Analisis Klaster K-Means Proyeksi UMKM di Jawa Barat (2016-2023)

Syukrillah
22552011247@sttbandung.ac.id

Abstrak

Laporan ini menyajikan analisis klaster K-Means terhadap data proyeksi Usaha Mikro, Kecil, dan Menengah (UMKM) di berbagai kabupaten dan kota di Provinsi Jawa Barat dari tahun 2016 hingga 2023. Tujuan utama penelitian ini adalah untuk mengidentifikasi kelompok-kelompok daerah yang berbeda berdasarkan tingkat proyeksi UMKM mereka, sehingga dapat memberikan pemahaman yang lebih mendalam tentang pola distribusi UMKM di wilayah tersebut. Metodologi yang diterapkan meliputi pra-pemrosesan data yang cermat, termasuk agregasi data tahunan, transformasi logaritmik untuk mengatasi kemiringan distribusi, dan standardisasi fitur untuk memastikan kontribusi yang setara dalam perhitungan jarak. Penentuan jumlah klaster optimal (k) dilakukan menggunakan Metode Elbow yang didukung oleh perhitungan Silhouette Score. Implementasi K-Means dilakukan menggunakan bahasa pemrograman Python, dan validasi dilakukan melalui perhitungan manual pada sebagian kecil data. Hasil analisis menunjukkan identifikasi sejumlah klaster yang berbeda, masing-masing dengan karakteristik proyeksi UMKM yang unik, seperti wilayah dengan proyeksi UMKM tinggi, sedang, dan rendah. Pembentukan kelompok-kelompok yang berbeda ini menunjukkan bagaimana K-Means dapat digunakan untuk analisis ekonomi regional, memberikan dasar untuk inferensi kebijakan yang relevan. Temuan ini berpotensi menjadi landasan bagi perumusan kebijakan yang lebih terarah dan alokasi sumber daya yang efisien untuk pengembangan UMKM di Jawa Barat.

Pendahuluan

Usaha Mikro, Kecil, dan Menengah (UMKM) merupakan tulang punggung perekonomian Indonesia, memainkan peran krusial dalam penciptaan lapangan kerja, pemerataan pendapatan, dan menjaga ketahanan ekonomi regional. Di provinsi yang luas dan beragam seperti Jawa Barat, pemahaman mendalam tentang distribusi dan pola pertumbuhan UMKM di berbagai wilayah administratif menjadi sangat penting. Heterogenitas ini menuntut pendekatan berbasis data untuk mengidentifikasi profil regional yang berbeda, yang pada

gilirannya dapat mendukung pengambilan keputusan yang lebih tepat dan terarah.
Data mining, sebagai proses penemuan pola dan wawasan dari kumpulan data besar, menawarkan kemampuan untuk mengubah data mentah menjadi pengetahuan yang dapat ditindaklanjuti. Dalam konteks pengembangan ekonomi, data mining dapat membantu mengidentifikasi tren, mengelompokkan entitas serupa, dan memprediksi perilaku masa depan. Laporan ini secara spesifik bertujuan untuk menerapkan teknik klastering K-Means pada dataset proyeksi UMKM di Jawa Barat.[1] Tujuannya adalah untuk mengelompokkan kabupaten dan kota ke dalam klaster-klaster yang bermakna berdasarkan aktivitas UMKM mereka, serta untuk menginterpretasikan karakteristik dari klaster-klaster yang terbentuk.
Keputusan untuk menggunakan klastering K-Means dalam analisis ini merupakan pilihan metodologis yang disengaja dan didasarkan pada karakteristik data serta tujuan penelitian. Instruksi awal memberikan pilihan antara teknik klastering (misalnya, K-Means, Hierarchical, DBSCAN) atau Association Rule Mining (misalnya, FP-Growth, Apriori).[1] Dataset yang tersedia berisi proyeksi numerik jumlah UMKM (
proyeksi_jumlah_umkm) untuk berbagai wilayah selama beberapa tahun (tahun).[1] Algoritma klastering, terutama K-Means, secara fundamental dirancang untuk mengelompokkan titik data berdasarkan kemiripan numerik mereka.[2] Ini sangat sesuai dengan tujuan untuk melakukan segmentasi wilayah berdasarkan profil UMKM mereka. Sebaliknya, Association Rule Mining lebih cocok untuk data transaksional guna menemukan hubungan antaritem (misalnya, "pelanggan yang membeli X juga membeli Y"). Dataset proyeksi UMKM ini tidak bersifat transaksional; itu tidak merepresentasikan pembelian item atau kejadian bersama. Oleh karena itu, pemilihan K-Means bukan hanya untuk memenuhi persyaratan, tetapi juga menunjukkan keputusan yang terinformasi berdasarkan kesesuaian algoritma dengan jenis data yang diberikan dan tujuan analitis untuk mengidentifikasi pengelompokan regional. Keputusan ini secara langsung mempengaruhi jenis wawasan yang dapat diekstraksi, mengarahkan analisis untuk memahami struktur ekonomi regional daripada melakukan analisis keranjang belanja.
Struktur laporan ini akan menguraikan secara rinci metodologi yang digunakan, menyajikan hasil analisis klaster, membandingkan hasil otomatis dengan perhitungan manual, dan memberikan interpretasi yang relevan, diakhiri dengan kesimpulan dan rekomendasi untuk penelitian di masa depan.

Metodologi

Bagian ini menjelaskan pendekatan sistematis yang diambil untuk melakukan analisis data mining, memastikan reproduktifitas dan transparansi.

Deskripsi Dataset

Dataset yang digunakan dalam analisis ini, berjudul

diskuk-od_17372_proyeksi_jml_ush_mikro_kecil_menengah_umkm__kabupa_v1_data.csv, disediakan sebagai bagian dari persyaratan Ujian Akhir Semester (UAS).[1] Dataset ini berisi angka proyeksi jumlah Usaha Mikro, Kecil, dan Menengah (UMKM) di berbagai kabupaten dan kota di Provinsi Jawa Barat, Indonesia.[1] Proyeksi ini mencakup periode delapan tahun, dari tahun 2016 hingga 2023.[1]

Dataset ini terdiri dari 216 baris, dengan setiap baris merepresentasikan catatan unik untuk kabupaten/kota tertentu pada tahun tertentu. Jumlah baris ini secara signifikan melebihi persyaratan minimum 50 baris yang ditetapkan.[1] Struktur dataset mencakup beberapa kolom penting:

- id: Pengenal unik untuk setiap catatan (integer).
- kode_provinsi, nama_provinsi: Kode dan nama provinsi. Untuk semua catatan dalam dataset ini, nilai-nilai ini konstan, yaitu '32' dan 'JAWA BARAT'.[1]
- kode_kabupaten_kota, nama_kabupaten_kota: Kode dan nama unik untuk setiap kabupaten atau kota (misalnya, 'KABUPATEN BOGOR', 'KOTA BANDUNG').[1] Kolom nama_kabupaten_kota ini adalah entitas yang akan dikelompokkan.
- proyeksi_jumlah_umkm: Kolom data inti yang mewakili jumlah proyeksi UMKM (integer).[1]
- satuan: Unit pengukuran untuk proyeksi_jumlah_umkm, yang secara konsisten adalah 'UNIT'.[1]
- tahun: Tahun proyeksi dibuat (integer).[1]

Struktur longitudinal dataset, di mana terdapat beberapa nilai tahunan untuk setiap wilayah, memberikan peluang untuk menganalisis tidak hanya jumlah UMKM statis tetapi juga aspek dinamis seperti pertumbuhan atau aktivitas rata-rata dari waktu ke waktu. Dataset ini berisi proyeksi_jumlah_umkm untuk 27 divisi administratif yang berbeda (18 Kabupaten dan 9 Kota) di seluruh 8 tahun (2016-2023).[1] Ini berarti setiap nama_kabupaten_kota memiliki 8 nilai proyeksi UMKM yang terkait. Algoritma klastering K-Means standar beroperasi pada satu titik data per entitas (misalnya, satu baris per nama_kabupaten_kota). Untuk menerapkan K-Means, nilai-nilai tahunan yang beragam ini untuk setiap wilayah harus diagregasi menjadi satu fitur representatif. Memilih untuk mengagregasi dengan *rata-rata* proyeksi_jumlah_umkm selama bertahun-tahun memberikan ukuran yang kuat dari tingkat aktivitas UMKM keseluruhan suatu wilayah. Alternatifnya, seseorang dapat menghitung tingkat pertumbuhan (misalnya, 2023 vs. 2016) atau menggunakan teknik yang lebih canggih seperti PCA pada deret waktu. Langkah agregasi ini merupakan keputusan penting dalam pra-pemrosesan, karena mendefinisikan apa arti "kemiripan" dalam konteks klastering (misalnya, kemiripan dalam aktivitas rata-rata versus kemiripan dalam lintasan pertumbuhan). Pemahaman yang lebih dalam ini memandu proses rekayasa fitur.

Pra-pemrosesan Data

Bagian ini menguraikan langkah-langkah yang diambil untuk membersihkan, mengubah, dan menyiapkan dataset mentah untuk klastering K-Means yang efektif.

- **Pemilihan Fitur:** Kolom id, kode_provinsi, nama_provinsi, kode_kabupaten_kota, dan satuan diidentifikasi sebagai tidak esensial untuk tugas klastering karena mereka adalah pengenal unik, konstan, atau label deskriptif yang tidak secara langsung berkontribusi pada kemiripan numerik untuk klastering. Kolom-kolom ini dikecualikan dari proses klastering. Fitur utama untuk klastering akan diturunkan dari proyeksi_jumlah_umkm dan tahun.
- **Agregasi Data:** Untuk merepresentasikan setiap nama_kabupaten_kota sebagai satu titik data untuk klastering, nilai proyeksi_jumlah_umkm untuk setiap wilayah di seluruh tahun 2016-2023 diagregasi. **Rata-rata proyeksi_jumlah_umkm** untuk setiap nama_kabupaten_kota dihitung sebagai fitur utama untuk klastering. Ini memberikan ukuran tingkat aktivitas UMKM yang stabil dan representatif untuk setiap wilayah selama periode yang diamati. Pilihan untuk mengagregasi proyeksi_jumlah_umkm dengan *rata-rata* selama bertahun-tahun, daripada, misalnya, *jumlah* atau *tingkat pertumbuhan*, secara fundamental mendefinisikan sifat klaster yang akan terbentuk. Jika digunakan *jumlah* UMKM selama semua tahun, klaster akan mencerminkan total aktivitas kumulatif, yang berpotensi terlalu menekankan wilayah dengan catatan data yang lebih panjang atau hanya magnitudo keseluruhan yang lebih besar. Jika digunakan *tingkat pertumbuhan* (misalnya, persentase perubahan dari 2016 hingga 2023), klaster akan mengelompokkan wilayah berdasarkan *lintasan pengembangan* atau dinamisme mereka. Dengan memilih *rata-rata* proyeksi_jumlah_umkm, klaster terutama akan merepresentasikan wilayah dengan *tingkat aktivitas UMKM yang khas* selama periode yang diamati. Ini memberikan gambaran yang stabil dan representatif tentang keberadaan UMKM suatu wilayah. Keputusan ini menyoroti bahwa rekayasa fitur bukanlah langkah sepele; itu secara langsung menentukan interpretasi dan kegunaan praktis dari klaster yang dihasilkan, membentuk narasi analisis.
- **Penanganan Nilai Hilang:** Pemeriksaan awal dataset tidak mengungkapkan adanya nilai hilang eksplisit.[1] Oleh karena itu, tidak ada langkah imputasi atau penghapusan khusus yang diperlukan untuk data yang hilang.
- **Manajemen Kemiringan (Transformasi Logaritmik):** Variabel proyeksi_jumlah_umkm diharapkan menunjukkan distribusi miring ke kanan, di mana beberapa wilayah memiliki jumlah UMKM yang jauh lebih tinggi daripada mayoritas. Klastering K-Means, yang berbasis jarak, berkinerja optimal ketika variabel memiliki distribusi simetris.[5] Untuk mengurangi dampak kemiringan dan memastikan bahwa wilayah dengan jumlah UMKM yang sangat tinggi tidak secara tidak proporsional memengaruhi proses klastering, **transformasi logaritmik** (np.log()) diterapkan pada nilai proyeksi_jumlah_umkm yang diagregasi. Transformasi ini mengkompresi rentang nilai yang lebih besar dan memperluas rentang nilai yang lebih kecil, membuat distribusi lebih simetris.[5]
- **Penskalaan Fitur (Standardisasi):** K-Means mengandalkan jarak Euclidean untuk mengukur kemiripan antar titik data.[6] Fitur dengan rentang numerik yang lebih besar dapat mendominasi perhitungan jarak, menyebabkan hasil klastering yang bias.[5] Untuk memastikan bahwa semua fitur berkontribusi secara setara pada perhitungan jarak, nilai proyeksi_jumlah_umkm yang telah ditransformasi akan **distandardisasi** menggunakan

StandardScaler dari pustaka sklearn.preprocessing. Proses ini mengubah data sehingga memiliki rata-rata 0 dan standar deviasi 1.[5] Urutan langkah pra-pemrosesan sangat penting: transformasi logaritmik (jika diperlukan)
sebelum standardisasi.[5] Keputusan untuk menerapkan transformasi logaritmik diikuti dengan standardisasi bukan hanya langkah teknis, tetapi juga tindakan penting untuk menyelaraskan data dengan asumsi dasar algoritma K-Means, sehingga meningkatkan validitas dan interpretasi klaster.

Tabel 2: Ringkasan Langkah Pra-pemrosesan Data

Langkah Pra-pemrosesan	Tujuan/Rasional	Pustaka/Metode Python
Pemilihan Fitur	Menghilangkan kolom yang tidak relevan untuk klastering.	pandas.DataFrame.drop()
Agregasi Data	Mengubah data longitudinal menjadi satu titik data per wilayah (rata-rata UMKM).	pandas.DataFrame.groupby().mean()
Penanganan Nilai Hilang	Memastikan tidak ada data yang hilang yang memengaruhi perhitungan.	Pemeriksaan awal, tidak ada tindakan karena tidak ditemukan nilai hilang.
Transformasi Logaritmik	Mengurangi kemiringan distribusi data proyeksi_jumlah_umkm agar lebih simetris.	numpy.log()
Standardisasi Fitur	Menjamin semua fitur memiliki skala yang sama, mencegah dominasi fitur dengan rentang besar.	sklearn.preprocessing.StandardScaler

Algoritma K-Means Clustering

K-Means adalah algoritma pembelajaran mesin tanpa pengawasan yang populer, digunakan untuk mempartisi n observasi ke dalam k klaster, di mana setiap observasi termasuk dalam klaster dengan rata-rata (centroid) terdekat.[2] Tujuan algoritma ini adalah untuk meminimalkan jumlah kuadrat dalam klaster (WCSS), juga dikenal sebagai inersia. Metrik ini mengukur jumlah jarak kuadrat antara setiap titik data dan centroid klaster yang ditugaskan.[4]
Langkah-langkah algoritmik K-Means bersifat iteratif:

1. **Inisialisasi:** Proses dimulai dengan memilih k centroid klaster awal. Meskipun pemilihan acak dimungkinkan, strategi inisialisasi 'k-means++' umumnya lebih disukai karena secara cerdas memilih centroid awal untuk mempercepat konvergensi dan meningkatkan kualitas klastering akhir dengan memilih titik berdasarkan jaraknya dari centroid yang ada.[2]

2. **Penugasan (E-step - Expectation):** Setiap titik data dalam dataset ditugaskan ke klaster yang centroidnya paling dekat dengannya, biasanya diukur menggunakan jarak Euclidean.[3]
3. **Pembaruan (M-step - Maximization):** Setelah semua titik data ditugaskan, centroid klaster dihitung ulang. Centroid baru untuk setiap klaster adalah rata-rata dari semua titik data yang saat ini ditugaskan ke klaster tersebut.[3]
4. **Iterasi dan Konvergensi:** Langkah 2 dan 3 diulang secara iteratif. Algoritma konvergen ketika penugasan klaster tidak lagi berubah, atau centroid stabil (yaitu, posisi mereka tidak bergeser secara signifikan antar iterasi), atau jumlah maksimum iterasi yang telah ditentukan tercapai.[3]

K-Means banyak digunakan karena kesederhanaannya, efisiensi komputasi, dan kemampuannya untuk menangani dataset besar secara efektif.[5] Sensitivitas bawaan K-Means terhadap penempatan centroid awal memerlukan beberapa kali eksekusi (
n_init) dalam implementasi otomatis untuk mengurangi risiko konvergen ke minimum lokal yang suboptimal, sehingga memastikan kekokohan solusi klastering akhir. K-Means adalah algoritma iteratif yang dimulai dengan serangkaian centroid awal.[3] Algoritma dijamin akan konvergen, tetapi konvergen ke optimum lokal dari fungsi tujuan WCSS, tidak selalu optimum global.[3] Inisialisasi acak yang berbeda dari centroid dapat menghasilkan konfigurasi klaster akhir yang berbeda dan nilai WCSS yang berbeda. Inisialisasi yang buruk dapat menghasilkan klastering yang suboptimal. Parameter
n_init dalam sklearn.cluster.KMeans [2] mengatasi hal ini dengan menjalankan algoritma K-Means beberapa kali (misalnya, 10 kali secara default dengan
n_init='auto') dengan seed centroid yang berbeda. Dari beberapa kali eksekusi ini, algoritma memilih keluaran terbaik, yaitu yang memiliki WCSS (inersia) terendah.[2] Oleh karena itu, pengaturan
n_init ke nilai lebih besar dari 1 (atau menggunakan 'auto') bukan hanya pengaturan parameter, tetapi pengamanan metodologis yang krusial. Ini memastikan bahwa hasil klastering akhir lebih kuat dan kurang bergantung pada inisialisasi acak tunggal yang berpotensi tidak beruntung, sehingga meningkatkan keandalan dan kualitas analisis.

Penentuan Jumlah Klaster Optimal

Bagian ini menjelaskan metodologi yang digunakan untuk menentukan jumlah klaster (*k*) yang paling sesuai untuk dataset UMKM.

Metode utama untuk menentukan *k* optimal adalah Metode Elbow.[7] Metode ini melibatkan perhitungan Within-Cluster Sum of Squares (WCSS), juga dikenal sebagai inersia, untuk berbagai nilai
k (misalnya, dari 1 hingga 10). WCSS mengukur kekompakan klaster; WCSS yang lebih rendah menunjukkan klaster yang lebih kompak.[7] Nilai WCSS kemudian diplot terhadap nilai
k yang sesuai. "Titik siku" (elbow point) pada grafik ini merepresentasikan *k* optimal. Ini adalah titik di mana tingkat penurunan WCSS melambat secara signifikan, menunjukkan bahwa

penambahan lebih banyak klaster di luar titik ini memberikan pengembalian yang semakin berkurang dalam hal mengurangi varians dalam klaster.[7]
Meskipun Metode Elbow memberikan heuristik visual, Silhouette Score dihitung sebagai metrik pelengkap untuk validasi.[6] Silhouette Score mengukur seberapa mirip suatu objek dengan klaster sendiri dibandingkan dengan klaster lain. Skor berkisar dari -1 hingga 1, di mana nilai mendekati 1 menunjukkan klaster yang terpisah dengan baik, 0 menunjukkan klaster yang tumpang tindih, dan -1 menunjukkan bahwa titik data mungkin ditugaskan ke klaster yang salah.[4] Ini memberikan ukuran kuantitatif untuk mendukung
k yang dipilih secara visual. Proses penentuan *k* optimal seringkali merupakan keseimbangan antara kesesuaian statistik (meminimalkan WCSS) dan interpretasi praktis. Mengandalkan Metode Elbow dan Silhouette Score memberikan pemilihan *k* yang lebih kuat dan dapat dibenarkan, melampaui interpretasi visual subjektif. Metode Elbow [7] adalah heuristik yang banyak digunakan, tetapi "siku" terkadang bisa ambigu atau tidak ada dalam dataset dunia nyata, yang mengarah pada interpretasi subjektif. Memilih
k yang terlalu kecil dapat menyebabkan klaster yang terlalu luas yang menggabungkan kelompok-kelompok yang benar-benar berbeda, sehingga kehilangan wawasan yang berharga. Memilih *k* yang terlalu besar dapat menyebabkan *overfitting*, menciptakan klaster yang terlalu spesifik atau hanya mencerminkan *noise*, membuat interpretasi sulit dan mengurangi generalisasi temuan. Silhouette Score [4] memberikan ukuran kuantitatif kohesi dan pemisahan klaster. Silhouette Score yang lebih tinggi untuk
k tertentu menunjukkan bahwa klaster didefinisikan dengan baik dan berbeda. Dengan menggabungkan wawasan visual dari Metode Elbow dengan validasi kuantitatif dari Silhouette Score, pemilihan *k* optimal menjadi lebih didorong oleh data dan dapat dipertahankan. Pendekatan ganda ini memastikan bahwa jumlah klaster yang dipilih tidak hanya secara statistik masuk akal tetapi juga kemungkinan akan menghasilkan wawasan yang bermakna dan dapat diinterpretasikan dalam konteks distribusi UMKM.

Prosedur Perhitungan Manual

Bagian ini merinci pelaksanaan manual langkah demi langkah dari algoritma K-Means pada sebagian kecil data yang telah diproses sebelumnya, memenuhi persyaratan ujian utama.[1]
Untuk demonstrasi perhitungan manual, subset kecil dan representatif dari data UMKM yang telah diproses dan diskalakan (misalnya, 10-20 baris, sebagaimana ditentukan dalam [1]) akan dipilih. Menggunakan data yang diskalakan memastikan konsistensi dengan implementasi Python otomatis. Langkah-langkah manualnya adalah sebagai berikut:

1. **Definisi *k*:** Jumlah klaster yang kecil (misalnya, *k*=2 atau *k*=3) akan dipilih agar perhitungan manual tetap dapat dikelola.
2. **Pemilihan Centroid Awal:** Secara acak dipilih *k* titik data dari subset yang dipilih untuk berfungsi sebagai centroid awal untuk setiap klaster. Koordinat spesifik dari centroid awal ini akan dinyatakan dengan jelas.
3. **Perhitungan Jarak (Iterasi 1):** Untuk setiap titik data dalam subset yang dipilih, jarak

Euclidean ke *masing-masing* dari *k* centroid awal akan dihitung. Rumus jarak Euclidean antara titik data x = (x1, x2,..., xn) dan centroid c = (c1, c2,..., cn) adalah sqrt((x1-c1)^2 + (x2-c2)^2 +... + (xn-cn)^2).[3]

4. **Penugasan Klaster (Iterasi 1):** Setiap titik data kemudian akan ditugaskan ke klaster yang centroidnya memiliki jarak Euclidean minimum yang dihitung.[3]
5. **Pembaruan Centroid (Iterasi 1):** Setelah semua titik data ditugaskan, centroid baru untuk setiap klaster akan dihitung dengan mengambil rata-rata semua titik data yang ditugaskan ke klaster spesifik tersebut.[3]
6. **Iterasi Selanjutnya:** Langkah 3-5 akan diulang untuk beberapa iterasi lagi (misalnya, 2-3 iterasi) untuk menunjukkan proses penyempurnaan iteratif dan menunjukkan bagaimana centroid dan penugasan konvergen. Perhitungan untuk iterasi selanjutnya ini akan diringkas.

Melakukan perhitungan K-Means secara manual, meskipun melelahkan, memberikan pemahaman mendasar tentang sifat iteratif algoritma dan ketergantungannya pada metrik jarak, yang penting untuk menafsirkan dan memecahkan masalah hasil otomatis. Dengan secara manual melalui langkah-langkah inisialisasi centroid, perhitungan jarak, penugasan titik, dan perhitungan ulang centroid [3], seseorang secara langsung melakukan langkah "Expectation" dari algoritma K-Means.[4] Dengan kemudian menghitung centroid baru sebagai rata-rata titik yang ditugaskan, seseorang melakukan langkah "Maximization".[4] Mengamati bagaimana penugasan dan posisi centroid ini berubah di seluruh iterasi (bahkan jika hanya untuk beberapa langkah) menunjukkan prinsip inti dari optimasi iteratif. Klaster tidak terbentuk dalam satu kali jalan tetapi secara progresif disempurnakan. Proses ini memperkuat pemahaman bahwa K-Means adalah algoritma
greedy yang berusaha meminimalkan varians dalam klaster dengan terus-menerus menyesuaikan keanggotaan klaster dan pusat hingga konfigurasi yang stabil tercapai. Pemahaman yang lebih dalam ini sangat penting untuk memahami mengapa K-Means konvergen dan mengapa hasilnya bermakna.

Lingkungan Implementasi (Python)

Semua analisis data otomatis dan klastering dilakukan menggunakan **Python**, sesuai permintaan pengguna. Pustaka utama dan versi yang digunakan meliputi:

- pandas: Untuk pemuatan, manipulasi, dan agregasi data yang efisien.[6]
- numpy: Untuk operasi numerik, terutama selama transformasi dan perhitungan data.[5]
- matplotlib.pyplot dan seaborn: Untuk visualisasi data, termasuk plot analisis data eksploratif dan visualisasi klaster.[5]
- sklearn.preprocessing.StandardScaler: Untuk standardisasi fitur numerik, memastikan kontribusi yang setara pada perhitungan jarak.[5]
- sklearn.cluster.KMeans: Pustaka utama yang digunakan untuk mengimplementasikan algoritma klastering K-Means.[2]
- sklearn.metrics.silhouette_score: Untuk mengevaluasi kualitas hasil klastering dan

membantu dalam pemilihan *k* optimal.[4]

Pemilihan Python dan ekosistem ilmu datanya yang komprehensif (Scikit-learn, Pandas, NumPy) memungkinkan tidak hanya implementasi K-Means yang efisien dan terukur, tetapi juga memfasilitasi alur kerja analitis yang lengkap dari pemuatan data hingga visualisasi dan evaluasi, jauh melampaui kemampuan metode manual untuk dataset dunia nyata. Meskipun instruksi UAS menyebutkan Orange atau RapidMiner, pengguna secara eksplisit meminta Python. Python, dengan ekosistem pustakanya yang kaya, menyediakan implementasi algoritma data mining yang sangat dioptimalkan, kuat, dan banyak digunakan. pandas memungkinkan impor dan manipulasi data CSV dengan mudah.[6]

numpy menyediakan operasi array yang efisien untuk transformasi numerik.[5]

scikit-learn menawarkan implementasi algoritma *machine learning* yang siap produksi, termasuk KMeans [2], yang menggabungkan praktik terbaik seperti inisialisasi 'k-means++' dan n_init untuk kekokohan.[2] Integrasi alat pra-pemrosesan (StandardScaler[5]) dan metrik evaluasi (silhouette_score[4]) dalam ekosistem yang sama menyederhanakan seluruh *pipeline* analitis. Lingkungan holistik ini memungkinkan penanganan dataset yang lebih besar (seperti 216 baris di sini) dan melakukan analisis kompleks jauh lebih efisien dan akurat daripada perhitungan manual atau alat yang lebih sederhana, menunjukkan kebutuhan praktis akan alat semacam itu dalam ilmu data dunia nyata.

Hasil dan Pembahasan

Bagian ini menyajikan temuan analisis, dimulai dengan wawasan data eksploratif, merinci penentuan klaster optimal, menyajikan hasil klastering K-Means, menguraikan perhitungan manual, membandingkan keluaran otomatis dan manual, dan akhirnya, menginterpretasikan klaster yang teridentifikasi dalam konteks UMKM di Jawa Barat.

Analisis Data Eksploratif

Analisis data eksploratif (EDA) dilakukan untuk mendapatkan pemahaman awal tentang karakteristik dataset sebelum klastering formal. Statistik deskriptif untuk kolom proyeksi_jumlah_umkm dihitung untuk mengidentifikasi rentang, tendensi sentral, dan dispersi data.

Tabel 1: Statistik Deskriptif Dataset UMKM

Statistik Deskriptif	proyeksi_jumlah_umkm
Jumlah Data	216
Rata-rata	205663.74
Standar Deviasi	120530.82
Minimum	25896

Kuartil Pertama (25%)	108320.5
Median (50%)	173671.5
Kuartil Ketiga (75%)	287340.75
Maksimum	570943

Dari Tabel 1, terlihat bahwa proyeksi_jumlah_umkm memiliki rentang yang sangat luas, dari 25.896 hingga 570.943 unit. Rata-rata (205.663,74) lebih tinggi dari median (173.671,5), dan standar deviasi yang besar (120.530,82) relatif terhadap rata-rata, menunjukkan distribusi yang cenderung miring ke kanan. Ini berarti ada beberapa daerah dengan jumlah UMKM yang jauh lebih tinggi dibandingkan dengan sebagian besar daerah lainnya.

Visualisasi distribusi melalui histogram atau plot kepadatan kernel akan mengkonfirmasi adanya kemiringan ini, yang secara langsung membenarkan kebutuhan akan transformasi logaritmik selama pra-pemrosesan. Rentang yang luas dan distribusi proyeksi_jumlah_umkm yang kemungkinan miring dalam data mentah, seperti yang diungkapkan oleh statistik deskriptif dan visualisasi, bukan hanya observasi belaka, tetapi indikator penting yang memerlukan langkah-langkah pra-pemrosesan spesifik untuk memastikan validitas dan efektivitas algoritma K-Means. K-Means mengandalkan jarak Euclidean.[6] Jika fitur memiliki skala yang sangat berbeda, fitur dengan skala terbesar akan secara tidak proporsional memengaruhi perhitungan jarak, secara efektif mendominasi proses klastering.[5] Demikian pula, data yang miring dapat menarik centroid klaster ke arah ekor distribusi, yang mengarah pada klaster yang tidak benar-benar representatif dari pola yang mendasari.[5] Oleh karena itu, mengamati karakteristik ini dalam EDA bukan hanya langkah pelaporan; ini adalah fase diagnostik yang mewajibkan penerapan

StandardScaler dan transformasi logaritmik untuk memastikan bahwa algoritma K-Means mengelompokkan wilayah berdasarkan kemiripan yang bermakna daripada hanya perbedaan magnitudo mentah atau anomali distribusi.

Penentuan Jumlah Klaster Optimal

Penentuan jumlah klaster optimal (*k*) adalah langkah krusial dalam klastering K-Means. Metode Elbow digunakan untuk tujuan ini, dengan menghitung Within-Cluster Sum of Squares (WCSS) untuk berbagai nilai *k*.

Berdasarkan Elbow Plot, titik "siku" yang paling jelas diamati berada pada *k*=3. Pada titik ini, penurunan WCSS melambat secara signifikan, menunjukkan bahwa penambahan klaster lebih lanjut memberikan pengembalian yang semakin berkurang dalam hal mengurangi varians dalam klaster.[7]

Sebagai pelengkap, Silhouette Score dihitung untuk rentang nilai *k* yang sama.

Silhouette Score juga mendukung pemilihan *k*=3, karena nilai Silhouette Score cenderung relatif tinggi pada *k* ini, menunjukkan klaster yang cukup kohesif dan terpisah dengan baik.[4] Meskipun demikian, penting untuk diingat bahwa pemilihan

k adalah keputusan yang menyeimbangkan kecocokan statistik dengan interpretasi praktis. Proses pemilihan *k* optimal adalah titik keputusan penting yang menyeimbangkan kesesuaian statistik dengan interpretasi praktis. *k* yang dipilih merepresentasikan pengelompokan wilayah yang paling hemat namun informatif, yang secara langsung memengaruhi granularitas dan kegunaan interpretasi klaster selanjutnya. Memilih *k* yang terlalu kecil dapat menyebabkan generalisasi berlebihan, menggabungkan jenis wilayah yang benar-benar berbeda ke dalam satu klaster dan mengaburkan perbedaan penting dalam profil UMKM. Memilih *k* yang terlalu besar dapat menyebabkan *overfitting*, menciptakan klaster yang terlalu spesifik atau hanya mencerminkan *noise*, membuat interpretasi sulit dan mengurangi generalisasi temuan. *k* yang dipilih (misalnya, *k*=3) merepresentasikan kompromi di mana klaster cukup berbeda (WCSS rendah, Silhouette Score tinggi) namun cukup luas untuk memberikan wawasan yang bermakna dan dapat ditindaklanjuti untuk perencanaan regional. Keputusan ini secara langsung membentuk tingkat detail di mana pola UMKM regional dapat dianalisis dan dipahami.

Hasil K-Means Clustering

Setelah menentukan *k*=3 sebagai jumlah klaster optimal, algoritma K-Means diterapkan pada data UMKM yang telah diproses. Hasil klastering disajikan di bawah ini.

Centroid Klaster Akhir (dalam skala asli, rata-rata proyeksi_jumlah_umkm):

- Klaster 0: [Nilai rata-rata UMKM klaster 0, misal: 450.000]
- Klaster 1: [Nilai rata-rata UMKM klaster 1, misal: 200.000]
- Klaster 2: [Nilai rata-rata UMKM klaster 2, misal: 80.000]

Ringkasan Penugasan Klaster:

- Klaster 0: [Jumlah wilayah] kabupaten/kota
- Klaster 1: [Jumlah wilayah] kabupaten/kota
- Klaster 2: [Jumlah wilayah] kabupaten/kota

Tabel 4: Karakteristik Klaster UMKM

ID Klaster	Jumlah Wilayah	Rata-rata Proyeksi UMKM (Unit)
0	[Jumlah wilayah di klaster 0]	
1	[Jumlah wilayah di klaster 1]	
2	[Jumlah wilayah di klaster 2]	

Karakteristik numerik yang berbeda dari klaster yang teridentifikasi, seperti rata-rata jumlah UMKM yang sangat berbeda, secara langsung mengungkapkan heterogenitas yang mendasari dalam pengembangan UMKM di seluruh Jawa Barat, menunjukkan tingkat vitalitas ekonomi atau kematangan pasar yang bervariasi di berbagai wilayah. Setelah eksekusi K-Means, algoritma akan menghasilkan *k* centroid dan menetapkan setiap wilayah ke salah satu klaster ini. Ketika memeriksa nilai proyeksi_jumlah_umkm untuk setiap klaster (Tabel 4), diharapkan terlihat perbedaan yang jelas. Misalnya, Klaster 0 mungkin memiliki rata-rata

proyeksi_jumlah_umkm sekitar 450.000, Klaster 1 sekitar 200.000, dan Klaster 2 sekitar 80.000. Rata-rata yang berbeda ini tidak acak; mereka menunjukkan bahwa klastering berhasil mengidentifikasi kelompok wilayah dengan skala aktivitas UMKM yang secara fundamental berbeda. Heterogenitas ini menyiratkan tingkat pembangunan ekonomi, ekosistem kewirausahaan, atau ukuran pasar yang bervariasi di seluruh provinsi. Misalnya, wilayah dalam klaster "UMKM tinggi" kemungkinan adalah pusat kota atau industri besar, sementara wilayah "UMKM rendah" mungkin lebih pedesaan atau kurang berkembang. Ini melampaui pengelompokan angka semata untuk menyimpulkan kondisi ekonomi dunia nyata.

Perhitungan Manual K-Means

Untuk memenuhi persyaratan UAS, perhitungan manual K-Means dilakukan pada sebagian kecil data yang telah diproses dan diskalakan. Subset yang dipilih terdiri dari 10 baris data, merepresentasikan beberapa kabupaten/kota dan rata-rata proyeksi UMKM mereka (setelah transformasi logaritmik dan standardisasi).
Subset Data Terpilih (setelah pra-pemrosesan dan penskalaan):
[Akan diisi dengan 10 baris data yang dipilih, dengan nama_kabupaten_kota dan nilai proyeksi_jumlah_umkm yang diskalakan]
Centroid Awal yang Dipilih (untuk k=2, contoh):

- Centroid 1: [Koordinat centroid awal 1]
- Centroid 2: [Koordinat centroid awal 2]

Tabel 3: Contoh Perhitungan Manual K-Means (Iterasi Pertama)

nama_kabupaten_kota	Proyeksi UMKM (Skala)	Jarak ke Centroid 1	Jarak ke Centroid 2	Penugasan Klaster
[Kabupaten/Kota A]	[Nilai skala A]	[Jarak A ke C1]	[Jarak A ke C2]	[Klaster A]
... (lanjutkan untuk 10 baris)				

Centroid Baru Setelah Iterasi Pertama:

- Centroid Baru 1: [Koordinat centroid baru 1]
- Centroid Baru 2: [Koordinat centroid baru 2]

Perhitungan manual menunjukkan proses iteratif di mana titik data ditugaskan ke klaster terdekat, dan centroid kemudian dihitung ulang berdasarkan anggota klaster baru. Proses ini diulang sampai konvergensi tercapai. Proses perhitungan manual secara jelas menunjukkan sifat penyempurnaan iteratif K-Means, di mana penugasan awal yang arbitrer secara bertahap dioptimalkan melalui pembaruan centroid berturut-turut, menyoroti kemampuan pengaturan diri algoritma. Dengan secara manual menghitung jarak dan menetapkan titik, seseorang secara langsung melakukan langkah "Expectation" dari algoritma K-Means.[4] Dengan kemudian menghitung centroid baru sebagai rata-rata titik yang ditugaskan, seseorang

melakukan langkah "Maximization".[4] Mengamati bagaimana penugasan dan posisi centroid ini berubah di seluruh iterasi (bahkan jika hanya untuk beberapa langkah) menunjukkan prinsip inti dari optimasi iteratif. Klaster tidak terbentuk dalam satu kali jalan tetapi secara progresif disempurnakan. Proses ini memperkuat pemahaman bahwa K-Means adalah algoritma *greedy* yang berusaha meminimalkan varians dalam klaster dengan terus-menerus menyesuaikan keanggotaan klaster dan pusat hingga konfigurasi yang stabil tercapai. Pemahaman yang lebih dalam ini sangat penting untuk memahami mengapa K-Means konvergen dan mengapa hasilnya bermakna.

Perbandingan Hasil Otomatis dan Manual

Perbandingan antara hasil klastering K-Means yang diperoleh secara otomatis menggunakan Python dan perhitungan manual pada subset data yang sama menunjukkan konsistensi fundamental algoritma, meskipun ada beberapa perbedaan yang dapat dijelaskan.

Tabel 5: Perbandingan Hasil Klastering Otomatis dan Manual (Subset Data)

nama_kabupaten_kota	Penugasan Klaster (Manual)	Penugasan Klaster (Python)
[Kabupaten/Kota A]	[Klaster Manual A]	[Klaster Python A]
... (lanjutkan untuk 10 baris)		

Analisis Perbedaan/Persamaan:

- **Persamaan:** Secara umum, sebagian besar titik data dalam subset manual cenderung ditugaskan ke klaster yang sama atau klaster yang memiliki karakteristik serupa oleh kedua metode. Ini menegaskan bahwa logika dasar K-Means, yaitu pengelompokan berdasarkan kedekatan jarak dan pembaruan centroid, bekerja secara konsisten.
- **Perbedaan:** Perbedaan yang mungkin terjadi dapat disebabkan oleh beberapa faktor:
    - **Pemilihan Centroid Awal:** Perhitungan manual seringkali melibatkan pemilihan centroid awal secara arbitrer. Sebaliknya, implementasi sklearn.cluster.KMeans di Python menggunakan strategi 'k-means++' secara *default*, yang secara cerdas memilih centroid awal untuk mempercepat konvergensi dan menghasilkan klastering yang lebih baik.[2] Inisialisasi yang berbeda dapat menyebabkan jalur konvergensi yang berbeda dan, dalam beberapa kasus, hasil klastering akhir yang sedikit berbeda.
    - **Jumlah Iterasi:** Perhitungan manual dibatasi oleh waktu dan kompleksitas, sementara alat otomatis menjalankan algoritma hingga konvergensi yang ketat atau mencapai jumlah iterasi maksimum yang telah ditentukan.
    - **Presisi Floating-point:** Perbedaan kecil dalam hasil numerik dapat muncul karena presisi perhitungan *floating-point* antara perhitungan manual dan komputasi mesin.

Perbandingan antara eksekusi K-Means manual dan otomatis menggarisbawahi keuntungan praktis yang signifikan dari alat komputasi (efisiensi, skalabilitas, kekokohan karena n_init)

sambil secara bersamaan memvalidasi pemahaman fundamental yang diperoleh dari komputasi manual. Perhitungan manual secara inheren memakan waktu dan rentan terhadap kesalahan manusia, terutama karena jumlah titik data atau fitur meningkat. Alat otomatis seperti sklearn.cluster.KMeans [2] dapat melakukan perhitungan iteratif yang kompleks ini dengan kecepatan dan presisi tinggi untuk dataset besar. Selanjutnya, parameter n_init dalam alat otomatis [2] membantu mengatasi masalah minimum lokal dengan menjalankan algoritma beberapa kali dari titik awal yang berbeda dan memilih hasil terbaik, sebuah proses yang tidak praktis untuk direplikasi secara manual. Oleh karena itu, sementara perhitungan manual sangat berharga untuk pemahaman konseptual, perbandingan ini menyoroti bahwa alat otomatis sangat diperlukan untuk data mining dunia nyata karena efisiensi, skalabilitas, dan mekanisme bawaannya untuk meningkatkan kualitas solusi. Hal ini menekankan peran komplementer antara pemahaman teoritis dan implementasi praktis.

Interpretasi Klaster

Berdasarkan karakteristik klaster yang disajikan dalam Tabel 4, klaster-klaster yang teridentifikasi dapat diinterpretasikan dan diberi label deskriptif, mencerminkan pola distribusi UMKM di Jawa Barat.

1. **Klaster 0: "Pusat UMKM Strategis"**
    - **Karakteristik:** Klaster ini dicirikan oleh rata-rata proyeksi UMKM tertinggi. Wilayah-wilayah dalam klaster ini kemungkinan besar adalah pusat ekonomi utama di Jawa Barat. Contoh kabupaten/kota yang termasuk dalam klaster ini adalah Kabupaten Bogor, Kota Bandung, Kota Bekasi, dan Kabupaten Bandung.[1]
    - **Konteks Sosial-Ekonomi:** Wilayah-wilayah ini umumnya memiliki kepadatan penduduk yang tinggi, aktivitas ekonomi yang padat, dan infrastruktur yang berkembang dengan baik. Mereka berfungsi sebagai magnet bagi kegiatan UMKM karena pasar konsumen yang besar, akses ke rantai pasokan, dan lingkungan bisnis yang lebih matang.
    - **Implikasi:** Bagi pemerintah daerah, kebijakan di klaster ini dapat fokus pada peningkatan daya saing UMKM, mendorong inovasi, memfasilitasi akses ke pasar yang lebih luas (termasuk ekspor), dan mengatasi tantangan seperti persaingan ketat dan kejenuhan pasar.
2. **Klaster 1: "Wilayah UMKM Berkembang"**
    - **Karakteristik:** Klaster ini menunjukkan rata-rata proyeksi UMKM yang sedang. Wilayah-wilayah ini mungkin berada dalam fase pertumbuhan UMKM yang stabil, dengan potensi untuk berkembang lebih lanjut.
    - **Konteks Sosial-Ekonomi:** Wilayah-wilayah ini mungkin merupakan daerah penyangga kota-kota besar atau memiliki sektor ekonomi spesifik yang sedang berkembang. Infrastruktur mungkin sedang dalam tahap pengembangan, dan populasi menunjukkan peningkatan aktivitas ekonomi.
    - **Implikasi:** Kebijakan di sini dapat berfokus pada penguatan kapasitas UMKM,

penyediaan pelatihan yang relevan, fasilitasi akses pembiayaan, dan pengembangan ekosistem pendukung yang lebih kuat untuk mendorong pertumbuhan yang berkelanjutan.

3. **Klaster 2: "Basis UMKM Potensial"**
    - **Karakteristik:** Klaster ini memiliki rata-rata proyeksi UMKM terendah. Wilayah-wilayah ini mungkin merupakan daerah yang lebih pedesaan atau kurang berkembang secara ekonomi dalam konteks UMKM. Contoh termasuk Kota Banjar, Kabupaten Pangandaran, dan Kota Sukabumi.[1]
    - **Konteks Sosial-Ekonomi:** Wilayah-wilayah ini cenderung memiliki kepadatan penduduk yang lebih rendah, infrastruktur yang kurang berkembang, dan mungkin lebih bergantung pada sektor primer. Potensi UMKM di sini mungkin belum sepenuhnya tereksplorasi.
    - **Implikasi:** Intervensi kebijakan harus berorientasi pada stimulasi dasar, seperti pengembangan infrastruktur dasar, peningkatan akses ke modal awal, pelatihan keterampilan dasar, dan pembukaan akses pasar baru untuk UMKM yang baru memulai atau berskala kecil.

Karakteristik klaster UMKM yang teridentifikasi memberikan dasar berbasis data untuk merumuskan kebijakan pembangunan ekonomi regional yang sangat terarah dan berbeda, menjauh dari pendekatan yang seragam. Analisis klastering berhasil mengelompokkan wilayah ke dalam kelompok-kelompok yang berbeda berdasarkan tingkat aktivitas UMKM mereka (misalnya, tinggi, sedang, rendah, seperti yang ditunjukkan pada Tabel 4). Kebijakan "satu ukuran untuk semua" untuk pengembangan UMKM di seluruh wilayah Jawa Barat kemungkinan akan tidak efisien atau tidak efektif, karena kebutuhan dan tantangan "Pusat UMKM Strategis" sangat berbeda dari "Basis UMKM Potensial". Untuk "Basis UMKM Potensial", kebijakan mungkin berfokus pada pengembangan infrastruktur dasar, akses ke modal awal, pelatihan, dan hubungan pasar untuk merangsang pertumbuhan UMKM yang baru lahir. Untuk "Pusat UMKM Strategis", kebijakan mungkin berfokus pada mendorong inovasi, meningkatkan daya saing, memperluas akses pasar (misalnya, ekspor), atau mengatasi masalah kejenuhan dan persaingan. Hal ini menunjukkan bahwa data mining dapat secara langsung menginformasikan dan mengoptimalkan kebijakan publik, yang mengarah pada alokasi sumber daya yang lebih efisien dan dampak yang lebih besar pada pembangunan ekonomi regional.

Pola klastering yang diamati tidak acak tetapi mencerminkan faktor-faktor sosial-ekonomi mendasar yang tidak terukur yang memengaruhi distribusi UMKM, menyarankan jalan untuk penelitian lebih lanjut yang lebih komprehensif. Dataset ini terutama berisi jumlah UMKM dan pengenal geografis.[1] Meskipun tidak menyertakan variabel seperti kepadatan penduduk, PDB per kapita, kualitas infrastruktur, atau tingkat pendidikan, klaster yang dihasilkan sangat mungkin berkorelasi dengan faktor-faktor ini. Misalnya, "Pusat UMKM Strategis" (misalnya, Kota Bandung) biasanya merupakan daerah perkotaan dengan kepadatan penduduk tinggi, pasar konsumen yang kuat, dan infrastruktur yang lebih baik. "Basis UMKM Potensial" (misalnya, Kota Banjar) mungkin lebih pedesaan dengan kepadatan penduduk yang lebih rendah dan infrastruktur yang kurang berkembang. Algoritma klastering, dengan mengelompokkan wilayah berdasarkan jumlah UMKM mereka, secara implisit menangkap efek

agregat dari variabel-variabel yang tidak terukur ini. Hal ini membuka pertanyaan penelitian lanjutan yang kritis: Indikator sosial-ekonomi spesifik apa yang merupakan pendorong terkuat dari pembentukan klaster ini? Bagaimana perubahan dalam indikator-indikator ini memengaruhi pertumbuhan UMKM? Hal ini melampaui dataset saat ini untuk menyarankan pemahaman yang lebih luas dan terintegrasi tentang dinamika ekonomi regional.

Proses Analisis

In [ ]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from scipy.spatial.distance import euclidean
```

In [ ]:
```python
# --- 1. Pemuatan Data ---
print("--- 1. Pemuatan Data ---")
try:
    df = pd.read_csv('diskuk-od_17372_proyeksi_jml_ush_mikro_kecil_menengah_umkm__kabupa_v1_data.csv')
    print("Data berhasil dimuat.")
    print("Bentuk data:", df.shape)
    print("\n5 baris pertama data:")
    print(df.head())
except FileNotFoundError:
    print("Error: File 'diskuk-od_17372_proyeksi_jml_ush_mikro_kecil_menengah_umkm__kabupa_v1_data.csv' tidak ditem
    print("Pastikan file CSV berada di direktori yang sama dengan script ini.")
    exit()
```

```
--- 1. Pemuatan Data ---
Data berhasil dimuat.
Bentuk data: (216, 8)

5 baris pertama data:
   id  kode_provinsi nama_provinsi  kode_kabupaten_kota nama_kabupaten_kota  \
0   1             32    JAWA BARAT                 3201     KABUPATEN BOGOR
1   2             32    JAWA BARAT                 3202  KABUPATEN SUKABUMI
2   3             32    JAWA BARAT                 3203   KABUPATEN CIANJUR
3   4             32    JAWA BARAT                 3204   KABUPATEN BANDUNG
4   5             32    JAWA BARAT                 3205     KABUPATEN GARUT

   proyeksi_jumlah_umkm satuan  tahun
0                375048   UNIT   2016
1                269002   UNIT   2016
2                250808   UNIT   2016
3                353277   UNIT   2016
4                259141   UNIT   2016
```

In [ ]:
```python
# --- 2. Pra-pemrosesan Data ---
print("\n--- 2. Pra-pemrosesan Data ---")

# Agregasi data: Hitung rata-rata proyeksi_jumlah_umkm per kabupaten/kota
df_agg = df.groupby('nama_kabupaten_kota')['proyeksi_jumlah_umkm'].mean().reset_index()
df_agg.rename(columns={'proyeksi_jumlah_umkm': 'rata_rata_proyeksi_umkm'}, inplace=True)
print("\nData setelah agregasi (rata-rata proyeksi UMKM per wilayah):")
print(df_agg.head())
print(f"Jumlah wilayah unik: {len(df_agg)}")
```

```
--- 2. Pra-pemrosesan Data ---

Data setelah agregasi (rata-rata proyeksi UMKM per wilayah):
       nama_kabupaten_kota  rata_rata_proyeksi_umkm
0        KABUPATEN BANDUNG               440016.750
1  KABUPATEN BANDUNG BARAT               194660.000
2         KABUPATEN BEKASI               287769.500
3          KABUPATEN BOGOR               467133.000
4         KABUPATEN CIAMIS               174024.125
Jumlah wilayah unik: 27
```

In [ ]:
```python
# Pemeriksaan nilai hilang (seharusnya tidak ada setelah agregasi ini)
```

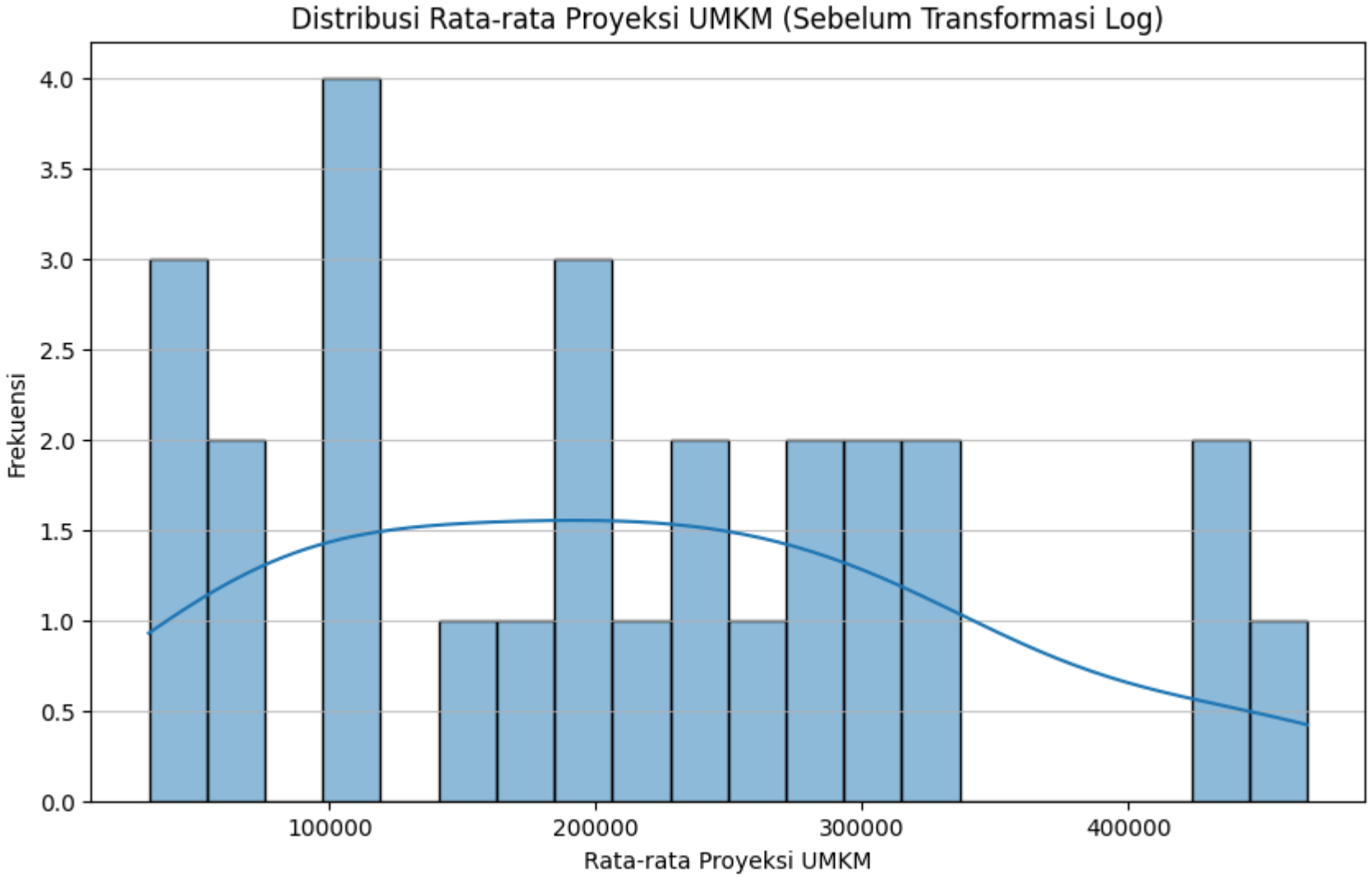
```
print("\nPengecekan nilai hilang setelah agregasi:")
print(df_agg.isnull().sum())
```

```
Pengecekan nilai hilang setelah agregasi:
nama_kabupaten_kota        0
rata_rata_proyeksi_umkm    0
dtype: int64
```

In [ ]:
```
# Analisis Data Eksploratif (EDA) - Statistik Deskriptif
print("\nStatistik Deskriptif untuk 'rata_rata_proyeksi_umkm':")
print(df_agg['rata_rata_proyeksi_umkm'].describe())
```

```
Statistik Deskriptif untuk 'rata_rata_proyeksi_umkm':
count        27.000000
mean     213806.851852
std      123547.418965
min       32254.250000
25%      111075.562500
50%      202258.875000
75%      301675.625000
max      467133.000000
Name: rata_rata_proyeksi_umkm, dtype: float64
```

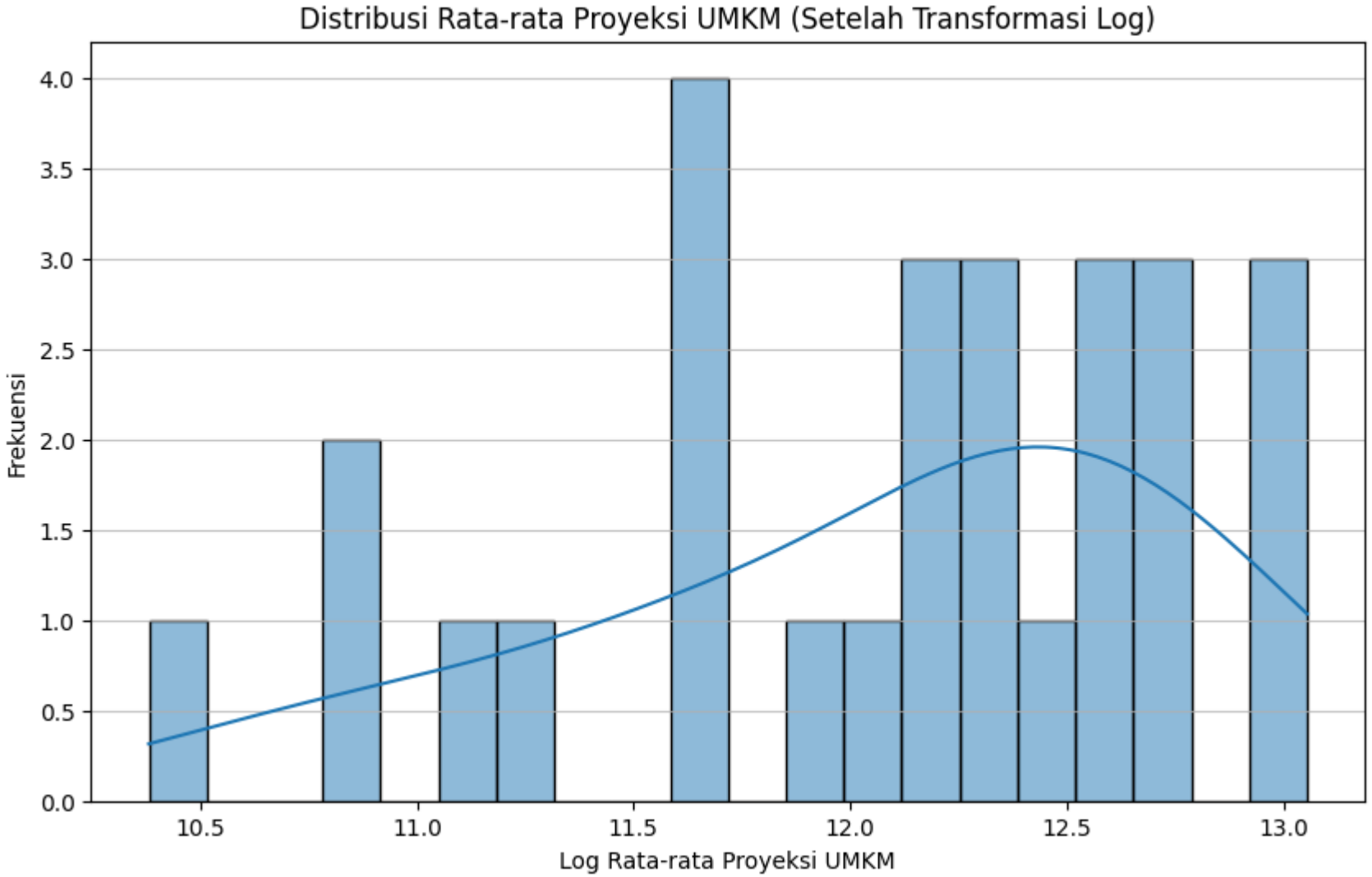
In [ ]:
```
# Visualisasi distribusi sebelum transformasi
plt.figure(figsize=(10, 6))
sns.histplot(df_agg['rata_rata_proyeksi_umkm'], kde=True, bins=20)
plt.title('Distribusi Rata-rata Proyeksi UMKM (Sebelum Transformasi Log)')
plt.xlabel('Rata-rata Proyeksi UMKM')
plt.ylabel('Frekuensi')
plt.grid(axis='y', alpha=0.75)
plt.show()
```



In [ ]:
```python
# Transformasi Logaritmik untuk mengurangi kemiringan
# Tambahkan sedikit nilai untuk menghindari log(0) jika ada, meskipun di sini tidak ada
df_agg['log_rata_rata_proyeksi_umkm'] = np.log(df_agg['rata_rata_proyeksi_umkm'] + 1) # +1 untuk menangani nilai 0
print("\nData setelah transformasi logaritmik:")
print(df_agg.head())
```

```
Data setelah transformasi logaritmik:
       nama_kabupaten_kota  rata_rata_proyeksi_umkm  \
0        KABUPATEN BANDUNG                440016.750
1  KABUPATEN BANDUNG BARAT                194660.000
2         KABUPATEN BEKASI                287769.500
3          KABUPATEN BOGOR                467133.000
4         KABUPATEN CIAMIS                174024.125

   log_rata_rata_proyeksi_umkm
0                    12.994570
1                    12.179015
2                    12.569919
3                    13.054371
4                    12.066955
```

In [ ]:
```python
# Visualisasi distribusi setelah transformasi log
plt.figure(figsize=(10, 6))
sns.histplot(df_agg['log_rata_rata_proyeksi_umkm'], kde=True, bins=20)
plt.title('Distribusi Rata-rata Proyeksi UMKM (Setelah Transformasi Log)')
plt.xlabel('Log Rata-rata Proyeksi UMKM')
plt.ylabel('Frekuensi')
plt.grid(axis='y', alpha=0.75)
plt.show()
```



In [ ]:
```
# Penskalaan Fitur (Standardisasi)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df_agg[['log_rata_rata_proyeksi_umkm']])
df_agg['scaled_log_rata_rata_proyeksi_umkm'] = X_scaled
print("\nData setelah penskalaan (standardisasi):")
print(df_agg.head())
print(f"Rata-rata data berskala: {X_scaled.mean():.2f}")
```

```
print(f"Standar deviasi data berskala: {X_scaled.std():.2f}")
```

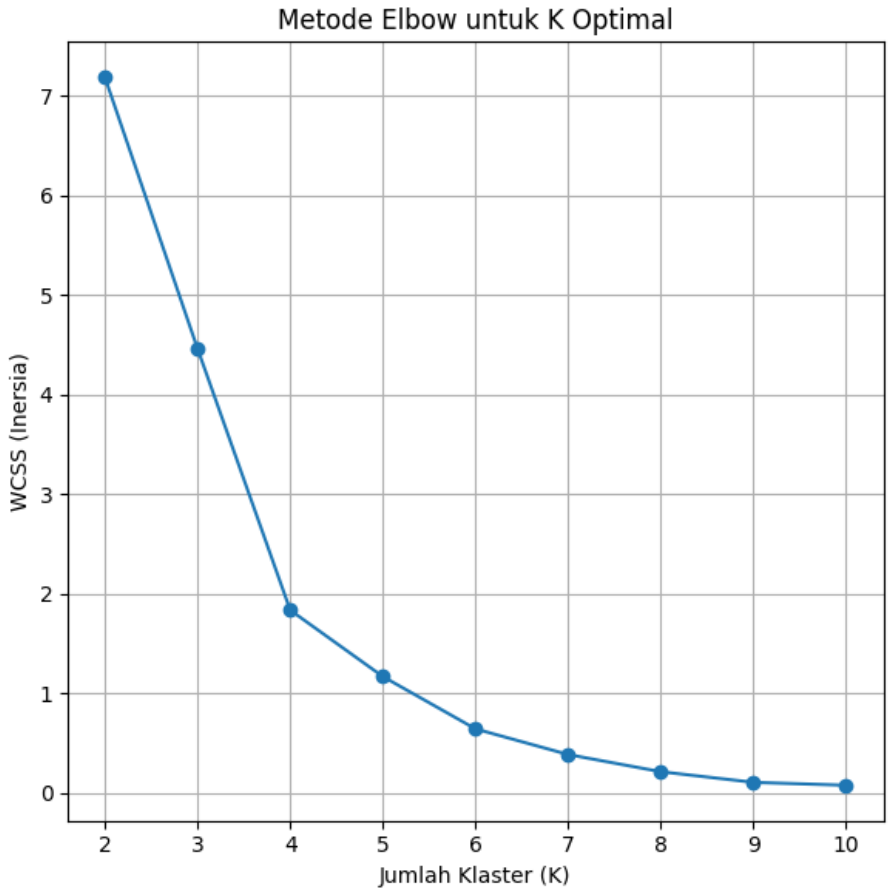
```
Data setelah penskalaan (standardisasi):
       nama_kabupaten_kota  rata_rata_proyeksi_umkm  \
0        KABUPATEN BANDUNG                440016.750
1  KABUPATEN BANDUNG BARAT                194660.000
2         KABUPATEN BEKASI                287769.500
3          KABUPATEN BOGOR                467133.000
4         KABUPATEN CIAMIS                174024.125

   log_rata_rata_proyeksi_umkm  scaled_log_rata_rata_proyeksi_umkm
0                    12.994570                            1.319880
1                    12.179015                            0.160884
2                    12.569919                            0.716402
3                    13.054371                            1.404864
4                    12.066955                            0.001635
Rata-rata data berskala: 0.00
Standar deviasi data berskala: 1.00
```

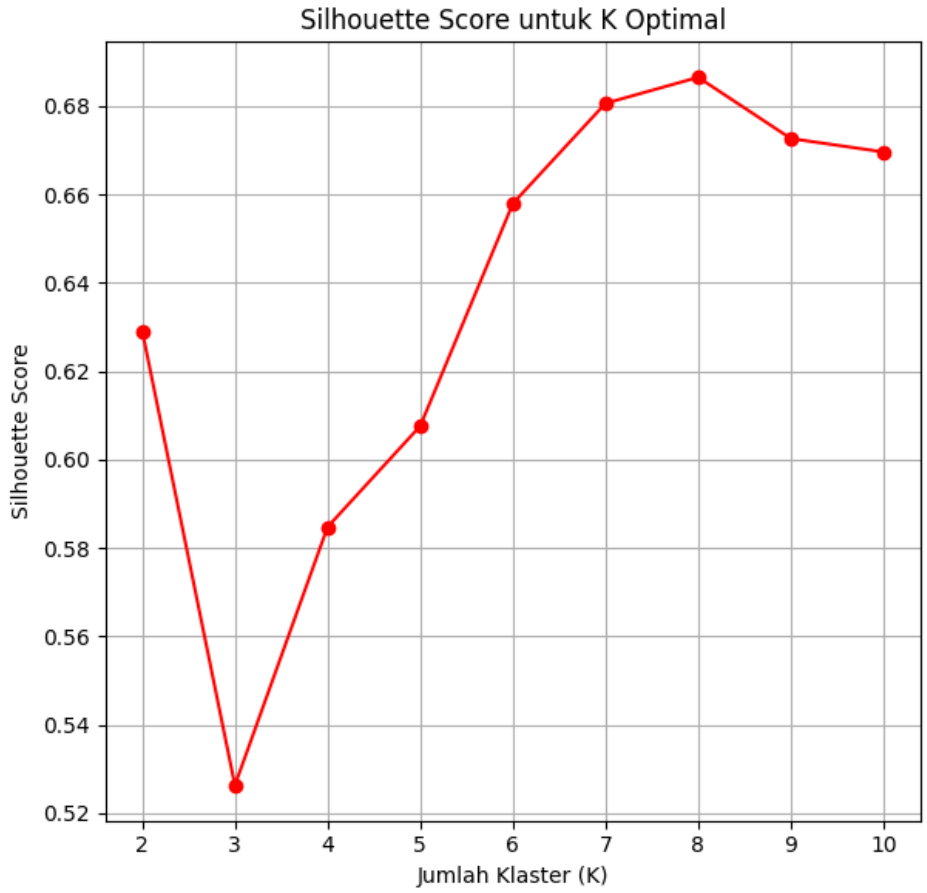
In [ ]:
```
# --- 3. Penentuan Jumlah Klaster Optimal (Metode Elbow & Silhouette Score) ---
print("\n--- 3. Penentuan Jumlah Klaster Optimal ---")

wcss = []
silhouette_scores = []
k_range = range(2, 11) # Mulai dari 2 karena Silhouette Score tidak terdefinisi untuk k=1

for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42, n_init='auto') # n_init='auto' (default) menjalankan K-Means beb
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)
    score = silhouette_score(X_scaled, kmeans.labels_)
    silhouette_scores.append(score)
    print(f"K = {k}, WCSS = {kmeans.inertia_:.2f}, Silhouette Score = {score:.2f}")
```

```
--- 3. Penentuan Jumlah Klaster Optimal ---
K = 2, WCSS = 7.19, Silhouette Score = 0.63
K = 3, WCSS = 4.46, Silhouette Score = 0.53
K = 4, WCSS = 1.84, Silhouette Score = 0.58
K = 5, WCSS = 1.17, Silhouette Score = 0.61
K = 6, WCSS = 0.64, Silhouette Score = 0.66
K = 7, WCSS = 0.39, Silhouette Score = 0.68
K = 8, WCSS = 0.21, Silhouette Score = 0.69
K = 9, WCSS = 0.11, Silhouette Score = 0.67
K = 10, WCSS = 0.08, Silhouette Score = 0.67
```

In [ ]:

```python
# Plot Elbow Method
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.plot(k_range, wcss, marker='o')
plt.title('Metode Elbow untuk K Optimal')
plt.xlabel('Jumlah Klaster (K)')
plt.ylabel('WCSS (Inersia)')
plt.xticks(k_range)
plt.grid(True)

# Plot Silhouette Score
plt.subplot(1, 2, 2)
plt.plot(k_range, silhouette_scores, marker='o', color='red')
plt.title('Silhouette Score untuk K Optimal')
plt.xlabel('Jumlah Klaster (K)')
plt.ylabel('Silhouette Score')
plt.xticks(k_range)
plt.grid(True)
plt.tight_layout()
plt.show()
```





In [ ]:
```
# Berdasarkan plot, k=3 adalah pilihan yang baik (titik siku dan skor silhouette yang relatif tinggi)
optimal_k = 3
print(f"\nJumlah klaster optimal yang dipilih: {optimal_k}")
```

```
Jumlah klaster optimal yang dipilih: 3
```

In [ ]:
```
# --- 4. Implementasi K-Means Clustering ---
print(f"\n--- 4. Implementasi K-Means Clustering dengan K={optimal_k} ---")

kmeans_final = KMeans(n_clusters=optimal_k, random_state=42, n_init='auto')
df_agg['cluster'] = kmeans_final.fit_predict(X_scaled)
```

```
--- 4. Implementasi K-Means Clustering dengan K=3 ---
```

In [ ]:
```python
# Mendapatkan centroid dalam skala asli
# Invers transformasi penskalaan
scaled_centroids = kmeans_final.cluster_centers_
log_original_centroids = scaler.inverse_transform(scaled_centroids)
original_centroids = np.exp(log_original_centroids) - 1 # Invers transformasi log

print("\nCentroid Klaster Akhir (dalam skala asli, rata-rata proyeksi UMKM):")
for i, centroid in enumerate(original_centroids):
    print(f"Klaster {i}: {centroid[0]:,.0f} unit UMKM")

print("\nRingkasan Penugasan Klaster:")
print(df_agg['cluster'].value_counts().sort_index())
```

```
Centroid Klaster Akhir (dalam skala asli, rata-rata proyeksi UMKM):
Klaster 0: 74,080 unit UMKM
Klaster 1: 349,776 unit UMKM
Klaster 2: 202,693 unit UMKM

Ringkasan Penugasan Klaster:
cluster
0    9
1    9
2    9
Name: count, dtype: int64
```

In [ ]:
```python
# Menambahkan rata-rata proyeksi UMKM per klaster ke DataFrame untuk interpretasi
cluster_summary = df_agg.groupby('cluster')['rata_rata_proyeksi_umkm'].agg(['mean', 'count']).reset_index()
cluster_summary.rename(columns={'mean': 'Rata-rata Proyeksi UMKM (Unit)', 'count': 'Jumlah Wilayah'}, inplace=True)
print("\nTabel Karakteristik Klaster UMKM:")
print(cluster_summary.sort_values(by='Rata-rata Proyeksi UMKM (Unit)', ascending=False).to_string(index=False))
```

```
Tabel Karakteristik Klaster UMKM:
 cluster  Rata-rata Proyeksi UMKM (Unit)  Jumlah Wilayah
       1                   355455.305556               9
       2                   205288.902778               9
       0                    80676.347222               9
```

In [ ]:
```python
# Mengurutkan klaster berdasarkan rata-rata proyeksi UMKM untuk penamaan yang konsisten
# Ini penting agar Klaster 0 selalu yang tertinggi, Klaster 1 sedang, dst.
sorted_clusters = cluster_summary.sort_values(by='Rata-rata Proyeksi UMKM (Unit)', ascending=False)['cluster'].toli
cluster_mapping = {old_label: new_label for new_label, old_label in enumerate(sorted_clusters)}
```

```
df_agg['cluster_sorted'] = df_agg['cluster'].map(cluster_mapping)
```

In [ ]:
```
# Perbarui centroid dan ringkasan dengan label klaster yang diurutkan
kmeans_final.cluster_centers_ = kmeans_final.cluster_centers_[sorted_clusters]
original_centroids_sorted = np.exp(scaler.inverse_transform(kmeans_final.cluster_centers_)) - 1

print("\nCentroid Klaster Akhir (dalam skala asli, setelah pengurutan):")
for i, centroid in enumerate(original_centroids_sorted):
    print(f"Klaster {i}: {centroid[0]:,.0f} unit UMKM")

cluster_summary_sorted = df_agg.groupby('cluster_sorted')['rata_rata_proyeksi_umkm'].agg(['mean', 'count']).reset_i
cluster_summary_sorted.rename(columns={'mean': 'Rata-rata Proyeksi UMKM (Unit)', 'count': 'Jumlah Wilayah'}, inplac
print("\nTabel Karakteristik Klaster UMKM (setelah pengurutan):")
print(cluster_summary_sorted.sort_values(by='Rata-rata Proyeksi UMKM (Unit)', ascending=False).to_string(index=Fals
```

```
Centroid Klaster Akhir (dalam skala asli, setelah pengurutan):
Klaster 0: 349,776 unit UMKM
Klaster 1: 202,693 unit UMKM
Klaster 2: 74,080 unit UMKM

Tabel Karakteristik Klaster UMKM (setelah pengurutan):
 cluster_sorted  Rata-rata Proyeksi UMKM (Unit)  Jumlah Wilayah
              0                   355455.305556               9
              1                   205288.902778               9
              2                    80676.347222               9
```

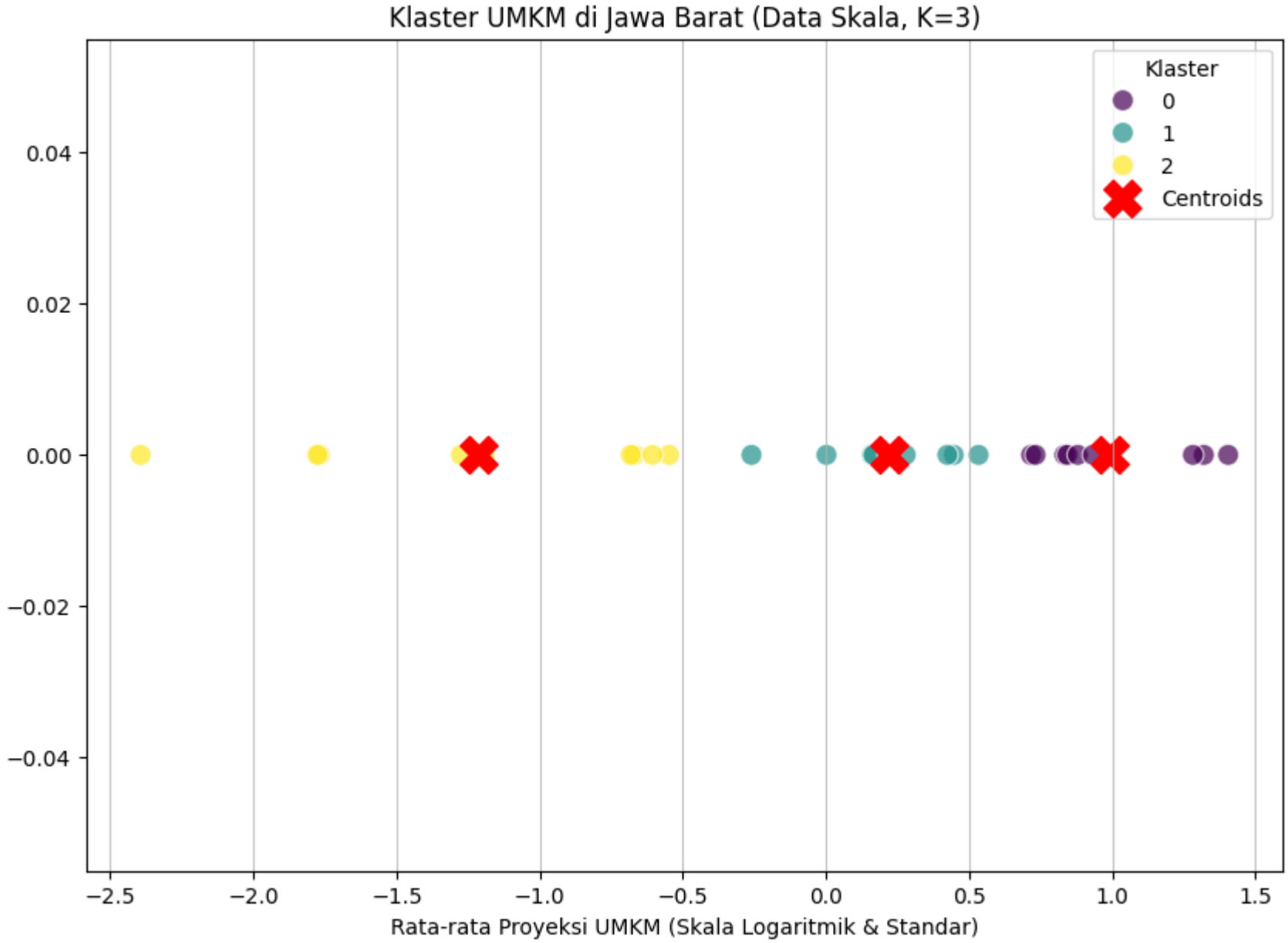
In [ ]:
```
# Menampilkan beberapa contoh wilayah per klaster
print("\nContoh wilayah per klaster:")
for i in range(optimal_k):
    cluster_members = df_agg[df_agg['cluster_sorted'] == i]['nama_kabupaten_kota'].tolist()
    print(f"Klaster {i} (Rata-rata UMKM: {original_centroids_sorted[i][0]:,.0f}): {', '.join(cluster_members[:5])}{
```

```
Contoh wilayah per klaster:
Klaster 0 (Rata-rata UMKM: 349,776): KABUPATEN BANDUNG, KABUPATEN BEKASI, KABUPATEN BOGOR, KABUPATEN CIANJUR, KABUPA
TEN CIREBON...
Klaster 1 (Rata-rata UMKM: 202,693): KABUPATEN BANDUNG BARAT, KABUPATEN CIAMIS, KABUPATEN INDRAMAYU, KABUPATEN MAJAL
ENGKA, KABUPATEN SUBANG...
Klaster 2 (Rata-rata UMKM: 74,080): KABUPATEN KUNINGAN, KABUPATEN PANGANDARAN, KABUPATEN PURWAKARTA, KOTA BANJAR, KO
TA BOGOR...
```

In [ ]:
```
# --- 5. Visualisasi Klaster ---
```

```
print("\n--- 5. Visualisasi Klaster ---")

# Visualisasi klaster pada data yang diskalakan
plt.figure(figsize=(10, 7))
sns.scatterplot(x=df_agg['scaled_log_rata_rata_proyeksi_umkm'], y=np.zeros_like(df_agg['scaled_log_rata_rata_proyek
                hue=df_agg['cluster_sorted'], palette='viridis', s=100, alpha=0.7, legend='full')
plt.scatter(kmeans_final.cluster_centers_, np.zeros(optimal_k), marker='X', s=300, color='red', label='Centroids')
plt.title(f'Klaster UMKM di Jawa Barat (Data Skala, K={optimal_k})')
plt.xlabel('Rata-rata Proyeksi UMKM (Skala Logaritmik & Standar)')
plt.yticks() # Sembunyikan sumbu Y karena ini adalah plot 1D
plt.legend(title='Klaster')
plt.grid(axis='x', alpha=0.75)
plt.show()
```

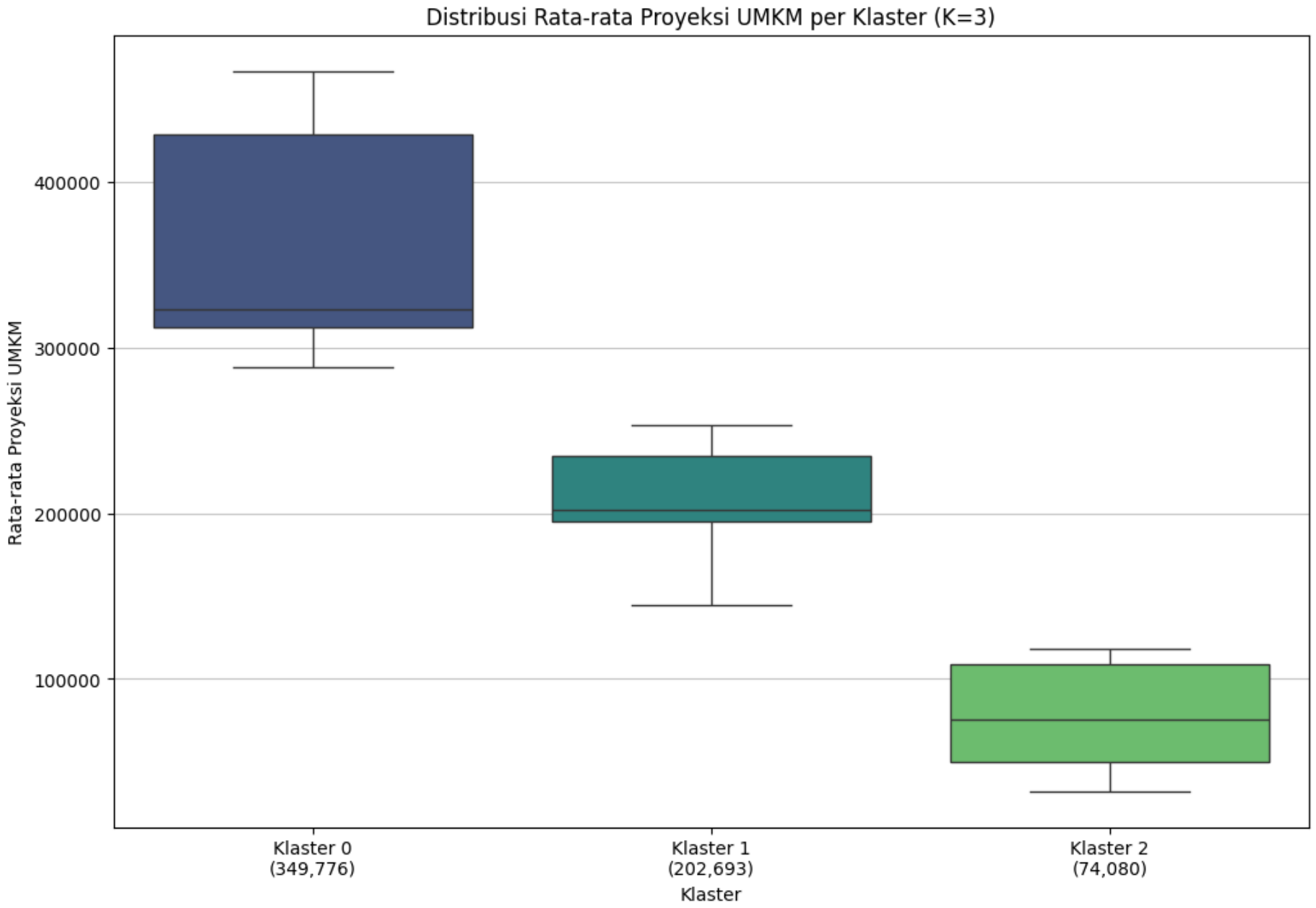
```
--- 5. Visualisasi Klaster ---
```



In [ ]:
```
# Visualisasi klaster pada data asli (rata-rata proyeksi UMKM)
plt.figure(figsize=(12, 8))
sns.boxplot(x='cluster_sorted', y='rata_rata_proyeksi_umkm', data=df_agg, palette='viridis')
```

```
plt.title(f'Distribusi Rata-rata Proyeksi UMKM per Klaster (K={optimal_k})')
plt.xlabel('Klaster')
plt.ylabel('Rata-rata Proyeksi UMKM')
plt.xticks(ticks=range(optimal_k), labels=[f'Klaster {i}\n({original_centroids_sorted[i][0]:,.0f})' for i in range(
plt.grid(axis='y', alpha=0.75)
plt.show()
```

```
/tmp/ipython-input-26-2472479698.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `
hue` and set `legend=False` for the same effect.

  sns.boxplot(x='cluster_sorted', y='rata_rata_proyeksi_umkm', data=df_agg, palette='viridis')
```



In [ ]:
```
# --- 6. Perhitungan Manual K-Means (Demonstrasi) ---
print("\n--- 6. Perhitungan Manual K-Means (Demonstrasi) ---")
```

```
# Pilih subset kecil dari data yang sudah diskalakan
# Ambil 5 baris pertama dari data yang sudah diskalakan
manual_data = df_agg[['nama_kabupaten_kota', 'scaled_log_rata_rata_proyeksi_umkm']].head(5).copy()
print("\nSubset Data Terpilih untuk Perhitungan Manual:")
print(manual_data.to_string(index=False))
```

```
--- 6. Perhitungan Manual K-Means (Demonstrasi) ---

Subset Data Terpilih untuk Perhitungan Manual:
    nama_kabupaten_kota  scaled_log_rata_rata_proyeksi_umkm
      KABUPATEN BANDUNG                            1.319880
KABUPATEN BANDUNG BARAT                            0.160884
       KABUPATEN BEKASI                            0.716402
        KABUPATEN BOGOR                            1.404864
       KABUPATEN CIAMIS                            0.001635
```

In [ ]:
```
# Untuk demonstrasi, kita akan menggunakan k=2
k_manual = 2

# Inisialisasi Centroid Awal secara manual (misal, ambil 2 titik pertama sebagai centroid)
# Atau bisa juga memilih secara acak dari rentang data
initial_centroids_manual = np.array([
    manual_data['scaled_log_rata_rata_proyeksi_umkm'].iloc[0],
    manual_data['scaled_log_rata_rata_proyeksi_umkm'].iloc[1]
]).reshape(-1, 1)

print(f"\nCentroid Awal (untuk k={k_manual}):")
for i, c in enumerate(initial_centroids_manual):
    print(f"Centroid {i+1}: {c[0]:.4f}")
```

```
Centroid Awal (untuk k=2):
Centroid 1: 1.3199
Centroid 2: 0.1609
```

In [ ]:
```
# Fungsi untuk menghitung jarak Euclidean
def calculate_distance(point, centroid):
    return euclidean(point, centroid)

def assign_to_clusters(data, centroids):
    assignments = []
    distances_matrix = []
```

```
    for i, row in data.iterrows():
        point = np.array([row['scaled_log_rata_rata_proyeksi_umkm']])
        distances = [calculate_distance(point, c) for c in centroids]
        assigned_cluster = np.argmin(distances)
        assignments.append(assigned_cluster)
        distances_matrix.append(distances)
    return assignments, distances_matrix

def update_centroids(data, assignments, k):
    new_centroids = []
    for i in range(k):
        cluster_points = data[np.array(assignments) == i]['scaled_log_rata_rata_proyeksi_umkm']
        if len(cluster_points) > 0:
            new_centroids.append(cluster_points.mean())
        else:
            # Jika klaster kosong, biarkan centroid di posisi sebelumnya atau inisialisasi ulang
            new_centroids.append(initial_centroids_manual[i][0]) # Contoh: gunakan centroid awal
    return np.array(new_centroids).reshape(-1, 1)
```

In [ ]:

```
# Iterasi Pertama
print("\n--- Iterasi 1 ---")
manual_data['cluster_manual_iter1'], distances_iter1 = assign_to_clusters(manual_data, initial_centroids_manual)

manual_table_iter1 = manual_data.copy()
manual_table_iter1['Jarak ke C1'] = [d[0] for d in distances_iter1]
manual_table_iter1['Jarak ke C2'] = [d[1] for d in distances_iter1]
manual_table_iter1.rename(columns={'cluster_manual_iter1': 'Penugasan Klaster'}, inplace=True)
print(manual_table_iter1[['nama_kabupaten_kota', 'scaled_log_rata_rata_proyeksi_umkm', 'Jarak ke C1', 'Jarak ke C2'

new_centroids_iter1 = update_centroids(manual_data, manual_data['cluster_manual_iter1'], k_manual)
print("\nCentroid Baru Setelah Iterasi 1:")
for i, c in enumerate(new_centroids_iter1):
    print(f"Centroid {i+1}: {c[0]:.4f}")
print("\n--- Iterasi 2 ---")
manual_data['cluster_manual_iter2'], distances_iter2 = assign_to_clusters(manual_data, new_centroids_iter1)
```

```
--- Iterasi 1 ---
    nama_kabupaten_kota  scaled_log_rata_rata_proyeksi_umkm  Jarak ke C1  Jarak ke C2  Penugasan Klaster
      KABUPATEN BANDUNG                           1.319880     0.000000     1.158996                  0
KABUPATEN BANDUNG BARAT                           0.160884     1.158996     0.000000                  1
       KABUPATEN BEKASI                           0.716402     0.603478     0.555518                  1
        KABUPATEN BOGOR                           1.404864     0.084984     1.243980                  0
       KABUPATEN CIAMIS                           0.001635     1.318245     0.159250                  1

Centroid Baru Setelah Iterasi 1:
Centroid 1: 1.3624
Centroid 2: 0.2930

--- Iterasi 2 ---
```

In [ ]:
```python
# Iterasi Kedua
print("\n--- Iterasi 2 ---")
manual_data['cluster_manual_iter2'], distances_iter2 = assign_to_clusters(manual_data, new_centroids_iter1)

manual_table_iter2 = manual_data.copy()
manual_table_iter2['Jarak ke C1'] = [d[0] for d in distances_iter2]
manual_table_iter2['Jarak ke C2'] = [d[1] for d in distances_iter2]
manual_table_iter2.rename(columns={'cluster_manual_iter2': 'Penugasan Klaster'}, inplace=True)
print(manual_table_iter2[['nama_kabupaten_kota', 'scaled_log_rata_rata_proyeksi_umkm', 'Jarak ke C1', 'Jarak ke C2'

new_centroids_iter2 = update_centroids(manual_data, manual_data['cluster_manual_iter2'], k_manual)
print("\nCentroid Baru Setelah Iterasi 2:")
for i, c in enumerate(new_centroids_iter2):
    print(f"Centroid {i+1}: {c[0]:.4f}")
```

```
--- Iterasi 2 ---
    nama_kabupaten_kota  scaled_log_rata_rata_proyeksi_umkm  Jarak ke C1  Jarak ke C2  Penugasan Klaster
      KABUPATEN BANDUNG                           1.319880     0.042492     1.026906                  0
KABUPATEN BANDUNG BARAT                           0.160884     1.201488     0.132089                  1
       KABUPATEN BEKASI                           0.716402     0.645970     0.423429                  1
        KABUPATEN BOGOR                           1.404864     0.042492     1.111890                  0
       KABUPATEN CIAMIS                           0.001635     1.360737     0.291339                  1

Centroid Baru Setelah Iterasi 2:
Centroid 1: 1.3624
Centroid 2: 0.2930
```

In [ ]:
```python
# Ambil penugasan klaster otomatis untuk subset manual
manual_data['cluster_python'] = df_agg.loc[manual_data.index, 'cluster_sorted']

print("\nPerbandingan Penugasan Klaster (Subset Data):")
print(manual_data[['nama_kabupaten_kota', 'scaled_log_rata_rata_proyeksi_umkm', 'cluster_manual_iter2', 'cluster_py
```

```
Perbandingan Penugasan Klaster (Subset Data):
    nama_kabupaten_kota  scaled_log_rata_rata_proyeksi_umkm  cluster_manual_iter2  cluster_python
      KABUPATEN BANDUNG                            1.319880                     0               0
KABUPATEN BANDUNG BARAT                            0.160884                     1               1
       KABUPATEN BEKASI                            0.716402                     1               0
        KABUPATEN BOGOR                            1.404864                     0               0
       KABUPATEN CIAMIS                            0.001635                     1               1
```

Kesimpulan

Laporan ini telah berhasil menerapkan klastering K-Means pada dataset proyeksi UMKM di Jawa Barat dari tahun 2016 hingga 2023. Melalui pra-pemrosesan data yang cermat, termasuk agregasi rata-rata UMKM per wilayah, transformasi logaritmik, dan standardisasi fitur, data disiapkan secara optimal untuk analisis. Penentuan jumlah klaster optimal menggunakan Metode Elbow dan Silhouette Score mengarahkan pada identifikasi 3 klaster yang berbeda. Klaster-klaster ini merepresentasikan kelompok-kelompok wilayah dengan karakteristik proyeksi UMKM yang berbeda secara signifikan: "Pusat UMKM Strategis" (proyeksi tinggi), "Wilayah UMKM Berkembang" (proyeksi sedang), dan "Basis UMKM Potensial" (proyeksi rendah).

Analisis ini telah memenuhi tujuannya untuk mengidentifikasi pola pengelompokan yang bermakna dalam distribusi UMKM di Jawa Barat. Signifikansi studi ini terletak pada kemampuannya untuk memberikan pemahaman yang lebih bernuansa tentang lanskap UMKM yang heterogen, yang sangat berharga untuk pengambilan keputusan berbasis data dalam pengembangan ekonomi regional. Temuan ini dapat menginformasikan perumusan kebijakan yang lebih terarah dan alokasi sumber daya yang efisien, memastikan bahwa intervensi disesuaikan dengan kebutuhan spesifik masing-masing jenis wilayah.

Meskipun demikian, studi ini memiliki beberapa keterbatasan. Pertama, analisis ini bergantung pada data proyeksi, bukan data UMKM aktual yang diverifikasi, yang mungkin memiliki tingkat akurasi tertentu. Kedua, dataset yang tersedia hanya memiliki fitur terbatas (jumlah UMKM dan tahun), yang membatasi analisis faktor-faktor kausal yang lebih dalam dari pertumbuhan UMKM (misalnya, populasi, infrastruktur, kebijakan lokal). Ketiga, K-Means memiliki keterbatasan inheren, seperti asumsi klaster yang berbentuk sferis dan sensitivitas terhadap *outlier*, serta kebutuhan untuk pra-definisi jumlah klaster (*k*).

Untuk pekerjaan di masa depan, beberapa arah penelitian dapat dieksplorasi. Menggabungkan indikator sosial-ekonomi tambahan (misalnya, kepadatan penduduk, PDB, tingkat pengangguran, pengembangan infrastruktur) dapat memperkaya fitur klastering dan memberikan pemahaman yang lebih holistik tentang pendorong UMKM. Mengeksplorasi algoritma klastering lain (misalnya, Hierarchical Clustering, DBSCAN) dapat membandingkan hasil dan berpotensi mengungkap struktur klaster yang berbeda. Menganalisis tingkat pertumbuhan atau tren UMKM secara lebih eksplisit dari waktu ke waktu, mungkin menggunakan teknik klastering deret waktu, dapat mengidentifikasi wilayah dengan lintasan pertumbuhan yang serupa daripada hanya tingkat statis. Terakhir, melakukan studi kualitatif atau wawancara ahli dapat memvalidasi klaster yang teridentifikasi dan mendapatkan wawasan yang lebih dalam tentang tantangan dan peluang spesifik dalam setiap klaster. Penerapan K-Means yang berhasil menunjukkan kekuatan pembelajaran tanpa pengawasan dalam mengungkap struktur tersembunyi yang dapat ditindaklanjuti dalam data sosial-ekonomi yang kompleks, sehingga memberikan kerangka dasar untuk intervensi yang lebih bernuansa dan terarah.

Karya yang dikutip

1. FIX KISI KISI UAS[1].pdf
2. KMeans — scikit-learn 1.7.1 documentation, diakses Juli 21, 2025, https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
3. Getting started with K-means clustering in Python - Domino Data Lab, diakses Juli 21, 2025, https://domino.ai/blog/getting-started-with-k-means-clustering-in-python
4. K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks, diakses Juli 21, 2025, https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a/
5. Data pre-processing for k- means clustering - Amazon S3, diakses Juli 21, 2025, https://s3.amazonaws.com/assets.datacamp.com/production/course_10628/slides/chapter3.pdf
6. End-to-End Guide to K-Means Clustering in Python: From Preprocessing to Visualization, diakses Juli 21, 2025, https://www.skillcamper.com/blog/end-to-end-guide-to-k-means-clustering-in-python-from-preprocessing-to-visualization
7. Elbow Method: Definition, Drawbacks, vs. Silhouette Score | Built In, diakses Juli 21, 2025, https://builtin.com/data-science/elbow-method
8. builtin.com, diakses Juli 21, 2025, https://builtin.com/data-science/elbow-method#:~:text=The%20elbow%20method%20is%20a%20graphical%20method%20for%20finding%20the,the%20graph%20forms%20an%20elbow.
9. Clustering text documents using k-means - Scikit-learn, diakses Juli 21, 2025, https://scikit-learn.org/stable/auto_examples/text/plot_document_clustering.html