E-ISSN: 2715-2731


# Analisis Data Penggunaan *Block storage* Untuk Rekomendasi Penyeimbangan Beban Kerja Aplikasi Telekomunikasi Menggunakan Klasterisasi

Fred Erick Soaloan Sembiring[1,*], Diana Purwitasari[2]

Program Studi Magister Manajemen Teknologi, Institut Teknologi Sepuluh Nopember, Indonesia

[1]fredericksembiring@gmail.com; [2]diana@its.ac.id

*penulis korespondensi

INFO ARTIKEL

**Sejarah Artikel**
Diterima: 9 Juli 2025
Direvisi: 28 Juli 2025
Diterbitkan: 31 Agustus 2025

**Kata Kunci**
Beban kerja
*Block storage*
DBSCAN
Klasterisasi
K-Means

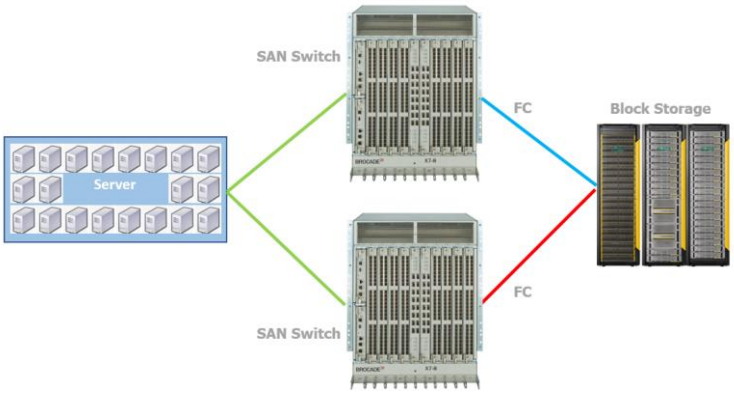
ABSTRAK

Perkembangan teknologi informasi yang pesat mendorong peningkatan kebutuhan akan sistem penyimpanan data yang andal dan efisien, khususnya pada sektor industri telekomunikasi. PT XYZ sebagai salah satu perusahaan telekomunikasi terbesar di Indonesia menghadapi tantangan terkait variasi beban kerja pada 32 *server block storage* yang mereka miliki. Perbedaan performa *server* akibat variasi metrik seperti IOPS, *service time*, dan *bandwidth* berpotensi menyebabkan ketidakseimbangan beban serta penurunan efisiensi infrastruktur TI. Menariknya PT XYZ belum memiliki alat atau metodologi yang efektif untuk menganalisis performa *server-server* tersebut. Penelitian ini bertujuan untuk menganalisis dan mengkategorikan beban kerja *server block storage* di PT XYZ menggunakan dua metode klasterisasi K-Means dan DBSCAN. Metode penelitian meliputi pengumpulan data performa *server*, proses preprocessing data, penerapan algoritma *clustering*, serta evaluasi hasil klasterisasi menggunakan ground truth sebagai acuan validasi. Hasil penelitian menunjukkan bahwa kedua metode mampu mengelompokkan *server* ke dalam tingkatan beban kerja, namun DBSCAN terbukti lebih akurat dengan tingkat akurasi mencapai 87,72%, dibandingkan K-Means yang hanya mencapai 23,60%. Selain itu, DBSCAN juga efektif dalam mengidentifikasi *server* dengan pola beban kerja anomali sebagai noise, yang tidak terdeteksi oleh K-Means. Kesimpulan dari penelitian ini adalah bahwa metode DBSCAN lebih direkomendasikan untuk analisis beban kerja *server block storage* di PT XYZ guna mendukung strategi penyeimbangan beban kerja dan optimalisasi penggunaan sumber daya TI secara lebih efisien.

## PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi (TIK) dalam beberapa dekade terakhir telah mengalami akselerasi yang luar biasa. Transformasi digital kini menjadi keniscayaan bagi berbagai sektor industri, mulai dari perbankan, manufaktur, kesehatan, hingga telekomunikasi. Salah satu dampak paling signifikan dari fenomena ini adalah meningkatnya volume data secara eksponensial, yang dikenal dengan istilah data explosion [1], [2]. Ledakan data ini memicu kebutuhan mendesak akan infrastruktur teknologi informasi yang mampu tidak hanya menyimpan, tetapi juga mengelola, memproses, dan menganalisis data dalam jumlah besar secara cepat dan efisien. Tanpa adanya manajemen data yang baik, organisasi akan kesulitan dalam menjaga efisiensi operasional dan meningkatkan mutu layanan. Salah satu management data yang baik adalah penggunaan *block storage* yang menyediakan akses data dalam bentuk blok-blok mentah, yang memungkinkan pengolahan data secara cepat dan efisien [3], [4]. Keunggulan ini membuat

*block storage* menjadi pilihan utama untuk infrastruktur basis data yang membutuhkan kecepatan tinggi. Pada penelitian ini *server* yang terhubung dengan *block storage* melalui SAN (*Storage Area Network*), SAN menggunakan *interface fiber channel* yang memiliki kapasitas hingga 16Ghz. Gambar 1 dibawah adalah gambaran umum topologi *server block storage* yang terhubung ke *server* aplikasi menggunakan teknologi SAN.




Gambar 1. Topologi *Server block storage* menggunakan *Storage Area Network*

Seiring meningkatnya kompleksitas dan volume data yang harus dikelola, muncul pula tantangan baru dalam optimalisasi kinerja *block storage*. Setiap aplikasi yang berjalan di atas infrastruktur penyimpanan memiliki karakteristik beban kerja yang berbeda, mulai dari pola input/output (I/O patterns), kebutuhan *throughput*, hingga sensitivitas terhadap latensi [5], [6]. Kapasitas *server* dalam memproses data ditentukan oleh beberapa metrik utama, seperti waktu layanan (*service time*), jumlah operasi input/output per detik (IOPS), dan lebar pita (*bandwidth*) yang tersedia [7], [8]. Variasi pada metrik-metrik ini sering kali menjadi penyebab utama terjadinya perbedaan kapabilitas antar *server*.

PT XYZ, sebuah perusahaan telekomunikasi terkemuka yang melayani lebih dari 56 juta pelanggan dan menempatkan transformasi digital sebagai bagian integral dari strategi bisnisnya, termasuk pengembangan aplikasi layanan pelanggan berbasis data dan digitalisasi proses internal. Dalam mewujudkan ini sangat bergantung pada infrastruktur TI yang andal, termasuk 32 *server block storage* yang menjadi wadah utama penyimpanan dan pemrosesan data. PT XYZ menghadapi permasalahan serius terkait variasi beban kerja antar *server block storage*, namun sayangnya perusahaan belum memiliki alat atau metodologi yang efektif untuk menganalisis performa *server-server* tersebut. Ketiadaan sistem monitoring dan analisis yang komprehensif mengakibatkan kurangnya pemahaman atas pola beban kerja aktual *server*, sehingga keputusan penempatan aplikasi atau pengalihan beban sering kali tidak didasarkan pada data yang valid, melainkan hanya mengandalkan asumsi atau pengalaman sebelumnya. Dampak dari kondisi tersebut dapat menyebabkan beberapa *server* mengalami kelebihan beban (*overutilized*), yang berdampak pada menurunnya performa aplikasi, sementara *server* lainnya justru kurang dimanfaatkan (*underutilized*), yang mengakibatkan pemborosan sumber daya TI [9], [10].

Untuk mengatasi permasalahan tersebut diperlukan adanya strategi penyeimbangan beban kerja antara *server* dengan menganalisis keseluruhan beban dalam membaca dan menulis sehingga dapat secara efektif meningkatkan response time, *throughput* dan waktu Iops [11]. Digunakan pendekatan berbasis data yang dapat menganalisa dengan cara mengidentifikasi dan mengkategorikan *server block storage* berdasarkan profil kinerjanya. Salah satu teknik yang dapat digunakan adalah metode clustering atau pengelompokan data[12]. Teknik ini telah banyak digunakan dalam berbagai studi untuk menemukan pola tersembunyi dan mengelompokkan objek berdasarkan kesamaan karakteristik [13], [14],

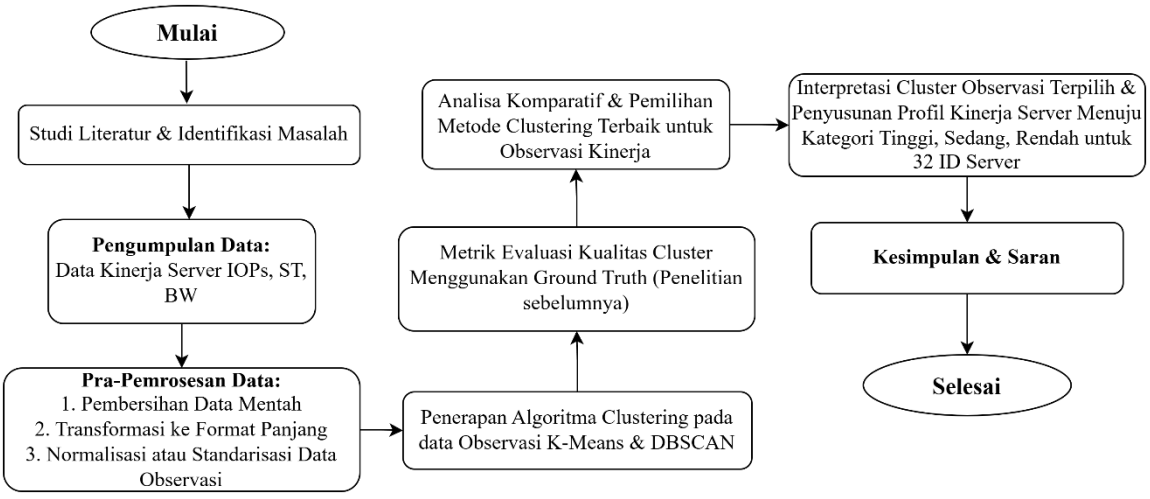
[15]. Dengan mengelompokkan *server* berdasarkan metrik performa seperti IOPS, *service time*, dan *bandwidth*, PT XYZ dapat memperoleh gambaran yang jelas mengenai tingkat beban kerja masing-masing *server*.

Penelitian ini secara khusus akan membandingkan dua algoritma clustering yaitu K-Means dan DBSCAN. K-Means merupakan algoritma partisi yang membagi data ke dalam sejumlah klaster yang telah ditentukan sebelumnya, dalam hal ini Tinggi, Sedang, dan Rendah, berdasarkan kemiripan atribut yang dimiliki setiap objek [16], [17]. Kelebihan utama K-Means adalah efisiensinya menangani dataset berukuran besar serta kemudahannya dalam interpretasi hasil. Sementara itu, DBSCAN merupakan algoritma clustering berbasis kepadatan yang tidak memerlukan penentuan jumlah klaster di awal yang mampu mengidentifikasi bentuk klaster yang tidak beraturan dan efektif dalam mendeteksi outlier atau *server* dengan pola beban kerja anomali [15], [18]. Selanjutnya hasil klaster di evaluasi akurasinya berdasarkan penelitian sebelumnya menggunakan ground truth [19]

Berdasarkan uraian tersebut maka rumusan masalah difokuskan pada tiga hal utama. Pertama, sejauh mana penerapan metode clustering K-Means dan DBSCAN efektif dalam mengkategorikan 32 *server block storage* di PT XYZ ke dalam tingkatan beban kerja Tinggi, Sedang, dan Rendah berdasarkan metrik IOPS, *service time*, dan *bandwidth*. Kedua, metode clustering mana yang menghasilkan pengelompokan *server* paling akurat jika dibandingkan dengan ground truth dari penelitan sebelumnya. Ketiga, bagaimana interpretasi profil beban kerja rata-rata dari setiap klaster dapat menjadi dasar dalam merumuskan strategi optimalisasi dan penyeimbangan beban kerja di PT XYZ. Melalui penelitian ini diharapkan dapat memberikan rekomendasi yang lebih akurat terkait penempatan aplikasi, strategi penyeimbangan beban kerja (load balancing), serta optimalisasi penggunaan sumber daya TI perusahaan dan juga rujukan bagi praktisi maupun akademisi dalam mengimplementasikan strategi pengelolaan beban kerja berbasis analisis data di lingkungan teknologi informasi.

## METODE

Berikut diagram alir penelitian merupakan diagram yang menunjukkan urutan setiap proses tahapan atau langkah instruksi pada penelitian untuk mencapai tujuan yang di perlihatkan oleh Gambar 2.




Gambar 2. Tahapan Penelitian

Metodologi penelitian ini dilakukan untuk mengkaji efektivitas algoritma klasterisasi K-Means dan DBSCAN dalam mengelompokkan *server block storage* di PT XYZ berdasarkan metrik performa IOPS, *Service time*, dan *Bandwidth*. Tahapan penelitian diawali dengan proses pengumpulan data yang dilakukan melalui kerja sama dengan tim IT SAN/Storage Infrastructure PT XYZ dengan memanfaatkan aplikasi *HPe Performance Tools* untuk menarik laporan performa dan log dump dari 32 *server block storage* selama periode dari 1 January 2024 hingga 31 Mei 2025. Data mentah yang diperoleh kemudian melalui tahap pra-pemrosesan untuk meningkatkan kualitas data, yang mencakup pembersihan data dari missing values dan pengecekkan nilai Finit, transformasi format data menjadi agregat per *server*, serta normalisasi atau standarisasi untuk menyamakan skala antar variabel. Setelah data siap, algoritma K-Means dan DBSCAN diterapkan. Tahapan awal K-Means adalah menentukan Jumlah Klaster optimal dengan pendekatan evaluasi *Elbow Method* dan *Silhouette Score*. Jumlah klaster ini diharapkan dapat mengkategori beban kerja Tinggi, Sedang, dan Rendah. Pada klaster DBSCAN minpts ditetapkan 5 merupakan nilai terbaik untuk data yang tidak terlalu tinggi, Lalu pencarian epsilon yang paling optimal berdasarkan evaluasi silhouette score. Kelebihan DBSCAN yaitu kemampuan klasterisasi berdasarkan kepadatan data dan mendeteksi potensi outlier atau *server* dengan perilaku anomali. Selanjutnya evaluasi kualitas hasil klasterisasi dilakukan menggunakan validasi eksternal berbasis ground truth dari penelitian terdahulu yang telah memetakan klasifikasi beban kerja *server* secara aktual [19]. Acuan validasi eksternal didapat berdasarkan spesikiasi hardware dan temuan pada penelitian sebelumnya. Penelitian ini menggunakan tipe disk SSD dimana memiliki kemampuan 5000 -10000 iops [20]. Kemampuan *service time* umumnya sekitar 10ms hingga 20ms untuk *service time* diatas 20ms sebagai ambang batas dimana pengguna akan merasakan kelambatan [21]. Untuk Kapasitas penggunaan *bandwith* secara konsisten mendekati atau melebihi 75-80% pada suatu link dapat menjadi indikasi kuat adanya *bottleneck* dan kondisi *overload* [22]. Ukuran karakteristik storage pada penelitian sebelumnya tersebut dijadikan acuan untuk label *ground truth*. Lalu dilakukan evaluasi akurasi yang dihitung berdasarkan tingkat kesesuaian label hasil klasterisasi dengan lab ground truth. Setelah kedua algoritma dievaluasi, dilakukan analisis komparatif untuk menentukan metode terbaik berdasarkan akurasi tertinggi. Terakhir, hasil klaster terbaik dianalisis lebih dalam untuk memprofilkan karakteristik beban kerja tiap kelompok *server* dan divisualisasikan guna memberikan rekomendasi praktis bagi PT XYZ dalam penyeimbangan beban kerja dan optimalisasi infrastruktur TI.

## HASIL DAN PEMBAHASAN

Pada tahapan ini akan dibahas bagaimana menganalisa karakteristik *server* blok storage dari awal pengumpulan data, pra pemrosesan data, analisa dengan 2 metoda klasterisasi, perbandingan akurasi, pemilihan metoda klaster terbaik hingga kesimpulan.

### Pengumpulan Data

Dataset yang digunakan dalam penelitian ini merupakan data historis performa 32 *server block storage* yang dioperasikan oleh PT XYZ dalam rentang waktu 1 Januari 2024 hingga 31 mei 2025. Berupa 96 kolom metrik (yaitu, 32 *server* dikalikan 3 metrik beban kerja per *server*) ditambah satu kolom untuk stempel waktu, sehingga totalnya menjadi 97 kolom mencakup tiga metrik utama yaitu IOPS, *Service time*, dan *Bandwidth* untuk masing-masing *server*. Struktur data disusun dalam format deret waktu (*time-series*) di mana setiap baris merepresentasikan pencatatan metrik pada waktu tertentu, sedangkan kolom-kolomnya terdiri dari identitas *server* dan ketiga metrik performa tersebut seperti terlihat pada Tabel 1.

Tabel 1. Data Awal Beban kerja *Server block storage*

| Waktu Pencatatan | *Server* Id 1(Total) IOs/Sec2 | *Server* Id 2 (Total) IOs/Sec2 | *Server* Id 3 (Total) IOs/Sec2 |
|---|---|---|---|
| Jan 1, 2024 12:00:00 AM | 277.5 | 51.7 | 502.4 |
| Jan 2, 2024 12:00:00 AM | 293.1 | 328.1 | 294.7 |
| Jan 3, 2024 12:00:00 AM | 281.0 | 41.5 | 268.5 |
| Jan 4, 2024 12:00:00 AM | 251.0 | 296.2 | 344.1 |

## Pra Pemrosesan Data

Langkah awal melakukan transformasi dan validasi pada kolom Time. Kolom ini merupakan kunci untuk memahami urutan kronologis data beban kerja *server*. Pada dataset awal, kolom Time teridentifikasi memiliki tipe data object. kolom Time ini dikonversi menjadi format datetime standar menggunakan fungsi to_datetime dari pustaka pandas dalam Python. Setelah konversi awal ke tipe datetime, dilakukan tahap koreksi tahun yang esensial pada sebagian data. Observasi menunjukkan bahwa data yang dikumpulkan mencakup periode transisi tahun, dan terdapat kebutuhan untuk memastikan konsistensi pencatatan tahun. Secara spesifik, untuk entri data yang dimulai dari indeks ke-366 dan seterusnya, logika pemrograman diterapkan untuk memeriksa tahun: jika tahun tercatat sebagai 2024, maka akan dikoreksi menjadi tahun 2025. Hasil akhir dari serangkaian transformasi ini adalah kolom Time yang telah valid, konsisten, dan direpresentasikan dalam format tanggal standar YYYY-MM-DD.

Tahapan berikutnya yaitu Transpose data dan agregasi fitur. Data beban kerja *server* awal, yang disajikan dalam format lebar (wide format) dengan banyak kolom metrik per stempel waktu, ditransformasikan menjadi format panjang (long format). Langkah ini krusial untuk menstrukturkan ulang data agar setiap baris merepresentasikan satu observasi unik dari ketiga metrik beban kerja (IOPS, *Service time*, *Bandwidth*) untuk satu *server* pada satu titik waktu tertentu. Proses ini mempermudah analisis dan agregasi fitur di tahap selanjutnya. Transformasi ini dilakukan menggunakan pustaka pandas di Python melalui dua langkah utama yaitu Identifikasi dan *Unpivot* (Melt) Kolom Metrik**:** Pertama, kolom-kolom yang berisi data IOPS, *Service time*, dan *Bandwidth* untuk masing-masing dari 32 *server* diidentifikasi. Kemudian, untuk setiap jenis metrik, dilakukan operasi melt (unpivot). Lalu penggabungan (*Merge*) Dataframe: Setelah ketiga jenis metrik (IOPS, *Service time*, *Bandwidth*) diubah ke format panjang secara terpisah, ketiga dataframe hasil melt tersebut digabungkan (merge) menjadi satu dataframe tunggal, yang dinamakan df_long. Penggabungan ini didasarkan pada kesamaan kolom Time dan *Server* ID. Kolom Time kemudian diubah namanya menjadi Time ID. Hasil utama dari tahap transformasi ini adalah terbentuknya dataframe df_long, yang menyajikan seluruh data beban kerja dalam struktur format panjang. Setiap baris pada df_long secara spesifik menunjukkan nilai IOPS, *Service time*, dan *Bandwidth* untuk satu *Server* ID pada satu Time ID tertentu.

Dataframe df_long yang telah terstruktur dalam format panjang ini menjadi tahap awal penting untuk langkah selanjutnya, yaitu agregasi fitur. Pada tahap agregasi tersebut, data beban kerja selama 1.5 tahun akan diringkas untuk menghasilkan satu set profil fitur unik (misalnya, rata-rata IOPS, rata-rata *Service time*, rata-rata *Bandwidth*) untuk masing-masing dari 32 *server*, yang kemudian akan digunakan untuk analisis klasterisasi.

Tahap berikutnya yaitu dilakukan cleaning dengan pengecekkan nilai hilang, Fungsi .isnull().sum() digunakan untuk menghitung jumlah nilai yang hilang (NaN) pada setiap kolom fitur yang relevan. Lalu pengecekan Nilai Finit (Finite Values): Fungsi numpy.isfinite() digunakan bersama dengan .all() untuk memverifikasi apakah semua nilai dalam kolom fitur merupakan angka finit (yaitu, bukan tak terhingga/infinity atau NaN).

Tahap terakhir dalam pra pemrosesan data yaitu normalisasi data menggunakan Standard Scaler. Dalam penelitian ini, metode normalisasi yang diterapkan pada 16.544 entri

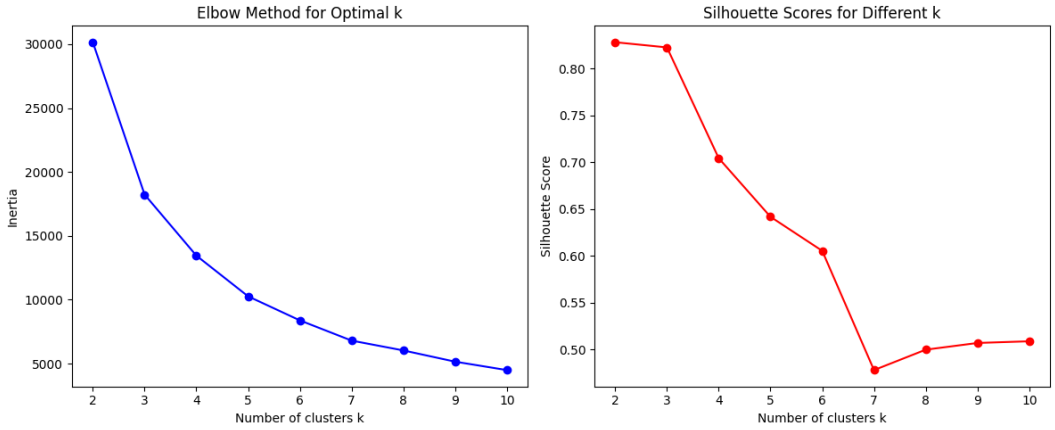
data dalam df_long_sorted adalah StandardScaler. Metode ini mentransformasi setiap fitur sehingga memiliki distribusi dengan rata-rata (mean) mendekati 0 dan standar deviasi (std) mendekati 1. Keberhasilan proses standardisasi ini diverifikasi melalui analisis statistik deskriptif dari data setelah transformasi. Data beban kerja per observasi waktu yang kini telah ternormalisasi ini kemudian diagregasi untuk setiap *Server* ID. Proses agregasi (misalnya, dengan menghitung nilai rata-rata dari fitur ternormalisasi untuk setiap *server* selama periode 1.5 tahun) akan menghasilkan matriks fitur akhir yang siap digunakan, yaitu dataset dengan dimensi 32 baris (mewakili *server*) dan 3 kolom (mewakili fitur beban kerja ternormalisasi dan teragregasi). Matriks inilah yang menjadi input final untuk penerapan algoritma klasterisasi K-Means dan DBSCAN. Struktur dari dataframe df_long ini diilustrasikan pada Tabel 2.

Tabel 2. Data Setelah Transformasi ke Format Panjang (df_long)

| Time ID | *Server* ID | IOPS | *Service time* | *Bandwidth* |
|---|---|---|---|---|
| 2024-01-01 00:00:00+07:00 | cbtcnb03rts18 | 277.5 | 0.46 | 7284.7 |
| 2024-01-02 00:00:00+07:00 | cbtcnb03rts18 | 293.1 | 0.48 | 7563.1 |
| 2024-01-03 00:00:00+07:00 | cbtcnb03rts18 | 281.0 | 0.47 | 7154.9 |

## Hasil Implementasi dan Evaluasi Klasterisasi Menggunakan Algoritma K-Means

Implementasi algoritma K-Means diawali dengan penentuan jumlah klaster optimal menggunakan Elbow Method dan Silhouette Score. Hasil analisis menunjukkan bahwa K=3 merupakan jumlah klaster terbaik seperti terlihat pada Gambar 3.



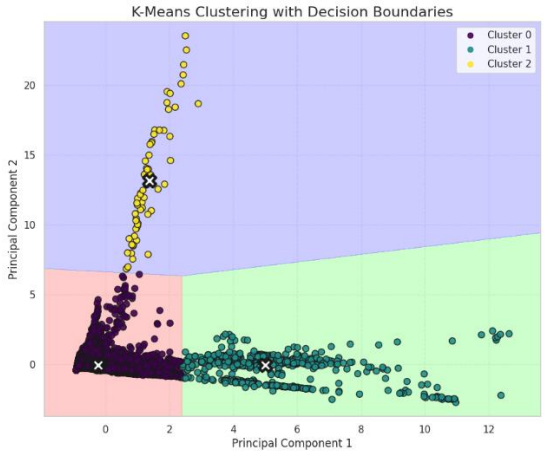

Gambar 3. Elbow Method dan Silhouette Scores untuk Nilai k yang Berbeda

Walaupun secara nilai, K=2 memang menghasilkan *Silhouette Score* yang sedikit lebih tinggi, namun pemilihan K=3 tetap dipertahankan sesuai dengan tujuan awal penelitian untuk menghasilkan tiga kelompok klaster. Selain itu, perbedaan nilai Silhouette Score antara K=2 (0.8284) dan K=3 (0.8228) juga relatif kecil sehingga penggunaan K=3 masih dapat diterima secara metodologis. Sejalan dengan tujuan penelitian yaitu mengelompokkan *server* ke dalam kategori Beban Kerja Tinggi, Menengah, dan Rendah. Proses klasterisasi dengan K=3 menghasilkan tiga kelompok *server* yang secara statistik memiliki perbedaan pada nilai rata-rata IOPS, *Service time* dan *Bandwidth* seperti terlihat pada Tabel 3.

Tabel 3. Ringkasan Profil Kerja dan Klasterisasi K-Means Beban Kerja

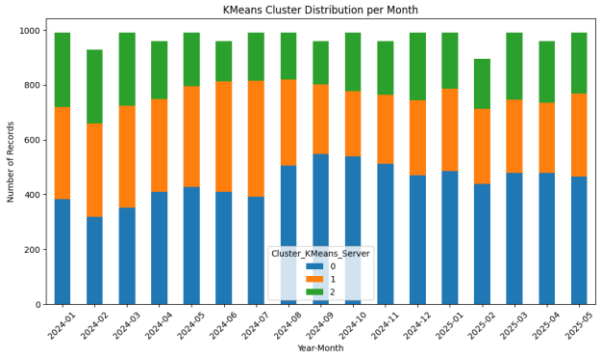
| Klaster ID | IOPS (Rata-rata) | *Service time* (Rata-rata) | *Bandwidth* (Rata-rata) | Klasifikasi Beban Kerja |
|---|---|---|---|---|
| Klaster 0 | 673.87 IOs/Sec | 0.71 ms | 29,450.36 KBs/Sec | Rendah |
| Klaster 1 | 3905.35 IOs/Sec | 1.15 ms | 521,302.96 KBs/Sec | Menengah |
| Klaster 2 | 323.85 IOs/Sec | 19.43 ms | 120,792.31 KBs/Sec | Tinggi |

Secara visual mengonfirmasi pembentukan tiga cluster yang berbeda dan bagaimana K-Means secara tegas membagi ruang fitur menjadi tiga wilayah yang tidak tumpang tindih, seperti terlihat pada Gambar 4.




Gambar 4. Visualisasi Cluster K-Means (K=3)

Klaster 1 yaitu Beban Kerja Menengah dan Paling Produktif, Klaster ini menunjukkan kapabilitas superior dengan nilai IOPS rata-rata yang mencapai 3905.35 IOs/Sec dan *throughput Bandwidth* tertinggi sebesar 521,302.96 KBs/Sec (sekitar 521 MB/s). Klaster 0 yaitu Beban Kerja Rendah Klaster ini menunjukkan profil yang sangat seimbang dan sehat, dengan IOPS yang solid (673.87 IOs/Sec), *service time* yang sangat cepat (0.71 ms), dan *throughput Bandwidth* yang juga signifikan (29,450.36 KBs/Sec). Terakhir, Klaster 2 diidentifikasi sebagai kelompok dengan Tingkat Beban Kerja Tinggi yang mengakibatkan degradasi kinerja. Faktor penentu utamanya adalah nilai *service time* rata-rata yang sangat tinggi, yaitu 19.43 ms, yang mendekati ambang batas masalah performa. Meskipun IOPS-nya paling rendah (323.85 IOs/Sec), latensi yang ekstrem ini ditambah dengan *Bandwidth* yang juga sangat tinggi (120,792.31 KBs/Sec) menegaskan bahwa *server-server* dalam klaster ini berada dalam kondisi *overload*.




Gambar 5. Tren Beban kerja Berdasarkan Bulan K-Means

Jika melihat tren beban kerja berdasarkan waktu sperti yang terlihat pada Gambar 5 menunjukkan adanya dinamika alokasi beban kerja antar *server*. *Server* yang sebelumnya tergolong Beban Kerja Tinggi mengalami penurunan aktivitas, sementara *server* dengan Beban Kerja Rendah justru dialihkan untuk menanggung beban lebih besar. Hal ini mencerminkan adanya strategi penyeimbangan beban kerja (*load balancing*) yang diterapkan oleh PT XYZ.

## Hasil Implementasi dan Evaluasi Klasterisasi Menggunakan Algoritma DBSCAN

Penerapan algoritma DBSCAN dilakukan dengan penyesuaian parameter epsilon dan MinPts berdasarkan analisis k-distance graph untuk memastikan pembentukan klaster yang optimal. MinPts ditetapkan 5, penetapan nilai ini umum digunakan sebagai awal yang baik untuk dataset banyak, terutama ketika dimensi data tidak terlalu tinggi (dalam kasus ini, 3 fitur kinerja). Hasil klasterisasi DBSCAN menunjukkan adanya tiga kelompok utama serta identifikasi outlier yang tidak dapat diklasifikasikan ke dalam kelompok manapun. Klaster-klaster ini berhasil mengelompokkan *server* berdasarkan kepadatan data, tanpa terikat pada bentuk geometri tertentu.

Tabel 4. Hasil Eksperimen Variasi Nilai Epsilon (ε) untuk DBSCAN

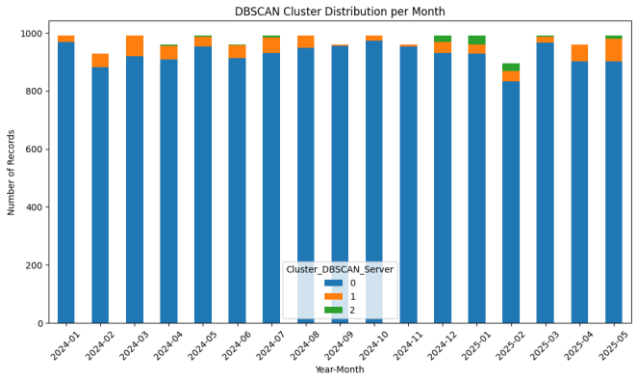
| **Nilai Epsilon (ε)** | **Jumlah *Klaster* Terbentuk (tidak termasuk *noise*)** | ***Silhouette Score*** |
|---|---|---|
| **1.10** | 4 | 0.8754 |
| **1.30** | 3 | 0.8930 |
| **1.50** | 3 | 0.8928 |

Berdasarkan data pada Tabel 4 terlihat titik optimal ditemukan pada saat nilai ε = 1.30. Pada konfigurasi ini, algoritma DBSCAN berhasil membentuk 3 klaster yang sesuai dengan target penelitian. Selain itu, pada titik ini pula dicapai nilai Silhouette Score yang sangat tinggi, yaitu 0.8930. Meskipun ada nilai ε lain yang juga menghasilkan 3 klaster (yaitu ε=1.50), Silhouette Score-nya sedikit lebih rendah. Oleh karena itu, nilai ε = 1.30 dipilih sebagai parameter optimal karena mampu menghasilkan jumlah klaster yang diinginkan dengan kualitas pengelompokan tertinggi. Dengan demikian, parameter final yang digunakan untuk analisis DBSCAN dalam penelitian ini adalah MinPts = 5 dan ε = 1.30.

Tabel 5. Ringkasan Profil Kerja dan Klasterisasi DBSCAN Beban Kerja

| Klaster ID | IOPS (Rata-rata) | *Service time* (Rata-rata) | *Bandwidth* (Rata-rata) | Klasifikasi Beban Kerja |
|---|---|---|---|---|
| Klaster 2 | 1174.85 IOs/Sec | 1.23 ms | 96,680.41 KBs/Sec | Tinggi |
| Klaster 1 | 800.17 IOs/Sec | 0.78 ms | 49,030.29 KBs/Sec | Menengah |
| Klaster 0 | 372.40 IOs/Sec | 1.48 ms | 63,920.62 KBs/Sec | Rendah |

Berdasarkan Tabel 5 Kelompok beban kerja Tinggi adalah Klaster 2, yang mendominasi dalam kapasitas pemrosesan beban kerja. Dengan nilai IOPS rata-rata tertinggi mencapai 1174.85 IOs/Sec dan *throughput Bandwidth* paling masif sebesar 96,680.41 KBs/Sec, klaster ini terbukti merupakan kelompok *server* yang paling sibuk dan produktif. Posisi Kinerja Menengah ditempati oleh Klaster 1. Keunggulan utama klaster ini terletak pada efisiensi dan responsivitasnya, dengan mencatatkan *service time* rata-rata terendah (0.78 ms). Kinerja latensi terbaik ini didukung oleh IOPS yang juga sangat tinggi (800.17 IOs/Sec), menjadikannya sangat ideal untuk tugas-tugas yang membutuhkan keseimbangan antara kecepatan transaksi dan respons. Terakhir, Klaster 0 diklasifikasikan sebagai kelompok Beban Kerja Rendah dalam konteks perbandingan ini. Meskipun memiliki *throughput Bandwidth* yang tinggi, nilai IOPS-nya adalah yang terendah (372.40 IOs/Sec) dan *service time*-nya adalah yang tertinggi di antara ketiganya (1.48 ms). Penting untuk dicatat bahwa klasifikasi "Rendah" ini bersifat relatif, karena secara absolut beban kerjanya masih tergolong sangat baik.




Gambar 6. Tren Beban kerja Berdasarkan Bulan DBSCAN

Berdasarkan Gambar 6 diatas mengungkapkan 3 tren pola beban kerja klaster yaitu Klaster Beban Kerja Rendah menunjukkan jumlah aktivitas yang sangat besar namun tetap stabil, mengindikasikan efisiensi kinerja yang optimal. Klaster Beban Kerja Menengah menunjukkan aktivitas yang fluktuatif namun konsisten, sedangkan Klaster Beban Kerja Tinggi terdiri atas *server* dengan aktivitas rendah namun kinerja yang ekstrem.

**Evaluasi Beban Kerja menggunakan Ground Truth**

Tahap Selanjutnya yaitu evaluasi yang bertujuan untuk mengukur secara kuantitatif seberapa efektif algoritma K-Means dan DBSCAN dalam menghasilkan pengelompokkan yang sesuai dengan kondisi nyata atau penelitian sebelumnya. Untuk mencapai tujuan ini, digunakan pendekatan validasi eksternal dengan membandingkan hasil Klasterisasi terhadap Ground Truth. Adapun acuan terhadap penelitan sebelumnya yaitu:

- IOPS: Pada penelitian ini menggunakan tipe disk SaS SSD dengan kemampuan 5000 hingga 10000 IOPS [20].
- *Service time* : latensi diatas 20ms sebagai ambang batas dimana pengguna akan merasakan kelambatan [21].
- Bandwith: Kapasitas mendekati atau melebihi 75-80% pada suatu link dapat menjadi indikasi kuat adanya *bottleneck* dan kondisi *overload* [22].
  Pada penelitian 32 *server* terhubung ke storage menggunakan 4 Port, dimana masing-masing port terhubung ke 8 *server*. Kapasitas per port adalah 2GBps dengan maksimum penggunaan adalah 80% yaitu 1,6Gbps. Apabila 1,6Gbps port terhubung ke 8 *server* artinya untuk menjaga stabilisasi performa masing-masing *server* bisa mengkonsumsi maksimal 200Mbps.

Acuan dari penelitain sebelumnya ini dijadikan ambang batas teratas terhadap label ground truth. Hasil evaluasi diberi label 'Tinggi', 'Menengah', atau 'Rendah' sebagai berikut:

- *Service time*: Merepresentasikan tingkat degradasi latensi.
  - Tinggi (Degradasi): > 20 ms
  - Menengah (Wajar): 5 - 20 ms
  - Rendah (Optimal): < 5 ms
- IOPS: Merepresentasikan kapabilitas pemrosesan transaksi. Dalam aturan ini, nilai IOPS yang tinggi menunjukkan beban kerja yang baik.
  - Tinggi (Optimal): ≥ 10,000 IOs/Sec
  - Menengah (Wajar): 5,000 - 10,000 IOs/Sec
  - Rendah (Degradasi): < 5,000 IOs/Sec
- *Bandwidth*: Merepresentasikan tingkat kejenuhan transfer data.
  - Tinggi (Jenuh/*Overload*): > 200,000 KBs/Sec (~200 MB/s)
  - Menengah (Utilisasi Tinggi): 100,000 - 200,000 KBs/Sec
  - Rendah (Utilisasi Rendah): < 100,000 KBs/Sec

Setiap data point *server* diberikan label ground truth yang secara objektif merepresentasikan kondisi beban kerjanya. Lalu dilakukan pemetaan (mapping) antara label cluster yang ditemukan dengan kelas ground truth yang paling sesuai. Pemetaan ini dilakukan dengan mencari kombinasi pasangan label yang menghasilkan jumlah *server* yang terklasifikasi benar secara maksimal. Hasil dari pengukuran terhadap ground truth untuk masing-masing klaster bisa kita lihat pada Tabel 6 dan Tabel 7.

Tabel 6. Hasil Pemetaan DBSCAN terhadap Ground Truth

| **Time ID** | ***Server* ID** | **IOPS (IOs/Sec)** | ***Service time* (ms2)** | ***Bandwidth* (KBs/Sec2)** | **Cluster_DBSCAN_Label** | **Ground_Truth_Label** |
|---|---|---|---|---|---|---|
| 2024-01-01 | cbt03dpdmsdb01 | 22.1 | 0.29 | 283.5 | Rendah | Rendah |
| 2024-01-02 | cbt03dpdmsdb01 | 21.8 | 0.27 | 283.5 | Rendah | Rendah |
| 2024-01-03 | cbt03dpdmsdb01 | 21.8 | 0.3 | 283.5 | Rendah | Rendah |

Tabel 7. Hasil Pemetaan K-Means terhadap Ground Truth

| **Time ID** | ***Server* ID** | **IOPS (IOs/Sec)** | ***Service time* (ms2)** | ***Bandwidth* (KBs/Sec2)** | **Cluster_K-Means_Label** | **Ground_Truth_Label** |
|---|---|---|---|---|---|---|
| 2024-01-01 | cbt03dpdmsdb01 | 22.1 | 0.29 | 283.5 | Menengah | Rendah |
| 2024-01-02 | cbt03dpdmsdb01 | 21.8 | 0.27 | 283.5 | Menengah | Rendah |
| 2024-01-03 | cbt03dpdmsdb01 | 21.8 | 0.3 | 283.5 | Menengah | Rendah |

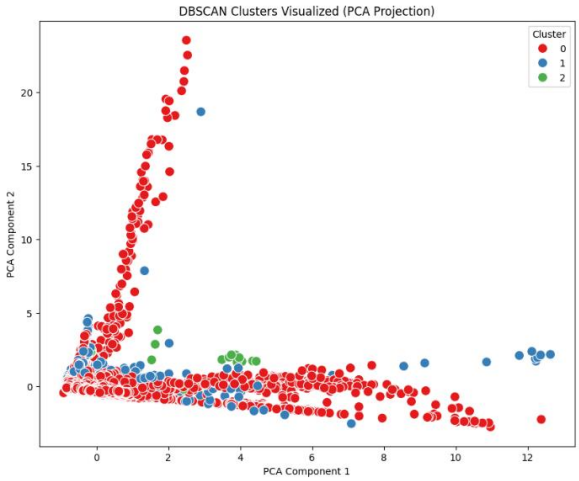
Setelah pemetaan optimal ditemukan, tingkat akurasi dihitung menggunakan Persamaan (1):

$$Akurasi = \frac{\text{Jumlah } Server \text{ yang Terklasifikasi Benar}}{Total\ Jumlah\ Server}\ x\ 100\% \qquad (1)$$

Pada Persamaan (1), "Jumlah *Server* yang Terklasifikasi Benar" adalah jumlah *server* di mana label cluster yang telah dipetakan cocok dengan label ground truth-nya. Setelah proses perhitungan dilakukan, lalu didapat bahwa akurasi K-Means sebesar 23,60% sedangkan DBSCAN mampu mencapai tingkat akurasi sebesar 87,72%. DBSCAN jauh lebih selaras dengan klasifikasi beban kerja fungsional yang ada di PT XYZ.

Adapun beberapa alasan mengapa akurasi K-means lebih rendah terhadap DBSCAN yaitu

1. K-Means sangat rentan terhadap outlier, karena outlier dapat secara signifikan memengaruhi posisi centroid.
2. K-Means memiliki bentuk sferis (bulat) dan ukuran yang relatif sama. Data kinerja *server* di dunia nyata seringkali tidak mengikuti asumsi ini. *Cluster* fungsional mungkin memiliki bentuk yang tidak beraturan atau memanjang, yang tidak dapat ditangkap dengan baik oleh K-Means.
3. Sebaliknya, DBSCAN memiliki keunggulan utama dalam menemukan klaster dengan bentuk arbitrer (tidak beraturan) dan secara inheren dapat mengidentifikasi outlier ataupun noise.



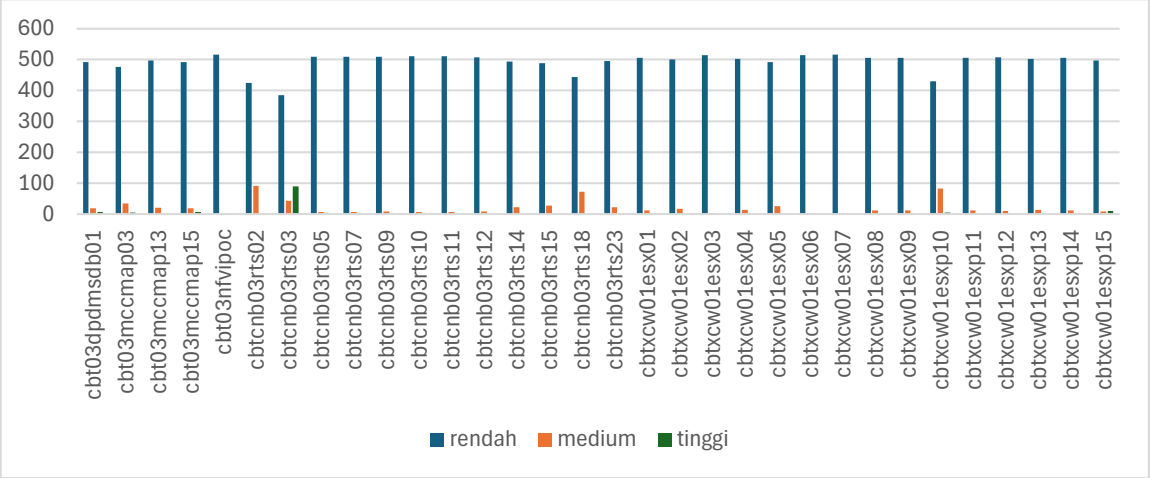

Gambar 7. Visualisasi PCA Hasil Klasterisasi DBSCAN

Visualisasi hasil klasterisasi DBSCAN menggunakan PCA pada Gambar 8 dapat kita lihat bahwa outlier ataupun noise dapat diklasifikasikan dengan baik. Klaster-klaster yang terbentuk tampak terpisah di ruang dua dimensi, terutama Klaster Beban Kerja Rendah yang membentuk kelompok yang sangat padat dan terpisah dari cluster lain. Analisis perilaku individual *server* menunjukkan konsistensi yang tinggi, di mana sebagian besar *server* menghabiskan waktu dalam kondisi Beban Kerja Rendah. Namun demikian, beberapa *server* teridentifikasi mengalami lonjakan beban kerja signifikan pada periode tertentu. Hasil ini juga menunjukkan bahwa DBSCAN efektif dalam mengidentifikasi pola yang kompleks dan tidak selalu mengikuti distribusi yang seragam, adanya *server* dengan perilaku yang tidak sesuai dengan mayoritas *server* lain yang jika tidak diidentifikasi dapat mengganggu kestabilan sistem secara keseluruhan. Temuan ini memberikan rekomendasi kuat bagi PT

XYZ untuk menggunakan pendekatan berbasis kepadatan seperti DBSCAN dalam upaya optimalisasi dan penyeimbangan beban kerja *server block storage* mereka.

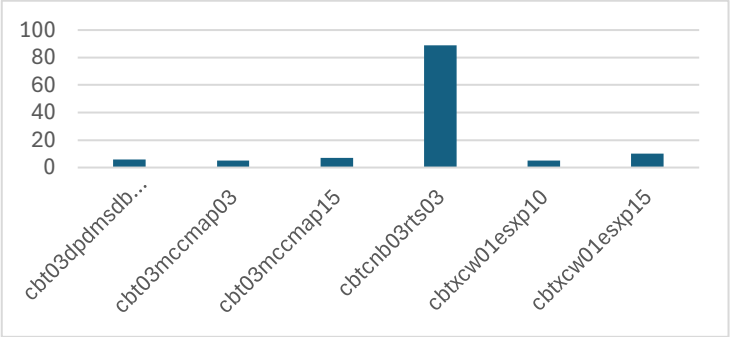
**Profil Perilaku Beban Kerja Individual *Server* Pada Cluster Terpilih**

Analisis DBSCAN ini bertujuan untuk melihat lebih dalam dari level klaster ke level *server* individual. Dengan menghitung berapa kali setiap *server* diklasifikasikan ke dalam kategori beban kerja 'Tinggi', 'Menengah', dan 'Rendah' selama keseluruhan periode observasi, kita dapat memahami profil perilaku dan konsistensi masing-masing *server*.




Gambar 8. Distribusi Kategori Beban Kerja per *Server*

Dari Gambar 9, observasi utama adalah bahwa mayoritas besar *server* menunjukkan perilaku yang sangat konsisten dan stabil. Lebih dari 25 dari 32 *server* menghabiskan hampir seluruh waktunya (lebih dari 95%) dalam kategori Beban Kerja Rendah. Hal ini mengindikasikan bahwa sebagian besar infrastruktur *server* beroperasi secara efisien dalam kondisi normal dan tidak mengalami *overutilized*.




Gambar 9. *Server* dengan Kategori Tinggi

Namun, terdapat beberapa *server* menunjukkan anomali terlihat pada Gambar 10 di mana *server* cbtcnb03rts03 tercatat masuk dalam kategori Beban Kerja Tinggi selama 3 bulan dan selebihnya ada 5 *server* yang masuk dalam kategori tinggi selama tidak lebih dari 2 minggu, meskipun saat ini kondisinya telah kembali normal. Temuan ini menggaris bawahi pentingnya pemantauan berkelanjutan untuk mengantisipasi lonjakan beban kerja yang tidak terduga dan rekomendasi penyeimbangan beban kerja dapat di lakukan pada saat tersebut.

## KESIMPULAN

Kesimpulan dari penelitian yaitu bahwa Efektivitas Metode Klasterisasi dengan kedua metode klasterisasi, K-Means dan DBSCAN, terbukti efektif dalam mengkategorikan 32 *server block storage* berdasarkan metrik kinerjanya . K-Means berhasil membagi *server* ke dalam tiga tingkatan beban kerja umum (Tinggi, Menengah, Rendah) yang intuitif. Sementara itu, DBSCAN menunjukkan efektivitas yang lebih mendalam dengan kemampuannya mengidentifikasi kelompok-kelompok *server* berdasarkan peran atau fungsi spesifiknya (Tinggi, Menengah, dan Rendah), serta kemampuannya mengisolasi *server* dengan anomali beban kerja sebagai noise. Untuk akurasi klasterisasi berdasarkan hasil evaluasi kuantitatif terhadap ground truth dari penelitian sebelumnya, DBSCAN merupakan metode yang lebih akurat secara signifikan. DBSCAN mencapai tingkat akurasi sebesar 87,72%, sementara K-Means hanya mencapai 23,60%. Hasil menunjukkan bahwa klaster DBSCAN jauh lebih selaras dengan klasifikasi beban kerja fungsional yang telah ada. Implikasi hasil Klasterisasi dapat digunakan untuk Interpretasi profil beban kerja rata-rata dari metode klasterisasi terpilih (DBSCAN) sehingga menjadi dasar yang kuat untuk merumuskan rekomendasi strategi, dengan memahami peran fungsional setiap klaster, PT XYZ dapat melakukan penyeimbangan beban (*load balancing*) dan juga strategi penempatan aplikasi yang cerdas *(intelligent application placement*). Pengetahuan bahwa ada kelompok *server* yang beban kerja overlaod dan kelompok lain yang beban kerja rendah memungkinkan alokasi sumber daya yang jauh lebih presisi dan efisien.

## REFERENSI

[1] R. Subekti *et al.*, 'TRANSFORMASI DIGITAL (Teori & implementasi Menuju Era Society 5.0)', 2024.

[2] C. Dale, 'The future of enterprise storage - the evolving landscape of enterprise storage. In Exertis Enterprise', *https://exertisenterprise.com/drive-the-future-of-enterprise-storage-the-evolving-landscape-of-enterprise-storage/?utm_source=chatgpt.com*, 2024.

[3] Louwrentius, 'Understanding Storage Performance - IOPS and Latency', *Https://Louwrentius.Com/Understanding-Storage-Performance-Iops-and-Latency.Html*, Mar. 2020.

[4] O. Karma Putri and Z. Saharuna, 'PENERAPAN TEKNOLOGI *BLOCK STORAGE* DALAM SISTEM VIRTUALISASI BERBASIS OPENSTACK', *In Journal of Informatics and Computer Engineering Research (JICER) (Vol. 1, Issue 1).*, 2024.

[5] S. Ahmadian, F. Taheri, and H. Asadi, 'Evaluating Reliability of SSD-Based I/O Caches in Enterprise Storage Systems', *IEEE Trans Emerg Top Comput*, vol. 9, no. 4, pp. 1914–1929, Oct. 2021, doi: 10.1109/TETC.2019.2945087.

[6] A. R. Al-Maeeni, A. Temirkhanov, A. Ryzhikov, and M. Hushchyn, 'Performance Modeling of Data Storage Systems Using Generative Models', *IEEE Access*, vol. 13, pp. 49643–49658, 2025, doi: 10.1109/ACCESS.2025.3552409.

[7] StorageExperts, 'HPE GreenLake for *Block storage*: Transform with cloud operations and self-service agility. HPE Community', https://community.hpe.com/t5/around-the-storage-block/hpe-greenlake-for-block-storage-transform-with-cloud-operations/ba-p/7162490.

[8] J. S. Saini and S. Architect, 'Block Size & its impact on Storage Performance', 2024.

[9] I. U. Akgun *et al.*, 'KML: Using Machine Learning to Improve Storage Systems', Nov. 2021, [Online]. Available: http://arxiv.org/abs/2111.11554

[10] A. H. Ali, 'Innovative Strategies for Enhancing Web Application Performance: A Contemporary Load Testing Approach', *International Journal of Engineering Trends and Technology*, vol. 72, no. 10, pp. 246–256, Oct. 2024, doi: 10.14445/22315381/IJETT-V72I10P124.

[11] X. Chen, S. Pang, H. Wang, and D. Zeng, 'Research on Load Balancing of Cloud Storage *Server* Based on Fully Connected Group', in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Aug. 2021. doi: 10.1088/1742-6596/1982/1/012134.

[12] S. K. Anand and S. Kumar, 'Experimental Comparisons of Clustering Approaches for Data Representation', *ACM Comput Surv*, vol. 55, no. 3, pp. 1–33, Mar. 2023, doi: 10.1145/3490384.

[13] J. Han, M. Kamber, and J. Pei, 'Data Mining: Concepts and Techniques (4rd ed.)', *Morgan Kaufmann Publishers Inc*, 2023.

[14] S. Chitta, C. Ravi, V. K. R. Vangoor, and S. M. Yellepeddi, 'AIOps: Integrating AI and Machine Learning into IT Operations. 4, 279.', 2024.
[15] M. Begum *et al.*, 'M-DBSCAN: Modified DBSCAN Clustering Algorithm for Detecting and Controlling Outliers', in *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing*, New York, NY, USA: ACM, Apr. 2024, pp. 1034–1035. doi: 10.1145/3605098.3636188.
[16] Raheela zaib and OURLIS Ourabah, 'Large Scale Data Using K-Means', *Mesopotamian Journal of Big Data*, vol. 2023, pp. 36–45, Feb. 2023, doi: 10.58496/MJBD/2023/006.
[17] M. Gagolewski, 'A framework for benchmarking clustering algorithms', *SoftwareX*, vol. 20, p. 101270, Dec. 2022, doi: 10.1016/j.softx.2022.101270.
[18] Y. Li, 'Network Anomaly Detection Algorithm Based on Deep Learning and Data Mining', in *Proceedings of the 2024 3rd International Conference on Cryptography, Network Security and Communication Technology*, New York, NY, USA: ACM, Jan. 2024, pp. 220–225. doi: 10.1145/3673277.3673316.
[19] G. M. Foody, 'Ground Truth in Classification Accuracy Assessment: Myth and Reality', *Geomatics*, vol. 4, no. 1, pp. 81–90, Feb. 2024, doi: 10.3390/geomatics4010005.
[20] B. Ganley, 'Busting solid-state storage myths', 2014.
[21] S. Allingham, 'How to improve application performance by decreasing disk latency like an IT engineer. Condusiv. ', https://condusiv.com/how-to-improve-application-performance-by-decreasing-disk-latency-like-an-it-engineer/.
[22] A. Thomasian, 'Storage Systems: Organization, Performance, Coding, Reliability, and Their Data Processing (1st ed.). Morgan Kaufmann', https://www.sciencedirect.com/book/9780323907965/storage-systems.