E-ISSN : 3031-9692 Vol. 1 No. 3 2024
Institut Teknologi dan Bisnis STIKOM Bali, 02-10-2024

Penerapan Algoritma Jaya Untuk Optimasi Pusat Klaster K-Means

Batara FS Simangunsong[1)], Ni Luh Gede Pivin Suwirmayanti[2)], I Made Ari Santosa[3)]
Sistem Informasi[1)],Studi Sistem Komputer[2,3)]
Institut Teknologi dan Bisnis STIKOM Bali
Denpasar, Indonesia
e-mail: 200030268@stikom-bali.ac.id[1)], pivin@stikom-bali.ac.id[2)], arisantosamade@gmail.com[3)]

Abstrak

K-Means merupakan teknik pengelompokan berbasis partisi yang merepresentasikan setiap klaster melalui nilai rata-rata dari anggotanya. Salah satu tantangan dalam teknik ini adalah bahwa proses iterasi yang berupaya mencapai optimalitas tidak selalu berhasil mencapai solusi global yang optimal, dikarenakan proses tersebut sangat bergantung pada pemilihan titik centroid. Untuk mengatasi permasalahan konvergensi suboptimal pada algoritma K-Means yang terkait dengan pemilihan awal centroid, penelitian ini mengusulkan pengintegrasian Algoritma Jaya. Algoritma Jaya, yang dikenal dengan kemampuannya dalam optimasi global yang cepat dan efektif, digunakan untuk mengoptimalkan centroid K-Means. Hasil pengujian pada tiga jumlah klaster berbeda menunjukkan bahwa Jaya K-Means secara konsisten mencatatkan nilai Sum of Squared Errors (SSE) yang lebih rendah dibandingkan dengan K-Means tradisional, namun dengan waktu eksekusi yang lebih lama. Untuk klaster=2, SSE Jaya K-Means memperoleh 306.74 dengan waktu 2.9 detik, sedangkan K-Means SSE 307.75 dengan waktu 0.06 detik. Pada klaster=4, SSE Jaya K-Means memperoleh 243.59 dengan waktu 3.7 detik, berbanding dengan K-Means dengan SSE 247.15 dengan waktu 0.12 detik. Untuk klaster=6, Jaya K-Means mencapai SSE 211.44 dengan waktu 3.56 detik, sedangkan K-Means SSE 215.79 dengan waktu 0.18 detik. Keseluruhan hasil ini menegaskan bahwa pendekatan hibrid K-Means dan Jaya menawarkan perbaikan signifikan dalam kinerja klastering, meskipun dengan pengorbanan pada kecepatan eksekusi.

Kata kunci: *K-Means, Jaya, Jaya K-Means, SSE, Breast Cancer*

1. Pendahuluan

Klastering adalah teknik penting dalam *Data Mining(DM)* [1]. Prosesnya dilakukan dengan data di kelompokkan ke dalam kelas yang berbeda-berbeda tanpa label. Objek dengan karakteristik yang mirip akan ditempatkan dalam kelas yang sama yang disebut dengan klaster. Dan objek yang tidak mirip dipisahkan ke dalam klaster yang berbeda [5]. Penerapan klastering telah banyak digunakan berbagai bidang seperti pariwisata, industri otomotif, pengembangan perkotaan, dan jaringan nirkabel seluler [2].

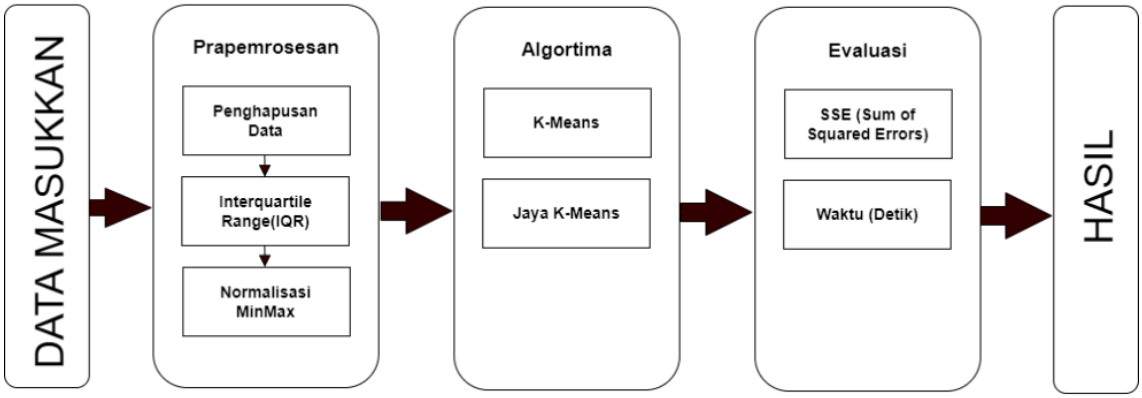
Berdasarkan prinsip dasar, klastering dibagi menjadi beberapa kategori seperti hirarkis, berbasis partisi, berbasis kerapatan (*density*), berbasis model, berbasis *grid*, dan klastering *modern* [3]. Klastering partisi memiliki beberapa metode, namun yang mendominasi adalah K-Means terutama karena kemudahan implementasi dalam pembelajaran tanpa pengawasan (*unsupervised learning*). Namun, K-Means memiliki beberapa kelemahan salah satunya adalah pemilihan acak pusat klaster (*centroid*) awal yang berpotensi masuk kedalam perangkap optimum lokal sehingga menghasilkan konvergensi prematur [4].

Penelitian terdahulu telah menunjukkan potensi integrasi antara K-Means dan berbagai algoritma metaheuristik. Peneliti [5] mengkombinasikan *Dynamic Artificial Cromosomes Genetic Algorithm* dengan K-Means dalam menentukan nilai *centroid* awal dan diuji dengan *sum of squared error (SSE)* dan *Davies Bouldin Index(DBI)*. Selanjutnya [6] menggunakan inisial *centroids* dari algoritma Genetika untuk K-Means dan membandingkan nilai DBI dan waktu eksekusi algoritma. Particle Swarm Optimization (PSO) digabungkan dengan K-Means oleh [7] untuk memberikan keseimbangan antara kemampuan pencarian solusi global dan kecepatan konvergensi yang dapat mempercepat proses segmentasi. Terakhir peneliti [8] mengoptimasi K-Means dengan *Invasive Weed Optimization (IWO)* dan *Genetic Algorithm (GA)* untuk menghindari klastering yang lemah yang disebabkan oleh pemilihan acak pusat klaster pada fase inisialisasi, dan untuk meningkatkan akurasi klastering dan diuji menggunakan SSE dan waktu eksekusi.

Dengan mempertimbangkan studi terdahulu tersebut, penelitian ini akan mengembangkan implementasi Algoritma Jaya untuk meningkatkan efektivitas dalam pengelompokan data, terutama dalam

menangani konvergensi lokal pada K-Means. Algoritma kemudian dievaluasi menggunakan SSE dan menghitung waktu eksekusi algoritma.

2. Metode Penelitian



Gambar 1. Desain Penelitian perbandingan K-Means dengan Jaya K-Means

2.1 *Dataset*

Dalam penelitian ini *dataset* yang digunakan adalah data publik yaitu *dataset Breast Cancer* yang dapat di akses dari website kaggle. Tabel 1 merupakan deskripsi dari *dataset* tersebut. *Dataset Breast Cancer* banyak digunakan dengan algoritma K-Means [9], untuk itu pengujian algoritma menggunakan *dataset* ini.

Tabel 1. Deskripsi Dataset

Dataset	Jumlah Fitur	Jumlah Data	Tipe Data	Deskripsi
Wisconsin Diagnostic Breast Cancer	33	569	Real	Tumor payudara dan hasil biopsi dari pasien.

2.2 Prapemrosesan *Dataset*

Sebelum *dataset* diolah menggunakan algoritma, penting untuk melakukan prapemrosesan untuk kesiapan dan meningkatkan kinerja algoritma. Menurut [10] beberapa langkah yang dapat dilakukan dalam prapmerosesan *dataset*. Kolom 'id' dapat dihapus karena tidak menyediakan informasi substantif mengenai karakteristik atau perilaku subjek data. Sedangkan Kolom yang berisi nilai *Not a Number (NAN)* merupakan data yang nilainya tidak terdefinisi, sehingga tidak bisa digunakan. Selanjutnya kolom 'Target' karena tujuan utama algoritma yang digunakan adalah klastering untuk menemukan kelompok-kelompok alami dalam data tanpa mempertimbangkan label atau hasil yang sudah diketahui, sehingga bisa dihapus.

Metode statistik yang kokoh untuk mendeteksi *outliers*, yang tidak terlalu sensitif terhadap keberadaan *outliers* itu sendiri adalah *Interquartile Range (IQR)* pada Rumus 1. Kuartil dari suatu *dataset* dibagi menjadi empat bagian, masing-masing mengandung 25% dari data.

$$IQR = Q3\text{-}Q1 \tag{1}$$

Pada Rumus 1, IQR sebagai selisih antara kuartil ketiga (Q3) dan pertama (Q1), yang digunakan untuk menentukan batas atas dan bawah dengan menambahkan atau mengurangi 1.5 kali IQR dari Q3 dan Q1, memungkinkan identifikasi *outlier* dengan efektif.

Terkahir, prapemrosesan data yang dilakukan adalah normalisasi Min–Max. Metode ini menormalisasi data dengan cara melihat seberapa besar nilai variabel melebihi nilai minimum ($min(X)$) dan menormalisasi perbedaan ini dengan rentang nilai menggunakan Rumus 2.

$$X^* = \frac{X - min(X)}{max(X) - min(X)} \tag{2}$$

Proses normalisasi nilai X menjadi X^* dengan mengurangi nilai minimum dan membaginya dengan rentang dari nilai minimum ke maksimum, menghasilkan nilai yang sudah dinormalisasi dalam

skala seragam. Proses ini memudahkan penanganan data dengan skala yang berbeda-beda dalam analisis data.

2.3 K-Means

K-Means adalah algoritma yang sederhana dan efektif untuk menemukan kelompok dalam data. [11], [12] menjelaskan langkah-langkah algoritma K-Means sebagai berikut:

1. Langkah 1: Masukkan berapa banyak klaster *K* yang diinginkan untuk membagi data.
2. Langkah 2: Secara acak menetapkan setiap nilai k sebagai lokasi awal *centroid*.
3. Langkah 3: Untuk setiap titik data, cari *centroid* yang paling dekat.
4. Langkah 4: Untuk setiap k klaster, hitung dan perbarui lokasi *centroid* ke nilai baru.
5. Langkah 5: Ulangi langkah 3 hingga 5 sampai konvergensi atau terminasi.

2.4 Jaya K-Means

Sebuah algoritma berbasis metaheuristik baru yang disebut algoritma Jaya menggabungkan fitur-fitur dari *Evolutionary Algorithms (EA)* dalam hal ketahanan (*the fittest*) serta *Swarm Intelligence (SI)* di mana sekawanan umumnya mengikuti pemimpin selama pencarian solusi optimal. Algoritma Jaya diusulkan oleh [13] pada tahun 2016. Berikut adalah bentuk *Pseudocode* Algoritma Jaya K-Means:

Pseudocode 1 : Algoritma Jaya K-Means

```
      Input:
      N : Ukuran populasi
      T : Jumlah iterasi/generasi
      K : Jumlah klaster
      Output: x_1 − centroid terbaik
 1:   Inisialisasi populasi N solusi secara acak
 2:   Hitung fitness f(X_i)                ∀_i= 1,2, … ,N
 3:   Urutkan populasi: (x_1 dan x_N) solusi terbaik dan terburuk
 4:   t = 1
 5:   while (t ≤ T) do
 6:     for i = 1, … , N do
 8:          Set r_1 ∈ [0,1]; Set r_2 ∈ [0,1]
 9:          x_i' = x_i + (r_1(x_1 − x_i) − r_2(x_N − x_i))
10:          if f(x_i') < f(x_i) then
11:              x_i = x_i'
12:          end if
13:     end for
14:     Urutkan populasi: (x_1 dan x_N)
15:     x_k−means ← K-Means(x_N)
16:     if f(x_k−means) < f(x_1) then
17:          x_1 = x_k−means
18:     end if
19:     t = t + 1
20:  end while
21:  return x_1
```

1. Inisialisasi Populasi

Proses inisialisasi populasi melibatkan pemilihan nilai *centroids* awal untuk setiap solusi secara acak distribusi seragam dari himpunan nilai $\{1,2,\cdots,K\}$. Populasi dibentuk sebanyak $N = 50$ yang berisikan solusi [14].

2. Fungsi *Fitness*

Fungsi *fitness* berperan penting dalam masalah optimasi. Fungsi *fitness* mengukur seberapa baik solusi yang dihasilkan, sehingga setiap solusi memiliki tujuan dan sering disebut dengan *objective function* . Solusi yang terbaik adalah solusi yang memiliki nilai minimal dari *sum of squared error (SSE)* [15] pada Rumus 3.

$$SSE = \sum_{i=1}^{n}\sum_{j=1}^{d}\left(x_{ij} - c_{kj}\right)^2 \quad (3)$$

Rumus 3 menambahkan kuadrat dari selisih antara setiap titik data x_{ij} dan *centroid cluster* c_{kj} dihitung untuk setiap dimensi d dari semua data n dalam klaster.

3. Operator Jaya

Proses evolusi Jaya dilakukan iterasi demi iterasi berdasarkan solusi terbaik dan terburuk, sehingga x_i dari semua solusi dalam *Jaya Memory(JM)* [13], [16] dan mengalami perubahan menggunakan operator Jaya yang dimodifikasi untuk klastering pada Rumus 4.

$$x_i' = x_i + (r1(x_1 - x_i) - r2(x_N - x_i)) \quad (4)$$

Rumus 4 x_i' menunjukkan nilai baru dari elemen ke-i yang diupdate dengan menggunakan pengaruh dari solusi terbaik x_1 dan terburuk x_N pada dimensi yang sama. $r1$ dan $r2$ adalah nilai acak antara 0 sampai 1.

4. K-Means

Melakukan K-Means pada solusi terburuk yang digunakan untuk meningkatkan efisiensi algoritma klastering dengan menghitung centroid baru pada solusi [14].

2.5 Evaluasi

Pengujian dilakukan dengan SSE dengan Rumus 3 dan performa eksekusi waktu di bandingkan [5], [8]. Pada penelitian ini dimulai dari klaster 2 sampai klaster ke 6 dengan beberapa iterasi untuk melihat konsistensi algoritma.

3. Hasil dan Pembahasan

Hasil pengujian algoritma ditampilkan perklaster untuk melihat nilia SSE dan waktu yang dibutuhkan untuk eksekusi algoritma. Kemudian dilanjutkan dengan nilai rata-rata kinerja masing-masing algoritma.

3.1 Klaster 2

Pada Tabel 2 , K-Means memiliki waktu komputasi yang lebih cepat dan konstan, sekitar 0.1 detik per iterasi, dengan sedikit fluktuasi dalam SSE. Sebaliknya, Jaya K-Means menunjukkan konsistensi yang lebih baik dalam SSE yang stabil di semua iterasi, tetapi dengan waktu komputasi yang lebih lama, berkisar antara 1.7 hingga 3.8 detik.

Tabel 2. Pengujian pada Klaster 2

	K-Means		Jaya K-Means	
Iterasi	SSE	Waktu(Detik)	SSE	Waktu(Detik)
1	308.003329647993155	0.1	306.742357513760226	2.3
2	308.003329647993155	0.1	306.742357513760226	3.8
3	306.746430683205233	0.1	306.742357513760226	3
4	308.007338324104467	0	306.742357513760226	1.7
5	308.003329647993155	0	306.742357513760226	3.7

3.2 Klaster 4

Pada Tabel 3, K-Means memiliki variasi SSE dari 244.45 hingga 252.35 dengan waktu komputasi yang singkat, sekitar 0.1 detik, sementara Jaya K-Means menunjukkan SSE yang stabil sekitar 243.59 dengan waktu komputasi yang lebih lama, antara 2.4 hingga 5.3 detik per iterasi. Terlihat algoritma K-Means menghasilkan SSE yang kurang stabil.

Tabel 3. Pengujian pada Klaster 4

	K-Means		Jaya K-Means	
Iterasi	SSE	Waktu(Detik)	SSE	Waktu(Detik)
1	248.585604205624179	0.2	243.595903451767612	2.7
2	244.447506448852323	0.1	243.588950639090740	2.4
3	245.664574505442232	0.1	243.595903451767583	4.6
4	252.351124924424141	0.1	243.588950639090740	3.5

5	244.709009183040621	0.1	243.588950639090740	5.3

3.3 Klaster 6

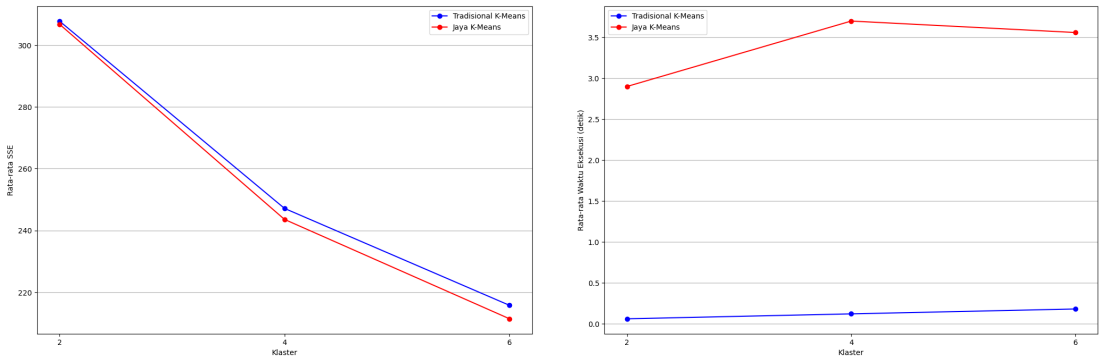
Pada Tabel 4, K-Means menunjukkan variasi SSE dari 214.55 hingga 218.57 dengan waktu komputasi yang cepat, antara 0.1 dan 0.4 detik per iterasi. Sebaliknya, Jaya K-Means memiliki SSE yang sangat stabil sekitar 211.43 tetapi dengan waktu komputasi yang lebih lama, berkisar dari 1.4 hingga 5.3 detik per iterasi.

Tabel 4. Pengujian pada Klaster 6

	K-Means		Jaya K-Means	
Iterasi	SSE	Waktu(Detik)	SSE	Waktu(Detik)
1	215.942935718390856	0.4	211.429198278640598	2.7
2	214.546005527241704	0.1	211.447594753406321	1.4
3	215.269840832936552	0.2	211.455160431150489	3.7
4	214.629193178942273	0.1	211.431796775528511	5.3
5	218.568309979867365	0.1	211.437204660147728	4.7

3.4 Rata-rata SSE dan Waktu(Detik) Eksekusi

K-Means dan Jaya K-Means menunjukkan perbedaan dalam SSE dan waktu eksekusi untuk tiga jumlah klaster yang berbeda. Untuk klaster=2, K-Means memiliki rata-rata SSE 307.75 dengan waktu 0.06 detik, sementara Jaya K-Means lebih rendah pada 306.74 dengan waktu yang lebih lama yaitu 2.9 detik. Pada klaster=4, K-Means mencatat SSE 247.15 dan waktu 0.12 detik, berbanding dengan Jaya K-Means yang mencapai 243.59 SSE dengan waktu 3.7 detik. Terakhir, untuk klaster=6, K-Means memiliki SSE 215.79 dan waktu 0.18 detik, sedangkan Jaya K-Means menunjukkan SSE lebih rendah yaitu 211.44 namun dengan waktu 3.56 detik. Secara umum, Jaya K-Means konsisten menunjukkan SSE lebih rendah namun dengan waktu eksekusi yang lebih panjang dibandingkan dengan K-Means seperti yang terlihat pada Gambar 2.



Gambar 2. Rata rata SSE dan Waktu Eksekusi Algoritma

4. Kesimpulan

Dalam evaluasi antara metode K-Means dan Jaya K-Means terhadap tiga kelompok klaster yang berbeda (K=2, K=4, K=6), Jaya K-Means konsisten unggul dalam menghasilkan nilai SSE yang lebih rendah dibandingkan dengan K-Means tradisional, menandakan kemampuannya yang lebih efektif dalam meminimalkan kesalahan pengelompokan. Secara khusus, penurunan nilai SSE terlihat pada semua kelompok klaster, menegaskan bahwa Jaya K-Means lebih akurat dalam mengelompokkan data sesuai dengan kesamaan karakteristiknya. Namun, keunggulan ini diimbangi dengan waktu eksekusi yang lebih lama pada Jaya K-Means, yang mencerminkan *trade-off* antara akurasi dan efisiensi waktu.

Daftar Pustaka

[1] A. E.-S. Ezugwu, M. B. Agbaje, N. Aljojo, R. Els, H. Chiroma, Dan M. A. Elaziz, "A Comparative Performance Study Of Hybrid Firefly Algorithms For Automatic Data Clustering," *Ieee Access*, Vol. 8, Hlm. 121089–121118, 2020, Doi: 10.1109/Access.2020.3006173.

[2] A. Belhadi, Y. Djenouri, K. Nørvåg, H. Ramampiaro, F. Masseglia, Dan J. C.-W. Lin, "Space–Time Series Clustering: Algorithms, Taxonomy, And Case Study On Urban Smart Cities," *Engineering*

Applications Of Artificial Intelligence, Vol. 95, Hlm. 103857, Okt 2020, Doi: 10.1016/J.Engappai.2020.103857.

[3] C. Zhang, W. Huang, T. Niu, Z. Liu, G. Li, Dan D. Cao, "Review Of Clustering Technology And Its Application In Coordinating Vehicle Subsystems," *Automot. Innov.*, Jan 2023, Doi: 10.1007/S42154-022-00205-0.

[4] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, Dan J. Heming, "K-Means Clustering Algorithms: A Comprehensive Review, Variants Analysis, And Advances In The Era Of Big Data," *Information Sciences*, Vol. 622, Hlm. 178–210, Apr 2023, Doi: 10.1016/J.Ins.2022.11.139.

[5] M. Mursalim, P. Purwanto, Dan M. A. Soeleman, "Penentuan Centroid Awal Pada Algoritma K-Means Dengan Dynamic Artificial Chromosomes Genetic Algorithm Untuk Tuberculosis Dataset," *Tc*, Vol. 20, No. 1, Hlm. 97–108, Feb 2021, Doi: 10.33633/Tc.V20i1.4230.

[6] R. Kurniati, O. Arsalan, Dan Y. Ramadhana, "Initial Centroid Determination Using Genetic Algorithm In Data Clustering," 2021.

[7] G. Himabindu, Ch. Raghu Kumar, Ch. Hemanand, Dan N. Rama Krishna, "Withdrawn: Hybrid Clustering Algorithm To Process Big Data Using Firefly Optimization Mechanism," *Materials Today: Proceedings*, Hlm. S2214785320378846, Des 2020, Doi: 10.1016/J.Matpr.2020.10.273.

[8] N. L. Gede Pivin Suwirmayanti, I. K. Gede Darma Putra, M. Sudarma, I. M. Sukarsa, Dan E. Setyaningsih, "Iwokm-Ga Hybrid Method To Improve Clustering Accuracy In Banking Data," *Int. J. Com. Dig. Sys.*, Jun 2024, Doi: Http://Dx.Doi.Org/10.12785/Ijcds/Xxxxxx.

[9] S. Naveen, N. V. Kashyap, V. P. Kulkarni, S. A, Dan M. S. Chakradhar, "Breast Cancer Prediction Using Unsupervised Learning Technique K-Means Clustering Algorithm," Dalam *2023 2nd International Conference On Vision Towards Emerging Trends In Communication And Networking Technologies (Vitecon)*, Vellore, India: Ieee, Mei 2023, Hlm. 1–6. Doi: 10.1109/Vitecon58111.2023.10157765.

[10] D. T. Larose, *Discovering Knowledge In Data: An Introduction To Data Mining*. Hoboken, N.J: Wiley-Interscience, 2005.

[11] N. L. G. P. Suwirmayanti Dan I. G. A. D. Saryanti, "Penerapan Teknik Clustering Untuk Pengelompokan Konsetrasi Mahasiswa Dengan Metode K-Means," *Sintesa*, Vol. 2, Nov 2019, Doi: 10.36002/Snts.V0i0.884.

[12] M. Ahmed, R. Seraj, Dan S. M. S. Islam, "The K-Means Algorithm: A Comprehensive Survey And Performance Evaluation," *Electronics*, Vol. 9, No. 8, Hlm. 1295, Agu 2020, Doi: 10.3390/Electronics9081295.

[13] R. Venkata Rao, "Jaya: A Simple And New Optimization Algorithm For Solving Constrained And Unconstrained Optimization Problems," *10.5267/J.Ijiec*, Hlm. 19–34, 2016, Doi: 10.5267/J.Ijiec.2015.8.004.

[14] K. Krishna Dan M. Narasimha Murty, "Genetic K-Means Algorithm," *Ieee Trans. Syst., Man, Cybern. B*, Vol. 29, No. 3, Hlm. 433–439, Jun 1999, Doi: 10.1109/3477.764879.

[15] D. Mustafi, A. Mustafi, Dan G. Sahoo, "A Novel Approach To Text Clustering Using Genetic Algorithm Based On The Nearest Neighbour Heuristic," *International Journal Of Computers And Applications*, Vol. 44, No. 3, Hlm. 291–303, Mar 2022, Doi: 10.1080/1206212x.2020.1735035.

[16] R. A. Zitar, M. A. Al-Betar, M. A. Awadallah, I. A. Doush, Dan K. Assaleh, "An Intensive And Comprehensive Overview Of Jaya Algorithm, Its Versions And Applications," *Arch Computat Methods Eng*, Vol. 29, No. 2, Hlm. 763–792, Mar 2022, Doi: 10.1007/S11831-021-09585-8.