BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1. Tinjauan Pustaka

Penelitian ini berfokus pada peningkatan kinerja algoritma K-Means dengan mengoptimalkan inisialisasi *centroid* menggunakan pendekatan hibrida metaheuristik. K-Means, meskipun populer karena kesederhanaannya, memiliki kelemahan signifikan terkait sensitivitasnya terhadap penentuan *centroid* awal, yang dapat menyebabkan konvergensi ke optima lokal (Al-Hafiz dkk., 2025; Shamas dkk., 2024; Vrouva dkk., 2025). Beberapa penelitian dalam lima tahun terakhir telah mengeksplorasi penggunaan algoritma optimasi seperti *Genetic Algorithm* (GA) dan *Particle Swarm Optimization* (PSO) baik secara tunggal maupun hibrida untuk menemukan posisi *centroid* yang lebih superior. Tinjauan berikut merangkum perkembangan terkini dari 15 studi relevan yang menjadi dasar bagi penelitian ini.

Tabel 2.1 berikut menyajikan ringkasan dari 15 penelitian relevan yang membahas penggunaan dan hibridisasi algoritma metaheuristik seperti *Genetic Algorithm* (GA) dan *Particle Swarm Optimization* (PSO) dalam konteks K-Means *clustering*. Penelitian-penelitian ini memberikan landasan teoretis dan empiris yang kuat untuk proposal penelitian yang diajukan, menunjukkan tren, efektivitas, dan potensi pengembangan metode hibrida untuk optimasi inisialisasi *centroid*.

Tabel 2.1 Ringkasan Penelitian Terdahulu yang Relevan

No.	Penelitian	Masalah Penelitian	Metode yang Digunakan	Dataset	Evaluasi & Hasil	GAP Analisis
1	Li dkk., 2022	Sensitivitas K-Means terhadap nilai awal (pusat *cluster* awal) pada dataset berdimensi tinggi.	Menggabungkan *fuzzy systems*, PSO, dan GA untuk mengidentifikasi pusat *clustering* awal (*initial clustering centers*) terbaik untuk K-Means.	GPS Taksi dan data urban multi-sumber.	Algoritma FPSO-GAK menunjukkan performa klastering terbaik dibandingkan GAK, PSOK, PSO-GAK, dan K-Means++, dengan nilai SC, SP, dan SSE yang unggul serta konvergensi cepat pada generasi ke-5.	Fokus utama pada data geospasial (*urban hotspots*); tujuannya adalah lokalisasi geografis, bukan segmentasi data multidimensi. Metode menggunakan sistem Fuzzy sebagai komponen tambahan.
2	Wang dkk., (2024)	Menurunkan biaya distribusi dan waktu pengiriman produk segar.	Hybrid APSOGA (GA + PSO) dengan K-Means untuk seleksi lokasi gudang.	Data koordinat distribusi wilayah.	Biaya distribusi berkurang 14,57% dari PSO dan 5,21% dari GA.	Fokus pada optimasi logistik, bukan segmentasi data; belum dievaluasi dengan indeks validasi klaster (Silhouette, DBI).

No.	Penelitian	Masalah Penelitian	Metode yang Digunakan	Dataset	Evaluasi & Hasil	GAP Analisis
3	Abdo dkk., (2024)	Keterbatasan algoritma konvensional (seperti K-Means) yang sensitif terhadap *centroid* awal dan cenderung terjebak dalam *local optima*.	Hybrid SA-PSO-GK++ (*Simulated Annealing*, PSO, *Gaussian Estimation*, dan K-Means++).	Data Medis (*Breast Cancer, Heart Dataset*), CMC, dan *benchmark* Iris.	Integrasi K-Means++ memastikan *centroid* awal yang tersebar, PSO membantu eksplorasi, dan SA membantu lolos dari *local minima*. Hasilnya diukur dengan *Sum of Euclidean Distances* (SED) dan *Normalized Mutual Information* (NMI).	Penelitian ini memvalidasi pada data medis, yang relevan dengan data gizi, tetapi tidak menggunakan GA dalam mekanisme hibrida inisialisasinya.
4	Karishma & Kumar (2024)	Optimasi penjadwalan tugas dalam sistem komputasi terdistribusi.	Model hibrida PSO-GA untuk penjadwalan tugas.	30 kasus dunia nyata dari metode sebelumnya.	*Performance Improvement Ratio* (PIR) meningkat 22,64%, efisiensi	Hybrid PSO-GA (HPSOGAK) digunakan untuk optimasi *task scheduling* pada komputasi terdistribusi,

No.	Penelitian	Masalah Penelitian	Metode yang Digunakan	Dataset	Evaluasi & Hasil	GAP Analisis
					6,93%, waktu respon turun 23,8%.	dengan K-Means sebagai subproses *task clustering* untuk mengurangi komunikasi antar-tugas.
5	Liu dkk., 2024	Memprediksi *output* daya surya PV secara akurat dan kuat, mengatasi sensitivitas K-Means terhadap *centroid* awal.	Model gabungan *optimized data clustering* berbasis iPSO (*Improved* PSO) dengan AdaLSTM *Network*.	Studi kasus dari dua stasiun PV nyata.	RMSE model yang diusulkan dapat berkurang 10,75% dibandingkan model yang menggunakan K-Means *clustering* standar.	Hanya menggunakan optimasi iPSO (tanpa GA) untuk mengoptimalkan *centroid clustering*. Fokus pada prediksi energi (data *time series*), bukan *clustering* multidimensi.
6	Lins Galdino & Da Silva, 2024	Membutuhkan prosedur *clustering* hibrida yang cocok untuk menangani dataset besar.	Hybrid K-Means (untuk membuat *pre-clusters*, e.g., 30) dan *Interval valued data-type*	Iris Dataset (Prosedur dirancang untuk data besar, contoh	Pendekatan *hybrid* terbukti efisien dan dapat menemukan kelompok representatif. Mencapai akurasi 90%	Menggabungkan K-Means dengan *Hierarchical Clustering*, bukan dengan algoritma optimasi *metaheuristik*

No.	Penelitian	Masalah Penelitian	Metode yang Digunakan	Dataset	Evaluasi & Hasil	GAP Analisis
			Hierarchical Clustering (IHCA).	500.000 observasi).	(menggunakan *Interval Ward clustering*).	(GA/PSO) untuk inisialisasi *centroid*. Fokus pada data bernilai interval.
7	Al-Hafiz dkk., (2025)	Meningkatkan akurasi klasifikasi (*phishing detection*) dan mengatasi tantangan pemilihan fitur.	Hybrid K-Means dan Genetic Algorithm (K-Gen PhishGuard). GA digunakan untuk pemilihan fitur yang optimal untuk K-Means *clustering*.	Dataset Kaggle (11.430 URL, 87 fitur).	GA berhasil memilih fitur terbaik, dan K-Means digunakan untuk mengelompokkan data, menghasilkan akurasi klasifikasi sebesar 99% dibandingkan 77,3% tanpa *feature selection*	GA di sini digunakan untuk pemilihan fitur (*feature selection*), yang merupakan tujuan yang berbeda dari inisialisasi *centroid*. Domainnya adalah *cybersecurity*.
8	Hassan dkk., (2025)	Kelemahan K-Means terhadap inisialisasi *centroid* acak dan penentuan nilai k yang optimal.	Hybrid K-Means++ PSO; K-Means++ untuk inisialisasi cerdas, PSO untuk optimasi global.	*Document Clustering* (20 *Newsgroups, Reuters,* WebKB)	Pendekatan ini mencapai *Purity* tinggi (0.95 pada *Reuters*). PSO memiliki kemampuan pencarian	Metode ini menggunakan inisialisasi cerdas (K-Means++) yang ditingkatkan dengan PSO, tetapi tidak

No.	Penelitian	Masalah Penelitian	Metode yang Digunakan	Dataset	Evaluasi & Hasil	GAP Analisis
				dan *Numeric* (Wine).	global, dan K-Means++ meningkatkan kinerja PSO dengan memberikan posisi awal yang lebih terperinci.	melibatkan *Genetic Algorithm* dalam hibrida tersebut. Domainnya fokus pada *text* dan *numeric* umum.
9	Vrouva dkk., 2025	Kurangnya algoritma klasifikasi dan prognosis yang akurat untuk pasien pasca-operasi (RTSA); mencari ukuran *training set* minimum.	Hybrid GAKmeans dan GAClust (*Genetic Algorithm-based Clustering*). GA digunakan untuk optimasi *training set*.	Data klinis pasien *reverse total shoulder arthroplasty* (RTSA).	Optimasi GA menghasilkan kinerja 100% pada *test set* dengan menggunakan hanya 35% data untuk *training* (dibandingkan KNN yang butuh 80%).	K-Means digunakan sebagai komponen dalam konteks klasifikasi dan prognosis medis, dan GA digunakan untuk meminimalkan ukuran *training set*.
10	Li dkk., 2025	Kinerja klasifikasi beban listrik dipengaruhi oleh redundansi data, kompleksitas seleksi	Metode Klasifikasi Beban Listrik Berbasis *Hybrid Clustering* Karakteristik Listrik Kontinu–Diskrit (*K*-	86 perusahaan industri pengguna listrik di Northwest China.	K-prototypes memiliki nilai DBI terkecil (1.397) (peningkatan 0.9% dari K-Means) dan nilai CH Index	Menggunakan hibrida K-prototypes untuk data campuran (kontinu–diskret), yang berbeda dari K-Means untuk data

No.	Penelitian	Masalah Penelitian	Metode yang Digunakan	Dataset	Evaluasi & Hasil	GAP Analisis
		fitur, dan keragaman perilaku konsumsi daya.	*prototypes*). GMM digunakan untuk *clustering* fitur diskrit.		terbesar (19.94) (peningkatan 0,2% dari K-Means).	kontinu murni. Tidak menggunakan GA atau PSO untuk inisialisasi.
11	Zhou & Yan, 2025	Perencanaan jalur tradisional tidak efektif di lingkungan pengiriman yang kompleks dan dinamis (*food delivery*).	Model perencanaan jalur cerdas menggabungkan K-means (untuk analisis *clustering* lokasi pelanggan) dan *Improved Genetic Simulated Annealing Algorithm* (GA-SA) untuk perencanaan jalur.	Data pengiriman makanan (*food delivery*).	Nilai *clustering* pesanan rata-rata 501 pesanan. Akurasi dan *recall* klasterisasi pelanggan masing-masing 90,87% dan 92,18%. Nilai F1 83% untuk perencanaan jalur.	Fokus pada optimasi logistik (perencanaan jalur) di mana K-Means hanya digunakan untuk *clustering* lokasi. Algoritma hibrida (GA-SA) digunakan untuk optimasi rute, bukan *centroid* K-Means
12	Rahimi dkk., 2025	Mengidentifikasi wilayah optimal untuk pembangunan pembangkit energi	Menggunakan algoritma *clustering* (K-Means, DBSCAN, Hierarchical, K-Medoids) yang diintegrasikan dengan	*Australian Towns List* (15.323 kota/desa) dan data potensi energi (*Solar*	Meskipun Genetic K-Means berkinerja terbaik berdasarkan metrik evaluasi *clustering*, Genetic K-	Fokus pada pemilihan lokasi energi (data geospasial/lingkungan). GA digunakan untuk mengidentifikasi metode

No.	Penelitian	Masalah Penelitian	Metode yang Digunakan	Dataset	Evaluasi & Hasil	GAP Analisis
		terbarukan di pedesaan Australia.	*Genetic Algorithm* untuk identifikasi klaster optimal.	*Irradiance, Wind Speed*).	Medoids menghasilkan *output* energi tertinggi (tetapi biaya finansial tertinggi).	*clustering* terbaik, bukan inisialisasi *centroid* GA-PSO.
13	Politi & Tanyel, 2025	Penelitian ini bertujuan meminimalkan keterlambatan pada persimpangan bersinyal berdekatan dengan mengoptimalkan *green time ratio* (g/c) agar antrian kendaraan tidak mengganggu aliran lalu lintas.	Pendekatan Hibrida K-Means *Clustering* dan *Genetic Algorithm* (GA),. K-Means membagi jarak antar persimpangan menjadi tiga klaster (misalnya, 110–160 m sebagai zona transisi),. GA mengoptimalkan rasio g/c untuk meminimalkan *delay*,.	Data lalu lintas sebanyak 930 titik disimulasikan, dikalibrasi, dan divalidasi dengan SIDRA *Intersection*, mencakup jarak (50–200 m), waktu siklus, dan volume kendaraan.	Rasio g/c optimal ditentukan dalam kisaran 0,58 hingga 0,69. Optimasi mengurangi *delay* sebesar 8,95% dan emisi CO2 sebesar 4,72% di persimpangan *downstream*,.	Fokus pada optimasi lalu lintas dan transportation engineering. K-Means digunakan untuk segmentasi fitur input (jarak), bukan untuk inisialisasi *centroid* K-Means. Optimasi Menggunakan K-Means + GA, bukan hibrida GA–PSO terintegrasi untuk optimasi *centroid*.

No.	Penelitian	Masalah Penelitian	Metode yang Digunakan	Dataset	Evaluasi & Hasil	GAP Analisis
14	Gálvez dkk., 2025	Rekonstruksi IFS citra fraktal berwarna; optimasi parameter IFS, warna, dan jumlah fungsi kontraktif optimal (N).	IMCH GAPSO (Hybrid GA-PSO Iteratif) + K-Means (untuk *clustering* warna).	Citra fraktal berwarna.	Rekonstruksi geometri dan warna sangat akurat; Optimasi N otomatis.	Domain berbeda (Rekonstruksi Citra vs. *Clustering* Data Gizi Multidimensi). K-Means digunakan untuk *pre-processing* segmentasi warna, bukan untuk optimasi inisialisasi *centroid*.
15	Suwirmayanti dkk., 2025	K-Means sensitif terhadap inisialisasi *centroid* awal, *outliers*, dan berkinerja buruk di ruang berdimensi tinggi.	Hybrid IWOKM-GA (mengintegrasikan *Invasive Weed Optimization* (IWO), K-Means, dan *Genetic Algorithm* (GA)).	Data perbankan (*German Credit Dataset*) dari UCI.	Menghasilkan Nilai Fungsi Biaya 2400,51, hampir tiga kali lebih baik (menurun hampir 60%) dari KM+GA. Waktu komputasi lebih lambat (328.08 detik).	Tidak melibatkan *Particle Swarm Optimization* (PSO), Fokus pada *clustering* data perbankan.

Berdasarkan tinjauan pustaka pada Tabel 2.1, terlihat jelas bahwa hibridisasi K-Means dengan algoritma metaheuristik adalah area riset yang aktif dan terbukti efektif di berbagai domain. Namun, tinjauan ini juga mengungkap celah penelitian (GAP) yang signifikan yang menjadi landasan penelitian ini. Sebagian besar penelitian yang ada tidak menggunakan hibrida GA atau PSO untuk mengoptimalkan inisialisasi *centroid* K-Means secara langsung. Sebaliknya, K-Means sering digunakan sebagai sub-proses untuk tujuan yang berbeda, seperti segmentasi lokasi untuk optimasi logistik (Wang dkk., 2024; Zhou & Yan, 2025) , segmentasi fitur input (Politi & Tanyel, 2025), atau pra-pemrosesan data citra (Gálvez dkk., 2025). Di sisi lain, penelitian yang menggunakan GA untuk optimasi clustering sering kali menerapkannya untuk tujuan selain inisialisasi *centroid*, seperti seleksi fitur (Al-Hafiz dkk., 2025) atau optimasi ukuran *training set* (Vrouva dkk., 2025).

Penelitian yang paling relevan yang berfokus pada optimasi *centroid* K-Means cenderung menggunakan kombinasi hibrida yang berbeda seperti *Simulated Annealing* (SA), *Invasive Weed Optimization* (IWO), *Differential Evolution* (DE). Meskipun Karishma & Kumar (2024) menggunakan hibrida GA-PSO-K-Means, fokus utamanya adalah optimasi penjadwalan tugas, bukan analisis *clustering* data. Dengan demikian, penelitian ini mengisi celah penelitian atau GAP dengan mengusulkan penerapan hibrida *Genetic Algorithm* (GA) dan *Particle Swarm Optimization* (PSO) secara spesifik dan murni untuk mengatasi masalah fundamental inisialisasi *centroid* K-Means, dan menerapkannya pada domain data gizi nasional multidimensi yang belum dieksplorasi oleh metode-metode hibrida tersebut serta mengevaluasi optimasi dengan *Silhouette Score* dan *Davies-Bouldin Index*.

2.2. Dasar Teori

Landasan teoretis penelitian mencakup konsep *data mining*, algoritma *clustering* K-Means, serta dua algoritma metaheuristik utama, yaitu *Genetic Algorithm* (GA), *Particle Swarm Optimization* (PSO), dan Dataset Gizi Indonesia: Prevalensi dan Konsumsi pada tahun 2023. Selain itu, dibahas pula hibridisasi

kedua algoritma serta metrik evaluasi yang digunakan untuk mengukur kinerja *clustering*.

2.2.1. Data Mining

Data mining adalah proses esensial untuk mengekstraksi informasi dan pola berharga dari kumpulan data besar (*big data*). Seiring meluasnya penerapan teknologi informasi seperti sensor dan perangkat IoT di berbagai sektor, volume data yang dihasilkan menjadi sangat besar. Data ini mencakup berbagai jenis, baik terstruktur maupun tidak terstruktur, yang analisis manualnya menjadi tidak efisien dan rentan terhadap kesalahan (Liu dkk., 2023). Oleh karena itu, teknik *data mining* diperlukan untuk mengubah data mentah menjadi wawasan yang dapat ditindaklanjuti, mendukung pengambilan keputusan di berbagai bidang seperti manufaktur (Liu dkk., 2023), kesehatan (Abdo dkk., 2024), keuangan (Xing, 2024), dan perencanaan pada pemerintahan (Yang & Zheng, 2025).

Secara umum, tugas-tugas dalam *data mining* dapat dikategorikan menjadi dua kelompok utama: prediktif dan deskriptif. Model prediktif bertujuan untuk memprediksi nilai masa depan atau yang tidak diketahui dari suatu variabel, yang mencakup tugas seperti estimasi, prediksi, dan klasifikasi. Sementara itu, model deskriptif bertujuan untuk menemukan pola atau hubungan yang melekat dalam data, yang mencakup tugas seperti klastering dan asosiasi. Terdapat beberapa teknik dalam *data mining* seperti:

1. *Estimation*

Estimation adalah proses memprediksi nilai dari suatu variabel numerik atau kontinu yang tidak diketahui. Tujuannya adalah untuk membangun model yang dapat memperkirakan nilai target berdasarkan sekumpulan variabel input. Sebagai contoh, dalam konteks sistem tenaga, estimasi dapat digunakan untuk memperkirakan konsumsi energi di masa depan berdasarkan data historis dan faktor-faktor relevan lainnya (Wang dkk., 2024). Contoh pada banyak kasus yaitu teknik regresi digunakan untuk tugas estimasi. Terdapat beberapa algoritma *data mining* untuk estimasi yaitu *Linear Regression*, *Neural Network*, dan *Support Vector Machine.*

2. *Prediction*

Prediction atau prediksi sangat mirip dengan estimasi dan klasifikasi, namun berfokus pada peramalan perilaku atau nilai di masa depan (Liu dkk., 2023). Contohnya termasuk memprediksi beban listrik jangka pendek (J. Li dkk., 2025), meramalkan permintaan listrik (Wang dkk., 2024), atau memprediksi hasil klinis pasca-operasi pada pasien (Vrouva dkk., 2025). Model prediksi sering kali memanfaatkan data deret waktu *(time-series)* untuk mengidentifikasi tren dan pola historis yang dapat diekstrapolasi ke masa depan (Mohammed & Ali, 2024). Adapun beberapa algoritma *data mining* untuk prediksi yaitu *Linear Regression, Neural Network,* dan *Support Vector Machine.*

3. *Classification*

Classification atau klasifikasi adalah salah satu tugas *supervised learning* yang bertujuan untuk memetakan item data ke dalam salah satu dari beberapa kelas atau kategori yang telah ditentukan sebelumnya (Lins Galdino & Da Silva, 2024). Proses ini melibatkan pembangunan model berdasarkan data pelatihan yang sudah memiliki label kelas. Model yang telah dilatih kemudian digunakan untuk memprediksi kelas dari data baru yang belum berlabel (Gu dkk., 2025). Beberapa algoritma klasifikasi yang populer diantaranya *Naïve Bayes, K-Nearest Neighbor*, C4.5, ID3, CART, *Linear Discriminant Analysis,* dan *Logistic Regression*.

4. *Clustering*

Clustering adalah tugas *unsupervised learning* yang bertujuan untuk mengelompokkan sekumpulan objek data ke dalam beberapa grup atau *cluster* (Hassan dkk., 2025). Pengelompokan ini didasarkan pada prinsip bahwa objek di dalam satu *cluster* harus memiliki kesamaan yang tinggi (kohesi intra-*cluster*), sementara objek di *cluster* yang berbeda harus memiliki ketidaksamaan yang tinggi (separasi inter-*cluster*) (Azevedo dkk., 2025). Tidak seperti klasifikasi, klastering tidak memerlukan data berlabel, sehingga sangat berguna untuk eksplorasi data dan penemuan pola (Yin dkk., 2024). Algoritma klastering dapat dibagi menjadi dua metode utama yaitu partisi dan hierarki:

a. Metode Partisi (*Partitioning Method*)

Metode partisi membagi dataset menjadi sejumlah *k cluster* yang tidak tumpang tindih, di mana setiap *cluster* direpresentasikan oleh sebuah titik pusat atau *centroid* (Seo dkk., 2025). Algoritma ini bersifat iteratif untuk mengoptimalkan penempatan *centroid* (Gálvez dkk., 2025). Algoritma data mining untuk klastering diantaranya *K-Means, K-Medoids, Self-Organizing Map (SOM),* dan *Fuzzy C-Means.*

b. Metode Hierarki (*Hierarchical Method*)

Metode hierarki menciptakan dekomposisi data secara bertingkat, yang dapat direpresentasikan dalam sebuah struktur pohon yang disebut dendrogram (Lins Galdino & Da Silva, 2024). Metode ini tidak memerlukan penentuan jumlah cluster di awal. Metode hierarki sangat berguna ketika struktur bertingkat dari data ingin dipahami. Namun, metode ini bisa jadi lebih intensif secara komputasi dibandingkan metode partisi, terutama untuk dataset besar.

5. Asosiasi

Asosiasi atau analisis keranjang pasar, adalah teknik *data mining* deskriptif yang digunakan untuk menemukan hubungan atau aturan asosiasi antar item dalam sebuah dataset besar (Liu dkk., 2023). Aturan ini biasanya diekspresikan dalam bentuk "Jika {A} maka {B}", yang berarti kemunculan itemset A sering kali diikuti oleh kemunculan itemset B. Contoh klasik adalah analisis data transaksi ritel untuk menemukan produk yang sering dibeli bersamaan. Adapun algoritma *data mining* untuk asosiasi yaitu *FP-Growth* dan *A Priori*.

2.2.2. K-Means *Clustering*

Algoritma K-Means adalah metode *clustering* partisi yang paling populer dan sederhana (Jia dkk., 2025; Karishma & Kumar, 2024). Algoritma ini bertujuan untuk mempartisi *n* data ke dalam *k cluster* di mana setiap data dimiliki oleh *cluster* dengan *mean (centroid)* terdekat (Hassan dkk., 2025; W. Wang dkk., 2025). Proses ini bersifat iteratif dan bertujuan meminimalkan *Sum of Squared Errors* (SSE) atau varians intra-*cluster* (Bouabdallaoui dkk., 2024; Morakis & Adamopoulos, 2024).

Berikut tahapan dalam algoritma K-Means (Yin dkk., 2024):

1. Menentukan jumlah klaster yang diinginkan.
2. Menentukan nilai *centroid* untuk setiap klaster. Menginisialisasi nilai *centroid* secara acak pada awal iterasi.
3. Gunakan Rumus *Euclidean* untuk menghitung jarak antara setiap data dengan *centroid* masing-masing, yang dapat dihitung dalam rumus (1) berikut:

$$d_{euclidean}(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (1)$$

   Dengan:

   d adalah jarak antara dua titik x dan y;

   n adalah jumlah dimensi/fitur data;

   i adalah indeks dimensi, mulai dari 1 sampai n;

   x_i adalah nilai komponen ke-i dari vektor x;

   y_i adalah nilai komponen ke-i dari vektor y.
4. Menentukan anggota klaster, objek dikelompokkan dengan mempertimbangkan jarak terdekat ke pusat klaster. Proses ini menghitung jarak antara setiap objek dan pusat klaster, dan objek yang memiliki jarak terdekat dengan pusat klaster dianggap sebagai anggota klaster.
5. Jika data masih beralih klaster di langkah kedua, proses akan dilanjutkan hingga tidak ada lagi data yang beralih klaster.

Jarak *Euclidean* adalah metrik jarak yang paling umum dan standar (default) yang digunakan dalam algoritma clustering berbasis *centroid* seperti K-Means (Vrouva dkk., 2025). Kelebihan utamanya terletak pada kesederhanaan matematis dan interpretasi geometrisnya yang intuitif. Metrik ini dirumuskan sebagai akar kuadrat dari jumlah kuadrat perbedaan antara koordinat yang sesuai dari dua titik dalam ruang n-dimensi (Li dkk., 2025). Karena K-Means berupaya mengelompokkan elemen data berdasarkan kemiripan *(similarity)* intrinsik, Jarak *Euclidean* secara efektif berfungsi sebagai ukuran standar untuk ketidakmiripan *(dissimilarity)* antar titik data atau antara titik data dengan *centroid* klaster (Azevedo dkk., 2025). Penerapan dalam algoritma clustering, Jarak *Euclidean* memungkinkan titik data untuk ditugaskan ke *centroid* yang paling dekat dengannya, suatu proses yang secara fundamental bertujuan meminimalkan total

kesalahan kuadrat (*Sum of Squared Errors* - SSE) atau Inersia/WCSS (*Within-Cluster Sum of Squares*) (Wang dkk., 2024).

Penggunaan jarak *Euclidean* secara langsung memperkuat tujuan fungsi objektif dari K-Means. Kualitas klaster dinilai dari seberapa rapat titik-titik data berada di sekitar *centroid* mereka; Jarak *Euclidean* yang lebih rendah mengindikasikan bahwa elemen-elemen klaster lebih rapat dan homogen (Abdo dkk., 2024). Oleh karena itu, bagi algoritma hibrida yang menggunakan K-Means (seperti K-Means + GA dan K-Means + PSO) untuk inisialisasi *centroid* atau untuk evaluasi solusi, Jarak *Euclidean* menjadi metrik yang efisien dan dapat diandalkan untuk data numerik (Wang dkk., 2025). Sebagai contoh, dalam *Hybrid Clustering-Enhanced Brain Storm Optimization* (HC-BSO) untuk perencanaan jalur robot, Jarak *Euclidean* digunakan untuk menghitung jarak antara titik tugas (Gálvez dkk., 2025). Selain itu, Jarak *Euclidean* juga merupakan salah satu dari berbagai metrik jarak yang digunakan oleh algoritma *K-Nearest Neighbors* (KNN) (Vrouva dkk., 2025). Implementasinya yang sederhana dan efisien ini menjadikan Jarak *Euclidean* metrik pilihan dalam model optimasi hibrida di berbagai domain.

K-Means memiliki kelemahan signifikan yaitu sangat sensitif terhadap pemilihan *centroid* awal (Suwirmayanti dkk., 2025; Jia dkk., 2025). Inisialisasi yang buruk dapat menyebabkan algoritma terjebak pada solusi optimal lokal, yang menghasilkan kualitas *cluster* yang suboptimal (Abdo dkk., 2024). Berbagai metode perbaikan telah diusulkan, seperti K-Means++, yang memilih *centroid* awal secara lebih cerdas untuk memastikan penyebaran yang lebih baik (Hassan dkk., 2025). Namun, K-Means++ tidak selalu menjamin solusi optimal global. Oleh karena itu, banyak peneliti beralih ke algoritma optimasi *metaheuristik* untuk menemukan *centroid* awal yang lebih baik (Karishma & Kumar, 2024).

Sebagai upaya memberikan pemahaman yang lebih mendalam, berikut adalah contoh perhitungan manual K-Means langkah demi langkah. Studi kasus algoritma K-means akan diuraikan pada contoh data yang tersaji seperti Tabel 2.2:

Tabel 2.2 Contoh Data yang Akan Diolah

Nama	X	Y
A1	2	10
A2	2	5
A3	8	4
A4	5	8
A5	7	5
A6	6	4
A7	1	2
A8	4	9

Tabel 2.2 menunjukkan contoh dataset yang akan diolah menggunakan algoritma K- means dengan kondisi awal memiliki 8 titik data dengan dua fitur (X, Y). Berikut adalah studi kasus pada tahapan algoritma K-means:

1. Menentukan jumlah partisi/*cluster* dari dataset yang akan dibentuk. Pada kasus ini data akan dikelompokkan menjadi 2 klaster (K=2).
2. Inisialisasi *centroid* awal yakni memilih dua titik data pertama secara acak sebagai *centroid* awal. Pada kasus ini C1 = A1(2, 10) C2 = A2(2, 5).
3. Hitung jarak Euclidean dari setiap titik data ke kedua *centroid* (C1 dan C2) dan menugaskannya ke *cluster* dengan jarak terdekat. Jarak Euclidean antara dua titik (x1, y1) dan (x2, y2) adalah $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$. Karena C1 dan C2 digunakan sebagai *centroid,* contoh penerapan pada C3 sebagai berikut:

   Jarak ke C1, d = $\sqrt{(8-2)^2 + (4-10)^2} = \sqrt{36+36} = \sqrt{72} \approx 8{,}49$

   Jarak ke C2, d = $\sqrt{(8-2)^2 + (4-5)^2} = \sqrt{36+1} = \sqrt{37} \approx 6{,}08$ (Cluster 2)

   Perhitungan terus dilakukan untuk semua data sehingga mendapatkan hasil klaster pada masing-masing *cluster*. Hasilnya ditunjukkan pada Tabel 2.3 berikut:

Tabel 2.3 Iterasi Pertama

Titik	Koordinat	Jarak ke C1(2,10)	Jarak ke C2(2,5)	*Cluster*
A1	(2,10)	0.00	5.00	C1
A2	(2,5)	5.00	0.00	C2
A3	(8,4)	8.49	6.08	C2
A4	(5,8)	3.61	4.24	C1
A5	(7,5)	7.07	5.00	C2
A6	(6,4)	7.21	4.12	C2
A7	(1,2)	8.06	3.16	C2
A8	(4,9)	2.24	4.47	C1

Tabel 2.3 menunjukkan proses iterasi pertama dimana telah teridentifikasi hasil dari perhitungan jarak menggunakan rumus Euclidean, yaitu pada kolom jarak terhadap *centroid* C1 dan C2. Penentuan klaster pada masing-masing objek dengan cara menghitung berdasarkan jarak minimum terhadap *centroid*. Pada data A7(1,2), jarak terdekat adalah dengan *centroid* C2(2,5), sehingga A7 masuk ke dalam *cluster* C2. Sementara itu, data A8(4,9) memiliki jarak terdekat dengan *centroid* C1(2,10), sehingga A8 masuk ke dalam *cluster* C1. Berdasarkan hasil iterasi pertama, diperoleh pembagian klaster yaitu *cluster* C1 terdiri dari (A1, A4, A8) dan *cluster* C2 terdiri dari (A2, A3, A5, A6, A7). Selanjutnya, posisi *centroid* diperbarui dengan menghitung rata-rata koordinat pada tiap klaster, sehingga diperoleh *centroid* baru C1(3.67, 9) dan C2(4.8, 4).

4. Pembaruan *centroid (update)* dihitung ulang posisi *centroid* dengan mengambil rata-rata dari anggota masing-masing cluster:

$$\text{C1 baru, d} = \left(\frac{2+5+4}{3}, \frac{10+8+9}{3}\right) = (3.67, 9)$$

$$\text{C2 baru, d} = \left(\frac{2+8+7+6+1}{5}, \frac{5+4+5+4+2}{5}\right) = (4.8, 4).$$

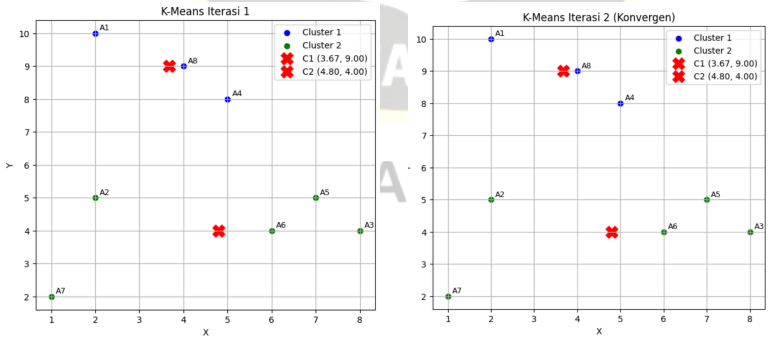
Jadi, *centroid* yang digunakan untuk iterasi selanjutnya adalah (3.67, 9) dan (4.8, 4). Perhitungan jarak terdekat selanjutnya sama dengan perhitungan jarak pada iterasi pertama. Jarak Euclidean ditunjukkan pada Tabel 2.4 berikut:

Tabel 2.4 Iterasi Kedua

Titik	Koordinat	Jarak ke C1(3.67,9)	Jarak ke C2(4.8,4)	*Cluster*
A1	(2,10)	1,94	6,53	C1
A2	(2,5)	4,54	3,00	C2
A3	(8,4)	6,64	3,20	C2
A4	(5,8)	1,67	4,02	C1
A5	(7,5)	5,25	2,42	C2
A6	(6,4)	5,51	1,20	C2
A7	(1,2)	7,46	4,31	C2
A8	(4,9)	0,33	5,06	C1

Tabel 2.4 menunjukkan bahwa pada iterasi kedua sudah tidak berubah antara Iterasi 1 dan Iterasi 2 artinya algoritma telah konvergen sehingga proses iterasi K-Means dihentikan.

5. Jika data sudah tidak terdapat yang berpindah dari satu klaster ke klaster lainnya, maka perhitungan K-Means dihentikan. Namun, jika masih terdapat data yang berpindah maka langkah perhitungan diulang hingga semua data stabil pada klasternya. Ilustrasi algoritma K-Means ditunjukkan pada Gambar 2.1 (a) dimana data A1–A8 telah dikelompokkan menjadi dua klaster. Gambar 2.1 (b) menunjukkan iterasi kedua setelah pembaruan *centroid*, dimana hasil klaster tetap sama seperti iterasi pertama. Hal ini menandakan algoritma telah mencapai konvergensi sehingga proses K-Means berhenti.



(a) (b)

Gambar 2.1 Ilustrasi Algoritma K-Means

2.2.3. Algoritma Optimasi *Metaheuristik*

Algoritma *metaheuristik* adalah metode optimasi stokastik yang terinspirasi dari alam dan mampu menemukan solusi mendekati optimal untuk masalah *NP*-

hard yang kompleks (Xing, 2024). Algoritma berbasis populasi *(swarm intelligence)* seperti *Genetic Algorithm* (GA) dan *Particle Swarm Optimization* (PSO) sangat populer karena kemampuan pencarian globalnya yang kuat (Politi & Tanyel, 2025; Šešum-Čavić dkk., 2024).

Genetic Algorithm (GA) atau Algoritma Genetika adalah sebuah metode pencarian dan optimasi metaheuristik yang terinspirasi oleh teori evolusi alamiah Charles Darwin, yaitu *"survival of the fittest"* (individu terkuat yang bertahan hidup) (Karishma & Kumar, 2024). GA termasuk dalam kategori algoritma evolusioner yang bekerja dengan memanipulasi sebuah populasi solusi potensial secara berulang untuk menemukan solusi optimal atau mendekati optimal dari suatu permasalahan (Al-Hafiz dkk., 2025). GA sangat cocok untuk memecahkan masalah optimasi yang kompleks, non-linear, dan memiliki ruang pencarian yang luas, di mana metode optimasi tradisional mungkin kesulitan (Schnellenbach-Held dkk., 2025).

Setiap solusi potensial dalam GA direpresentasikan sebagai individu atau kromosom. Kromosom ini terdiri dari serangkaian gen yang menyandikan parameter-parameter dari solusi tersebut. Kualitas atau "kebaikan" dari setiap individu diukur menggunakan fungsi *fitness*, yang pada dasarnya adalah fungsi objektif dari masalah yang ingin dioptimalkan (Vrouva dkk., 2025). Individu dengan nilai *fitness* yang lebih tinggi dianggap sebagai solusi yang lebih baik dan memiliki probabilitas lebih besar untuk bereproduksi dan mewariskan sifat genetiknya ke generasi berikutnya (Karishma & Kumar, 2024).

Evolusi populasi dari satu generasi ke generasi berikutnya terjadi melalui tiga operator genetik utama yaitu seleksi (*selection*), pindah silang (*crossover*), dan mutasi (*mutation*) (Zhu dkk., 2024). Melalui proses iteratif ini, populasi secara bertahap bergerak menuju solusi yang lebih optimal. Tahapan yang diperlukan pada saat menggunakan *genetic algorithm* adalah sebagai berikut (Schnellenbach-Held dkk., 2025):

1. Inisialisasi Populasi

Inisialisasi populasi *(population initiation)* merupakan langkah pertama yang fundamental dalam *Genetic Algorithm*, yang berfungsi sebagai titik awal

proses optimasi. Tahap ini melibatkan pembentukan populasi awal individu atau kromosom yang umumnya dibangkitkan secara acak *(randomly formed)* (Suwirmayanti dkk., 2025), tujuannya adalah agar populasi awal dapat meliputi wilayah yang luas dari ruang solusi *(covers a large region of the solution space)*. Setiap kromosom merepresentasikan satu solusi potensial yaitu seperangkat *centroid* awal K-Means yang lengkap (Xing, 2024). Solusi ini harus dienkode ke dalam format kromosom yang sesuai; misalnya, untuk parameter yang berada di ruang kontinu (seperti posisi *centroid*), pengkodean bilangan riil digunakan, berbeda dengan masalah diskret seperti seleksi fitur yang sering menggunakan vektor biner.

Ukuran Populasi (*Npop*) sebesar 10-100 individu merupakan parameter kunci yang secara langsung memengaruhi keragaman solusi (*diversity of solutions*) yang dieksplorasi (Grefenstette, 1986). Pemilihan nilai ini didasarkan pada pertimbangan efisiensi komputasi mengingat ruang pencarian centroid pada penelitian ini relatif terdefinisi dengan baik, sehingga populasi berukuran kecil dinilai cukup untuk menghasilkan kandidat solusi yang representatif tanpa beban komputasi yang berlebihan (Saifullah & Drężewski, 2024). Meskipun populasi berukuran lebih besar secara teoritis mampu meningkatkan keragaman genetik, nilai 10 dipilih sebagai titik keseimbangan antara eksplorasi ruang solusi yang memadai dan efisiensi waktu eksekusi, sejalan dengan pendekatan yang diterapkan pada studi optimasi hibrida berskala dataset serupa (Bouabdallaoui dkk., 2024; Suwirmayanti dkk., 2025).

2. Evaluasi *Fitness*

Evaluasi nilai *fitness* dilakukan dengan menilai setiap individu dalam populasi yaitu kandidat solusi berupa set *centroid*, menggunakan fungsi *fitness*. Fungsi ini berperan untuk mengukur seberapa baik kualitas solusi yang diwakili oleh individu tersebut (Hassan dkk., 2025). Konteks penelitian ini yaitu clustering, di mana tujuan utamanya adalah meminimalkan nilai *Sum of Squared Errors* (SSE) atau *Within-Cluster Sum of Squares* (WCSS) (Li dkk., 2025) nilai fitness yang lebih rendah justru menandakan solusi yang lebih baik (Schnellenbach-Held dkk., 2025). Fungsi *fitness* yang digunakan identik dengan fungsi objektif, dan proses optimasi

difokuskan pada pencarian nilai minimum dari fungsi tersebut. Oleh karena itu, tidak diperlukan transformasi inversi terhadap fungsi objektif, karena pendekatan yang digunakan secara langsung bertujuan untuk meminimalkan error dalam pembentukan klaster (Wang dkk., 2024). Berikut adalah rumus (2) untuk *Sum of Squared Errors:*

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i^t||^2 \quad (2)$$

Dengan:

SSE adalah *Sum of Squared Errors* atau total *varians intra-klaster*;

k adalah jumlah klaster yang ditentukan;

C_i adalah himpunan titik data yang termasuk dalam klaster ke-i;

x adalah titik data individual yang merupakan anggota dari klaster C_i;

μ_i^t adalah titik pusat *(centroid)* dari klaster ke-i;

$||x - \mu_i^t||^2$adalah jarak kuadrat *(squared Euclidean distance)* antara titik data x dengan *centroid* klaster μ_i^t.

3. Seleksi *(Selection)*

Seleksi *(selection)* merupakan operator genetik utama dalam *Genetic Algorithm* yang menerapkan prinsip *"Survival of the Fittest"* (Karishma & Kumar, 2024). Proses ini menentukan kromosom yang akan dipertahankan dan diregenerasi berdasarkan nilai *fitness*, sehingga individu dengan kualitas lebih tinggi memiliki peluang lebih besar untuk menjadi *parent* bagi generasi berikutnya (Xing, 2024). Tujuannya adalah menjaga ukuran populasi sambil meningkatkan proporsi individu berkualitas, sebagai bagian dari rangkaian operasi genetik bersama *crossover* dan mutasi untuk membentuk populasi yang lebih optimal (Suwirmayanti dkk., 2025).

Terdapat beberapa metode yang umum digunakan untuk melakukan pemilihan kromosom induk. Salah satu pendekatan yang diidentifikasi adalah pendekatan seleksi *roulette-wheel (roulette-wheel selection approach)*, yang digunakan dalam berbagai penelitian (Liu dkk., 2023). Metode lain yang sering diterapkan dan dianggap tangguh dalam optimasi adalah Seleksi Turnamen *(tournament selection)*, yang digunakan sebagai algoritma seleksi dalam beberapa studi, termasuk pembaruan model elemen hingga *(Finite Element Model Updating)* (Schnellenbach-Held dkk., 2025). Di samping metode utama ini, mekanisme

Elitisme juga sering diaktifkan, di mana individu terbaik dari suatu generasi secara otomatis menghasilkan klon *(clones)* untuk dipertahankan di generasi berikutnya (Morakis & Adamopoulos, 2024). Operator seleksi bekerja berdasarkan kriteria seleksi yang telah ditentukan *(predefined selection criteria)* (Politi & Tanyel, 2025a), memastikan bahwa gen-gen terbaik memiliki peluang terbesar untuk membentuk generasi yang akan datang.

4. Pindah Silang (*Crossover*)

Crossover adalah operator genetik yang sangat penting dalam *Genetic Algorithm*, berfungsi sebagai mekanisme utama untuk menghasilkan kromosom keturunan *(offspring)* melalui penggabungan informasi genetik dari dua kromosom induk *(parents)* yang telah dipilih. Tujuan teoretis dari operasi ini adalah memastikan pewarisan gen unggul dari solusi yang ada (eksploitasi) dan secara simultan meningkatkan keragaman populasi *(population diversity)*. Peningkatan keragaman genetik ini sangat krusial karena memungkinkan algoritma untuk menjelajahi ruang solusi yang lebih luas dan kompleks yang merupakan prasyarat vital untuk mendukung kapabilitas pencarian global GA dan mencegah algoritma terjebak dalam optima lokal. Probabilitas *Crossover (Pc)* merupakan parameter berskala antara 0 dan 1, secara langsung mengontrol frekuensi di mana operasi penggabungan genetik ini diterapkan pada kromosom di setiap generasi.

Beberapa literatur menunjukkan bahwa nilai crossover yang efektif umumnya berada pada rentang 0,6 hingga 0,95 (Xing, 2024), karena rentang tersebut mampu menjaga keseimbangan antara eksplorasi dan eksploitasi (Hassan dkk., 2025). Penetapan *Pc* = 0,7 (70%) dipilih untuk memaksimalkan rekombinasi genetik antar individu dengan fitness yang baik, sehingga proses pencarian solusi dapat menjelajahi ruang solusi lebih luas dan menemukan kombinasi yang lebih beragam (Gálvez dkk., 2025). Nilai ini dianggap ideal karena tidak terlalu rendah hingga memperlambat konvergensi, namun juga tidak terlalu tinggi hingga kehilangan stabilitas, sehingga algoritma dapat mencapai konvergensi yang cepat dan hasil yang lebih konsisten terhadap solusi berkualitas tinggi (Karishma & Kumar, 2024).

5. Mutasi (*Mutation*)

Mutasi *(mutation)* merupakan operator genetik sekunder yang memiliki fungsi krusial dalam memelihara keragaman genetik *(genetic diversity)* di antara populasi dari satu generasi ke generasi berikutnya (Gálvez dkk., 2025). Fungsi utama mutasi adalah memperkenalkan perubahan acak yang kecil *(small changes)* pada kromosom. Mekanisme eksplorasi tersebut sangat penting karena tujuannya adalah untuk mencegah algoritma berkonvergensi secara prematur *(prevent premature convergence)* dan membantu melarikan diri dari optima lokal *(escape local optima)* yang mungkin telah ditemukan (Chen dkk., 2025). Mutasi memastikan bahwa semua ruang pencarian memiliki probabilitas bukan-nol untuk dieksplorasi (Gálvez dkk., 2025). Namun, operator ini harus digunakan dengan sangat hati-hati, sebab jika Probabilitas Mutasi *(Pm)* terlalu tinggi, hal itu akan merusak solusi yang baik yang telah ditemukan, berpotensi mengubah GA menjadi pencarian acak *(random search)*. Oleh karena itu, *Pm* harus dipertahankan pada nilai yang sangat rendah (Politi & Tanyel, 2025b). Berbagai studi menunjukkan *Pm* sebagai konstanta yang rendah, seperti 0,01 atau 0,05, atau setinggi 0,1 dalam kasus tertentu (Zhu dkk., 2024).

Penetapan Probabilitas Mutasi menggunakan persamaan 3 berikut:

$$Pm = \frac{1}{5 \times len(bounds)} \quad (3)$$

Penetapan tersebut mengimplementasikan strategi penyesuaian yang cermat terhadap dimensi masalah. Konsep tersebut sejalan dengan prinsip teoretis GA di mana *Pm* yang efektif sering kali diambil secara berbanding terbalik terhadap ukuran populasi (Gálvez dkk., 2025) (dalam konteks ini, dimensi masalah atau *len(bounds)* bertindak sebagai penanda kompleksitas ruang solusi). Konteks optimasi, solusi direpresentasikan oleh vektor posisi atau parameter yang memiliki batas atas *(upper bounds)* dan batas bawah *(lower bounds)* (Tsipi dkk., 2025). Ketika dimensi masalah *(len(bounds))* meningkat, kromosom menjadi lebih panjang dan ruang pencarian menjadi lebih luas. Melalui implementasi *Pm* secara terbalik proporsional terhadap dimensi, algoritma memastikan bahwa tingkat perubahan acak yang diperkenalkan oleh mutasi per gen tetap sangat rendah (Gálvez dkk., 2025). Strategi ini secara efektif mengendalikan gangguan pada

kromosom, menjaga stabilitas dan kualitas solusi terbaik yang ada, sambil tetap memberikan peluang yang diperlukan untuk eksplorasi minor yang terkendali, yang merupakan kunci untuk mencapai konvergensi yang cepat dan stabil (Schnellenbach-Held dkk., 2025).

6. Pembentukan Generasi Baru dan Terminasi

Individu-individu baru yang dihasilkan dari *crossover* dan mutasi membentuk populasi untuk generasi berikutnya. Seluruh proses dari Tahap 2 hingga 5 diulangi secara iteratif. Proses ini akan berhenti jika salah satu kriteria terminasi terpenuhi, seperti: jumlah generasi maksimum telah tercapai, nilai *fitness* dari solusi terbaik tidak menunjukkan perbaikan signifikan selama beberapa generasi, dan solusi yang memuaskan telah ditemukan. Setelah terminasi, individu dengan nilai *fitness* tertinggi dalam populasi akhir dianggap sebagai solusi terbaik (atau mendekati optimal) untuk masalah tersebut.

Sedangkan *Particle Swarm Optimization* (PSO) adalah algoritma optimasi metaheuristik global yang terinspirasi oleh perilaku sosial kawanan burung atau ikan dalam mencari makanan. Diperkenalkan pertama kali oleh Kennedy dan Eberhart pada tahun 1995, PSO mensimulasikan bagaimana individu-individu dalam sebuah kelompok (kawanan atau *swarm*) berkolaborasi dan berbagi informasi untuk menemukan solusi optimal dari suatu permasalahan. Setiap solusi potensial dalam ruang pencarian direpresentasikan sebagai sebuah partikel, dan keseluruhan partikel ini membentuk sebuah kawanan (*swarm*).

Setiap partikel memiliki dua atribut utama yaitu posisi (yang merepresentasikan sebuah solusi) dan kecepatan (yang menentukan arah dan besaran pergerakan partikel di ruang pencarian). Proses pencarian yang bersifat iteratif, setiap partikel menyesuaikan gerakannya berdasarkan dua jenis informasi penting:

1. Pengalaman pribadi terbaik (*personal best* atau *pbest*): posisi terbaik yang pernah dicapai oleh partikel itu sendiri sejauh ini.
2. Pengalaman global terbaik (*global best* atau *gbest*): posisi terbaik yang pernah dicapai oleh seluruh kawanan partikel. Melalui mekanisme ini, partikel-partikel secara kolektif bergerak menuju wilayah yang paling menjanjikan dalam ruang

pencarian, menyeimbangkan antara eksplorasi (pencarian di area baru) dan eksploitasi (penyempurnaan solusi di area yang sudah diketahui baik).

Proses PSO bersifat iteratif. Pada setiap iterasi, kecepatan dan posisi setiap partikel diperbarui. Misalkan dalam ruang pencarian berdimensi d, kecepatan dan posisi partikel ke-i pada iterasi ke-k+1 diperbarui menggunakan rumus berikut (K. Li dkk., 2024):

Rumus (4) Pembaruan Kecepatan:

$$v_i^{k+1} = w \cdot v_i^k + c_1 \cdot r_1 \cdot (pbest_i - x_i^k) + c_2 \cdot r_2 \cdot (gbest - x_i^k) \quad (4)$$

Rumus (5) Pembaruan Posisi:

$$x_i^{k+1} = x_i^k + v_i^{k+1} \quad (5)$$

Dengan:

v_i^k adalah kecepatan partikel ke-i pada iterasi ke-k;

x_i^k adalah posisi partikel ke-i pada iterasi ke-k;

w adalah faktor inersia (mengontrol kontribusi kecepatan sebelumnya);

c_1, c_2 adalah koefisien akselerasi (untuk *cognitive* dan *social component*);

r, r_2 adalah bilangan acak *uniform* [0,1];

$pbest_i$ adalah posisi terbaik yang pernah dicapai partikel ke-i;

$gbest$ adalah posisi terbaik global yang dicapai oleh seluruh partikel.

Setelah posisi diperbarui, kualitas solusi dievaluasi menggunakan fungsi *fitness*. Fungsi ini bergantung pada masalah yang dioptimalkan. Berdasarkan konteks inisialisasi *centroid* K-Means, *fitness* bisa berupa *Sum of Squared Errors* (SSE), di mana nilai yang lebih rendah menunjukkan solusi yang lebih baik (Suwirmayanti dkk., 2025). Proses ini diulang hingga kriteria berhenti (misalnya, jumlah iterasi maksimum atau konvergensi *fitness*) terpenuhi.

Algoritma PSO telah diterapkan secara luas dalam berbagai domain, termasuk penjadwalan (Karishma & Kumar, 2024), analisis citra medis (Saifullah & Drężewski, 2024), dan optimasi jaringan (Tsipi dkk., 2025), karena beberapa keunggulannya. Selain itu, PSO memiliki struktur yang lugas dan mudah diimplementasikan dibandingkan algoritma metaheuristik lainnya seperti *Genetic Algorithm*. Melalui mekanisme sosialnya, PSO mampu menjelajahi ruang solusi yang luas untuk menemukan solusi mendekati optimal global. Umumnya, PSO

memiliki laju konvergensi yang lebih cepat dibandingkan GA, membuatnya cocok untuk masalah yang membutuhkan solusi dalam waktu singkat (Karishma & Kumar, 2024).

Kelemahan utama PSO adalah kecenderungannya untuk terjebak pada solusi optimal lokal, terutama pada masalah yang kompleks atau multimodal. Hal itu terjadi karena partikel cenderung terlalu cepat tertarik ke arah *gbest* (Wang dkk., 2024). Menyeimbangkan antara eksplorasi global dan eksploitasi lokal adalah tantangan. Pengaturan parameter seperti bobot inersia (ω) sangat krusial, jika tidak diatur dengan baik, algoritma bisa gagal menemukan solusi yang baik (Karishma & Kumar, 2024).

Menggabungkan PSO dengan GA adalah pendekatan yang populer untuk memanfaatkan kekuatan kedua algoritma. GA, dengan operator *crossover* dan mutasinya, dapat meningkatkan keragaman populasi dan membantu PSO keluar dari optima lokal (Chen dkk., 2025). Penelitian dalam konteks *clustering*, PSO sering digunakan untuk mengoptimalkan pemilihan *centroid* awal K-Means, yang merupakan kelemahan utama dari K-Means standar (Bouabdallaoui dkk., 2024). PSO dapat mencari ruang solusi untuk menemukan set *centroid* awal yang superior, sehingga K-Means dapat mencapai hasil *cluster* yang lebih akurat dan stabil. Misalnya, beberapa penelitian menggunakan K-Means++ (versi K-Means yang lebih cerdas) untuk menginisialisasi partikel dalam kerangka PSO, yang kemudian digunakan untuk optimasi lebih lanjut (Šešum-Čavić dkk., 2024). Penelitian lain, seperti oleh Karishma & Kumar (2024), secara eksplisit menggunakan hibrida PSO-GA untuk memperbarui *centroid*, yang kemudian digunakan sebagai inisialisasi untuk K-Means, terbukti meningkatkan efisiensi dan kualitas hasil.

Secara keseluruhan, PSO adalah teknik optimasi berbasis *swarm intelligence* yang kuat dan serbaguna. Meskipun memiliki beberapa keterbatasan, kinerja dan keandalannya dapat ditingkatkan secara signifikan melalui berbagai modifikasi dan hibridisasi dengan teknik lain, menjadikannya alat yang sangat berharga dalam penelitian ini untuk mengoptimalkan inisialisasi *centroid* K-Means.

2.2.4. Hibridisasi *Genetic Algorithm* dan *Particle Swarm Optimization* (GA-PSO) K-Means

Hibridisasi metaheuristik bertujuan untuk menggabungkan kekuatan dari dua atau lebih algoritma untuk mengatasi kelemahan masing-masing. Pendekatan hibrida GA-PSO telah terbukti efektif dalam berbagai domain, mulai dari penjadwalan tugas hingga perencanaan rute, karena mampu menyeimbangkan eksplorasi dan eksploitasi dengan lebih baik. Karakteristik masing-masing algoritma tersebut disajikan dalam Tabel 2.5 berikut:

Tabel 2.5 Perbandingan Karakteristik GA dan PSO

Karakteristik	***Genetic Algorithm* (GA)**	***Particle Swarm Optimization* (PSO)**
Kekuatan Utama	Eksplorasi global yang kuat, menjaga keragaman populasi (Karishma & Kumar, 2024).	Konvergensi cepat, eksploitasi lokal yang efisien (Saifullah & Drężewski, 2024).
Kelemahan	Konvergensi cenderung lambat (Karishma & Kumar, 2024).	Rentan terhadap konvergensi prematur ke optimal lokal (Hassan dkk., 2025).
Operator Utama	Seleksi, *Crossover*, Mutasi (Karishma & Kumar, 2024).	Pembaruan kecepatan dan posisi berdasarkan *pbest* dan *gbest* (K. Li dkk., 2024).
Aliran Informasi	Antar-individu dalam satu generasi (melalui *crossover*) (Karishma & Kumar, 2024).	Searah menuju partikel terbaik (*gbest*) (Šešum-Čavić dkk., 2024).

Penelitian ini menggunakan GA untuk menjelajahi ruang solusi secara luas, sementara PSO dapat digunakan untuk menyempurnakan pencarian di dalam wilayah-wilayah tersebut untuk menemukan solusi optimal secara cepat. Sinergi ini

mengatasi konvergensi lambat GA dan kecenderungan PSO untuk terjebak pada optima lokal.

Solusi terbaik GA tersebut kemudian digunakan sebagai inisialisasi untuk PSO. Melalui proses tersebut, PSO tidak lagi bergerak dari posisi acak, melainkan diarahkan pada wilayah solusi yang lebih potensial. Selanjutnya, PSO menjalankan mekanisme pembaruan iteratif yang melibatkan *personal best* (pbest) dan *global best* (gbest) untuk menyempurnakan posisi *centroid* secara lebih presisi. Hasil akhir dari PSO berupa gbest, yaitu *centroid* yang paling optimal berdasarkan evaluasi fungsi objektif. *Centroid* ini kemudian digunakan sebagai titik awal K-Means *clustering*. Melalui hasil *centroid* yang lebih representatif, K-Means dijalankan hingga konvergen untuk menghasilkan partisi data yang lebih stabil, akurat, dan *robust*.

Evaluasi hasil *clustering* dalam penelitian ini dilakukan menggunakan metrik validasi internal, mengingat dataset kecukupan gizi tidak memiliki label kelas *(ground truth)*. Metrik yang digunakan meliputi *Silhouette Score, Sum of Squared Errors (SSE),* dan *Davies-Bouldin Index (DBI)*. Hasil dari model hybrid GA–PSO kemudian dibandingkan dengan K-Means konvensional, K-Means + GA, dan K-Means + PSO untuk menilai peningkatan akurasi, stabilitas, serta kualitas partisi data yang dihasilkan.

2.2.5. Metrik Evaluasi *Clustering*

Clustering adalah metode *unsupervised learning*, evaluasi kinerjanya tidak dapat menggunakan metrik seperti akurasi yang memerlukan label kebenaran. Oleh karena itu, digunakan metrik validasi internal yang mengukur kualitas struktur *cluster* berdasarkan data itu sendiri (Seo dkk., 2025). Metrik yang umum digunakan yaitu *Silhouette Score* dan *Davies-Bouldin Index (DBI).*

Silhouette Score adalah metrik yang digunakan untuk mengevaluasi kualitas hasil dari algoritma *clustering* (pengelompokan data) (Hassan dkk., 2025). Tujuan utamanya adalah untuk mengukur seberapa baik sebuah titik data (atau objek) dikelompokkan dalam clusternya sendiri dibandingkan dengan cluster lain yang terdekat (Abdo dkk., 2024).

Metrik ini memberikan gambaran tentang dua hal penting dalam sebuah cluster:

1. Kohesi *(Cohesion)*: Seberapa mirip sebuah objek dengan anggota lain dalam clusternya sendiri. Diukur sebagai jarak rata-rata antara objek tersebut dengan semua titik lain dalam cluster yang sama (Tsipi dkk., 2025).
2. Pemisahan *(Separation)*: Seberapa berbeda atau jauh sebuah objek dengan cluster tetangga terdekatnya. Diukur sebagai jarak rata-rata objek tersebut ke semua titik di cluster terdekat lainnya (Mohammed & Ali, 2024).

Nilai *silhouette score* berkisar antara -1 hingga +1. Nilai mendekati +1 menunjukkan bahwa objek tersebut sangat cocok dengan klasternya dan sangat berbeda dari *cluster* tetangga. Hal tersebut adalah hasil pengelompokan yang baik. Nilai mendekati 0 menandakan bahwa objek berada sangat dekat dengan batas antara dua *cluster* (*cluster* yang tumpang tindih). Sedangkan nilai mendekati -1 menunjukkan bahwa objek tersebut kemungkinan besar ditempatkan di cluster yang salah (Surampudi & Kumudham, 2024).

Rumus (6) *Silhouette Score,* untuk setiap titik data tunggal i, *silhouette score* $s(i)$ dihitung dengan rumus berikut (Hassan dkk., 2025):

$$s(i) = \frac{b(i)-a(i)}{\max(a(i),b(i))} \tag{6}$$

di mana rumus (7) $a(i)$ adalah rata-rata jarak titik iii ke semua titik dalam cluster yang sama (kohesi).

$$a(i) = \frac{1}{|C_i|-1}\sum_{j\in C_i, i\neq j} d(i,j) \tag{7}$$

rumus (8) $b(i)$ adalah jarak rata-rata terendah dari titik iii ke semua titik di cluster lain (pemisahan).

$$b(i) = min_{k\neq i}\frac{1}{|C_k|}\sum_{j\in C_k} d(i,j) \tag{8}$$

$|C_i|$ adalah jumlah titik data dalam *cluster* C_i.

Silhouette score untuk keseluruhan dataset adalah nilai rata-rata dari $s(i)$ untuk semua titik data. Secara keseluruhan, semakin tinggi skor siluet rata-rata, semakin baik kualitas pengelompokannya, karena ini menunjukkan bahwa cluster yang terbentuk padat secara internal (*compact*) dan terpisah dengan baik (*well-separated*) satu sama lain.

Metrik selanjutnya yaitu *Davies-Bouldin Index* (DBI) adalah metrik internal yang digunakan untuk mengevaluasi kualitas algoritma *clustering* (pengelompokan data) (Seo dkk., 2025). Berbeda dengan *Silhouette Score* yang mengukur kualitas cluster dari perspektif setiap titik data, DBI mengevaluasi kualitas pengelompokan secara keseluruhan dengan mengukur kemiripan rata-rata antara setiap cluster dengan cluster lain yang paling mirip dengannya (Li dkk., 2025).

Prinsip utama di balik DBI adalah bahwa cluster yang baik haruslah padat secara internal (*compact*) dan terpisah dengan baik (*well-separated*) dari cluster lainnya. DBI bekerja dengan cara (Šešum-Čavić dkk., 2024):

1. Mengukur kepadatan internal *(intra-cluster similarity)* yaitu menghitung seberapa tersebar titik-titik data di dalam sebuah cluster dari pusatnya (sentroid). Semakin kecil nilai ini, semakin padat clusternya.
2. Mengukur pemisahan antar-cluster *(inter-cluster separation)* yakni mengukur jarak antara pusat (sentroid) dari dua cluster yang berbeda. Semakin besar jarak ini, semakin baik pemisahannya.

Nilai DBI yang lebih rendah menunjukkan hasil pengelompokan yang lebih baik, karena ini berarti cluster-cluster yang terbentuk lebih padat dan lebih jauh terpisah satu sama lain. Tidak seperti *Silhouette Score*, DBI tidak memiliki batas atas yang pasti, namun nilai terendahnya adalah nol. Metrik ini sangat berguna ketika label kelas sebenarnya tidak tersedia, sehingga evaluasi harus dilakukan secara internal (Seo dkk., 2025).

Rumus untuk *Davies-Bouldin Index* (DBI) dihitung menggunakan rumus (9) (Šešum-Čavić dkk., 2024):

$$DBI = \frac{1}{n}\sum_{i=1}^{n} max_{j \neq i} \left(\frac{S_i + S_j}{d(c_i, c_j)}\right) \quad (9)$$

di mana;

n adalah jumlah klaster

i dan j adalah indeks untuk setiap klaster, dari 1 hingga n.

S_i adalah kerapatan *intra-cluster* (rata-rata jarak) dari semua titik data dalam *cluster* i ke pusatnya C_i.

S_j adalah kerapatan *intra-cluster* dari semua titik data dalam cluster j ke pusatnya

(C_j)

$d(c_i, c_{j)}$ adalah pemisahan antar-*cluster*, yaitu jarak (biasanya *Euclidean*) antara pusat *(centroid)* cluster i dan j.

Setiap *cluster* yaitu mencari *cluster* lain yang "paling mirip" dengannya (rasio kepadatan terhadap pemisahan yang tertinggi), lalu merata-ratakan nilai-nilai terburuk ini untuk semua *cluster*.