IMPLEMENTASI LVQ DENGAN OPTIMASI ALGORITMA GENETIK PADA KLASIFIKASI DIAGNOSIS PENYAKIT DIABETES

Cornelius Justin S. H. (2006529796), Rafi Alvanzah (2006528736), M. Hanif Pramudya Z. (2006487566), Javier Bintoro (2006572150), Muhammad Daffa (2006568626), Daffa Al Ghifary (2006463420), Tulus Setiawan (2006568802)

Program Studi Matematika, Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam

Universitas Indonesia

Kampus UI Depok, 16424, Indonesia

ABSTRAK

Penyakit diabetes ditandai dengan kadar gula (glukosa) darah yang tinggi atau di atas nilai normal. Jika diabetes tidak dikontrol dengan baik, berbagai komplikasi yang dapat membahayakan nyawa pasien dapat muncul. Analisis data diabetes merupakan masalah yang menantang karena sebagian besar data medis bersifat nonlinier, tidak normal, berstruktur korelasi, dan bersifat kompleks. Sistem berbasis ML dapat digunakan sebagai teknik pemilihan fitur (FST) dan pengklasifikasi. Sistem menggunakan metode klasifikasi yaitu *Learning Vector Quantization* (LVQ). Pada beberapa kasus klasifikasi, LVQ memiliki kelemahan yaitu akurasi yang lemah sehingga dibutuhkan metode optimasi yaitu Algoritma Genetika (AG) yang mampu meningkatkan akurasi. Vektor bobot LVQ akan dioptimasi oleh AG melalui proses genetika hingga dihasilkan vektor bobot optimal yang akan digunakan LVQ untuk *training* dan *testing*. Pengujian dilakukan terhadap LVQ dan LVQ-AG, kemudian LVQ menghasilkan akurasi 75.73% sedangkan setelah dioptimasi oleh AG menjadi 79% dengan parameter terbaik antara lain ukuran populasi 100, *crossover rate* 0.9, *probability mutation* 0.5, jumlah generasi 100, *learning rate* 0.001, *decrease learning rate* 0.1, *epoch* maksimal 200, dan minimal *learning rate* 0,0001.

Kata Kunci: Klasifikasi, Diabetes, Algoritma Genetik, Learning Vector Quantizer

1. PENDAHULUAN

Diabetes adalah penyakit jangka panjang atau kronis gangguan metabolisme yang ditandai dengan gula darah tinggi.Penyakit diabetes ditandai dengan kadar gula (glukosa) darah yang tinggi atau di atas nilai normal. Glukosa yang menumpuk di dalam darah akibat tidak diserap sel tubuh dengan baik dapat menyebabkan banyak penyakit rumit jangka panjang yang serius seperti penyakit kardiovaskuler, stroke, gagal ginjal, serangan jantung, penyakit arteri perifer, pembuluh darah, dan saraf. Jika diabetes tidak dikontrol dengan baik, berbagai komplikasi yang dapat membahayakan nyawa pasien dapat muncul. [1]

Learning Vector Quantizer (LVQ) adalah suatu metode klasifikasi pola yang masing-masing unit output mewakili kategori atau kelompok tertentu. Pemrosesan yang terjadi pada setiap neuron adalah mencari jarak terdekat antara suatu vektor masukan ke bobot yang bersangkutan. Kelebihan metode ini adalah selain mencari jarak terdekat, selama pembelajaran unit output diposisikan dengan mengatur dan memperbaharui bobot melalui pembelajaran yang terawasi untuk memperkirakan keputusan klasifikasi. Sedangkan untuk LVQ, dua vektor (pemenang dan *runner-up*) diperbarui jika beberapa kondisi dipenuhi. Ide pengembangan algoritma LVQ adalah jika input memiliki taksiran jarak yang sama dengan vektor pemenang dan runner-up, maka masing-masing vektor tersebut harus melakukan pembelajaran. [2]

Genetic Algorithm (GA) adalah algoritma optimasi yang memaksimalkan atau meminimalkan fungsi yang diberikan. Operator seleksi menjadi ciri khusus dalam algoritma genetika karena merupakan salah satu yang terutama menentukan ruang pencarian evolusioner. Ini digunakan untuk meningkatkan peluang kelangsungan hidup individu yang paling cocok. Ada banyak mekanisme seleksi tradisional yang digunakan dan banyak mekanisme seleksi yang ditentukan pengguna khusus untuk definisi masalah. [3]

Analisis data diabetes merupakan masalah yang menantang karena sebagian besar data medis bersifat nonlinier, tidak normal, berstruktur korelasi, dan bersifat kompleks. Sistem berbasis ML dapat digunakan sebagai teknik pemilihan fitur (FST) dan pengklasifikasi. Ini juga membantu orang untuk mendiagnosis diabetes secara akurat dan pengklasifikasi terbaik adalah masalah terpenting untuk stratifikasi risiko diabetes yang akurat. Hal tersebut mendorong penulis mengenai konstruksi Model berbasis Machine Learning yang akurat dalam mendiagnosis diabetes.

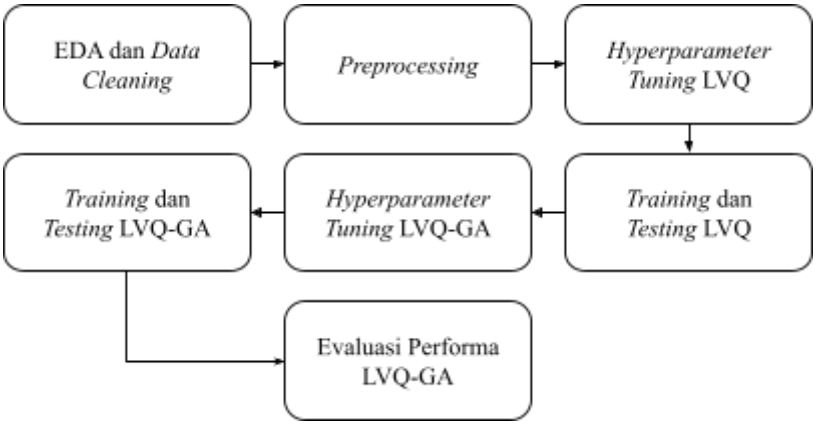
Rumusan Masalah

1. Bagaimana implementasi *Learning Vector Quantizer* dengan optimasi *Genetic Algorithm* pada klasifikasi diagnosis penyakit diabetes?
2. Bagaimana performa *Learning Vector Quantizer* dengan optimasi *Genetic Algorithm* terhadap masalah klasifikasi diagnosis penyakit diabetes berdasarkan *accuracy, precision, recall,* dan *f1 score*?

Tujuan

1. Mengimplementasikan *Learning Vector Quantizer* dengan optimasi *Genetic Algorithm* pada klasifikasi diagnosis penyakit diabetes.
2. Menganalisis performa *Learning Vector Quantizer* dengan optimasi *Genetic Algorithm* terhadap masalah klasifikasi diagnosis penyakit diabetes berdasarkan *accuracy, precision, recall,* dan *f1 score.*

2. METODE



Gambar 1: Gambaran proses penentuan parameter terbaik untuk model LVQ-GA.

Gambar 1 menunjukan proses yang digunakan untuk menentukan parameter terbaik dan bobot terbaik untuk mengklasifikasi diagnosis penyakit diabetes. Dataset yang digunakan terdiri dari 768 sampel yang berisi fitur terkait penyakit diabetes yang didapat dari Kaggle.

***Exploratory Data Analysis* (EDA)**

Exploratory Data Analysis (EDA) adalah suatu proses yang digunakan untuk mengeksplorasi dan memahami data secara lebih mendalam. Tujuan dari EDA adalah untuk menemukan pola, menguji hipotesis, dan menemukan korelasi atau hubungan antar fitur (variabel) dalam data.

EDA biasanya dilakukan sebelum melakukan analisis statistik yang lebih lanjut atau sebelum membuat model *machine learning*. EDA dapat dilakukan dengan menggunakan beberapa teknik, seperti visualisasi data, statistik deskriptif, dan uji hipotesis.

Data Cleaning

Data cleaning adalah proses pembersihan data dari kesalahan atau kekurangan yang ada sebelum data tersebut digunakan untuk tujuan analisis atau pemodelan. Data yang tidak bersih dapat menyebabkan hasil analisis atau pemodelan yang tidak akurat atau tidak dapat diandalkan.

Data cleaning biasanya meliputi beberapa tahap, seperti mengidentifikasi dan menghapus data yang tidak valid atau tidak relevan, mengisi kembali data yang hilang atau kosong, dan mengubah data yang tidak sesuai dengan format yang diinginkan. *Data cleaning* juga dapat meliputi tahap seperti menghapus data duplikat, menyatukan data yang terpisah, dan menstandarkan data yang tidak sesuai.

Preprocessing

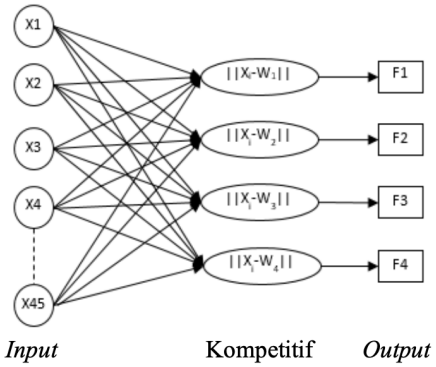
Preprocessing adalah tahap yang dilakukan sebelum melakukan pemodelan atau analisis data dengan menggunakan *machine learning*. *Preprocessing* merupakan tahap yang sangat penting karena data yang tidak bersih atau tidak sesuai dengan format yang diinginkan dapat menyebabkan hasil pemodelan atau analisis yang tidak akurat.

Preprocessing dapat meliputi beberapa tahap, seperti pembersihan data, pengolahan data, dan pengubahan data ke dalam format yang sesuai dengan kebutuhan pemodelan atau analisis. *Preprocessing* juga dapat meliputi tahap seperti pemilihan fitur (variabel), normalisasi data, dan pemecahan data menjadi data latih (*training data*) dan data uji (*test data*).

***Learning Vector Quantizer* (LVQ)**

Learning Vector Quantization (LVQ) adalah sebuah algoritma *machine learning* yang digunakan untuk melakukan klasifikasi data. LVQ merupakan salah satu jenis algoritma yang termasuk dalam kelompok algoritma *Neural Network*.

LVQ beroperasi dengan cara mengelompokkan data ke dalam kelas-kelas yang telah ditentukan sebelumnya. LVQ melakukan klasifikasi dengan cara menemukan vektor yang terdapat pada setiap kelas yang disebut dengan *prototype vector. Prototype vector* ini merupakan representasi dari kelas tersebut dan digunakan untuk membandingkan dengan data yang akan diklasifikasikan.



Gambar 2 : Arsitektur LVQ.

Arsitektur LVQ terdiri dari :

1. Lapisan *Input* yang merupakan lapisan yang menerima informasi (*input*) terkait permasalahan.
2. Lapisan Kompetitif yang merupakan lapisan yang memproses jarak kedekatan *input* dengan bobot yang berpadanan untuk setiap *output*. Jarak terdekat akan menentukan kelas suatu *input*.
3. Lapisan *Output* yang merupakan lapisan yang menampilkan *output* dari kelas yang sudah ditentukan di kelas kompetitif.

Langkah proses LVQ [10]:

1. Penghitungan jarak terhadap masing-masing kelas dengan menggunakan *Euclidean Distance.*

$$D_{i,j} = ||X_i - W_j|| \qquad (1)$$

Keterangan

$D_{i,j}$ = Jarak *Euclidean Distance input* ke-i dengan bobot kelas ke-j

X_i = Data *sample* atau *input* ke-i

W_j = Bobot kelas ke-j

2. Menentukan kelas dari data ke-i berdasarkan jarak minimum terhadap seluruh bobot.

$$C_i = \begin{cases} 0, \ \min(D_{i,0}, D_{i,1}) = D_{i,0} \\ 1, \ \min(D_{i,0}, D_{i,1}) = D_{i,1} \end{cases} \qquad (2)$$

Keterangan

C_i= kelas ke-i

$D_{i,0}$= *Euclidean Distance* antara data *sample* ke-i dengan bobot dengan target 0

$D_{i,1}$= *Euclidean Distance* antara data *sample* ke-i dengan bobot dengan target 1

3. Melakukan *update* bobot W_j dengan syarat :

   a. Apabila $C_i = T_i$ maka

$$W_j(baru) = W_j(lama) + \alpha(X_i - W_j(lama)) \qquad (3)$$

b. Apabila $C_i \neq T_i$ maka

$$W_j(baru) = W_j(lama) + \alpha(X_i - W_j(lama)) \quad (4)$$

Keterangan :

T_i = Target kelas pada data *sample*

α = *Learning rate*, nilai acak antara 0 hingga 1

4. Melakukan *update learning rate (α)*

$$\alpha(baru) = \alpha(lama) \times dec\,\alpha \quad (5)$$

Keterangan

$dec\,\alpha$ = *decreasing learning rate*

5. Ulangi langkah (1-4) hingga tercapai epoch maksimum.

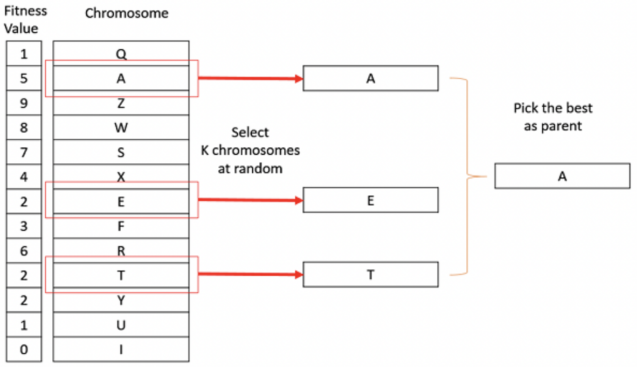
$$Akurasi = \frac{jumlah\ data\ dengan\ kelas\ sesuai\ target}{jumlah\ seluruh\ data} \times 100 \quad (6)$$

***Genetic Algorithm* (GA)**

Algoritma Genetik (*Genetic Algorithm*) adalah sebuah algoritma yang digunakan untuk menyelesaikan masalah optimasi dengan cara menggunakan prinsip-prinsip seleksi alam. Algoritma genetik mengadopsi cara kerja evolusi dalam mencari solusi terbaik untuk masalah yang dihadapi.

Algoritma genetik memiliki beberapa tahap utama, yaitu [10]:

1. Inisialisasi populasi: Membuat sejumlah individu atau kromosom yang merupakan solusi sementara untuk masalah yang dihadapi.
2. Hitung *fitness* : Menghitung *fitness* dari setiap kromosom di mana nilai *fitness* nya merupakan akurasi dari proses LVQ (persamaan (6)).
3. Seleksi *parent*: Memilih individu-individu terbaik dari populasi untuk dijadikan *parent* dan akan dilakukan proses *crossover.* Seleksi *parent* yang digunakan adalah *random selection* dan *tournament rank selection* .
   a. Pada *random selection* akan dipilih k-kromosom dari populasi secara acak. Proses ini dilakukan pada generasi pertama.
   b. Pada *tournament rank selection* akan dipilih k-kromosom dari populasi secara acak dan pilih yang terbaik dari ini untuk menjadi *parent*. Proses yang sama diulangi untuk memilih *parent* berikutnya.



Gambar 3 : Ilustrasi *tournament rank selection.*

4. Crossover: Reproduksi yang melibatkan 2 parent untuk menghasilkan offspring atau kromosom baru yang gen nya merupakan persilangan kedua parent. Metode crossover yang digunakan adalah Two-Point Crossover.

$$O1 = [P1(1),\ldots,P1(p),P2(p+1),\ldots,P2(q),P1(q+1),\ldots,P1(n)]$$
$$O2 = [P2(1),\ldots,P2(p),P1(p+1),\ldots,P1(q),P2(q+1),\ldots,P2(n)] \quad (7)$$
$$0 < p < q < n$$

Keterangan

$O1$ = *Offspring 1*

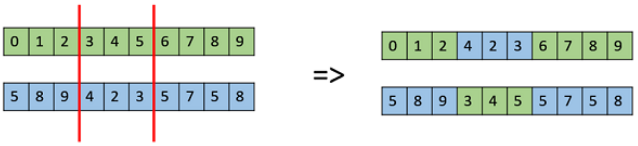
$O2$ = *Offspring 2*

$P1$ = *Parent 1*

$P1$ = *Parent 2*

n = Jumlah gen pada kromosom

p = Nilai acak antara 0 dan n-1
q = Nilai acak antara p dan n



Gambar 4 : Ilustrasi *two point crossover.*

5. Mutasi: Menimbulkan perubahan kecil pada individu atau kromosom baru yang dihasilkan dari proses *crossover* untuk memberi variasi. Metode mutasi yang digunakan adalah *random mutation.*

$$G_n = G_n + r\,(max_n - min_n) \qquad (8)$$

Keterangan
G_n = Gen ke-n
r = Nilai acak [-0.1 , 0.1]
max_n = Nilai maksimal pada gen ke-n
min_n = Nilai minimal pada gen ke-n

6. Penilaian atau seleksi *fitness*: Menilai kromosom baru yang dihasilkan dan mengganti kromosom yang kurang baik dengan kromosom baru yang lebih baik. Metode seleksi *fitness* yang digunakan adalah *elitism selection.* Pada metode *elitism selection* akan dipilih k-kromosom terbaik untuk menjadi populasi di generasi berikutnya.

Kromosom	*Fitness*
Kromosom 1	75
Kromosom 2	50
Kromosom 3	92
Kromosom 4	60
Kromosom 5	40
Kromosom 6	33

→

Kromosom	*Fitness*
Kromosom 3	92
Kromosom 1	75
Kromosom 4	60
Kromosom 2	50

Gambar 5 : Ilustrasi *elitism selection.*

Langkah-langkah tersebut terus dilakukan hingga solusi yang diinginkan ditemukan atau hingga kondisi tertentu terpenuhi.

Hyperparameter Tuning

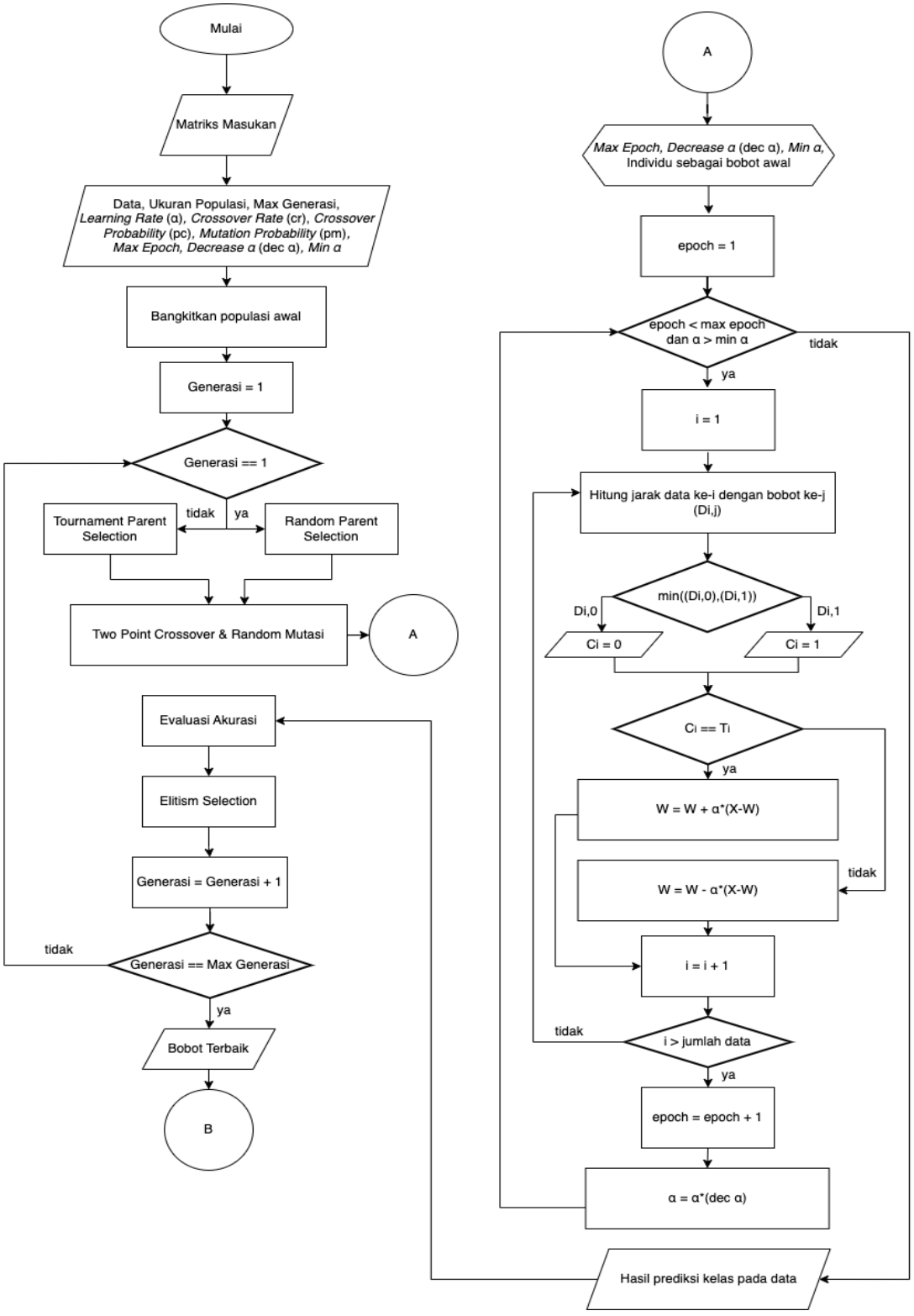
Hyperparameter tuning adalah proses menyesuaikan nilai-nilai yang tidak dapat diperoleh dari data latih (*training data*) dalam algoritma *machine learning*. Nilai-nilai tersebut disebut dengan *hyperparameter*. *Hyperparameter* bisa berupa nilai yang mempengaruhi kinerja algoritma, seperti *learning rate* dalam algoritma *gradient descent* atau jumlah titik pemisahan dalam algoritma *decision tree.*

Hyperparameter tuning dilakukan untuk mencari nilai *hyperparameter* yang paling optimal agar algoritma dapat bekerja dengan baik pada data yang akan diprediksi. Proses *hyperparameter tuning* biasanya dilakukan dengan cara mencoba berbagai nilai *hyperparameter* dan mengevaluasi kinerja algoritma dengan menggunakan data uji (*test data*).

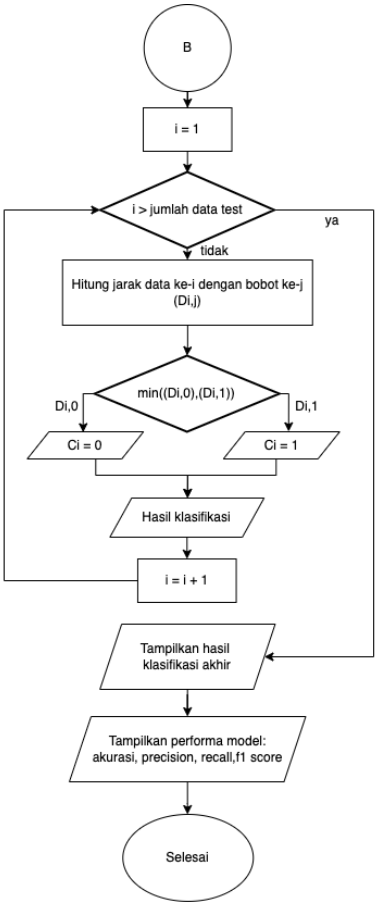
Proses Pembentukan Model

Pada penelitian ini, digunakan gabungan antara dua model yaitu, model *learning vector quantizer* (LVQ) dan *genetics algorithm* (GA). LVQ digunakan untuk menentukan *fitness* dari bobot yang akan di-*update* oleh GA sehingga tercipta bobot paling optimal, proses ini bisa dilihat pada Gambar 6. Selanjutnya, bobot yang didapat dari proses LVQ-GA akan digunakan untuk melakukan klasifikasi dengan LVQ yang bisa dilihat pada Gambar 7.

Sebelum membentuk model, akan dilakukan EDA, *data cleaning, preprocessing*, dan *hyperparameter tuning* sesuai dengan Gambar 1. Pada proses *hyperparameter tuning* LVQ akan digunakan *randomized search* dengan 5-*fold cross validation* dan iterasi sebanyak 200 kali sehingga didapat parameter terbaik untuk model LVQ. Selanjutnya, parameter terbaik tersebut akan digunakan untuk proses penentuan parameter terbaik dan bobot terbaik dari LVQ-GA. Setelah didapat bobot terbaik yang didapat dari LVQ-GA dan parameter terbaik dari proses *hyperparameter tuning* LVQ, barulah akan tercipta model klasifikasi LVQ dengan optimisasi GA yang diharapkan lebih optimal dibandingkan model klasifikasi LVQ tanpa optimisasi GA.



Gambar 6 : *Flowchart Training* LVQ-GA.



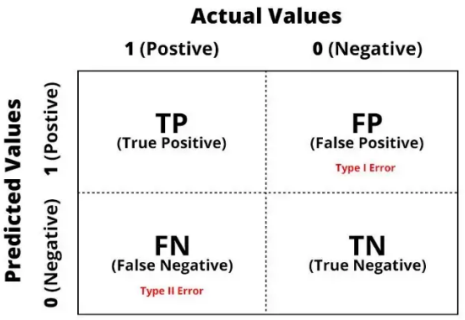
Gambar 7 : *Flowchart Testing* LVQ-GA.

Keterangan parameter :

1. Populasi: Sekumpulan kromosom yang menyimpan solusi. Setiap kromosom dalam populasi mewakili sebuah solusi dan setiap kromosom terdiri dari sekumpulan nilai atau variabel yang disebut gen.
2. Generasi: Generasi pada *genetic algorithm* mengacu pada satu iterasi pada algoritma. Setiap generasi menghasilkan solusi berdasarkan generasi sebelumnya.
3. *Learning Rate* : Seberapa besar bobot pada neuron diubah pada setiap iterasi.
4. *Crossover Rate* (CR): Tingkat kemungkinan terjadinya penggabungan dua individu atau kromosom sehingga terbentuk individu baru yang memiliki sifat-sifat dari kedua individu tersebut.
5. *Probability Crossover* (PC): Parameter yang menentukan probabilitas dipilihnya dua solusi (*parent solution*) untuk menghasilkan solusi baru (*child solution*).
6. *Probability Mutation* (PM): Parameter yang menentukan probabilitas solusi (*chromosome*) akan dimodifikasi.
7. *Decreasing Learning Rate* (Dec α): Laju penurunan *learning rate* pada setiap iterasi.
8. Minimum *Learning Rate* (min α): *Learning rate* minimum. Jika tercapai *learning rate* minimum iterasi akan dihentikan.
9. Epoch: Ketika seluruh dataset sudah melalui proses training pada *Neural Network* sampai dikembalikan ke awal untuk sekali putaran.
10. Bobot: Bobot dari setiap kriteria yang akan dijadikan perhitungan dalam penentuan label kelas data.

Evaluasi Performa LVQ-GA

Dalam mengevaluasi performa dari algoritma LVQ-GA dibutuhkan suatu *matrix* evaluasi. *Matrix* evaluasi yang biasa digunakan pada kasus *supervised learning* seperti klasifikasi adalah *confusion matrix*. Pada dasarnya *confusion matrix* memberikan informasi perbandingan hasil klasifikasi yang dilakukan oleh model dengan hasil klasifikasi sebenarnya dalam bentuk tabel matriks [12].



Gambar 8 : Tabel *Confusion Matrix.*

- *True Positif* (TP) : Merupakan data positif yang diprediksi benar.
- *True Negative* (TN) : Merupakan data negatif yang diprediksi benar.
- *False Positive* (FP) — *Type I Error* : Merupakan data negatif namun diprediksi sebagai data positif.
- *False Negative* (FN) — *Type II Error* : Merupakan data positif namun diprediksi sebagai data negatif.

Berdasarkan *confusion matrix*, kita bisa menentukan *accuracy, precision, recall,* dan *F1-score* [11].

1. *Accuracy:* Rasio prediksi benar (positif dan negatif) dibandingkan dengan keseluruhan data. Akan tetapi pada algoritma LVQ-GA *accuracy* yang digunakan adalah pada persamaan 6 (bersifat serupa).

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (9)$$

2. *Precision*: Rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif.

$$Precision = \frac{TP}{TP + FP} \quad (10)$$

3. *Recall:* Rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif.

$$Recall = \frac{TP}{TP + FN} \quad (11)$$

4. *F1-Score:* Mengukur keseimbangan antara *Precision* – *Recall*. Nilai terbaik *F1-Score* adalah 1.0 dan nilai terburuknya adalah 0.

$$F1\ Score = \frac{2 \times Recall \times Precision}{Recall + Precision} \quad (12)$$

3. DESKRIPSI DATA

Dataset yang digunakan dalam penelitian ini berupa data sekunder yang berjudul Diabetes Klasifikasi yang memiliki fitur meliputi *pregnancies*, *glucose*, *blood pressure*, *skin thickness*, *insulin*, BMI, *diabetes pedigree function*, *age*, dan *outcome*. Dataset berikut akan digunakan sebagai *input* baik pada proses pelatihan model maupun pengujian model dengan perbandingan antara data latih dan data uji 85:15.

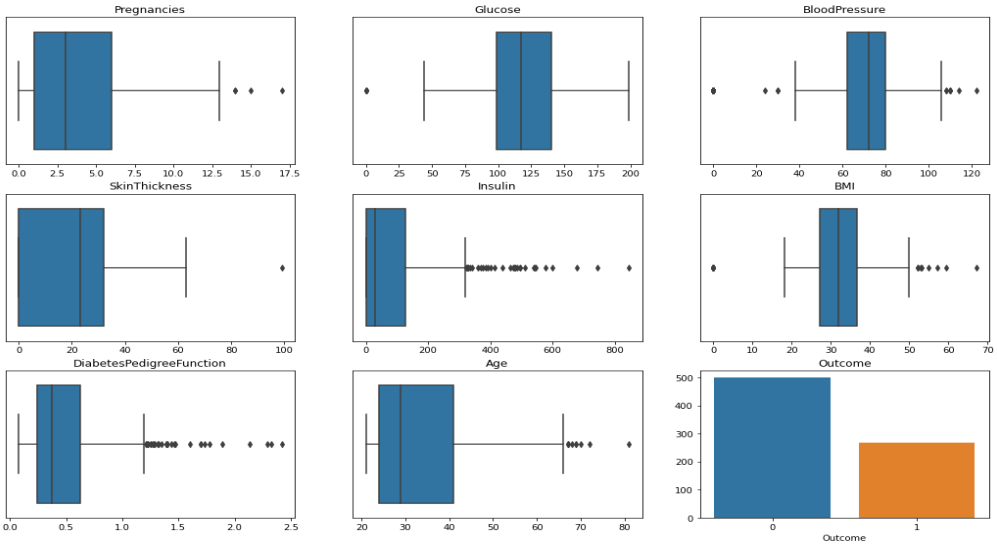
Dataset terdiri dari 768 sampel dan 9 fitur. Fitur yang terdapat dalam *dataset* memiliki jenis yang berbeda-beda. Fitur *pregnancies*, *glucose*, *blood pressure*, *skin thickness*, *insulin*, BMI, *diabetes pedigree function*, dan *age* merupakan data numerik, sedangkan *outcome* merupakan data kategorik. *Outcome* merupakan fitur untuk mengklasifikasi apakah individu tersebut diabetes yang dilabelkan dengan 1 atau tidak diabetes yang dilabelkan dengan 0.

Dataset Diabetes Klasifikasi tidak memiliki *missing value* dan juga *duplicated value*, tetapi dataset memiliki cukup banyak *outliers* jika dilihat dari Gambar 7. Oleh karena itu, perlu dilakukan penanganan *outliers* dengan metode IQR (*Interquartile Range*). Selanjutnya, perlu juga dilakukan *feature scaling* dengan teknik standarisasi agar model mendapat performa yang lebih baik karena dataset memiliki jarak dan satuan yang berbeda-beda jika dilihat dari Tabel 1 dan Gambar 7.

	Fitur	**Keterangan**
1	*Pregnancies*	Jumlah kehamilan
2	*Glucose*	Konsentrasi glukosa plasma

3	*Blood Pressure*	Tekanan darah diastolik (mm/Hg)
4	*Skin Thickness*	Ketebalan lipatan kulit trisep (mm)
5	*Insulin*	Insulin (U/mL)
6	BMI	Indeks berat badan (kg/m2)
7	*Diabetes Pedigree Function*	Nilai kecenderungan diabetes berdasarkan riwayat keluarga
8	*Age*	Usia (tahun)
9	*Outcome*	Diabetes/Tidak Diabetes

Tabel 1: Keterangan fitur dalam dataset.



Gambar 9: Distribusi data dari dataset.

4. IMPLEMENTASI DAN ANALISA HASIL

Preprocessing

Berdasarkan hasil deskripsi data akan dilakukan *preprocessing* terlebih dahulu pada dataset. Penanganan *outliers* dilakukan dengan metode IQR (*Interquartile Range*). IQR merupakan sebuah cara untuk memahami penyebaran sekumpulan angka. IQR memperhitungkan kuartil atas (25% teratas), kuartil bawah (25% terbawah), batas bawah, dan batas atas pada masing-masing fitur. Berikut perhitungan yang terdapat dalam metode IQR:

1. IQR $= Q_3 - Q_1$
2. UB $= Q_3 + (1.5\ x\ IQR)$

Tabel 2 merupakan nilai maksimum dari setiap fitur untuk sebelum dan sesudah dilakukan penanganan *outliers*.

	Pregnancies	***Glucose***	***Blood Pressure***	***Skin Thickness***	***Insulin***	**BMI**	***Diabetes Pedigree Function***	***Age***
Sebelum	17	199	122	99	846	67.1	2.42	81
Sesudah	13.5	199	107	80	318.1	50.5	1.2	66.5

Tabel 2: Sebelum dan sesudah penanganan *outliers*.

Setelah dilakukan penanganan *outliers*, akan dilakukan standarisasi data untuk meningkatkan kualitas data dengan cara menciptakan kekonsistenan di dalam seluruh nilai fitur data sehingga meningkatkan performa model dalam mengklasifikasi penderita diabetes.

Pembagian Data Latih dan Data Uji

Pembagian data latih dan data uji yang akan dilakukan pada penelitian ini adalah dengan rasio 85:15, di mana 85% (652 data) digunakan sebagai data latih dan 15% (116 data) digunakan sebagai sebagai data uji. Pemilihan rasio 85:15 ini dipilih karena jumlah data yang kami miliki tidak banyak dan agar model dapat mendeteksi pola lebih baik sehingga menghasilkan performa model yang lebih baik pula.

Hyperparameter Tuning LVQ

Dengan melakukan *hyperparameter tuning* untuk mencari nilai *hyperparameter* yang paling optimal agar model LVQ dan model LVQ-GA dapat bekerja dengan baik pada data yang akan diprediksi. *Hyperparameter tuning* LVQ dilakukan dengan menginisiasi nilai dari *hyperparameter* seperti pada tabel di bawah ini:

Hyperparameter	**Nilai yang akan di-*tuning***
Learning rate (α)	Bilangan real dalam rentang (0.001, 1)
Decrease alpha (dec α)	Bilangan real dengan rentang (0.1, 0.9)
Minimum alpha (min α)	Bilangan real dengan rentang (10-100, 10-4)
Epochs	Bilangan bulat dengan rentang (50, 200)

Tabel 3 : Rentang nilai *hyperparameter* untuk di-*tuning.*

Proses *Tuning* dilakukan dengan menggunakan *Randomized Search* dengan *5-fold cross validation* dan iterasi sebanyak 200, sehingga dihasilkan 5 nilai parameter mean akurasi terbaik dari data *cross validation* yaitu;

alpha	***dec alpha***	***min alpha***	**epochs**	**mean akurasi**
0.074589	0.1	0.0001	200	75.73%
0.05423	0.491641	0.000056	197	75.72%
0.067427	0.223951	0.000094	193	75.57%
0.06424	0.483451	0.000093	200	75.56%
0.136618	0.132077	0.000034	197	75.41%

Tabel 4 : Hasil 5 nilai parameter terbaik pada proses LVQ.

Pembentukan Model *Learning Vector Quantizer*

Menggunakan metode *learning vector quantizer* dalam pembentukan model, proses *model fit* data latih, dan proses *model predict* data uji menghasilkan performa sebagai berikut:

Accuracy	***F1-Score***	***Precision***	***Recall***
74%	62%	65%	60%

Tabel 5 : Hasil metrik evaluasi menggunakan *hyperparameter* teroptimal saat tuning LVQ.

Pembentukan Model *Learning Vector Quantizer - Genetics Algorithm*

Menggunakan modifikasi model *learning vector quantizer* dengan algoritma optimasi *genetics algorithm* untuk mencari inisiasi bobot awal. Akan digunakan tiga kombinasi *hyperparameter* GA yang berbeda, diperoleh metrik evaluasi pada *data test* sebagai berikut ;

Ukuran Populasi	*Max* Generasi	CR	PC	PM	*Accuracy*	*F1-Score*	*Precision*	*Recall*
50	100	0.5	0.9	0.5	75%	51%	79%	38%
30	20	0.9	0.9	0.1	76%	69%	64%	75%
100	100	0.9	0.9	0.5	79%	70%	70%	70%

Tabel 6 : Hasil metrik evaluasi pada 3 kombinasi *hyperparameter* GA yang berbeda.

Analisa Hasil

Didapat *hyperparameter* terbaik dari proses *hyperparameter tuning* adalah *hyperparameter* dengan *mean* akurasi tertinggi, yaitu 75.73%, dengan parameter sebagai berikut ;

Hyperparameter	***alpha***	***dec alpha***	***min alpha***	**epochs**
Nilai setelah Tuning	0.074589	0.1	0.0001	200

Tabel 7 : *Hyperparameter* terbaik dari hasil *Randomized Search* dengan *5-fold cross validation tuning* dengan 200 iterasi.

Dari Tabel 6 pada pembentukan model *learning vector quantizer - genetics algorithm* didapat hasil metrik evaluasi terbaik ketika parameter GA memiliki ukuran populasi 100, maksimum generasi 100, *crossover rate* 0.9, *probability crossover* 0.9, dan *probability mutation* 0.5.

Berikut adalah perbandingan performa hasil model LVQ dengan model LVQ-GA yang dapat dilihat pada Table 8. Dapat dilihat bahwa model LVQ-GA memiliki performa yang lebih baik jika dibandingkan model LVQ tanpa optimasi bobot.

Hasil Klasifikasi	LVQ	LVQ - GA
Accuracy	74%	79%
F1-Score	62%	70%
Precision	65%	70%
Recall	60%	70%

Tabel 8 : Perbandingan metrik akurasi dari model LVQ dan LVQ-GA

5. KESIMPULAN

Pada proyek ini, telah dikonstruksi model akhir dari LVQ - GA yang dapat diimplementasikan untuk klasifikasi berupa diagnosis penyakit diabetes berdasarkan dataset yang ada. Algoritma LVQ yang merupakan algoritma klasifikasi memerlukan bobot awal dalam proses *training* modelnya. Penentuan bobot awal yang optimal merupakan hal yang esensial demi efektivitas berjalannya algoritma. Oleh karena itu, dengan pengaplikasian algoritma GA yang merupakan algoritma optimasi dapat menghasilkan bobot awal tersebut dengan baik. Hal ini dibuktikan dengan model LVQ - GA yang memiliki performa hasil yang lebih baik dibandingkan dengan model LVQ saja yang dibuktikan dari nilai *accuracy*, *F1-Score, precision,* dan *recall.* Dengan ini, didapatkan hasil akhir dari konstruksi model sehingga menjawab tujuan awal dari proyek.

DAFTAR PUSTAKA

[1] Md. Maniruzzaman, Md. Jahanur Rahman, Benojir Ahammed, and Md. Menhazul Abedin(2020) .Classification and prediction of diabetes disease using machine learning paradigm

[2] Elvia Budianita, Widodo Prijodiprodjo (2013). .Penerapan Learning Vector Quantization (LVQ) untuk Klasifikasi Status Gizi Anak.

[3] I. Abuiziah, N. Shakarneh (2013). A Review of Genetic Algorithm Optimization: Operations and Applications to Water Pipeline Systems.

[4] McKinney, W. (2012). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. O'Reilly Media.

[5] Kim, J., & Marron, J. S. (2015). Data Cleaning: Problems and Current Approaches. Annual Review of Statistics and Its Application, 2, 1-20. doi:10.1146/annurev-statistics-031014-092757

[6] Kohonen, T. (1995). Self-Organizing Maps. Springer-Verlag.

[7] Shmueli, G., Patel, N. R., & Bruce, P. C. (2017). Data Preparation for Data Mining. Morgan Kaufmann.

[8] Mitchell, M. (1998). An Introduction to Genetic Algorithms. MIT Press.

[9] Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. Journal of Machine Learning Research, 13, 281-305.

[10] Raissa Arniantya, Budi Darma Setiawan, dan Putra Pandu Adikara (2018). Optimasi Vektor Bobot Pada *Learning Vector Quantization* Menggunakan Algortime Genetika Untuk Identifikasi Jenis *Attention Deficit Hyperactivity Disorder* Pada Anak.

[11] Resika Arthana (2019). Mengenal Accuracy, Precision, Recall dan Specificity serta yang diprioritaskan dalam Machine Learning.

[12] Kuncahyo Setyo Nugroho (2019). Confusion Matrix untuk Evaluasi Model pada Supervised Learning.

Lampiran

Berikut lampiran program konstruksi model klasifikasi diagnosis penyakit diabetes
https://colab.research.google.com/drive/1aYvkKD3O4JxPP-UKACmw4RqX8INRM3gF?usp=sharing