,6 | 189          | 2        |
| 124 | 52   | 157                 | 30,6 | 92,6  | 169          | 2        |
| 125 | 51   | 194                 | 40,3 | 129,5 | 180          | 2        |
| 126 | 43   | 199                 | 35,4 | 131,4 | 161          | 2        |
| 127 | 52   | 176                 | 40,1 | 104,5 | 157          | 2        |
| 128 | 61   | 267                 | 41,5 | 188,1 | 187          | 2        |
| 129 | 49   | 212                 | 40,5 | 139,7 | 159          | 2        |
| 130 | 57   | 168                 | 40,8 | 88,8  | 192          | 2        |
| 131 | 60   | 203                 | 45,1 | 124,9 | 165          | 2        |
| 132 | 28   | 185                 | 42,3 | 111,5 | 156          | 2        |
| 133 | 40   | 224                 | 45,6 | 193   | 186          | 2        |
| 134 | 52   | 319                 | 57,2 | 228   | 169          | 2        |
| 135 | 29   | 213                 | 36,7 | 145,3 | 155          | 2        |
| 136 | 37   | 239                 | 40   | 267,9 | 187          | 2        |
| 137 | 46   | 207                 | 38,2 | 136,8 | 160          | 2        |
| 138 | 57   | 226                 | 42,5 | 145,5 | 190          | 2        |
| 139 | 54   | 186                 | 36,6 | 109,8 | 198          | 2        |
| 140 | 45   | 249                 | 44,2 | 170,6 | 171          | 2        |
| 141 | 52   | 230                 | 36,2 | 107,6 | 181          | 2        |
| 142 | 54   | 199                 | 42,2 | 119,2 | 188          | 2        |
| 143 | 50   | 67                  | 42,1 | 112,1 | 164          | 2        |
| 144 | 47   | 249                 | 42,3 | 119,7 | 150          | 2        |
| 145 | 41   | 209                 | 37,1 | 138,3 | 168          | 2        |
| 146 | 56   | 418                 | 45,7 | 338,1 | 171          | 2        |
| 147 | 54   | 202                 | 38,1 | 129,5 | 172          | 2        |
| 148 | 57   | 176                 | 39,8 | 105,6 | 153          | 2        |
| 149 | 47   | 235                 | 45,2 | 152   | 189          | 2        |
| 150 | 55   | 169                 | 33,9 | 95,8  | 198          | 2        |
| 151 | 65   | 254                 | 38,6 | 183,6 | 159          | 2        |
| 152 | 49   | 302                 | 52,4 | 212   | 187          | 2        |
| 153 | 32   | 237                 | 46,4 | 159   | 158          | 2        |
| 154 | 51   | 166                 | 34,1 | 100,5 | 157          | 2        |

| No  | Umur | Cholestrol<br>Total | HDL  | LDL   | Trigliserida | Kelompok |
|-----|------|---------------------|------|-------|--------------|----------|
| 155 | 50   | 189                 | 35,6 | 119,2 | 171          | 2        |
| 156 | 53   | 245                 | 45,7 | 56,5  | 214          | 3        |
| 157 | 44   | 291                 | 46,7 | 180,5 | 319          | 3        |
| 158 | 53   | 223                 | 39,4 | 132,2 | 252          | 3        |
| 159 | 26   | 266                 | 40,2 | 181,2 | 223          | 3        |
| 160 | 67   | 276                 | 51,4 | 188,2 | 182          | 3        |
| 161 | 79   | 340                 | 31,2 | 208,9 | 162          | 3        |
| 162 | 38   | 287                 | 51,6 | 172,2 | 316          | 3        |
| 163 | 53   | 266                 | 45,7 | 165,5 | 274          | 3        |
| 164 | 62   | 162                 | 36,2 | 81    | 224          | 3        |
| 165 | 45   | 265                 | 47,6 | 150,2 | 241          | 3        |
| 166 | 57   | 212                 | 36,9 | 102,9 | 361          | 3        |
| 167 | 44   | 268                 | 43,6 | 161,2 | 316          | 3        |
| 168 | 85   | 278                 | 45,3 | 140,5 | 261          | 3        |
| 169 | 69   | 163                 | 32,5 | 100   | 152          | 3        |
| 170 | 41   | 173                 | 37,3 | 113,5 | 281          | 3        |
| 171 | 33   | 259                 | 34,1 | 166,3 | 293          | 3        |
| 172 | 52   | 287                 | 36,9 | 193,3 | 284          | 3        |
| 173 | 56   | 189                 | 32,1 | 91,7  | 326          | 3        |
| 174 | 49   | 324                 | 34   | 252,2 | 189          | 3        |
| 175 | 68   | 202                 | 37,8 | 133   | 156          | 3        |
| 176 | 55   | 180                 | 36,7 | 102,5 | 204          | 3        |
| 177 | 65   | 226                 | 36,1 | 145,3 | 223          | 3        |
| 178 | 42   | 228                 | 38,6 | 135,8 | 268          | 3        |
| 179 | 49   | 204                 | 33,6 | 38,3  | 461          | 3        |
| 180 | 51   | 235                 | 41,6 | 146,2 | 236          | 3        |
| 181 | 56   | 260                 | 42,2 | 170,8 | 235          | 3        |
| 182 | 50   | 180                 | 35,1 | 68,7  | 381          | 3        |
| 183 | 29   | 201                 | 36,1 | 147,3 | 226          | 3        |
| 184 | 58   | 259                 | 44,2 | 174,6 | 201          | 3        |
| 185 | 53   | 246                 | 44,9 | 153,5 | 238          | 3        |
| 186 | 56   | 261                 | 40,6 | 176,2 | 221          | 3        |
| 187 | 50   | 187                 | 36,1 | 91,1  | 295          | 3        |
| 188 | 64   | 192                 | 36,1 | 101,9 | 260          | 3        |
| 189 | 53   | 264                 | 36,9 | 159,3 | 319          | 3        |
| 190 | 51   | 253                 | 39,3 | 162,5 | 256          | 3        |
| 191 | 48   | 201                 | 39,5 | 116,7 | 224          | 3        |
| 192 | 50   | 196                 | 34,5 | 105,1 | 282          | 3        |
| 193 | 41   | 193                 | 35,7 | 113,1 | 221          | 3        |

| No  | Umur | Cholestrol<br>Total | HDL  | LDL   | Trigliserida | Kelompok |
|-----|------|---------------------|------|-------|--------------|----------|
| 194 | 91   | 212                 | 30,4 | 167,9 | 155          | 3        |
| 195 | 69   | 240                 | 48,3 | 160,7 | 155          | 3        |
| 196 | 80   | 247                 | 40,5 | 159,9 | 233          | 3        |
| 197 | 81   | 233                 | 40,7 | 159,5 | 164          | 3        |
| 198 | 46   | 250                 | 40,8 | 169   | 201          | 3        |
| 199 | 70   | 175                 | 35,1 | 99,7  | 201          | 3        |
| 200 | 52   | 514                 | 32,8 | 318,6 | 813          | 3        |

**Keterangan :**

- Kelompok 1 = Normal
- Kelompok 2 = Rentan
- Kelompok 3 = Mengkhawatirkan



## LAMPIRAN B DATA LATIH DAN DATA UJI

| No | Data Ke-1  |          | Data Ke-2  |          | Data Ke-3  |          | Data Ke-4  |          | Data Ke-5  |          |
|----|------------|----------|------------|----------|------------|----------|------------|----------|------------|----------|
|    | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji |
| 1  | 10         | 105      | 63         | 94       | 112        | 47       | 53         | 76       | 77         | 108      |
| 2  | 29         | 49       | 37         | 76       | 63         | 16       | 72         | 62       | 45         | 2        |
| 3  | 79         | 41       | 8          | 102      | 15         | 28       | 42         | 56       | 46         | 50       |
| 4  | 28         | 76       | 96         | 89       | 61         | 34       | 17         | 93       | 114        | 58       |
| 5  | 5          | 24       | 75         | 3        | 13         | 14       | 2          | 79       | 31         | 104      |
| 6  | 23         | 107      | 47         | 92       | 49         | 40       | 34         | 12       | 102        | 34       |
| 7  | 46         | 36       | 38         | 5        | 22         | 58       | 90         | 106      | 115        | 14       |
| 8  | 66         | 91       | 62         | 95       | 106        | 69       | 88         | 101      | 99         | 1        |
| 9  | 44         | 58       | 112        | 29       | 76         | 94       | 32         | 15       | 36         | 81       |
| 10 | 115        | 61       | 56         | 9        | 9          | 8        | 26         | 80       | 59         | 39       |
| 11 | 69         | 116      | 57         | 61       | 101        | 96       | 86         | 71       | 61         | 40       |
| 12 | 4          | 111      | 46         | 12       | 105        | 7        | 24         | 47       | 27         | 73       |
| 13 | 21         | 8        | 115        | 22       | 53         | 88       | 63         | 5        | 29         | 89       |
| 14 | 25         | 117      | 65         | 118      | 31         | 104      | 14         | 41       | 17         | 64       |
| 15 | 30         | 1        | 43         | 114      | 100        | 46       | 55         | 48       | 100        | 55       |
| 16 | 81         | 2        | 50         | 48       | 111        | 25       | 18         | 58       | 20         | 12       |
| 17 | 65         | 85       | 83         | 40       | 41         | 19       | 27         | 19       | 53         | 13       |
| 18 | 52         | 12       | 15         | 52       | 72         | 83       | 77         | 33       | 107        | 5        |
| 19 | 13         | 9        | 82         | 51       | 71         | 33       | 54         | 59       | 116        | 62       |
| 20 | 14         | 42       | 79         | 77       | 116        | 20       | 117        | 82       | 90         | 80       |
| 21 | 109        | 64       | 26         | 20       | 98         | 12       | 11         | 4        | 54         | 24       |
| 22 | 103        | 74       | 18         | 110      | 4          | 1        | 81         | 100      | 6          | 52       |
| 23 | 26         | 80       | 36         | 41       | 51         | 62       | 44         | 116      | 69         | 112      |
| 24 | 60         | 62       | 80         | 72       | 2          | 73       | 65         | 102      | 42         | 22       |
| 25 | 83         | 122      | 32         | 150      | 91         | 155      | 7          | 128      | 35         | 120      |
| 26 | 47         | 121      | 31         | 133      | 24         | 128      | 97         | 135      | 118        | 136      |
| 27 | 67         | 130      | 99         | 155      | 97         | 124      | 73         | 142      | 92         | 141      |
| 28 | 95         | 137      | 6          | 124      | 5          | 122      | 49         | 120      | 37         | 126      |
| 29 | 84         | 141      | 74         | 151      | 81         | 119      | 36         | 149      | 84         | 134      |
| 30 | 114        | 143      | 81         | 131      | 59         | 138      | 118        | 154      | 56         | 153      |
| 31 | 108        | 138      | 78         | 141      | 110        | 141      | 16         | 150      | 16         | 125      |
| 32 | 97         | 156      | 98         | 199      | 68         | 164      | 92         | 175      | 96         | 159      |
| 33 | 82         | 184      | 2          | 169      | 3          | 196      | 10         | 162      | 78         | 160      |
| 34 | 55         | 160      | 10         | 195      | 84         | 184      | 38         | 167      | 88         | 157      |
| 35 | 73         | 158      | 104        | 173      | 108        | 167      | 110        | 176      | 41         | 177      |
| 36 | 51         | 170      | 23         | 200      | 42         | 162      | 35         | 185      | 60         | 184      |
| 37 | 27         | 200      | 28         | 163      | 75         | 160      | 91         | 157      | 111        | 200      |
| 38 | 70         | 183      | 93         | 196      | 45         | 156      | 31         | 192      | 7          | 174      |
| 39 | 39         | 186      | 49         | 171      | 30         | 179      | 64         | 199      | 4          | 197      |
| 40 | 118        | 179      | 67         | 183      | 67         | 183      | 105        | 194      | 70         | 164      |
| 41 | 104        |          | 44         |          | 38         |          | 69         |          | 113        |          |
| 42 | 33         |          | 116        |          | 44         |          | 43         |          | 8          |          |

| No | Data Ke-1  |          | Data Ke-2  |          | Data Ke-3  |          | Data Ke-4  |          | Data Ke-5  |          |
|----|------------|----------|------------|----------|------------|----------|------------|----------|------------|----------|
|    | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji |
| 43 | 77         |          | 69         |          | 77         |          | 60         |          | 72         |          |
| 44 | 15         |          | 24         |          | 43         |          | 74         |          | 103        |          |
| 45 | 3          |          | 30         |          | 118        |          | 1          |          | 106        |          |
| 46 | 88         |          | 71         |          | 66         |          | 109        |          | 66         |          |
| 47 | 31         |          | 85         |          | 93         |          | 3          |          | 33         |          |
| 48 | 90         |          | 106        |          | 115        |          | 98         |          | 110        |          |
| 49 | 113        |          | 16         |          | 89         |          | 68         |          | 51         |          |
| 50 | 112        |          | 45         |          | 17         |          | 87         |          | 109        |          |
| 51 | 102        |          | 33         |          | 27         |          | 21         |          | 87         |          |
| 52 | 56         |          | 53         |          | 65         |          | 52         |          | 9          |          |
| 53 | 45         |          | 27         |          | 29         |          | 9          |          | 83         |          |
| 54 | 43         |          | 109        |          | 11         |          | 29         |          | 74         |          |
| 55 | 92         |          | 97         |          | 52         |          | 30         |          | 67         |          |
| 56 | 87         |          | 7          |          | 87         |          | 113        |          | 101        |          |
| 57 | 99         |          | 84         |          | 82         |          | 57         |          | 49         |          |
| 58 | 16         |          | 60         |          | 64         |          | 46         |          | 95         |          |
| 59 | 59         |          | 19         |          | 39         |          | 103        |          | 28         |          |
| 60 | 68         |          | 103        |          | 113        |          | 23         |          | 30         |          |
| 61 | 53         |          | 86         |          | 32         |          | 107        |          | 48         |          |
| 62 | 11         |          | 108        |          | 23         |          | 22         |          | 98         |          |
| 63 | 71         |          | 111        |          | 21         |          | 114        |          | 75         |          |
| 64 | 100        |          | 73         |          | 80         |          | 61         |          | 19         |          |
| 65 | 6          |          | 11         |          | 79         |          | 66         |          | 10         |          |
| 66 | 54         |          | 54         |          | 6          |          | 37         |          | 71         |          |
| 67 | 96         |          | 68         |          | 70         |          | 78         |          | 94         |          |
| 68 | 48         |          | 13         |          | 55         |          | 75         |          | 11         |          |
| 69 | 37         |          | 1          |          | 74         |          | 94         |          | 23         |          |
| 70 | 89         |          | 25         |          | 10         |          | 84         |          | 76         |          |
| 71 | 7          |          | 87         |          | 78         |          | 70         |          | 85         |          |
| 72 | 32         |          | 35         |          | 90         |          | 50         |          | 117        |          |
| 73 | 106        |          | 91         |          | 60         |          | 39         |          | 47         |          |
| 74 | 38         |          | 113        |          | 107        |          | 13         |          | 3          |          |
| 75 | 35         |          | 17         |          | 18         |          | 99         |          | 43         |          |
| 76 | 50         |          | 55         |          | 57         |          | 85         |          | 26         |          |
| 77 | 22         |          | 70         |          | 92         |          | 112        |          | 38         |          |
| 78 | 94         |          | 39         |          | 99         |          | 108        |          | 65         |          |
| 79 | 110        |          | 21         |          | 86         |          | 8          |          | 97         |          |
| 80 | 98         |          | 105        |          | 36         |          | 40         |          | 91         |          |
| 81 | 20         |          | 101        |          | 35         |          | 95         |          | 25         |          |
| 82 | 17         |          | 4          |          | 56         |          | 111        |          | 44         |          |
| 83 | 93         |          | 64         |          | 85         |          | 51         |          | 32         |          |
| 84 | 86         |          | 58         |          | 50         |          | 25         |          | 15         |          |
| 85 | 57         |          | 117        |          | 109        |          | 115        |          | 63         |          |
| 86 | 40         |          | 59         |          | 103        |          | 6          |          | 57         |          |

| No  | Data Ke-1  |          | Data Ke-2  |          | Data Ke-3  |          | Data Ke-4  |          | Data Ke-5  |          |
|-----|------------|----------|------------|----------|------------|----------|------------|----------|------------|----------|
|     | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji |
| 87  | 78         |          | 14         |          | 37         |          | 20         |          | 86         |          |
| 88  | 19         |          | 90         |          | 114        |          | 104        |          | 105        |          |
| 89  | 101        |          | 42         |          | 26         |          | 89         |          | 82         |          |
| 90  | 18         |          | 107        |          | 95         |          | 83         |          | 79         |          |
| 91  | 63         |          | 100        |          | 102        |          | 28         |          | 18         |          |
| 92  | 75         |          | 88         |          | 117        |          | 67         |          | 93         |          |
| 93  | 34         |          | 34         |          | 48         |          | 45         |          | 21         |          |
| 94  | 72         |          | 66         |          | 54         |          | 96         |          | 68         |          |
| 95  | 133        |          | 144        |          | 153        |          | 137        |          | 131        |          |
| 96  | 146        |          | 149        |          | 127        |          | 132        |          | 123        |          |
| 97  | 140        |          | 127        |          | 154        |          | 141        |          | 133        |          |
| 98  | 151        |          | 140        |          | 135        |          | 148        |          | 129        |          |
| 99  | 150        |          | 126        |          | 152        |          | 119        |          | 144        |          |
| 100 | 124        |          | 137        |          | 150        |          | 144        |          | 124        |          |
| 101 | 142        |          | 153        |          | 149        |          | 125        |          | 147        |          |
| 102 | 149        |          | 122        |          | 142        |          | 145        |          | 145        |          |
| 103 | 139        |          | 143        |          | 134        |          | 146        |          | 152        |          |
| 104 | 134        |          | 147        |          | 129        |          | 140        |          | 149        |          |
| 105 | 127        |          | 148        |          | 131        |          | 126        |          | 130        |          |
| 106 | 136        |          | 123        |          | 132        |          | 124        |          | 138        |          |
| 107 | 128        |          | 128        |          | 137        |          | 127        |          | 146        |          |
| 108 | 129        |          | 142        |          | 147        |          | 143        |          | 135        |          |
| 109 | 126        |          | 130        |          | 136        |          | 147        |          | 139        |          |
| 110 | 154        |          | 135        |          | 148        |          | 138        |          | 151        |          |
| 111 | 148        |          | 134        |          | 151        |          | 151        |          | 142        |          |
| 112 | 123        |          | 139        |          | 121        |          | 133        |          | 137        |          |
| 113 | 144        |          | 145        |          | 126        |          | 121        |          | 155        |          |
| 114 | 125        |          | 119        |          | 146        |          | 153        |          | 121        |          |
| 115 | 152        |          | 146        |          | 133        |          | 139        |          | 119        |          |
| 116 | 131        |          | 125        |          | 139        |          | 122        |          | 143        |          |
| 117 | 132        |          | 120        |          | 143        |          | 136        |          | 127        |          |
| 118 | 135        |          | 132        |          | 145        |          | 155        |          | 128        |          |
| 119 | 145        |          | 121        |          | 120        |          | 130        |          | 150        |          |
| 120 | 120        |          | 152        |          | 140        |          | 152        |          | 140        |          |
| 121 | 155        |          | 136        |          | 144        |          | 129        |          | 154        |          |
| 122 | 147        |          | 138        |          | 130        |          | 131        |          | 132        |          |
| 123 | 119        |          | 154        |          | 125        |          | 134        |          | 148        |          |
| 124 | 153        |          | 129        |          | 123        |          | 123        |          | 122        |          |
| 125 | 189        |          | 189        |          | 188        |          | 198        |          | 188        |          |
| 126 | 197        |          | 165        |          | 165        |          | 196        |          | 163        |          |
| 127 | 196        |          | 172        |          | 186        |          | 187        |          | 161        |          |
| 128 | 187        |          | 176        |          | 174        |          | 189        |          | 190        |          |
| 129 | 198        |          | 193        |          | 198        |          | 164        |          | 186        |          |
| 130 | 191        |          | 198        |          | 187        |          | 200        |          | 182        |          |

| No  | Data Ke-1  |          | Data Ke-2  |          | Data Ke-3  |          | Data Ke-4  |          | Data Ke-5  |          |
|-----|------------|----------|------------|----------|------------|----------|------------|----------|------------|----------|
|     | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji | Data Latih | Data Uji |
| 131 | 199        |          | 167        |          | 163        |          | 166        |          | 181        |          |
| 132 | 164        |          | 168        |          | 191        |          | 163        |          | 158        |          |
| 133 | 188        |          | 191        |          | 194        |          | 182        |          | 198        |          |
| 134 | 194        |          | 166        |          | 173        |          | 193        |          | 191        |          |
| 135 | 169        |          | 192        |          | 176        |          | 178        |          | 194        |          |
| 136 | 180        |          | 161        |          | 199        |          | 183        |          | 185        |          |
| 137 | 161        |          | 185        |          | 172        |          | 156        |          | 199        |          |
| 138 | 177        |          | 186        |          | 161        |          | 159        |          | 169        |          |
| 139 | 185        |          | 188        |          | 178        |          | 174        |          | 162        |          |
| 140 | 193        |          | 175        |          | 197        |          | 172        |          | 172        |          |
| 141 | 159        |          | 160        |          | 180        |          | 191        |          | 195        |          |
| 142 | 168        |          | 194        |          | 169        |          | 161        |          | 183        |          |
| 143 | 176        |          | 184        |          | 168        |          | 181        |          | 175        |          |
| 144 | 163        |          | 197        |          | 177        |          | 165        |          | 193        |          |
| 145 | 171        |          | 170        |          | 195        |          | 180        |          | 171        |          |
| 146 | 173        |          | 177        |          | 182        |          | 173        |          | 180        |          |
| 147 | 167        |          | 187        |          | 181        |          | 188        |          | 189        |          |
| 148 | 166        |          | 190        |          | 193        |          | 190        |          | 187        |          |
| 149 | 175        |          | 178        |          | 192        |          | 186        |          | 170        |          |
| 150 | 162        |          | 156        |          | 159        |          | 195        |          | 166        |          |
| 151 | 157        |          | 180        |          | 166        |          | 158        |          | 165        |          |
| 152 | 165        |          | 157        |          | 171        |          | 168        |          | 192        |          |
| 153 | 190        |          | 182        |          | 175        |          | 184        |          | 156        |          |
| 154 | 172        |          | 164        |          | 190        |          | 160        |          | 179        |          |
| 155 | 174        |          | 158        |          | 158        |          | 179        |          | 167        |          |
| 156 | 178        |          | 174        |          | 185        |          | 177        |          | 168        |          |
| 157 | 182        |          | 159        |          | 157        |          | 170        |          | 178        |          |
| 158 | 195        |          | 162        |          | 200        |          | 197        |          | 196        |          |
| 159 | 181        |          | 179        |          | 189        |          | 169        |          | 176        |          |
| 160 | 192        |          | 181        |          | 170        |          | 171        |          | 173        |          |

**Keterangan:**

- Setiap Nilai pada Data Latih dan Data Uji merupakan ID dari Dataset