EVALUASI CLUSTERING *K-MEANS* DAN *K-MEDOID* PADA PERSEBARAN COVID-19 DI INDONESIA DENGAN METODE *DAVIES BOULDIN INDEX* (DBI)

THESIS

Oleh:

FATHURRAHMAN
NIM. 200605210023

PROGRAM STUDI MAGISTER INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGRI MAULANA MALIK IBRAHIM
MALANG
2023

EVALUASI CLUSTERING *KMEANS* DAN *KMEDOID* PADA PERSEBARAN COVID-19 DI INDONESIA DENGAN METODE *DAVIES BOULDIN INDEX* (DBI)

THESIS

Diajukan kepada:

Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Magister Komputer (M.Kom)

Oleh :

FATHURRAHMAN

NIM. 200605210023

PROGRAM STUDI MAGISTER
INFORMATIKA FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2023

EVALUASI CLUSTERING *K-MEANS* DAN *K-MEDOID* PADA PERSEBARAN COVID-19 DI INDONESIA DENGAN METODE *DAVIES BOULDIN INDEX* (DBI)

THESIS

Oleh:
FATHURRAHMAN
NIM. 200605210023

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal 26 Mei 2023

Pembimbing I,	Pembimbing II,
Dr. Sri Harini, M.Si	Dr. Ririen Kusumawati, M.Kom
NIP. 19731014 200112 2 002	NIP. 19720309 200501 2 002

Mengetahui,
Ketua Program Studi Magister Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 2009011 008

EVALUASI CLUSTERING *K-MEANS* DAN *K-MEDOID* PADA PERSEBARAN COVID-19 DI INDONESIA DENGAN METODE *DAVIES BOULDIN INDEX* (DBI)

THESIS

Oleh:
FATHURRAHMAN
NIM. 200605210023

Telah Dipertahankan di Depan Dewan Penguji Thesis dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan Untuk Memperoleh Gelar Magister Komputer (M.Kom.)
Tanggal 31 Mei 2023

Susunan Dewan Penguji			Tanda Tangan
Penguji Utama	:	Dr. Cahyo Crysdian NIP. 19740424 2009011 008	( )
Ketua Penguji	:	Dr. M. Amin Hariyadi, M.T NIP. 19670018 200501 1 001	( )
Sekretaris Penguji	:	Dr. Sri Harini, M.Si NIP. 19731014 200112 2 002	( )
Anggota Penguji	:	Dr. Ririen Kusumawati, M.Kom NIP. 19720309 200501 2 002	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Magister Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 2009011 008

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama	: Fathurrahman
NIM	: 200605210023
Program Studi	: Magister Informatika
Fakultas	: Sains dan Teknologi

Menyatakan dengan sebenarnya bahwa Thesis yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambilalihan data, tulisan atau pikiran orang lain yang saya akui sebagai tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka. Apabila dikemudian hari terbukti atau dapat dibuktikan Thesis ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 26 Mei 2023
Yang membuat pernyataan,

Fathurrahman
NIM. 200605210023

MOTTO

"Menuntut ilmu adalah takwa.
Menyampaikan ilmu adalah ibadah.
Mengulang-ulang ilmu adalah zikir.
Mencari ilmu adalah jihad."
– Abu Hamid Al Ghazali

HALAMAN PERSEMBAHAN

Dengan mengucapkan syukur Alhamdulillahi rabbil alamin, Thesis ini saya persembahkan untuk :

1. Seluruh Keluarga tercinta yang selalu memberikan doa dan semangat.
2. Seluruh Civitas Akademika Universitas Islam Negeri Maulana Malik Ibrahim Malang yang telah memberikan kesempatan untuk menambah ilmu teknologi dan agama.
3. Seluruh rekan-rekan mahasiswa Magister Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang semua angkatan atas kerjasamanya selama ini.
4. Bapak, ibu, saudara dan rekan-rekan sekalian yang tidak bisa saya sebutkan satu persatu dalam mendukung Thesis ini hingga bisa diselesaikan.

KATA PENGANTAR

Assalamu'alaikum Wr. Wb

Syukur Alhamdulillah penulis haturkan kehadirat Allah SWT yang telah melimpahkan Rahmat dan Hidayah-Nya, sehingga penulis dapat menyelesaikan studi di Program Studi Magister Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang sekaligus menyelesaikan Thesis ini dengan baik. Selanjutnya penulis haturkan ucapan terima kasih seiring do'a dan harapan *jazakumullah ahsanal jaza'* kepada semua pihak yang telah membantu terselesaikannya Thesis ini. Ucapan terima kasih ini penulis sampaikan kepada:

1. Dr. Sri Harini, M.Si. dan Dr. Ririen Kusumawati, M.Kom, selaku dosen pembimbing Thesis, yang telah banyak memberikan pengarahan dan pengalaman yang berharga.
2. Segenap civitas akademika Program Studi Magister Informatika, terutama seluruh Bapak / Ibu dosen, terima kasih atas segenap ilmu dan bimbingannya.
3. Keluarga tercinta yang senantiasa memberikan do'a dan semangat
4. Semua rekan-rekan seperjuangan yang ikut mendukung dan membantu.

Penulis menyadari bahwa dalam penyusunan Thesis ini masih terdapat kekurangan dan penulis berharap semoga Thesis ini bisa memberikan manfaat kepada para pembaca khususnya bagi penulis secara pribadi. Amiinn Yaa Rabbal Alamin.

Wasalamu'alaikum Wr. Wb

Malang, 26 Mei 2023

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Fathurrahman, Fathurrahman, 2023, Evaluasi Clustering K-Means dan K-Medoid pada Persebaran Covid-19 di Indonesia dengan Metode davies Bouldin index (DBI), Program Magister Informatika Universitas Islam Negeri Maulana Malik Ibrahim, Pembimbing: (1) Dr. Sri Harini, M.Si. (2) Dr. Ririen Kusumawati, M.Kom

Kata Kunci: Covid-19, Davies-Bouldin Index, K-Means Clustering, K-Medoids Clustering

Tingginya persebaran Covid-19 di Indonesia, dan persebaran di tiap-tiap daerah yang berbeda-beda, menjadikan perlu adanya pengelompokan daerah dengan masing-masing tingkat penyebarannya, untuk mengetahui kemiripan karakteristik atau kriteria dari setiap daerah dengan tingkat penyebaran Covid-19 yang akan terkumpul dalam suatu cluster tertentu. Penelitian ini menggunakan komparasi analisis cluster menggunakan K-Means dan K-Medoid untuk menganalisis perseberan virus Covid-19 di indonesia. Analisis komparasi kedua algoritma dibuktikan dengan adanya nilai davies bouldin index (DBI) sebagai parameter evaluasi menggunakan Bahasa Pemrograman Python Version 3 yang dijalankan pada tools Jupyter Notebook. Langkah penelitian dimulai dari import library atau modul yang digunakan dalam berbagai tahapan dalam penelitian ini. Tahapan yang dilakukan antara lain melakukan pre-processing berupa proses binning data hingga normalisasi data. Selanjutnya, menampilkan visualisasi data sebaran Covid-19. Kemudian, melakukan modeling Algoritma K-Means dan K-Medoids. Hingga diakhiri dengan langkah terakhir berupa evaluasi menggunakan Davies-Bouldin Index (DBI). Setelah dilakukan evaluasi DBI, K-Means mendapatkan nilai 0.9762331449809145, sedangkan K-Medoids mendapatkan nilai 0.9809235412405508. Karena K-Means memiliki nilai DBI yang lebih rendah dibandingkan K-medoids, maka dapat dikatakan K-Means menghasilkan klasterisasi yang lebih baik dalam klasterisasi data sebaran Covid-19 di Indonesia.

ABSTRACT

Fathurrahman, Fathurrahman, 2023, Clustering Evaluation of K-Means and K-Medoid on the Distribution of Covid-19 in Indonesia using Davies Bouldin index (DBI), Program Magister Informatika Universitas Islam Negeri Maulana Malik Ibrahim, Pembimbing: (1) Dr. Sri Harini, M.Si. (2) Dr. Ririen Kusumawati, M.Kom

Kata Kunci: Covid-19, Davies-Bouldin Index, K-Means Clustering, K-Medoids Clustering

The high spread of Covid-19 in Indonesia, and the distribution in each region which is different, makes it necessary to group regions with each level of spread, to find out the similarity of characteristics or criteria from each region with the level of spread of Covid-19 which will be collected in a certain cluster. This study uses a comparative cluster analysis using K-Means and K-Medoid to analyze the spread of the Covid-19 virus in Indonesia. Comparative analysis of the two algorithms is proven by the Davies Bouldin index (DBI) value as an evaluation parameter using the Python Version 3 Programming Language which is run on Jupyter Notebook tools. The research step starts from the import library or modules used in various stages of this research. The steps taken include pre-processing in the form of data binning to data normalization. Next, displays a visualization of data on the distribution of Covid-19. Then, modeling the K-Means and K-Medoids Algorithms. Until it ends with the last step in the form of an evaluation using the Davies-Bouldin Index (DBI). After the DBI evaluation, K-Means got a score of 0.976233144980914 5, while K-Medoids got a score of 0.9809235412405508. Because K-Means has a lower DBI value than K-medoids, it can be said that K-Means produces better clustering in clustering data on the spread of Covid-19 in Indonesia.

خلاصة

فتح الرحمن، فتح الرحمن، 2023، تقييم مجموعات ك ـ مينس وك ـ ميدويد حول الانتشار كوفيد-19 في إندونيسيا باستخدام طريقة مؤشر ديفيز بولدين(DBI) ، برنامج الماجستير في جامعة مولانا مالك إبراهيم الإسلامية الحكومية، المشرف: (1) د. . سري هاريني، ماجستير. (2) د. ريرين كوسوماواتي، م.كوم

الكلمة الرئيسية : كوفيد-19، مؤشر ديفيز بولدين، تجميع ك ـ مينس ، تجميع ك ـ ميدويد

لمعرفة أوجه التشابه في خصائص أو معايير كل منطقة مع مستوى انتشار كوفيد-19 والتي سيتم جمعها في مجموعة معينة . يستخدم هذا البحث التحليل العنقودي المقارن باستخدام ك ـ مينس وك ـ ميدويد لتحليل انتشار فيروس كوفيد 19-في إندونيسيا. تم إثبات التحليل المقارن للخوارزميتين من خلال قيمة مؤشر دافيد بوديس (DBI) كمعلمة تقييم باستخدام لغة البرمجة بيتون الإصدار 3 والتي يتم تشغيلها على أداة Jupyter Notebook.تبدأ خطوات البحث من استيراد المكتبات أو الوحدات المستخدمة في المراحل المختلفة لهذا البحث. تشمل المراحل التي يتم تنفيذها المعالجة المسبقة في شكل تجميع البيانات لتطبيع البيانات. بعد ذلك، يتم عرض تصور لبيانات توزيع كوفيد 19-ثم قم بتصميم خوارزميات ك ـ مينس وك ـ ميدويدس .حتى تنتهي بالخطوة الأخيرة وهي التقييم باستخدام مؤشر ديفيز بولدين .(DBI). بعد تقييمDBI ، حصلت ك ـ مينس على قيمة 0.9762331449809145، بينما حصلت وك ـ ميدويدس على قيمة 0.9809235412405508. نظرًا لأن ك ـ مينس لها قيمة DBI أقل من وك ـ ميدويدس ، يمكن القول أن ك ـ مينس تنتج تجميعًا أفضل في تجميع البيانات حول انتشار كوفيد 19-في إندونيسيا.

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Penganalisisan klaster adalah suatu metode yang digunakan untuk mengelompokkan objek atau pengamatan yang berbeda ke dalam kelompok (klaster) dengan tujuan mengidentifikasi kesamaan yang maksimal di antara objek dalam sebuah klaster dan kesamaan minimal antara klaster yang berbeda. Metode ini merupakan sebuah pendekatan statistik multivariat yang memungkinkan pengelompokan pengamatan berdasarkan beberapa variabel. Penganalisisan klaster juga merupakan salah satu teknik dalam pembelajaran mesin yang termasuk dalam kategori pembelajaran tanpa pengawasan. (Wang et al., 2020).

Secara keseluruhan, ada dua jenis metode yang berbeda dalam analisis klaster, yaitu metode hirarki dan metode non hirarki. Metode hirarki melibatkan pembentukan klaster secara bertahap atau bertingkat, mirip dengan struktur pohon. Metode ini cocok digunakan ketika jumlah klaster yang diinginkan tidak diketahui dan cocok untuk pengamatan yang tidak terlalu banyak. Di sisi lain, metode non hirarki melibatkan penentuan klaster yang ingin dibentuk sebelumnya atau secara manual. Dalam metode ini, objek-objek akan dikelompokkan ke dalam k klaster yang telah ditentukan. (Suyanto, 2017).

Dua diantara metode *non-hierarchical clustering* yang populer dan banyak diterapkan adalah metode K-Means dan K-Medoid (Rodriguez et al., 2019). Mengacu pada penelitiannya, Rodriguez menyatakan bahwa K-Means dan K-Medoid populer digunakan karena keduanya memiliki kemiripan konsep.

Diantaranya adalah kemiripan dalam alur pembacaan data, penentuan k kelompok serta kesamaan metode jarak yang digunakan, yaitu (*euclidean distance*). Selain itu, K-Means dan K-Medoid memiliki kelebihan masing-masing. Mengacu pada penelitian Deswiaqsa, dkk (Deswiaqsa et al., 2022), K-Means adalah suatu algoritma pengelompokan yang melakukan partisi data ke dalam sejumlah k-*cluster* yang sudah ditentukan di awal. K-Means juga dapat digunakan untuk mempartisi data ke dalam bentuk beberapa *cluster* atau kelompok data yang terbentuk memiliki karakteristik yang sama (Putra & Kadyanan, 2021). Sedangkan, K-Medoid atau sering juga disebut sebagai PAM (*Partioning Around Medoid*) adalah sebuah algoritma clustering yang memiliki kemiripan dengan K-Means. Salah satu perbedaan utama antara kedua algoritma ini terletak pada penggunaan representasi objek (medoid) sebagai pusat klaster untuk setiap klaster dalam K-Medoid, sedangkan K-Means menggunakan nilai rata-rata (mean) sebagai pusat klaster (Andini & Arifin, 2020). Oleh karena itu, mengacu pada *literature review* soal kesamaan dan perbedaan konsep antara K-Means dan K-Medoid, maka penelitian untuk mengkomparasikan kinerja K-Means dan K-Medoid menjadi *research challage* yang menarik dan menantang.

Terdapat beragam implementasi aplikatif dari K-Means dan K-Medoid dalam analisis *cluster,* salah satunya adalah untuk mengelompokan persebaran penyakit Covid-19 sebagaimana penelitian Virgantari, dkk (Virgantari & Faridhan, 2020) dan Sindi, dkk (Sindi et al., 2020). Covid-19 merupakan suatu penyakit yang dapat menyebar dari orang ke orang dan disebabkan oleh suatu varian virus corona yang baru. Beberapa individu yang terinfeksi dapat mengalami gejala pernapasan yang

ringan hingga parah. Penyakit ini telah menjadi suatu ancaman bagi kesehatan masyarakat secara global. Pasalnya, penyebaran penyakit ini telah mencapai hingga ke 199 negara di seluruh dunia sepanjang tahun 2020, sehingga menjadikan status pandemi secara global. Berdasarkan laporan dari Our World in Data, pada tanggal 17 Maret 2022, tingkat kematian (case fatality rate/CFR) akibat Covid-19 di Indonesia mencapai 2,58%. Persentase ini menempatkan Indonesia sebagai negara dengan tingkat kematian tertinggi kedua di kawasan Asia Tenggara (Annur, 2022).

Senada dengan isyarat dalam al-Qur'an, terdapat ayat yang membahas suatu musibah datangnya atas kodrat dan kehendak Allah. Termasuk dalam hal ini adalah musibah berupa wabah pandemi covid-19 yang sedang melanda. Pada QS At-Taubah ayat 126, Allah SWT berfirman:

مَآ أَصَابَ مِن مُّصِيبَةٍ إِلَّا بِإِذْنِ ٱللَّهِ ۗ وَمَن يُؤْمِنۢ بِٱللَّهِ يَهْدِ قَلْبَهُۥ ۚ وَٱللَّهُ
بِكُلِّ شَىْءٍ عَلِيمٌ ﴿١١﴾

Artinya:

"Tidak ada suatu musibah pun yang menimpa seseorang kecuali dengan izin Allah dan barangsiapa yang beriman kepada Allah niscaya Dia akan memberi petunjuk kepada hatinya. Dan Allah Maha Mengetahui segala sesuatu". (At Taghabun:11)

Ibnu Katsir, dalam kutipan yang mengacu pada perkataan Ibnu Abbas tentang ayat tersebut, menjelaskan bahwa segala musibah datang atas takdir dan kehendak Allah. Bagi mereka yang beriman kepada Allah ketika mereka mengalami musibah, dan mereka menyadari bahwa hal tersebut merupakan ketentuan Allah, lalu mereka bersabar dan tunduk pada kehendak-Nya, Allah akan memberikan petunjuk kepada

mereka. Allah juga akan menggantikan apa yang telah hilang dengan sesuatu yang serupa atau bahkan yang lebih baik.

Tingginya persebaran Covid-19 di Indonesia, dan persebaran di tiap-tiap daerah yang berbeda-beda, menjadikan perlu adanya pengelompokan daerah dengan masing-masing tingkat penyebarannya, untuk mengetahui kemiripan karakteristik atau kriteria dari setiap daerah dengan tingakat penyebaran Covid-19 yang akan terkumpul dalam suatu *cluster* tertentu. Berdasarkan uraian-uraian yang telah dijabarkan, maka peneliti akan menggunakan komparasi analisis *cluster* menggunakan K-Means dan K-Medoid untuk menganalisis perseberan virus Covid-19 di indonesia. Analisis komparasi kedua algoritma dibuktikan dengan adanya nilai *Davies-Bouldin Index* (DBI) sebagai parameter evaluasi. Seperti yang diketahui, analisis klaster memiliki kelemahan ketika titik klaster dipilih secara acak, sehingga hasil pengelompokan data dapat bervariasi. Jika nilai tersebut rendah, pengelompokan yang dihasilkan akan kurang optimal. Oleh karena itu, dalam penelitian ini, peneliti melakukan evaluasi terhadap kedua algoritma menggunakan indeks Davies Bouldin (DBI). Dengan menggunakan DBI, suatu klaster dianggap optimal jika nilai DBI minimal dan mendekati nol, yang menunjukkan tingkat optimalitas klaster tersebut.

1.2 Pernyataan Masalah

Berdasarkan latar belakang tersebut, permasalahan penelitian ini adalah bagaimana mengestimasi kelompok K-Means dan K-Medoid penyebaran Covid-19 di Indonesia dengan menggunakan metode Davies-Bouldin Index (DBI).

1.3 Tujuan Penelitian

Dari pernyataan masalah, maka tujuan dari penelitian adalah :

1. Mendapatkan dan menganalisis model *cluster* persebaran Covid-19 di Indonesia dengan menggunakan *clustering* K-Means dan K-Medoid.
2. Mendapatkan evaluasi metode K-Means dan K-Medoid dalam menganalisis persebaran virus Covid-19 di Indonesia dengan menggunakan DBI.

1.4 Manfaat Penelitian

Kegunaan penelitian yang diharapkan adalah sebagai referensi dan kontribusi ilmiah untuk penelitian selanjutnya yang menganalisis kelompok Covid-19 di Indonesia.

1.5 Batasan Masalah

1. Data yang digunakan pada penelitian ini merupakan data persebaran Covid-19 di Indonesia pada tahun 2020-2022 yang diambil dari basis data Kaggle.
2. Bahasa pemrograman Python digunakan untuk olah data, implementasi metode K-Means dan K-Medoid serta visualisasi grafis.

1.6 Sistematika Penulisan

Sistematika penelitian ini diantaranya adalah :

BAB I PENDAHULUAN

Bab ini merupakan bab pertama dari penelitian yang meliputi latar belakang, topik, tujuan penelitian dan manfaat penelitian, dengan fokus bab ini dimaksudkan sebagai dasar dilakukannya penelitian.

BAB II KAJIAN PUSTAKA

Bab II merupakan bab yang membahas landasan teori yang peneliti gunakan sebagai acuan dalam penelitian.

BAB III METODE PENELITIAN

Bab ini meliputi jenis penelitian, variabel penelitian, jenis data, metode penelitian, dan metode penelitian.

BAB IV HASIL DAN PEMBAHASAN

Bagian ini membahas hasil penelitian dan menjelaskan jawabannya di bagian pendahuluan

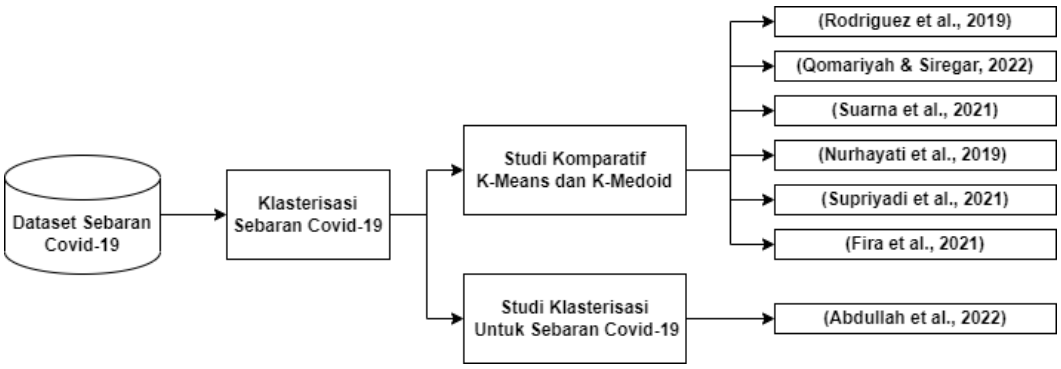
BAB V PENUTUP

Bagian terakhir berfokus pada bagaimana kesimpulan yang ditarik berdasarkan hasil model yang diperoleh sebelumnya dapat menanggapi masalah penelitian dan proposal penelitian yang dibuat sehingga dapat dikembangkan lebih luas lagi.

BAB II
STUDI PUSTAKA

2.1 Analisis *Cluster* Menggunakan K-Means dan K-Medoids

Analisis *cluster* yang digunakan dalam penelitian ini adalah metode K-Means dan K-Medoid. Subbab ini menyajikan penelitian terkait sebagai *state of the art* penelitian. Pembahasan *state of the art* difokuskan pada algoritma K-Means dan K-Medoid yang secara umum yang telah dilakukan untuk menyelesaikan permasalahan *clustering*. Selain itu, penelitian terkait pada subbab ini juga membahas mengenai klasterisasi K-Means dan K-Medoid yang diimplementasikan secara khusus untuk Covid-19. *State of the art* dalam penelitian ini dapat dinyatakan dalam kerangka teori pada Gambar 2.1.



Gambar 2.1 Kerangka Teori

Pertama, peneliti mengkaji penelitian komparatif antar algoritma klastering yang dilakukan oleh Rodriguez (Rodriguez et al., 2019). Dua diantara algoritma klasterisasi yang dibandingkan adalah K-Means dan K-Medoid. Mengacu pada penelitiannya, Rodriguez menyatakan bahwa K-Means dan K-Medoid populer

digunakan karena keduanya memiliki kemiripan konsep. Diantaranya adalah kemiripan dalam alur pembacaan data, penentuan k kelompok serta kesamaan metode jarak yang digunakan, yaitu (*euclidean distance*). Sementara itu, terdapat perbedaan mendasar dari kedua algoritma ini. K-Medoid menggunakan objek sebagai perwakilan (*medoid*) sebagai pusat *cluster* untuk setiap *cluster*, sedangkan K-Means menggunakan nilai rata-rata (*mean*) sebaga pusat *cluster*. Oleh karena itu, penelitian untuk mengkomparasikan kinerja K-Means dan K-Medoid menjadi *research challage* yang menarik dan menantang. Hal ini untuk menguji secara komparatif perihal performa K-Means dan K-Medoid untuk objek klasterisasi tertentu.

Berikutnya, peneliti mengkaji penelitian terdahulu yang secara komparatif telah membandingkan K-Means dengan K-Medoid untuk klasterisasi objek yang berbeda. Hasilnya, terbukti bahwa K-Medoid belum tentu lebih unggal daripada K-Means, pun juga sebaliknya. Penelitian komparatif tentang K-Means dan K-Medoid oleh Qomariyah (Qomariyah & Siregar, 2022) dan Suarna (Suarna et al., 2021) menyatakan bahwa K-Means lebih baik daripada K-Medoid berdasarkan evaluasi DBI. Namun, penelitian Nurhayati (Nurhayati et al., 2019) menyimpulkan bahwa K-Medoid lebih unggal dari K-Means berdasarkan nilai akurasi, waktu eksekusi (*execution time*) dan kompleksitas waktu (*time complexity*). Lebih rinci, Qomariyah (Qomariyah & Siregar, 2022) menggunakan perbandingan K-Means dan K-Medoid untuk klasterisasi mahasiswa. Kemudian berdasarkan evaluasi DBI, K-Means menghasilkan nilai 0,781, sedangkan K-Medoid mendapatkan nilai 0,929. Pada evaluasi DBI, nilai yang mendekati 0 adalah nilai yang terbaik. Penelitian

berikutnya yang dilakukan oleh Suarna (Suarna et al., 2021) juga membuktikan bahwa K-Means lebih unggal dibandingkan K-Medoid untuk klasterisasi *fish cooking menu*. Disimpulkan bahwa, K-Means menghasilkan nilai DBI sebesar -1,535, sedangkan K-Medoid menghasilkan nilai -1,777. Keunggulan K-Medoid atas K-Means dibuktikan oleh Nurhayati (Nurhayati et al., 2019) melalui penelitian menggunakan teknologi *big data.* Keunggulan K-Medoid dinyatakan berdasarkan akurasi, waktu eksekusi (*execution time*) dan kompleksitas waktu (*time complexity*). K-Medoid memperoleh akurasi lebih baik dengan nilai 63,24%, sedangkan 52,11% untuk akurasi K-Means. Waktu eksekusi (*execution time*) dari K-Medoid juga menunjukkan performa yang lebih baik. Tercatat bahwa rata-rata kecepatan eksekusi untuk K-Medoid adalah 3,1 ms, sedangkan K-Means sebesar 3,45 ms. Mengacu ke kompleksitas waktu (*time complexity*), penelitian tersebut menggunakan pembuktian melalui Big O (n2). Disimpulkan bahwa rata-rata nilai yang dihasilkan oleh K-Medoid adalah 310,157 sedangkan K-Means sebesar 377,886. Ketiga penelitan tersebut membuktikan bahwa studi komparatif antara K-Means dan K-Medoid tetap relevan untuk dilakukan mengingat objek penelitian saat ini berbeda dengan penelitian terdahulu, yakti sebaran Covid-19 di Indonesia.

Penelitian klasterisasi Covid-19 pernah dilakukan oleh Abdullah (Abdullah et al., 2022), yaknik melakukan klasterisasi provinsi di Indonesia yang beresiko terpapar Covid-19 berdasarkan data yang terkonfirmasi positif Covid-19, kematian, serta pemulihan. Penelitian tersebut menggunakan algoritma K-Means yang membentuk 3 klaster provinsi.

Kemudian, penelitian yang dilakukan oleh Silvi (Silvi, 2018) membandingkan hasil analisis klaster menggunakan metode titik sentral dan k-means dengan outlier dalam klasterisasi indikator HIV/AIDS di Indonesia. Studi ini menggunakan statistik kesenjangan untuk menentukan jumlah klaster ideal yang mengelompokkan kabupaten menjadi 7 klaster. Hasil penelitian ini menyimpulkan bahwa untuk data dengan outlier, menggunakan centroid link memberikan hasil yang lebih masuk akal dibandingkan dengan metode k. Jumlah kuadrat dalam/jumlah kuadrat antara metode-k adalah 0,12232, sedangkan jumlah kuadrat dalam/jumlah kuadrat dari metode tautan adalah 0,067307. Metode centroid linkage membuat grup lebih homogen, sehingga nilai rasio yang dihasilkan lebih rendah. Hal ini berarti metode centroid linking memiliki kualitas akurasi kelompok yang lebih baik dibandingkan dengan K-Means. Yang kedua adalah studi oleh Supriyadi (Supriyadi et al., 2021) yang meneliti perbandingan algoritma K-Means dan K-Medoid untuk mengelompokkan armada truk berdasarkan produktivitas. Dalam penelitian ini, kedua metode clustering digunakan untuk mengklasifikasikan setiap rangkaian kendaraan berdasarkan produktivitas kinerjanya. Penelitian ini membahas tentang perbandingan K-Medizo dan K-Medoid, dilanjutkan dengan uji validasi terhadap hasil cluster yang terbentuk. Indeks Davies-Bouldin yang digunakan sebagai metode evaluasi dalam analisis cluster memberikan nilai validitas 0,67 untuk K-Medium dan 1,78 untuk K-Medoid.

Penelitian ketiga adalah sebagaimana yang dilakukan oleh Fira (Fira et al., 2021). Penelitian tersebut mengelompokan penyebaran Coivd-19 di Indonesia. Tujuan dari penelitian ini adalah untuk mengelompokkan provinsi-provinsi di

Indonesia berdasarkan tingkat tinggi dan rendahnya penyakit Covid-19. Penelitian ini juga membandingkan penggunaan dua metode algoritma, yaitu K-Means dan K-Medoid. Hasil penelitian menunjukkan bahwa dengan menggunakan metode K-Means, terdapat cluster 1 yang terdiri dari 2 wilayah dan dikategorikan sebagai tinggi, sedangkan cluster 2 terdiri dari 32 wilayah dan dikategorikan sebagai rendah. Sementara itu, dengan menggunakan algoritma K-Medoid, terdapat cluster 1 yang terdiri dari 4 wilayah dan dikategorikan sebagai tinggi, sedangkan cluster 2 terdiri dari 30 wilayah dan dikategorikan sebagai rendah. Dalam perbandingan ini, diperoleh nilai koefisien silhouette dengan metode K-Means sebesar 0,207, sedangkan nilai koefisien silhouette dengan metode K-Medoid sebesar 0,347. Terakhir, adalah penelitian dari Adha (Adha et al., 2021) yang bertujuan untuk mengelompokkan negara-negara berdasarkan pola kasus Covid-19 di seluruh dunia. Hasil pengelompokan ini dapat digunakan sebagai referensi dan memberikan gambaran mengenai negara-negara dengan tingkat pemulihan yang rendah dapat mempelajari proses pemulihan negara-negara dengan tingkat pemulihan yang tinggi yang berada dalam kelompok yang sama. Hasil penelitian ini menunjukkan bahwa metode K-Means lebih baik daripada metode DBSCAN dalam mengelompokkan kasus Covid-19. Algoritma K-Means memiliki nilai indeks siluet (SI) terbaik sebesar 0.6902, yang tercapai pada eksperimen dengan nilai k=8.

2.2 Clustering

Clustering adalah salah satu metode paling terkenal dalam penambangan data. Dalam konteks data mining, clustering mengacu pada proses pengelompokan beberapa data atau objek ke dalam kelompok (cluster) sedemikian rupa sehingga

setiap cluster berisi data yang mirip satu sama lain dan berbeda dari objek di cluster lain. Hingga saat ini peneliti masih terus berusaha untuk menyempurnakan model clustering dan menghitung jumlah cluster yang optimal untuk menghasilkan cluster terbaik. Clustering adalah tentang menempatkan objek yang mirip (dekat) ke dalam sebuah cluster dan menjaga jarak antar cluster seluas mungkin.

Suyanto (Suyanto, 2017) menyatakan bahwa clustering adalah proses pengelompokan objek data ke dalam kelompok yang saling mirip, dimana setiap kelompok terdiri dari objek data yang mirip dan berbeda dengan objek data kelompok lain. Clustering adalah tentang menempatkan objek serupa (berjarak dekat) ke dalam kelompok (cluster) sambil menjaga jarak antar kelompok sekecil mungkin. Ada beberapa metode hoarding yang dikembangkan oleh para ahli dan masing-masing metode memiliki karakteristik, kelebihan dan kekurangannya masing-masing. Clustering dapat dibedakan berdasarkan struktur cluster, keanggotaan data cluster, dan kekompakan data cluster. Metode clustering dapat dibagi menjadi dua kelompok berdasarkan strukturnya, yaitu clustering hirarkis dan clustering partisi. Pengelompokan hierarkis mengikuti aturan bahwa kumpulan data dapat dilihat sebagai grup, dan dua atau grup yang lebih kecil dapat digabungkan menjadi satu grup besar, dan seterusnya, hingga semua data dikelompokkan. Metode pengelompokan hierarki adalah satu-satunya metode yang terdapat dalam kelas pengelompokan hierarki. Di sisi lain, metode clustering membagi data menjadi beberapa kelompok yang tidak tumpang tindih, artinya semua data hanya dimiliki oleh satu kelompok. Clustering mengelompokkan data bersama tanpa mempertimbangkan kategori data yang telah ditentukan sebelumnya. Proses ini

berbeda dengan proses klasifikasi, dimana kelas data harus didefinisikan pada awal proses. Oleh karena itu, clustering sering disebut sebagai unclustered data. Cluster memiliki ciri-ciri sebagai berikut:
a) Homogenitas yang tinggi antar anggota suatu cluster (dalam cluster). b) Heterogenitas tinggi antar klaster (cluster).

2.3 Algoritma *Clustering* (*Clustering* Algorithm)

Clustering adalah salah satu metode dalam Data Mining yang termasuk dalam kategori pembelajaran tanpa pengawasan (unsupervised learning). K-Means adalah salah satu metode non-hirarki dalam clustering data yang bertujuan untuk membagi data ke dalam satu atau lebih klaster. Metode ini melakukan partisi data ke dalam klaster sehingga data dengan karakteristik yang serupa dikelompokkan ke dalam klaster yang sama, sementara data dengan karakteristik yang berbeda dikelompokkan ke klaster lainnya. Tujuan dari data clustering ini adalah untuk meminimalkan fungsi objektif yang ditentukan dalam proses clustering, yang pada umumnya bertujuan untuk mengurangi variasi di dalam sebuah klaster dan memaksimalkan variasi antara klaster. *cluster* (Dewi & Pramita, 2019).

Pada dasarnya clustering adalah suatu metode pencarian dan pengelompokan data yang memiliki kesamaan karakteristik (similarity) antar data. Clustering adalah salah satu metode penambangan data tanpa pengawasan, yang berarti bahwa metode ini tidak memerlukan pelatihan dan tidak memerlukan tujuan keluaran yang telah ditentukan sebelumnya. Dalam data mining, digunakan dua jenis metode pengelompokan untuk mengelompokkan data, yaitu pengelompokan hierarkis (hierarchical clustering) dan pengelompokan non-hierarkis (non-hierarchical clustering) (Suyanto, 2017).

2.4 K-Means Clustering

K-Means Clustering adalah suatu algoritma pengelompokan iteratif yang melakukan partisi data ke dalam sejumlah cluster yang telah ditetapkan sebelumnya. Algoritma K-Means clustering memiliki sifat yang sederhana untuk diimplementasikan dan dieksekusi, serta cenderung cepat dan adaptif. Algoritma ini umum digunakan dalam praktek, dan secara historis, K-Means clustering menjadi salah satu algoritma yang paling penting dalam bidang data mining (Paembonan & Abduh, 2021).

K-Means Clustering merupakan sebuah metode pengelompokan berdasarkan jarak yang membagi data ke dalam sejumlah cluster. Algoritma ini hanya bekerja dengan atribut numerik. K-Means clustering termasuk dalam kategori partitioning clustering, yang memisahkan data ke dalam daerah-daerah terpisah. Algoritma K-Means terkenal karena kemudahan implementasinya serta kemampuannya dalam mengelompokkan data yang besar dan mengatasi data outlier dengan cepat. Dalam algoritma K-Means, setiap data akan termasuk dalam satu cluster tertentu pada tahap awal, namun pada tahap-tahap berikutnya, data tersebut dapat pindah ke cluster lain jika terdapat perubahan yang lebih sesuai (Abdullah et al., 2022).

Pada tahap awal K-Means clustering, beberapa komponen dari populasi data dipilih secara acak sebagai pusat cluster awal. Setelah itu, algoritma K-Means menguji setiap komponen data dalam populasi dan menentukan ke pusat cluster mana komponen tersebut akan dikelompokkan, berdasarkan jarak minimum antara komponen tersebut dengan setiap pusat cluster. Posisi pusat cluster kemudian

dihitung ulang berdasarkan komponen-komponen data yang telah dikelompokkan, dan proses ini berulang sampai semua komponen data tergolong dalam cluster-cluster yang sesuai. Pada akhirnya, akan terbentuk cluster-cluster baru yang telah didefinisikan.

2.5 Langkah *Clustering* K-Means

Berdasarkan Rahayu (Rahayu et al., 2019), proses *clustering* dengan menggunakan algoritma K-Means *clustering* memiliki tahapan antara lain:

a. Tahap Inisialisasi: Menentukan jumlah cluster yang diinginkan (K) dan tingkat perbedaan (jarak) yang diinginkan. Jika diperlukan, menetapkan ambang batas perubahan fungsi objektif dan ambang batas perubahan posisi centroid.
b. Memilih K titik data acak dari set data X sebagai centroid awal.
c. Mengelompokkan semua data ke dalam centroid terdekat berdasarkan metrik jarak yang telah ditentukan (memperbarui ID setiap data).
d. Menghitung kembali posisi centroid (C) berdasarkan data yang termasuk dalam setiap cluster.
e. Mengulangi langkah c dan d sampai kondisi konvergensi tercapai, yaitu: (a) perubahan fungsi objektif sudah di bawah ambang batas yang diinginkan, atau (b) tidak ada data yang berpindah ke cluster lain, atau (c) perubahan posisi centroid sudah di bawah ambang batas yang ditetapkan.

2.6 K-Medoid Clustering

K-Medoids clustering, juga dikenal sebagai *Partitioning Around Medoids* (PAM), merupakan variasi dari metode K-Means. Metode ini berbeda dalam penggunaan medoids sebagai representasi pusat cluster daripada menggunakan

mean dari setiap cluster. Tujuan penggunaan medoids adalah untuk mengurangi sensitivitas partisi terhadap nilai ekstrem yang terdapat dalam dataset. (Andini & Arifin, 2020).

Menurut Kauffman dan Rousseeuw dalam Nadliyah (Nahdliyah et al., 2019), algoritma yang digunakan dalam clustering K-Medoids didasarkan pada pencarian objek yang mewakili k (sejumlah cluster) objek dalam data. Objek yang mewakili suatu cluster atau k disebut medoid. Setelah menemukan sekumpulan objek untuk bertindak sebagai medoid, langkah selanjutnya adalah membuat cluster dengan menugaskan setiap objek data (non-medoid) ke medoid terdekat.

Algoritma K-Medoids bertujuan untuk mengatasi kelemahan clustering K-Medoids yang sensitif terhadap outlier. Dalam K-Means, elemen dengan nilai besar dapat menyimpang secara signifikan dari distribusi data, sehingga memengaruhi hasil pengelompokan secara keseluruhan. Namun, dengan algoritma K-Medoids, pusat klaster asli masih dipilih secara acak, seperti halnya dengan K-means. Proses iteratif selanjutnya dalam algoritma K-Medoids adalah secara terus-menerus mengganti objek yang berada di tengah cluster dengan objek lain. Tujuan dari iterasi ini adalah untuk terus meningkatkan kualitas cluster yang dihasilkan dengan mengukur kualitas tersebut menggunakan fungsi rata-rata perbedaan (jarak) antar objek di pusat cluster. Namun, perlu diingat bahwa karena objek pusat cluster terus berubah, stabilitas hasil cluster mungkin tidak tercapai dan kualitas clustering mungkin menjadi tidak stabil. Kualitas ini dievaluasi menggunakan fungsi perbedaan rata-rata (jarak) antara objek dan pusat cluster (Sindi et al., 2020).

Menurut Septiani (Septiani et al., 2022), Berikut adalah tahapan K-Medoids clustering:

a. Memilih secara acak K objek dari total N objek sebagai objek representatif (medoids).
b. Menghitung jarak antara setiap objek dengan masing-masing medoids.
c. Menetapkan setiap objek ke cluster yang sesuai dengan medoids terdekat, dan menghitung fungsi objektif yang merupakan jumlah ketidakmiripan (dissimilarity) antara setiap objek dengan medoids terdekat (yang memiliki jarak minimum).
d. Memilih secara acak objek yang bukan medoids (non-medoids).
e. Menghitung jarak antara setiap objek dengan masing-masing non-medoids.
f. Menetapkan setiap objek ke cluster yang memiliki jumlah ketidakmiripan minimum antara objek tersebut dengan non-medoids terdekat.
g. Menghitung selisih (perbedaan) fungsi objektif sebelum dan sesudah perubahan.
h. Jika pertukaran objek representatif (medoids) dengan non-medoids mengurangi fungsi objektif, maka melakukan penggantian tersebut.
i. Mengulangi langkah d-h sampai tidak ada perubahan pada objek representatif.
j. Analisis dianggap selesai jika tidak ada perubahan pada objek representatif (medoids).

Tahapan ini menjelaskan bagaimana K-Medoids clustering dilakukan untuk membangun cluster dengan objek representatif yang lebih stabil dan tahan terhadap outlier.

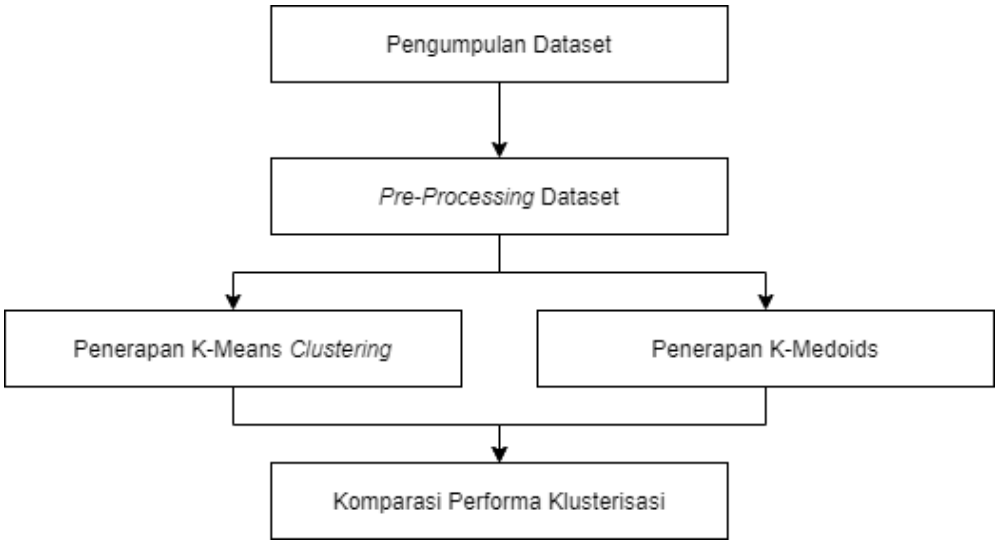
2.7 Davies Bouldin Index

Davies Bouldin Index adalah salah satu metode evaluasi *clustering* yang digunakan untuk mengaevaluasi dan mengukur seberapa baik *clustering* yang telah dilakukan melalui metric kohesi dan metric separasi, Evaluasi *Clustering* terbagi menjadi dua yaitu, Internal Evaluation dan External evaluation, untuk internal mengevaluasi menggunakan data yang ada, contoh metode nya antara lain, *davies bouldin index, dunn index,* sedangkan exterrnal evaluation menggunakan informasi terdahulu mengenai data contoh metode nya antara lain *Rand Measure, f-measure,jaccard index,* pada penelitian ini peneliti akan menggunakan Davies-Bouldin Index (DBI), DBI merupakan salah satu metode validasi *cluster* untuk evaluasi kuantitatif dari hasil *clustering*. Metode ini ditemukan pertama kali oleh David L. Davies dan Donald. Pada tahun 1979, Pengukuran ini bertujuan memaksimalkan jarak inter*cluster* antara satu *cluster* dengan *cluster* yang lain. Dalam penelitian ini DBI akan digunakan untuk mendeteksi outlier pada masing masing *cluster* yang terbentuk (Singh et al., 2020).

BAB III
METODE PENELITIAN

3.1 Prosedur Penelitian

Prosedur penelitian untuk membandingkan kinerja Algoritma K-Means *Clustering* dan K-Medoids ditunjukkan pada Gambar 3.1. Penelitian ini dimulai dari pengumpulan dataset. Kemudian, dilakukan *pre-processing* dataset untuk memastikan bahwa dataset siap digunakan sebagai masukan kedua algoritma. Setelahnya, dilakukan perhitungan Algoritma K-Means *Clustering* dan K-Medoids. Perbandingan kinerja kedua algoritma lantas diukur menggunakan metode evaluasi internal, yakni Davies Bouldin Index (DBI). Tiap langkah dalam prosedur penelitian ini akan dijelaskan secara rinci pada subbab berikutnya.



Gambar 3.1 Prosedur penelitian

3.2 Pengumpulan Dataset

Prosedur penelitian diawali dengan tahapan pengumpulan dataset. Dataset yang digunakan adalah dataset yang diunduh melalui laman https://www.kaggle.com/datasets/hendratno/covid19-indonesia. Dataset ini menunjukkan sebaran Covid-19 di Indonesia yang disajikan secara *time series*, terhitung mulai tanggal 1 Maret 2020 hingga 17 September 2022. Jumlah keseluruhan atribut dalam dataset adalah 38 dan jumlah *instances* adalah 21760.

3.3 Pre-Processing Dataset

Database preprocessing adalah proses yang mengubah data mentah menjadi format yang digunakan sebagai masukan untuk suatu algoritma. Proses ini penting karena data mentah seringkali tidak memiliki bentuk yang teratur. Selain itu, algoritma clustering K-Means dan K-Medoids juga tidak mampu mengolah data mentah. Oleh karena itu, proses ini sangat penting untuk memudahkan proses selanjutnya yaitu menganalisis data menggunakan algoritma. Dalam penelitian ini, pre-processing dilakukan dengan cara mereduksi atribut yang tidak berkaitan langsung dengan perhitungan algoritma, serta mereduksi *instances* yang memiliki *missing value*. Untuk melakukan deteksi missing value, digunakan perkakas bantu data mining, yakni Waikato Environtmen for Knowledge Analysis yang dikenal dengan WEKA. Tabel 3.1 menunjukkan atribut-atribut yang dilakukan pre-processing berupa reduksi atribut dan reduksi *instances* tertentu yang memiliki *missing value*. Mengacu ke Tabel 3.1, atribut provinsi menjadi output dari klasterisasi. Hal ini dilakukan atas dasar bahwa Pemerintah Indonesia, melalui situs resminya, covid19.go.id menginformasikan sebaran Covid-19 berdasarkan provinsi di Indonesia. Selain itu, penelitian ini juga dikuatkan leh penelian terdahulu yang

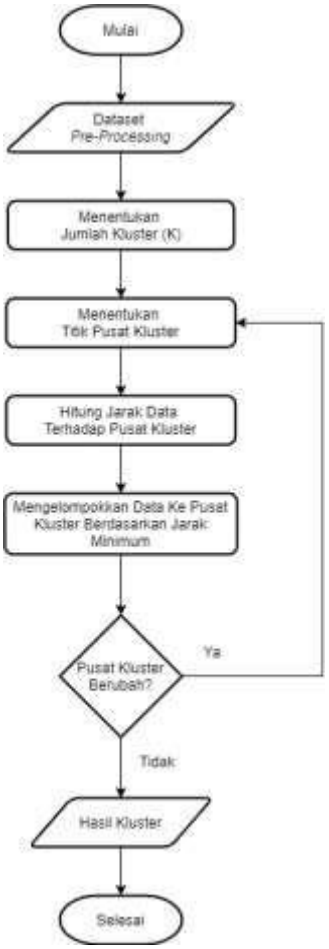
melakukan klsterisasi berdasarkan provinsi, sebagaimana penelitian Abdullah (Abdullah et al., 2022).

Tabel 3.1 *Pre-processing* pada atribut tertentu

No.	Atribut	Tipe Atribut	Pre-Processing
1.	Date	Date	Reduksi Atribut
2.	Location ISO Code	Kategorikal	Reduksi Atribut
3.	Location	Kategorikal	Reduksi Atribut
4.	New Cases	Numerik	
5.	New Deaths	Numerik	
6.	New Recovered	Numerik	
7.	New Active Cases	Numerik	
8.	Total Cases	Numerik	
9.	Total Deaths	Numerik	
10.	Total Recovered	Numerik	
11.	Total Active Cases	Numerik	
12.	Location Level	Kategorikal	Reduksi Atribut
13.	City or Regency	Kategorikal	Reduksi Atribut
14.	Province	Kategorikal	Reduksi 642 *instances* yang *missing value*
15.	Country	Kategorikal	Reduksi Atribut
16.	Continent	Kategorikal	Reduksi Atribut
17.	Island	Kategorikal	Reduksi 642 missing value
18.	Time Zone	Time	Reduksi Atribut
19.	Special Status	Kategorikal	Reduksi Atribut
20.	Total Regencies	Numerik	
21.	Total Cities	Numerik	Reduksi 614 *instances* yang *missing value*
22.	Total Districts	Numerik	
23.	Total Urban Villages	Numerik	Reduksi 617 *instances* yang *missing value*
24.	Total Rural Villages	Numerik	Reduksi 642 *instances* yang *missing value*
25.	Area	Numerik	
26.	Population	Numerik	
27.	Population Density	Numerik	
28.	Longitude	Numerik	
29.	Latitude	Numerik	
30.	New Cases per Million	Numerik	
31.	Total Cases per Million	Numerik	
32.	New Deaths per Million	Numerik	
33.	Total Deaths per Million	Numerik	

34.	Total Deaths per 100 rb	Numerik	
35.	Case Fatality Rate	Numerik	
36.	Case Recovered Rate	Numerik	
37.	Growth Factor of New Cases	Numerik	Reduksi 1187 *instances* yang *missing value*
38.	Growth Factor of New Deaths	Numerik	Reduksi 2467 *instances* yang *missing value*

3.4 Perhitungan Algoritma K-Means



Gambar 3.2 Diagram alir Algoritma K-Means

Gambar 3.2 menunjukkan diagram alir yang menggambarkan urutan langkah dari Algoritma K-Means. Merujuk penelitian Abdullah (Abdullah et al., 2022), langkah-langkah Algoritma K-Means dalam penelitian ini antara lain:

a. Mendapatkan nilai (k) sebagai jumlah kluster yang dibentuk. Nilai k optimal mengacu pada Metode Elbow.
b. Menentukan centroid sebagai titik pusat kluster awal secara acak pada dataset sebaran Covid-19.
c. Menghitung jarak antara pusat kluster dan seluruh *instances* pada dataset sebaran Covid-19 menggunakan rumus *Euclidience distance* sesuai Persamaan (1).

$$d(x,y) = \sqrt{\sum^{n}_{i=1}(xi - yi)^2)} \qquad (1)$$

Keterangan :

Xi = pusat titik centroid

Yi = data *instances*

i = atribut data Covid-19

d. Mengalokasikan masing-masing objek dalam centroid terdekat
e. Melakukan iterasi, serta mendapatkan centroid baru menggunakan Persamaan (2).

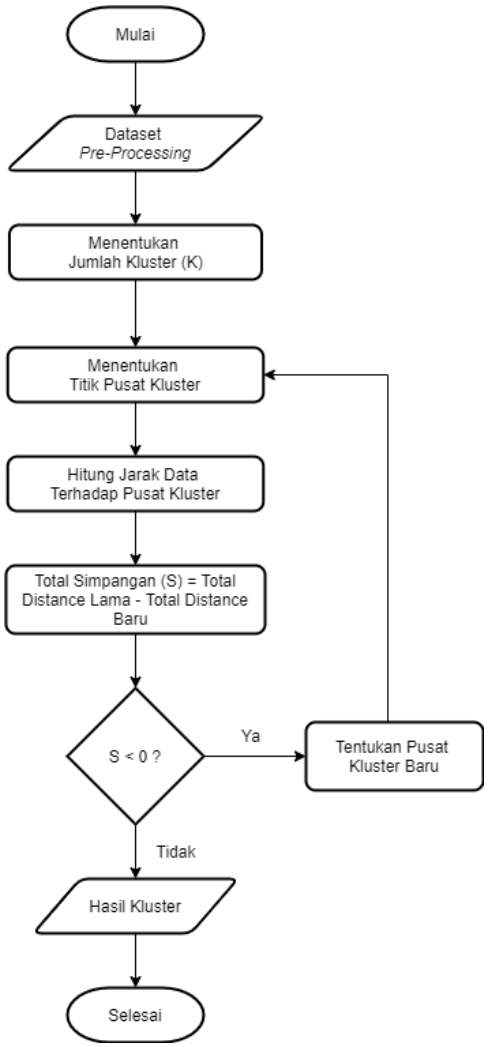
$$v = \frac{\sum_{i=1}^{n} x_i}{n} \; ; 1,2,3, \ldots n \qquad (2)$$

f. Mengulangi langkah ketiga jika centroid baru tidak sama dengan centroid lama.

3.5 Perhitungan Algoritma K-Medoids

Gambar 3.2 merupakan diagram alir dari perhitungan algoritma K-Medoids. Proses perhitungan dimulai dari menententukan jumlah kluster, inisialisasi jumlah kluster, menghitung jarak *euclidean*, serta menghitung total jarak simpangan (S). Jarak simpangan (S) diperoleh dengan menghitung selisih total jarak baru – total

jarak lama. Jika jarak simpangan (S) > 0 maka proses perhitungan selesai. Akan tetapi, jika jarak simpangan (S) < 0 maka lakukan perhitungan sampai menemukan hasil > 0 kemudian proses perhitungan selesai.



Gambar 3.3. Diagram Alir Algoritma K-Medoids

Adapun urutan langkah-langkah algoritma K-Medoids berdasarkan penelitian Septiani (Septiani et al., 2022), secara rinci sebagai berikut:

a. Memilih secara acak medoid awal sebanyak k dari n *instances* pada dataset sebaran Covid-19. Pada tahap ini dilakukan pemilihan medoid awal secara acak dari dataset yang telah di-*preprocessing*.
b. mengelompokkan setiap data Covid-19 ke *cluster* terdekat menggunakan pendekatan *Euclidian Distance* untuk menghitung jarak antar data Covid-19 dengan rumus pada Persamaan (3):

$$d(x, y) = \sqrt{\sum^{n}_{i=1}(xi - yi)^2)} \qquad (3)$$

Keterangan :

Xi = pusat titik medoid

Yi = data *instances*

i = atribut data Covid-19

c. Menandai jarak k terdekat objek data ke medoid dan menghitung totalnya. Untuk menghitung nilai kedekatannya, dengan cara mencari nilai minimum dari cost 1 dan cost 2 kemudian menghitung total kedekatannya.
d. Menentukan anggota *cluster* ke medoid sementara. Rumus untuk perhitungan *cluster* yaitu jika cost 1 = Minimal (Cost 1 dan Cost 2) maka bernilai (1,2)
e. Melakukan iterasi medoid. Untuk mencari iterasi medoid, langkah-langkah yang dilakukan yaitu memilih medoid sementara dan ikuti langkah-langkah diatas mulai dari langkah b, c dan d.
f. Kemudian menghitung jarak simpangan total (S) dengan cara menghitung nilai jarak total baru – jarak total lama. Jika S<0>0.

3.6 Komparasi Kinerja Algoritma

Komparasi kinerja dari klusterisasi K-Means *Clustering* dan K-Medoids menggunakan Davies Bouldin Index sebagai metode evaluasinya. Mengacu pada penelitian Singh (Singh et al., 2020), metode evaluasi DBI memiliki kelebihan dalam hal mengukur evaluasi *cluster* pada suatu metode pengelompokan disebabkan nilai kohesi dan separasi yang dihasilkannya. Kohesi didefinisikan sebagai jumlah dari kedekatan data terhadap centroid dari *cluster* yang diikuti, dikenal sebagai *sum of square within cluster*. Pada saat yang sama, pemisahan didasarkan pada jarak antara pusat cluster, yang disebut jumlah kuadrat antar cluster. Melalui nilai kohesi dan separasi inilah yang menjadikan semakin kecil nilai DBI yang didapatkan (mendekati 0 atau sama dengan 0) maka menunjukkan semakin baik *cluster* dari hasil pengelompokan.

Perhitungan DBI diawali dengan mencari nilai melalui perhitungan nilai *sum of square within cluster* (SSW), yaitu untuk mengetahui matrik kohesi dalam sebuah *cluster* ke-*i* yang dirumuskan pada Persamaan (4).

$$SSW_i = \frac{1}{m_i} \sum_{j=1}^{m_i} d(x_j, c_i) \qquad (4)$$

Keterangan :

SSW : sum of squere within *cluster*

$\boldsymbol{m_i}$: Jumlah data dalam kluster ke- *i*

$\boldsymbol{c_i}$: *Centroid cluster* ke-i

$\mathbf{d}(\boldsymbol{x_j}, \boldsymbol{c_i})$: Jarak dari data ke-i ke titik *cluster i*

Berikutnya, dilakukan langkah selanjutnya menggunakan Persamaan (5) untuk mengetahui separasi antar *cluster*.

$$SSB_{ij} = d(i,j) \qquad (5)$$

Keterangan :

SSB : sum of squere between *cluster*

d(*i,j*) : Jarak *Euclidience distance* data ke i dan data ke-j

Selanjutnya, yaitu mencari Rasio dari hasil perhitungan SSW dan SSB seperti pada Persamaan (6):

$$R_{ij} = \frac{SSW_i + SSW_j}{SSB_{ij}} \qquad (8)$$

Untuk proses perhitungan tahap akhir, Persamaan (9) digunakan untuk memperoleh nilai DBI.

$$DBI = \frac{1}{k}\sum_{i=1}^{k} \max i \neq j \, (R_{i,j}) \quad (9)$$

Keterangan :

k : Jumlah *cluster* yang dihitung

Semakin kecil nilai DBI yang didapatkan, yakni nilai yang mendekati 0 atau sama dengan 0 maka menunjukkan semakin baik *cluster* dari hasil pengelompokan. Nantinya, nilai DBI didapatkan dari Algoritma K-Means dan algoritma K-Medoids. Kedua nilai DBI dikomparasi sebagai evaluasi kinerja dari klusterisasi.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Deskripsi Penelitian

Penelitian ini menggunakan Bahasa Pemrograman Python *Version 3* yang dijalankan pada *tools* Jupyter Notebook. Langkah penelitian dimulai dari *import library* atau modul yang digunakan dalam berbagai tahapan dalam penelitian ini. Tahapan yang dilakukan antara lain melakukan *pre-processing* berupa proses binning data hingga normalisasi data. Selanjutnya, menampilkan visualisasi data sebaran Covid-19. Kemudian, melakukan modeling Algoritma K-Means dan K-Medoids. Hingga diakhiri dengan langkah terakhir berupa evaluasi menggunakan *Davies-Bouldin Index* (DBI).

4.2 *Import Library* pada Python

Tabel 4.1. *Import Library* pada python untuk implementasi K-Means

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.preprocessing import MinMaxScaler
```

Tabel 4.2. *Import Library* pada python untuk implementasi K-Medoids

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
import seaborn as sns
from sklearn_extra.cluster import KMedoids
from sklearn.preprocessing import MinMaxScaler
```

Tabel 4.1 dan Tabel 4.2 menunjukkan *source code* untuk *import* library atau modul yang digunakan dalam implementasikan Algoritma K-Means dan K-Medoids. Berikut adalah penjelasan untuk setiap baris *source code*:

1. **import matplotlib.pyplot as plt**: meng-*import* modul matplotlib.pyplot dan memberi alias plt, yang digunakan untuk membuat visualisasi grafik dan plot di Python. Visualisasi grafik nantinya digunakan untuk menvisualisasikan data sebaran Covid-19 ataupun grafik hasil klasterisasi.
2. **import numpy as np:** meng-*import* modul numpy dan memberi alias np, yang digunakan untuk operasi numerik pada Python seperti array dan matriks.
3. **import pandas as pd:** meng-*import* modul pandas dan memberi alias pd, yang digunakan untuk manipulasi dan analisis data terstruktur seperti tabel dan spreadsheet.
4. **import plotly.express as px:** meng-*import* modul plotly.express dan memberi alias px. Modul ini memiliki fungsi yang hampir sama seperti modul matplotlib.pyplot. Yakni untuk membuat visualisasi data. Bedanya visualisasi yang dihasilkan lebih interaktif.
5. **import plotly.graph_objects as go:** meng-*import* modul plotly.graph_objects dan memberi alias go, yang digunakan untuk membuat visualisasi data yang kompleks dan kustom.
6. **import seaborn as sns:** meng-*import* modul seaborn, yang digunakan untuk membuat visualisasi data statistik yang menarik dan informatif.

7. **from sklearn.cluster *import* KMeans:** meng-*import* kelas KMeans dari modul sklearn.cluster, yang digunakan untuk melakukan analisis cluster pada data sebaran Covid-19.
8. **from sklearn_extra.cluster *import* KMedoids:** meng*import* kelas KMedoids dari modul sklearn_extra.cluster, yang digunakan untuk melakukan analisis cluster pada data sebaran Covid-19.
9. **sklearn.preprocessing *import* MinMaxScaler:** meng-*import* kelas MinMaxScaler dari modul sklearn.preprocessing, yang digunakan untuk melakukan *pre-processing* data seperti normalisasi dan scaling.

Dengan meng-*import* modul-modul ini, maka berbagai fungsi yang terdapat pada setiap modul dapat digunakan. Dalam penelitian ini, fungsi tersebut digunakan untuk menyelesaikan *pre-processing data*, pembuatan visualisasi hingga modelling algoritma K-Means dan K-Medoids di Python.

4.3 Pre-Processing Data

Source code pada Tabel 4.3 mendefinisikan sebuah kelas Python bernama **ColumnData** yang digunakan untuk merepresentasikan nama-nama kolom pada data sebaran Covid-19 di Indonesia. Tahapan ini adalah langkah pertama yang dilakukan untuk *pre-processing* data.

Tabel 4.3. *Source code* untuk merepresentasikan nama-nama kolom pada data sebaran Covid-19

```
class ColumnData:
    date = 'Date'
    province = 'Province'
    island = 'Island'
    cases = 'Total Cases'
    deaths = 'Total Deaths'
    recovered = 'Total Recovered'
```

```
    actives_cases = 'Total Active Cases'
    population = 'Population'
    area = 'Area (km2)'
    mortality = 'Mortality'
    density = 'Population Density'
```

Pengubahan nama-nama kolom yang dilakukan antara lain:

1. **date = 'Date':** merepresentasikan nama kolom untuk tanggal atau waktu, yang menunjukkan kapan kasus Covid-19 terjadi.
2. **province = 'Province':** merepresentasikan nama kolom untuk provinsi atau wilayah di Indonesia di mana kasus Covid-19 terjadi.
3. **island = 'Island':** merepresentasikan nama kolom untuk pulau di Indonesia di mana kasus Covid-19 terjadi.
4. **cases = 'Total Cases'**: merepresentasikan nama kolom untuk jumlah kasus Covid-19 di suatu wilayah pada suatu waktu tertentu.
5. **deaths = 'Total Deaths':** merepresentasikan nama kolom untuk jumlah kematian akibat Covid-19 pada suatu wilayah pada suatu waktu tertentu.
6. **recovered = 'Total Recovered':** merepresentasikan nama kolom untuk jumlah kesembuhan dari Covid-19 pada suatu wilayah pada suatu waktu tertentu.
7. **actives_cases = 'Total Active Cases':** merepresentasikan nama kolom untuk jumlah kasus Covid-19 aktif yang masih terjadi pada suatu wilayah pada suatu waktu tertentu.
8. **population = 'Population':** merepresentasikan nama kolom untuk jumlah populasi suatu wilayah di Indonesia.

9. **area = 'Area (km2)':** merepresentasikan nama kolom untuk luas wilayah suatu wilayah di Indonesia dalam kilometer persegi.
10. **mortality = 'Mortality':** merepresentasikan nama kolom untuk angka kematian akibat Covid-19 pada suatu wilayah, yang dihitung dengan membagi jumlah kematian dengan jumlah kasus.
11. **density = 'Population Density':** merepresentasikan nama kolom untuk kepadatan penduduk suatu wilayah di Indonesia, yang dihitung dengan membagi jumlah populasi dengan luas wilayah.

Dengan mendefinisikan kelas **ColumnData** seperti ini, peneliti dapat menggunakannya sebagai referensi untuk mengakses kolom-kolom dalam dataset kasus Covid-19 di Indonesia, sehingga membuat *source code* lebih mudah dibaca dan dipahami. Misalnya, jika mengakses kolom jumlah kasus pada suatu wilayah pada suatu waktu tertentu, maka dapat menuliskan **ColumnData.cases.**

Tabel 4.4. Binning data

```
def create_bins(df, columns, q=5):
    for column in columns:
        df[column] = pd.qcut(df[column], q,
duplicates='drop').cat.codes
```

Tabel 4.4 menunjukkan *source code* untuk melakukan tahapan *pre-processing* berikutnya, yakni melakukan binning data atau mengelompokkan data ke dalam bagian-bagian yang lebih kecil yang disebut bin berdasarkan kriteria tertentu. Pada *source code*, terdapat **create_bins**, yang merupakan sebuah fungsi yang menerima tiga parameter, yaitu **df**, **columns**, dan **q**. **df** adalah sebuah objek

pandas dataframe yang akan diolah. Selain itu objek **df** inilah yang dalam penelitian ini berfungsi untuk menyimpan data sebaran Covid-19. **columns** adalah sebuah *list* yang berisi nama kolom atau atribut dalam *dataframe* yang akan diolah. **q** adalah sebuah integer yang menentukan jumlah bin atau kelompok yang dihasilkan. Fungsi ini mengiterasi setiap kolom yang ada di **columns**, kemudian memanggil fungsi **pd.qcut** dari *library Pandas* pada setiap kolom tersebut. Fungsi **pd.qcut** digunakan untuk membagi setiap kolom menjadi beberapa bin dengan jumlah yang sama berdasarkan kuantil yang sama. Setelah kolom dibagi menjadi beberapa bin, fungsi **cat.codes** akan dipanggil untuk mengubah setiap nilai pada kolom menjadi bin yang sesuai. Bin yang dihasilkan ditentukan berdasarkan urutan kuantil dari terkecil hingga terbesar. *Source code* ini dapat digunakan untuk memudahkan proses analisis data pada data sebaran Covid-19 di Indonesia, seperti untuk membagi data ke dalam beberapa kelompok berdasarkan kuantil, sehingga dapat mempermudah analisis dan visualisasi data secara lebih efektif.

Tabel 4.5. *Source code* untuk normalisasi data

```
def normalize_data(df, columns):
    minMaxScaler = MinMaxScaler()
    df[columns] =
minMaxScaler.fit_transform(d[columns])
```

Tabel 4.5 menunjukkan *source code* untuk melakukan tahapan *pre-processing* berikutnya, yakni normalisasi data yang di ditampung dalam fungsi **normalize_data**. **normalize_data** merupakan sebuah fungsi yang menerima dua parameter, yaitu **df** dan **columns**. **df** adalah sebuah objek *pandas dataframe* yang akan dinormalisasi. **columns** adalah sebuah *list* yang berisi nama kolom atau atribut dalam *dataframe* yang akan dinormalisasi. Fungsi ini menggunakan objek

MinMaxScaler dari *library scikit-learn* untuk melakukan normalisasi data pada kolom-kolom yang terdapat dalam parameter **columns**. Normalisasi ini mengubah setiap nilai pada kolom menjadi nilai yang berada dalam rentang 0 sampai 1.

Pada fungsi ini, objek **MinMaxScaler** diinisialisasi terlebih dahulu, kemudian diaplikasikan pada kolom-kolom yang terdapat dalam parameter **columns** dengan menggunakan method **fit_transform**. Setelah normalisasi, kolom-kolom dalam *dataframe* **df** akan di-*update* dengan nilai-nilai yang telah dinormalisasi. *Source code* **normalize_data** sebagaimana pada Gambar 4.5 dapat digunakan setelah fungsi **create_bins**. Dalam penelitian ini **normalize_data** berperan dalam melakukan normalisasi pada data sebaran Covid-19 di Indonesia setelah data dibagi ke dalam beberapa kelompok berdasarkan kuantil. Dengan melakukan normalisasi, nilai-nilai dalam kolom yang berbeda skala atau rentang dapat dibandingkan dan diolah secara lebih efektif.

Tabel 4.6. Membaca data sebaran Covid-19 di Indonesia

```
data =
pd.read_csv('covid_19_indonesia_time_series_all.csv
')
data.head()
```

Tabel 4.6 berisikan *source code* untuk membaca *file* csv **covid_19_indonesia_time_series_all.csv** dan memasukkannya dalam sebuah *dataframe* menggunakan *library pandas*. Fungsi **read_csv()** dari pandas digunakan untuk membaca *file* csv dan mengonversinya ke dalam sebuah dataframe. *File* csv **covid_19_indonesia_time_series_all.csv** berisi data kasus COVID-19 di Indonesia sebagaimana yang telah dijelaskan pada bab sebelumnya. Setelah data dimuat ke dalam *dataframe*, *source code* **data.head()** digunakan untuk

menampilkan 5 baris pertama dari **dataframe** tersebut. Fungsi **head()** dari pandas digunakan untuk menampilkan beberapa baris teratas dari *dataframe*, dengan jumlah baris yang ditampilkan secara *default* adalah 5. Tujuannya adalah untuk memastikan data telah dimuat dengan benar dan melihat seperti apa format data yang digunakan dalam tahapan analisis berikutnya. Luaran dari *source code* pada Tabel 4.6 tertera pada Gambar 4.1.

	Date	Location ISO Code	Location	New Cases	New Deaths	New Recovered	New Active Cases	Total Cases	Total Deaths	Total Recovered	...	Latitude	New Cases per Million	Total Cases per Million	New Deaths per Million	Total Deaths per Million	Total Deaths per 100rb	Case Fatality Rate
0	3/1/2020	ID-JK	DKI Jakarta	2	0	0	2	39	20	75	...	-6.204699	0.18	3.60	0.0	1.84	0.18	51.28%
1	3/2/2020	ID-JK	DKI Jakarta	2	0	0	2	41	20	75	...	-6.204699	0.18	3.78	0.0	1.84	0.18	48.78%
2	3/2/2020	IDN	Indonesia	2	0	0	2	2	0	0	...	-0.789275	0.01	0.01	0.0	0.00	0.00	0.00%
3	3/2/2020	ID-RI	Riau	1	0	0	1	1	0	1	...	0.511548	0.16	0.16	0.0	0.00	0.00	0.00%
4	3/3/2020	ID-JK	DKI Jakarta	2	0	0	2	43	20	75	...	-6.204699	0.18	3.96	0.0	1.84	0.18	46.51%

5 rows × 38 columns

Gambar 4.1. Data sebaran Covid-19 yang telah di-*load*

Tabel 4.7. Pemilihan kolom tertentu pada *dataframe*

```
data = data[[
    ColumnData.date,
    ColumnData.province,
    ColumnData.island,
    ColumnData.cases,
    ColumnData.deaths,
    ColumnData.recovered,
    ColumnData.actives_cases,
    ColumnData.population,
    ColumnData.area,
    ColumnData.density
]]
```

Pada *source code* di Tabel 4.7, *pre-processing* berlanjut dengan melakukan pemilihan kolom tertentu pada *dataframe* yang berisikan data sebaran Covid-19, yaitu **ColumnData.date**, **ColumnData.province**, **ColumnData.island**,

ColumnData.cases, **ColumnData.deaths**, **ColumnData.recovered**, **ColumnData.actives_cases**, **ColumnData.population**, **ColumnData.area**, dan **ColumnData.density**. Pemilihan kolom dilakukan dengan menggunakan *operator double square brackets* ("[[" dan "]]") yang mengindikasikan pemilihan beberapa kolom sekaligus. Pemilihan kolom dilakukan berdasarkan kolom-kolom yang berkontribusi dalam klasterisasi menggunakan K-Means dan K-Medoids.

Tabel 4.8. *Row deletion* pada *dataframe*

```
data = data.dropna(axis=0, how="any")
```

Tabel 4.8 berisikan *source code* yang berperan vital dalam *pre-processing* data sebaran Covid-19, yakni operasi penghapusan baris (*row deletion*) pada *dataframe* yang berisi data sebaran Covid-19 di Indonesia. Hal ini dilakukan karena *dataframe* memiliki beberapa baris yang memiliki nilai kosong atau *missing value* (NaN). Oleh karena itu, dilakukan penghapusan semua baris yang memiliki setidaknya satu nilai kosong. Fungsi **dropna()** digunakan untuk melakukan penghapusan baris yang memiliki nilai kosong. Parameter pertama **axis=0** menunjukkan bahwa penghapusan dilakukan pada baris (*row*), sedangkan parameter kedua **how="any"** menunjukkan bahwa baris dihapus jika memiliki setidaknya satu nilai kosong. Dengan melakukan operasi penghapusan baris yang memiliki nilai kosong, maka data yang tersisa dalam *dataframe* tersebut sudah pasti lengkap (tidak ada nilai kosong), sehingga data sebaran Covid-19 di Indonesia siap digunakan untuk analisis atau pemodelan menggunakan Algoritma K-Means maupun K-Medoids.

Tabel 4.9. Pengubahan tipe data pada kolom “Date”

```
data[ColumnData.date] = pd.to_datetime(data.Date,
infer_datetime_format=True).dt.date
```

Selanjutnya, pada dataframe terdapat sebuah kolom bernama **Date**. *Source code* pada Tabel 4.9 digunakan digunakan untuk mengubah tipe data pada kolom "Date" menjadi tipe data tanggal (*date*) dengan format yang tepat, kemudian menyimpan hasilnya pada **ColumnData.date**. Dengan melakukan transformasi tersebut, maka data pada kolom **Date** yang awalnya dalam format string atau objek diubah menjadi tipe data tanggal yang lebih mudah untuk diproses dan dianalisis. Transformasi ini juga mempermudah adanya penggunaan berbagai operasi terkait tanggal, seperti perhitungan selisih tanggal atau pengelompokkan data berdasarkan tanggal pada proses modeling algorima K-Means maupun K-Medoids.

Tabel 4.10. Perhitungan *mortality*

```
data[ColumnData.mortality] = data[ColumnData.deaths] /
data[ColumnData.cases]
```

Berikutnya adalah melakukan perhitungan **ColumnData.mortality**. Pada *dataframe* terdapat kolom **deaths** yang berisi jumlah kematian akibat Covid-19, dan kolom **cases** yang berisi jumlah kasus positif Covid-19. *Source code* pada Tabel 4.10 digunakan untuk menghitung tingkat kematian (*mortality rate*) dari Covid-19 pada setiap baris data, kemudian menyimpan hasilnya pada **ColumnData.mortality**. Oleh karena itu, maka data sebaran Covid-19 pada *dataframe* saat ini ditambahkan informasi terkait tingkat kematian pada setiap baris

data. Informasi tersebut berguna untuk memperoleh gambaran yang lebih komprehensif terkait dampak dari Covid-19 pada setiap provinsi di Indonesia.

Tabel 4.11. Transformasi dan agregasi pada *dataframe*

```
dfl = data[
    [ColumnData.date, ColumnData.cases,
ColumnData.deaths, ColumnData.recovered]
].groupby(ColumnData.date).sum().reset_index()

dfl = dfl[(dfl[ColumnData.cases] >=
100)].melt(id_vars=ColumnData.date,

value_vars=[ColumnData.cases, ColumnData.deaths,
ColumnData.recovered])
```

Langkah *pre-processing* terakhir adalah melakukan transformasi dan agregasi pada kolom tertentu di *dataframe*. Terdapat beberapa kolom yang dipilih pada *dataframe*, yaitu kolom **date** yang berisi tanggal terkait kasus Covid-19, kolom **cases** yang berisi jumlah kasus positif Covid-19, kolom **deaths** yang berisi jumlah kematian akibat Covid-19, dan kolom **recovered** yang berisi jumlah pasien yang sembuh dari Covid-19. *Source code* pada Tabel 4.11 menunjukkan beberapa tahapan proses dalam transformasi dan agregasi, antara lain:

1. Dilakukan seleksi kolom untuk memperoleh kolom-kolom yang dibutuhkan untuk analisis, yaitu kolom **date**, **cases**, **deaths**, dan **recovered**.
2. Kemudian, dilakukan operasi **groupby** pada kolom **date**, yang digunakan untuk mengelompokkan data berdasarkan tanggal, dan melakukan operasi agregasi (sum) pada kolom-kolom lainnya untuk memperoleh total jumlah kasus, kematian, dan kesembuhan pada setiap tanggal.

3. Setelah itu, dilakukan reset index pada *dataframe* hasil agregasi untuk mereset indeks baris dan membuat kolom **date** menjadi kolom biasa.
4. Selanjutnya, dilakukan seleksi data untuk memperoleh data hanya pada tanggal-tanggal tertentu dimana jumlah kasus Covid-19 melebihi atau sama dengan 100.
5. Terakhir, dilakukan operasi **melt** pada *dataframe* hasil seleksi, dengan menggunakan kolom **date** sebagai **id_vars**, dan kolom-kolom **cases**, **deaths**, dan **recovered** sebagai **value_vars**. Operasi **melt** digunakan untuk mengubah *dataframe* dari format *wide* menjadi format *long*, sehingga data dapat lebih mudah untuk dianalisis dan divisualisasikan.

Dengan melakukan transformasi dan agregasi data tersebut, maka data sebaran Covid-19 di Indonesia diolah menjadi format yang lebih mudah untuk dianalisis dan divisualisasikan.

4.4 Visualisasi Data

Tabel 4.12. *Source code* untuk visualisasi grafik baris berdasarkan kesembuhan dan kematian akibat Covid-19

```
vis_lp = px.line(dfl, x=ColumnData.date, y='value',
color='variable')
vis_lp.update_layout(title='COVID-19 in Indonesia:
total number of cases over time',
                     xaxis_title='Indonesia',
yaxis_title='Number of cases',
                     legend=dict(x=0.02, y=0.98))
vis_lp.show()
```

Setelah seluruh tahapan *pre-processing* selesai dilakukan, maka tahapan berikutnya adalah melakukan visualisasi data dalam berbagai bentuk grafik dengan tujuan mempermudah penyampaian informasi awal soal data sebaran Covid-19 di

Indonesia sebelum dilakukan modelling algoritma K-Means dan K-Modoids. Tabel 4.12 menunjukkan *source code* yang digunakan untuk membuat visualisasi grafik garis (*line chart*) dari *dataframe* **dfl** yang telah diolah pada *source code* sebelumnya. Visualisasi grafik garis menampilkan perkembangan jumlah kasus Covid-19, jumlah kematian, dan jumlah kesembuhan di Indonesia dari waktu ke waktu.

Pada baris pertama, *source code* menggunakan *library* **plotly express** (px) untuk membuat visualisasi grafik garis. Fungsi **line()** digunakan untuk membuat grafik garis. Kemudian**, x=ColumnData.date** digunakan untuk menentukan sumbu x pada grafik untuk series waktu, dan **y='value'** yang digunakan untuk menentukan sumbu y pada grafik yang menunjukkan jumlah kasus, kematian, dan kesembuhan. Setelah itu, **color='variable'** digunakan untuk memberikan warna yang berbeda pada setiap garis yang merepresentasikan jumlah kasus, kematian, dan kesembuhan.

Selanjutnya, pada baris kedua, *source code* berfungsi untuk menambahkan judul grafik, label sumbu x dan y, dan posisi legend. Terdapat **legend=dict(x=0.02, y=0.98)** yang digunakan untuk menentukan posisi legend pada grafik. Pada baris terakhir, digunakan fungsi **show()** untuk menampilkan grafik yang telah dibuat. Sehingga, tampil grafik pada Gambar 4.2.

Gambar 4.2. Grafik Covid-19 di Indonesia dari waktu ke waktu

Gambar 4.2 menyajikan informasi berupa grafik baris mengenai perkembangan Covid-19 di Indonesia dari waktu ke waktu, mulai dari Maret 2020 hingga September 2022. Terdapat lonjakan yang signifikan pada total kasus yang diiringi oleh total kesembuhan. Untuk total kematian, terdapat lonjakan cukup signifikan pada bulan Juli 2021. Melalui grafik ini, dapat diketahui bahwa terdapat lonjakan kasus Covid-19 di Indonesia pada periode bulan-bulan tertentu.

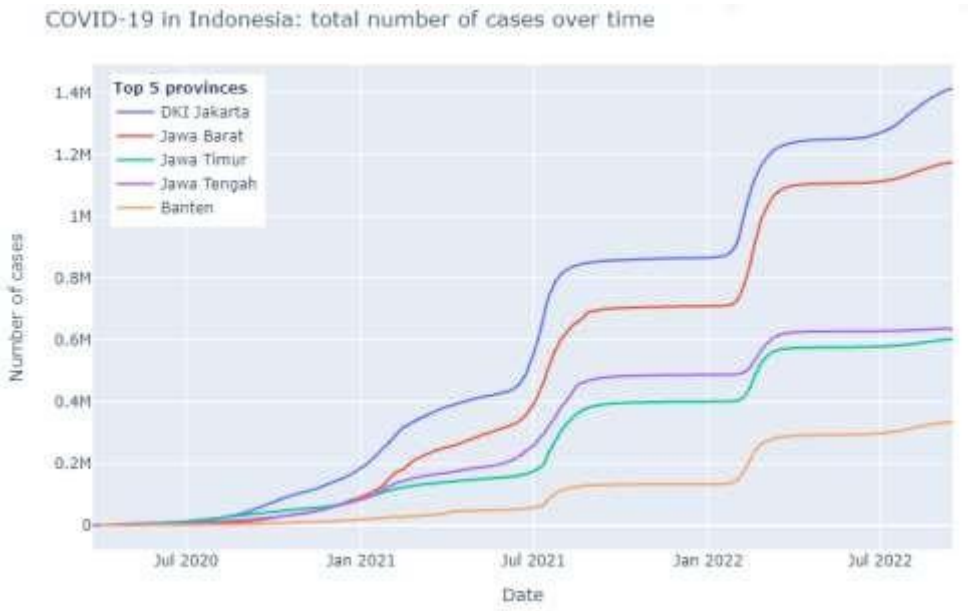
Tabel 4.13. *Source code* untuk menyajikan jumlah kasus Covid-19 di beberapa provinsi di Indonesia dalam grafik garis

```
pd.options.mode.chained_assignment = None
limit = 5
group = data.groupby(ColumnData.province)
t = group.tail(1).sort_values(ColumnData.cases,
ascending=False).set_index(ColumnData.province).drop(
    columns=[ColumnData.date])

s = data[(data[ColumnData.province].isin([i for i in
t.index[:limit]]))]
```

```
s = s[(s[ColumnData.cases] >= 100)]

# vis_lp = visualization line plot
vis_lp2 = px.line(s, x=ColumnData.date,
y=ColumnData.cases, color=ColumnData.province)
vis_lp2.update_layout(title='COVID-19 in Indonesia:
total number of cases over time',
                      xaxis_title=ColumnData.date,
yaxis_title='Number of cases',
                      legend_title='<b>Top          %s
provinces</b>' % limit,
                      legend=dict(x=0.02, y=0.98))
vis_lp2.show()
```

Source code pada Tabel 4.13 digunakan untuk membuat visualisasi grafik garis (*line chart*) dari *dataframe* data yang merepresentasikan jumlah kasus Covid-19 di beberapa provinsi di Indonesia dari waktu ke waktu. Pada baris pertama, *source code* menggunakan **pd.options.mode.chained_assignment** untuk menonaktifkan warning yang muncul saat melakukan *assignment* pada *dataframe*. Pada baris kedua *source code*, digunakan fungsi **groupby()** untuk mengelompokkan data berdasarkan nama provinsi pada kolom **ColumnData.province**. Setelah itu, fungsi **tail()** digunakan untuk memilih baris terakhir dari setiap kelompok provinsi (yang merepresentasikan data terbaru dari setiap provinsi), kemudian hasilnya diurutkan berdasarkan jumlah kasus dengan fungsi **sort_values().** Kemudian kolom **ColumnData.date** dihapus dengan fungsi **drop()** karena tidak digunakan pada visualisasi yang akan dibuat. Hasilnya disimpan dalam variabel **t**. Pada baris ketiga, diambil baris-baris pada *dataframe* **data** yang merepresentasikan provinsi-provinsi dengan jumlah kasus tertinggi (berdasarkan variabel limit yang diatur pada baris pertama), dan jumlah kasus di atas atau sama dengan 100. Hasilnya disimpan dalam variabel **s**. Pada baris

keempat, digunakan **plotly express** untuk membuat visualisasi grafik garis dengan fungsi **line()**. Argumen **x=ColumnData.date** digunakan untuk menentukan sumbu x pada grafik (tanggal), argumen **y=ColumnData.cases** digunakan untuk menentukan sumbu y pada grafik (jumlah kasus), dan argumen **color=ColumnData.province** digunakan untuk memberikan warna yang berbeda pada setiap garis yang merepresentasikan setiap provinsi. Pada baris kelima, dilakukan *update* pada *layout* grafik dengan menambahkan judul grafik, label sumbu x dan y, judul legend, dan posisi legend. Pada baris ini, judul grafik ditentukan dengan **title='COVID-19 in Indonesia: total number of cases over time'**, sedangkan label sumbu x dan y ditentukan dengan argumen **xaxis_title** dan **yaxis_title**. Judul *legend* ditentukan dengan argumen **legend_title**, sedangkan posisi *legend* ditentukan dengan argumen **legend**. Pada baris terakhir, digunakan fungsi **show()** untuk menampilkan grafik yang telah dibuat. Tampilan grafik ditunjukkan oleh Ganbar 4.3.



Gambar 4.3. Grafik Covid-19 di Indonesia dari waktu ke waktu berdasarkan lima

provinsi dengan kasus Covid-19 tertinggi.

Tabel 4.14. *Source code* untuk visualisasi data dalam bentuk heatmap mengenai jumlah kasus Covid-19 di Indonesia dalam waktu tertentu

```
heatmap = data[(data[ColumnData.cases] >=
100)].sort_values([ColumnData.date,
ColumnData.province])
vis_hmap = go.Figure(data=go.Heatmap(
    z=heatmap[ColumnData.cases],
    x=heatmap[ColumnData.date],
    y=heatmap[ColumnData.province],
    colorscale='Viridis'))

vis_hmap.update_layout(
    title='COVID-19 in Indonesia: number of cases
over time', xaxis_nticks=45)

vis_hmap.show()
```

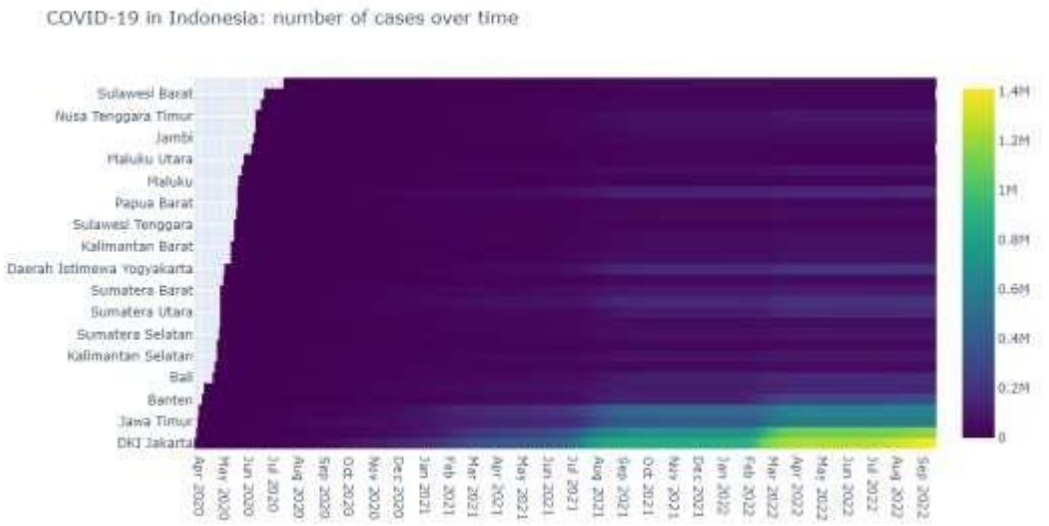
Source code pada Tabel 4.14 berfungsi untuk menghasilkan *heatmap* yang menunjukkan jumlah kasus COVID-19 di Indonesia selama periode waktu tertentu. Penjelasan *source code* tersebut secara detail adalah sebagai berikut: Pertama-tama, data yang digunakan diolah untuk memfilter data yang memiliki jumlah kasus lebih dari atau sama dengan 100, dan diurutkan berdasarkan tanggal dan provinsi menggunakan method **sort_values().** Hasilnya disimpan ke dalam variabel *heatmap*. Selanjutnya, dibuat sebuah objek **Figure** dari *library* **plotly.graph_objects** dengan memanggil method **go.Figure()** dan menyimpannya ke dalam variabel **vis_hmap**. Pada objek **Figure** tersebut, dibuat sebuah *heatmap* dengan memanggil method **go.Heatmap()** dan menyertakan argumen-argumen seperti:

- **z**: data jumlah kasus yang diambil dari variabel *heatmap* dan kolom **ColumnData.cases**
- **x**: data tanggal yang diambil dari variabel *heatmap* dan kolom **ColumnData.date**
- **y**: data provinsi yang diambil dari variabel *heatmap* dan kolom **ColumnData.province**
- **colorscale**: skala warna yang digunakan dalam *heatmap*, dalam hal ini *Viridis*.

Setelah itu, layout dari objek **Figure** diperbarui dengan memanggil method **update_layout()** dan menyertakan argumen-argumen seperti:

- **title:** judul dari heatmap
- **xaxis_nticks:** jumlah tick pada sumbu x

Terakhir, *heatmap* ditampilkan dengan memanggil method **show()** pada variabel **vis_hmap**. Output dari *source code* tersebut adalah sebuah *heatmap* yang menunjukkan jumlah kasus COVID-19 di setiap provinsi di Indonesia selama periode waktu tertentu. Sumbu horizontal menunjukkan tanggal, sumbu vertikal menunjukkan nama provinsi, dan warna di setiap kotak menunjukkan jumlah kasus di provinsi tersebut pada tanggal tertentu. Semakin terang warnanya, semakin banyak kasus di suatu provinsi pada tanggal tersebut. Gambar 4.4 menunjukkan luaran dari diagram



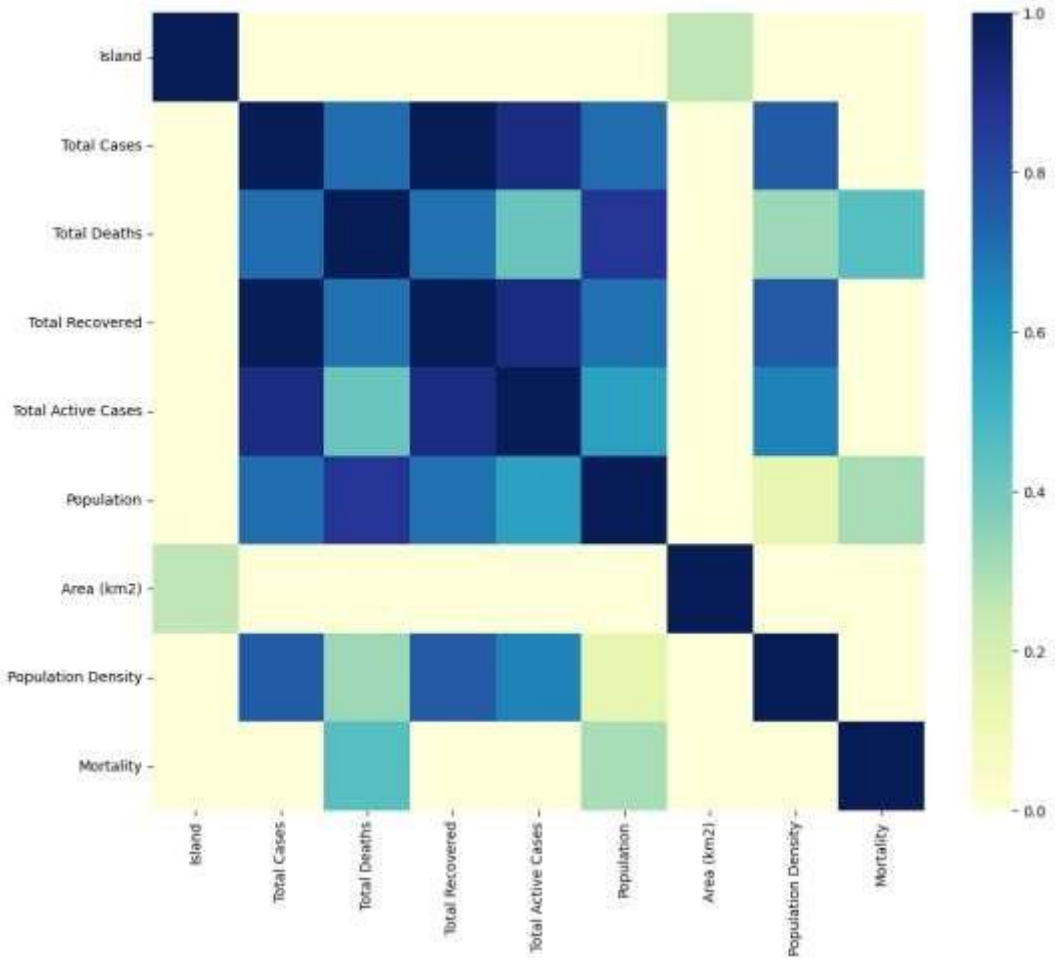
Gambar 4.4. Visualisasi data dalam bentuk heatmap mengenai jumlah kasus Covid-19 di Indonesia dalam waktu tertentu

Tabel 4.15. *Source code* untuk visualisasi data dalam bentuk heatmap mengenai korelasi antar data pada setiap fitur

```
t.replace({'Jawa': 0, 'Sulawesi': 1, 'Kalimantan': 2,
'Sumatera': 3, 'Maluku': 4, 'Papua': 5, 'Nusa
Tenggara': 6},
          inplace=True)

fig, ax = plt.subplots(figsize=(12, 10))
sns.heatmap(t.corr(), vmin=0, cmap='YlGnBu')
plt.show()
```

Source code pada Tabel 4.15 bertujuan untuk membuat *heatmap* yang menampilkan korelasi antara data pada setiap fitur (kolom) dalam dataset **t**. Dataset **t** yang dimaksud adalah data sebaran Covid-19. Korelasi menunjukkan seberapa kuat hubungan linier antara dua fitur dalam dataset. Pertama, terdapat *code* untuk mengganti nama provinsi dengan angka (0 sampai 6) menggunakan metode *replace*. Angka ini digunakan sebagai label untuk sumbu x dan y pada *heatmap*. Selanjutnya, dengan menggunakan *library seaborn*, *heatmap* dibuat dengan memanggil **sns.heatmap().** Fungsi ini sebagai argumen matriks korelasi dari fitur-

fitur dalam dataset **t**. Argumen **vmin=0** digunakan untuk menentukan nilai minimum pada skala warna *heatmap*. Argumen **cmap='YlGnBu'** menentukan palet warna yang digunakan pada *heatmap*. Terakhir, visualisasi *heatmap* ditampilkan dengan memanggil **plt.show().** Ukuran *figure* (lebar dan tinggi) dapat diatur menggunakan **figsize.** Luaran *source code* pada Tabel 4.15 tertera pada Gambar 4.5.



Gambar 4.5. Visualisasi data dalam bentuk *heatmap* mengenai korelasi antar data pada setiap fitur

Tabel 4.16. *Source code* untuk pemilihan fitur

```
corr = t.corr().iloc[[0, 1]].transpose()
corr       =       corr[(corr[ColumnData.cases]       >
0.25)].sort_values(ColumnData.cases, ascending=False)
features = corr.index.tolist()
features.append(ColumnData.mortality)
print('Selected features:', features)

d = t[features].copy()
d.head()
```

Tabel 4.16 merupakan *source code* untuk proses pemilihan fitur (*feature selection*) pada dataset **t** yang sebelumnya telah dilakukan transformasi label encoding pada kolom '**island**' dan juga di-*drop* kolom '**date**'. Pertama, dilakukan perhitungan korelasi antara kolom '**cases**' dan '**recovered**' pada setiap daerah di Indonesia menggunakan **t.corr().** Hasil korelasi tersebut diambil hanya untuk kolom '**cases**' dan '**recovered**' dengan **iloc[[0, 1]].transpose()**. Selanjutnya, dilakukan seleksi fitur dengan mengambil hanya daerah-daerah yang memiliki korelasi di atas 0.25 dengan **corr[(corr[ColumnData.cases] > 0.25)].sort_values(ColumnData.cases, ascending=False)**. Daerah-daerah tersebut kemudian ditampung dalam variabel **features**. Terakhir, dilakukan pengambilan data pada **t** hanya pada kolom-kolom yang terpilih dalam **features** dengan **d = t[features].copy()**. Hasilnya merupakan *dataframe* baru yang hanya berisi data pada kolom-kolom terpilih tersebut, sebagaimana ditampilkan pada Gambar 4.6

Province	Total Cases	Total Recovered	Total Active Cases	Population Density	Total Deaths	Population	Mortality
DKI Jakarta	1412511	1386134	10864	16334.31	15513	10846145	0.010983
Jawa Barat	1173731	1144358	13436	1276.55	15937	45161325	0.013578
Jawa Tengah	636409	601517	1403	1108.64	33489	36364072	0.052622
Jawa Timur	601545	569003	778	846.78	31764	40479023	0.052804
Banten	333875	328482	2443	1109.64	2950	10722374	0.008836

Gambar 4.6. Daftar fitur-fitur yang terpilih

Hasil yang dibuat oleh kode sumber pada Tabel 4.16 adalah hasil yang ditunjukkan pada Gambar 4.6. Gambar 4.6 menunjukkan daftar fungsi yang dipilih , yaitu **'Total Cases', 'Total Recovered', 'Total Active Cases', 'Population Density', 'Total Deaths', 'Population',** dan **'Mortality'**, serta 5 baris pertama dari data sebaran Covid-19 yang hanya menggunakan fitur-fitur tersebut.

Tabel 4.17. Normalisasi data

```
create_bins(d, [
    ColumnData.cases,
    ColumnData.deaths,
    ColumnData.recovered,
    ColumnData.actives_cases,
    ColumnData.population,
    ColumnData.mortality,
    ColumnData.density
], q=7)

normalize_data(d, d.columns)
d.head()
```

Berdasarkan fitur yang dipilih pada *source code* di Tabel 4.17 dilakukan binning menjadi 7 bagian dan selanjutnya dilakukan normalisasi data sehingga mengurangi adanya *outliers* dan data lebih mudah dipahami dengan jarak antara 0-

1. Proses normalisasi dilakukan melalui *source code* pada Tabel 4.17, menghasilkan luaran sebagaimana pada Gambar 4.7.

Province	Total Cases	Total Recovered	Total Active Cases	Population Density	Total Deaths	Population	Mortality
DKI Jakarta	1.0	1.0	1.0	1.000000	1.000000	1.000000	0.0
Jawa Barat	1.0	1.0	1.0	1.000000	1.000000	1.000000	0.0
Jawa Tengah	1.0	1.0	1.0	1.000000	1.000000	1.000000	1.0
Jawa Timur	1.0	1.0	1.0	0.833333	1.000000	1.000000	1.0
Banten	1.0	1.0	1.0	1.000000	0.666667	0.833333	0.0

Gambar 4.7. Luaran normalisasi data

4.5. Analisis Menggunakan K-Means

Tabel 4.18. Seleksi fitur pada proses modelling K-Means

```
# Modelling
# Seleksi Fitur
X = d[['Mortality', 'Total Cases','Total Active
Cases', 'Population Density', 'Population', 'Total
Deaths']]
```

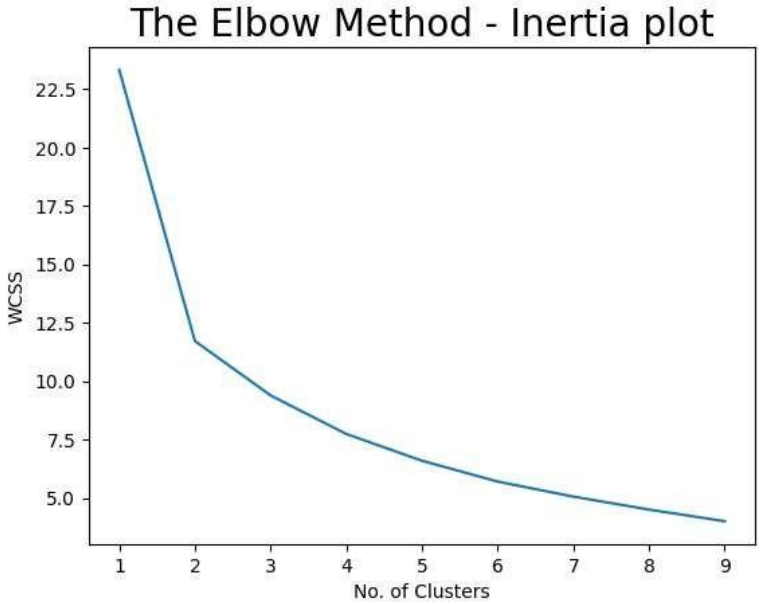
Tabel 4.18 merupakan *source code* untuk tahap seleksi fitur pada proses modelling dengan menggunakan dataset Covid-19 di Indonesia. Fitur-fitur yang dipilih untuk dilibatkan dalam model ini adalah **Mortality, Total Cases, Total Active Cases, Population Density, Population**, dan **Total Deaths**. Fitur-fitur tersebut dipilih karena mempengaruhi perkembangan kasus Covid-19 di Indonesia. Dalam *source code* tersebut, variabel X diisi dengan data dari fitur-fitur tersebut yang sudah dipilih. Data fitur-fitur tersebut digunakan sebagai input dalam model K-Means yang dibangun untuk memprediksi kasus Covid-19 di Indonesia.

Tabel 4.19. Metode Elbow untuk penentuan jumlah klaster optimal pada K-Means

```
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

#Elbow Method - Inertia plot
inertia = []
#looping the inertia calculation for each k
for k in range(1, 10):
    #Assign KMeans as cluster_model
    cluster_model = KMeans(n_clusters = k,
random_state = 24)
    #Fit cluster_model to X
    cluster_model.fit(X)
    #Get the inertia value
    inertia_value = cluster_model.inertia_
    #Append the inertia_value to inertia list
    inertia.append(inertia_value)
##Inertia plot
plt.plot(range(1, 10), inertia)
plt.title('The Elbow Method - Inertia plot', fontsize
= 20)
plt.xlabel('No. of Clusters')
plt.ylabel('WCSS')
plt.show()
```

Tabel 4.19 merupakan implementasi dari metode elbow untuk menentukan jumlah klaster yang optimal pada data sebaran Covid-19. Metode elbow merupakan teknik penentuan jumlah cluster yang optimal pada saat pengelompokan data menggunakan algoritma k-means. Pada *source code* tersebut, dilakukan looping dari k=1 hingga k=9, dimana pada setiap iterasi dibuat objek KMeans dengan jumlah klaster k yang bersesuaian. Kemudian objek tersebut dilatih dengan fitur X dan dihitung nilai inertia (*Within Cluster Sum of Squares*/WCSS) nya. Nilai inertia tersebut kemudian di-append ke dalam list inertia. Setelah proses loop selesai, nilai inertia diplot pada grafik Inertia plot dengan sumbu-x merepresentasikan jumlah klaster, dan sumbu-y merepresentasikan nilai inertia. Untuk menentukan nilai k

yang optimal adalah tujuan dari plot ini, yaitu ketika penurunan nilai inertia mulai mengalami perubahan yang lebih lambat (bentuk seperti siku), sehingga sering disebut juga sebagai "elbow point", sebagaimana ditunjukkan pada Gambar 4.8.



Gambar 4.8. Grafik hasil metode Elbow untuk penentuan jumlah klaster optimal pada K-Means

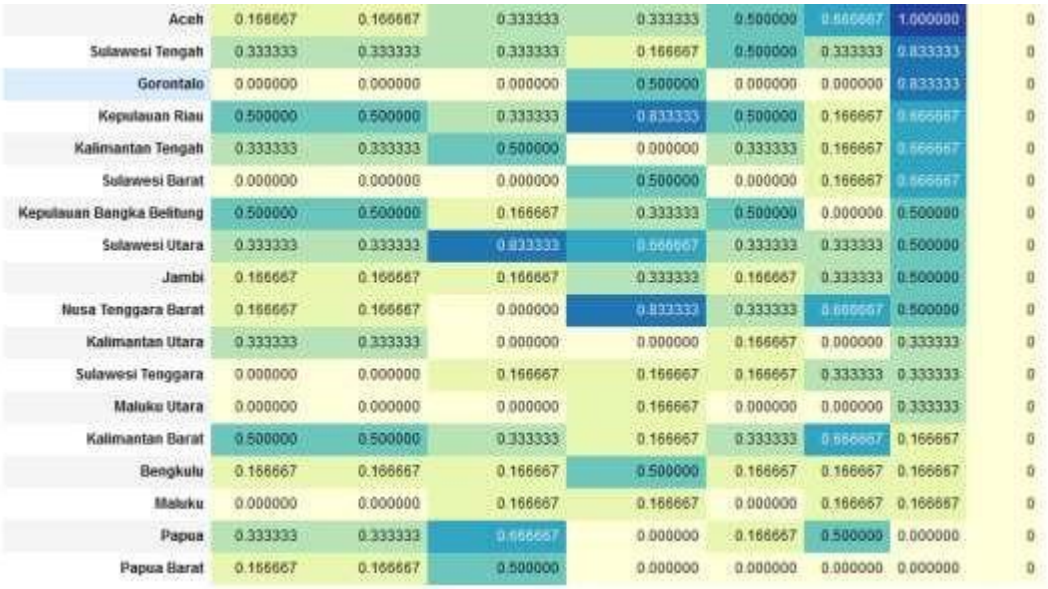
Berdasarkan Gambar 4.8, penentuan nilai k sebagai jumlah klaster yang optimal didapatkan ketika penurunan nilai inertia mulai mengalami perubahan yang lebih lambat, sehingga k yang optimal didapatkan pada nilai k = 2. Nilai k tersebut digunakan dalam proses selanjutnya, yakni modelling algoritma K-Means.

Tabel 4.20. Klastering menggunakan K-Means

```
kmeans = KMeans(n_clusters=2)
pred = kmeans.fit_predict(d[d.columns])
t['K-means'], d['K-means'] = [pred, pred]
d[d.columns].sort_values(['K-means',
ColumnData.mortality, ColumnData.cases,
ColumnData.actives_cases, ColumnData.density],
ascending=False).style.background_gradient(
    cmap='YlGnBu', low=0, high=0.2)
```

Gambar 4.20 adalah *source code* untuk melakukan *clustering* dengan menggunakan K-Means dengan jumlah cluster sebanyak 2 (n_clusters=2) pada data COVID-19 Indonesia yang telah diolah sebelumnya. Selain itu, terdapat proses untuk menambahkan kolom baru bernama **'K-means'** pada data **t** dan **d**, yang berisi hasil prediksi klaster dari K-Means. Luaran dari *source code* tersebut adalah Gambar 4.29 yang menampilkan data **d** yang diurutkan berdasarkan klaster **'K-means', 'Mortality', 'Total Cases', 'Total Active Cases'**, dan **'Population Density'**. Pada Gambar 4.9 semakin gelap warna hijau pada sel data, semakin besar nilainya. Dari gambar tersebut, dapat dilihat bagaimana data terbagi menjadi dua kelompok (klaster) yang dihasilkan oleh K-Means.

Province	Total Cases	Total Recovered	Total Active Cases	Population Density	Total Deaths	Population	Mortality	K-means
Jawa Tengah	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1
Jawa Timur	1.000000	1.000000	1.000000	0.833333	1.000000	1.000000	1.000000	1
Sumatera Selatan	0.666667	0.666667	0.833333	0.333333	0.833333	0.833333	1.000000	1
Lampung	0.500000	0.500000	0.666667	0.833333	0.833333	0.833333	1.000000	1
Bali	0.833333	0.833333	0.833333	0.833333	0.833333	0.500000	0.833333	1
Riau	0.833333	0.833333	0.666667	0.333333	0.833333	0.833333	0.833333	1
Kalimantan Selatan	0.666667	0.666667	0.500000	0.500000	0.666667	0.500000	0.833333	1
Daerah Istimewa Yogyakarta	0.833333	0.833333	0.833333	1.000000	1.000000	0.500000	0.666667	1
Kalimantan Timur	0.833333	0.833333	0.666667	0.000000	0.833333	0.333333	0.666667	1
Sumatera Utara	0.833333	0.833333	0.833333	0.666667	0.666667	1.000000	0.333333	1
Sumatera Barat	0.666667	0.666667	0.666667	0.666667	0.666667	0.666667	0.333333	1
Sulawesi Selatan	0.666667	0.666667	0.500000	0.666667	0.666667	0.833333	0.166667	1
Nusa Tenggara Timur	0.666667	0.666667	0.333333	0.666667	0.333333	0.666667	0.166667	1
DKI Jakarta	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	0.000000	1
Jawa Barat	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	0.000000	1
Banten	1.000000	1.000000	1.000000	1.000000	0.666667	0.833333	0.000000	1



Aceh	0.166667	0.166667	0.333333	0.333333	0.500000	0.666667	1.000000	0
Sulawesi Tengah	0.333333	0.333333	0.333333	0.166667	0.500000	0.333333	0.833333	0
Gorontalo	0.000000	0.000000	0.000000	0.500000	0.000000	0.000000	0.833333	0
Kepulauan Riau	0.500000	0.500000	0.333333	0.833333	0.500000	0.166667	0.666667	0
Kalimantan Tengah	0.333333	0.333333	0.500000	0.000000	0.333333	0.166667	0.666667	0
Sulawesi Barat	0.000000	0.000000	0.000000	0.500000	0.000000	0.166667	0.666667	0
Kepulauan Bangka Belitung	0.500000	0.500000	0.166667	0.333333	0.500000	0.000000	0.500000	0
Sulawesi Utara	0.333333	0.333333	0.833333	0.666667	0.333333	0.333333	0.500000	0
Jambi	0.166667	0.166667	0.166667	0.333333	0.166667	0.333333	0.500000	0
Nusa Tenggara Barat	0.166667	0.166667	0.000000	0.833333	0.333333	0.666667	0.500000	0
Kalimantan Utara	0.333333	0.333333	0.000000	0.000000	0.166667	0.000000	0.333333	0
Sulawesi Tenggara	0.000000	0.000000	0.166667	0.166667	0.166667	0.333333	0.333333	0
Maluku Utara	0.000000	0.000000	0.000000	0.166667	0.000000	0.000000	0.333333	0
Kalimantan Barat	0.500000	0.500000	0.333333	0.166667	0.333333	0.666667	0.166667	0
Bengkulu	0.166667	0.166667	0.166667	0.500000	0.166667	0.166667	0.166667	0
Maluku	0.000000	0.000000	0.166667	0.166667	0.000000	0.166667	0.166667	0
Papua	0.333333	0.333333	0.666667	0.000000	0.166667	0.500000	0.000000	0
Papua Barat	0.166667	0.166667	0.500000	0.000000	0.000000	0.000000	0.000000	0

Gambar 4.9. Hasil klasterisasi data sebaran Covid-19 di Indonesia menggunakan K-Means

Berdasarkan hasil klasterisasi data sebaran Covid-19 di Indonesia menggunakan K-Means, didapatkan bahwa sebaran Covid-19 di Indonesia terbagi menjadi 2 klaster. Jika dipetakan menurut provinsi, maka dapat dibagi sebagai berikut:

a. **Klaster 0**, antara lain: Aceh, Sulawesi Tengah, Gorontalo, Kepulauan Riau, Kalimantan Tengah, Kepulauan Bangka Belitung, Sulawesi Utara, Jambi, Nusa Tenggara Barat, Kalimantan Utara, Sulawesi Tenggara, Maluku Utara, Kalimantan Barat, Bengkulu, Maluku, Papua, Papua Barat
b. **Klaster 1,** antara lain: Jawa Tengah, Jawa Timur, Sumatera Selatan, Lampung, Bali, Riau, Kalimantan Selatan, Daerah Istimewa Yogyakarta, Kalimantan Timur, Sumatera Utara, Sumatera Barat, Sulawesi Selatan, Nusa Tenggara Timur, DKI Jakarta, Jawa Barat, Banten

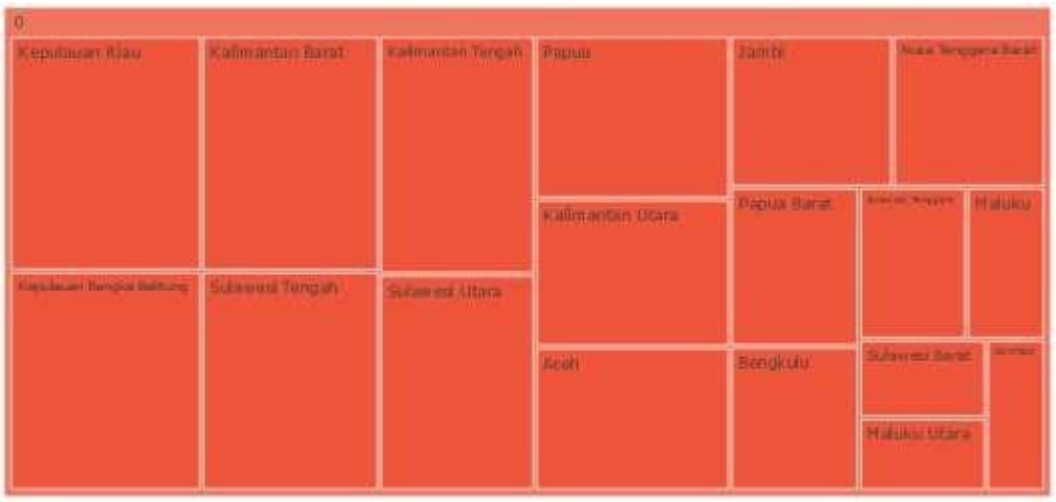
Dalam klasterisasi sebaran Covid-19 menggunakan algoritma K-Means, provinsi di Indonesia masuk dalam klaster yang sama jika atribut-atribut yang digunakan untuk klasterisasi memiliki kemiripan yang cukup tinggi di antara provinsi-provinsi tersebut sebagaimana ditampilkan pada Gambar 4.9. Kemiripan ini diukur berdasarkan jarak euclidean antara titik data. Misalnya, jika menggunakan atribut-atribut seperti **Total Cases, Total Recovered, Total Active Cases, Population Density, Total Deaths, Population**, dan **Mortality** sebagai fitur untuk klasterisasi, maka provinsi-provinsi yang memiliki nilai-nilai serupa untuk atribut-atribut ini cenderung masuk ke dalam klaster yang sama. Sebagai contoh, terdapat dua provinsi yang memiliki jumlah kasus Covid-19, angka kematian, jumlah kasus aktif, dan kepadatan penduduk yang hampir sama seperti Provinsi Jawa Timur dan Provinsi Jawa Tengah, algoritma K-Means menganggap kedua provinsi tersebut memiliki kemiripan yang tinggi dan akan menempatkannya dalam klaster yang sama

Tabel 4.21. *Source code treemap* untuk visualisasi hasil klasterisasi menggunakan K-Means

```
vis_tmap = px.treemap(t.reset_index(), path=['K-
means', ColumnData.province],
values=ColumnData.cases)
vis_tmap.update_layout(title='K-means clusters')
vis_tmap.show()
```

Source code pada Tabel 4.21 digunakan untuk membuat visualisasi *Treemap* menggunakan library *Plotly Express*. *Treemap* adalah jenis grafik yang menampilkan hierarki data dengan menggunakan persegi panjang dan subpersegi panjang untuk merepresentasikan setiap kategori. Visualisasi treemap

menunjukkan jumlah kasus COVID-19 untuk setiap provinsi di Indonesia yang diklasifikasikan ke dalam dua cluster hasil dari algoritma K-means sebelumnya. Pada axis **horizontal (x)**, kita dapat melihat kedua klaster, yaitu 0 dan 1, sedangkan pada axis **vertical (y)**, kita dapat melihat nama-nama provinsi. Ukuran persegi panjang menunjukkan jumlah kasus COVID-19 di masing-masing provinsi. Luaran dari *source code* ini adalah sebuah visualisasi *treemap* yang menunjukkan dua klaster hasil dari algoritma K-means yang ditampilkan pada Gambar 4.10. Setiap persegi panjang pada *treemap* menunjukkan jumlah kasus COVID-19 untuk setiap provinsi di Indonesia.



Gambar 4.10. Visualisasi hasil klasterisasi K-Means menggunakan *treemap*

Tabel 4.22. *Source code treemap* untuk visualisasi *mortality* berdasarkan hasil klasterisasi dengan algoritma K-means

```
vis_tmap = px.treemap(t.reset_index(), path=['K-
means', ColumnData.province],
values=ColumnData.mortality)
vis_tmap.update_layout(title='K-means clusters')
vis_tmap.show()
```

Tabel 4.22 menunjukkan *source code* yang digunakan untuk membuat visualisasi treemap yang menunjukkan nilai kematian (*mortality*) pada setiap provinsi di Indonesia berdasarkan hasil klasterisasi dengan algoritma K-means. Luarannya berupa treemap yang terdiri dari beberapa kotak berukuran berbeda, masing-masing mewakili satu provinsi di Indonesia, yang dikategorikan berdasarkan klaster K-means yang telah dihasilkan. Ukuran kotak menunjukkan nilai kematian (*mortality*) pada setiap provinsi, sehingga semakin besar kotaknya, semakin tinggi angka kematian di provinsi tersebut. Luaran ditunjukkan pada Gambar 4.11.

Gambar 4.11. Visualisasi *mortality* berdasarkan hasil klasterisasi dengan algoritma K-means

Tabel 4.23. *Source code* visualisasi untuk menampilkan jumlah kasus Covid-19 di setiap provinsi di Indonesia

```
c = t.sort_values(['K-means', ColumnData.cases],
ascending=False)
data = [go.Bar(x=c[(c['K-means'] == i)].index,
y=c[(c['K-means'] == i)][ColumnData.cases],
               text=c[(c['K-means'] ==
i)][ColumnData.cases], name=i) for i in range(0, 6)]

vis_bar = go.Figure(data=data)
vis_bar.update_layout(title='K-means Clustering:
number of cases by cluster',
                      xaxis_title='Indonesia State',
yaxis_title='Deaths per case')
vis_bar.show()
```

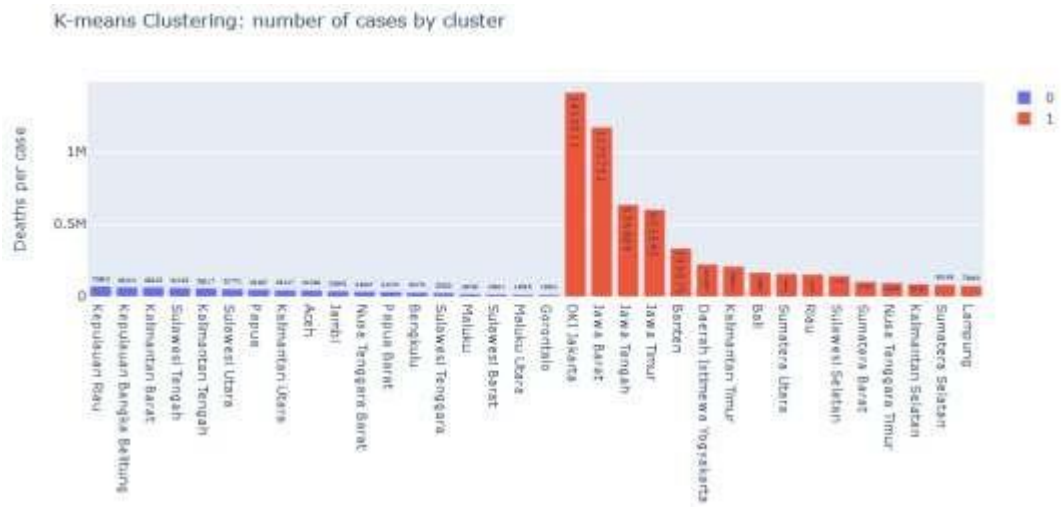
Source code pada Tabel 4.23 digunakan untuk membuat visualisasi yang menunjukkan jumlah kasus Covid-19 di setiap provinsi di Indonesia yang diurutkan berdasarkan klaster hasil klaster K-means. Di bawah ini adalah penjelasan baris demi baris dari *source code*:

1. **c = t.sort_values(['K-means', ColumnData.cases], ascending=False):** Kode ini mengurutkan dataframe t berdasarkan kolom **'K-means'** dan **'cases'** secara menurun (*descending*) dan hasilnya disimpan dalam variabel **c**. Pengurutan ini bertujuan untuk mempersiapkan data agar dapat diplot dalam bentuk bar chart dengan urutan yang sesuai.
2. **data = [go.Bar(x=c[(c['K-means'] == i)].index, y=c[(c['K-means'] == i)][ColumnData.cases], text=c[(c['K-means'] == i)][ColumnData.cases], name=i) for i in range(0, 6)]**: Kode ini melakukan iterasi dari 0 hingga 5 (6 angka), dan untuk setiap iterasi, data yang sesuai dengan nilai **'K-means'** diambil dari *dataframe* **c**. Kemudian, nilai indeks dari data tersebut digunakan sebagai nilai sumbu-x dalam bar chart, nilai **'cases**' digunakan sebagai nilai sumbu-y, dan juga sebagai teks yang ditampilkan di atas setiap bar. Setiap iterasi menghasilkan sebuah objek go.Bar yang direpresentasikan dalam format list, dan keseluruhan list tersebut disimpan dalam variabel data.
3. **vis_bar = go.Figure(data=data)**: Kode ini membuat objek **go.Figure** yang akan digunakan untuk menyusun visualisasi data. Data yang digunakan dalam visualisasi diambil dari variabel data yang sudah dibuat sebelumnya.
4. **vis_bar.update_layout(title='K-means Clustering: number of cases by cluster', xaxis_title='Indonesia State', yaxis_title='Deaths per case'):** Kode

ini mengatur layout (tata letak) dari visualisasi. Judul visualisasi adalah "**K-means Clustering: number of cases by cluster**", sumbu-x diberi label "**Indonesia State**", dan sumbu-y diberi label "**Deaths per case**".

5. **vis_bar.show():** Kode ini menampilkan visualisasi yang sudah dibuat menggunakan objek **vis_bar**.

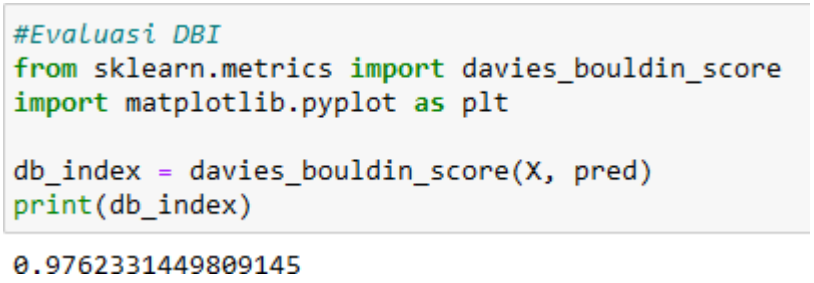
Pada hasil outputnya, terdapat *bar chart* yang menampilkan jumlah kasus Covid-19 di setiap provinsi di Indonesia, yang diurutkan berdasarkan cluster hasil dari K-means *clustering*. Setiap *cluster* ditampilkan dengan warna yang berbeda pada plot bar tersebut. Pada sumbu x terdapat daftar provinsi di Indonesia, sedangkan pada sumbu y terdapat jumlah kasus Covid-19. Terdapat juga nilai jumlah kasus yang ditampilkan pada setiap bar pada plot tersebut, sebagaimana ditampilkan pada Gambar 4.12.



Gambar 4.12. Visualisasi jumlah kasus Covid-19 di setiap provinsi di Indonesia

Tabel 4.24. *Source code* evaluasi DBI untuk hasil klasterisasi menggunakan K-Means

```
#Evaluasi DBI
from sklearn.metrics import davies_bouldin_score
import matplotlib.pyplot as plt

db_index = davies_bouldin_score(X, pred)
print(db_index)
```



Gambar 4.13. Hasil evaluasi DBI untuk hasil klasterisasi menggunakan K-Means

Hasil evaluasi model *clustering* dengan menggunakan metode Davies-Bouldin Index (DBI) diperoleh sebagaimana pada Gambar 4.13 dengan menggunakan *source code* pada Tabel 4.24. DBI digunakan untuk mengukur kualitas pengelompokan data pada metode *clustering*. Semakin kecil nilai DBI, semakin baik pengelompokan data yang terjadi. Pada hasil output, didapatkan nilai DBI sebesar 0.9762331449809145. Nilai ini menunjukkan kualitas pengelompokan data yang baik karena mendekati 0. Semakin mendekati 0, semakin baik hasil pengelompokan data. Oleh karena itu, dapat dikatakan bahwa model *clustering* yang digunakan pada *source code* tersebut telah memberikan hasil yang baik dalam melakukan klasterisasi data sebaran Covid-19 di Indonesia.

4.6. Analisis Algoritma K-Medoids

Tabel 4.25. Seleksi fitur pada proses modelling K-Medoids

```
# Modelling
# Seleksi Fitur
X = d[['Mortality', 'Total Cases','Total Active
Cases', 'Population Density', 'Population', 'Total
Deaths']]
```

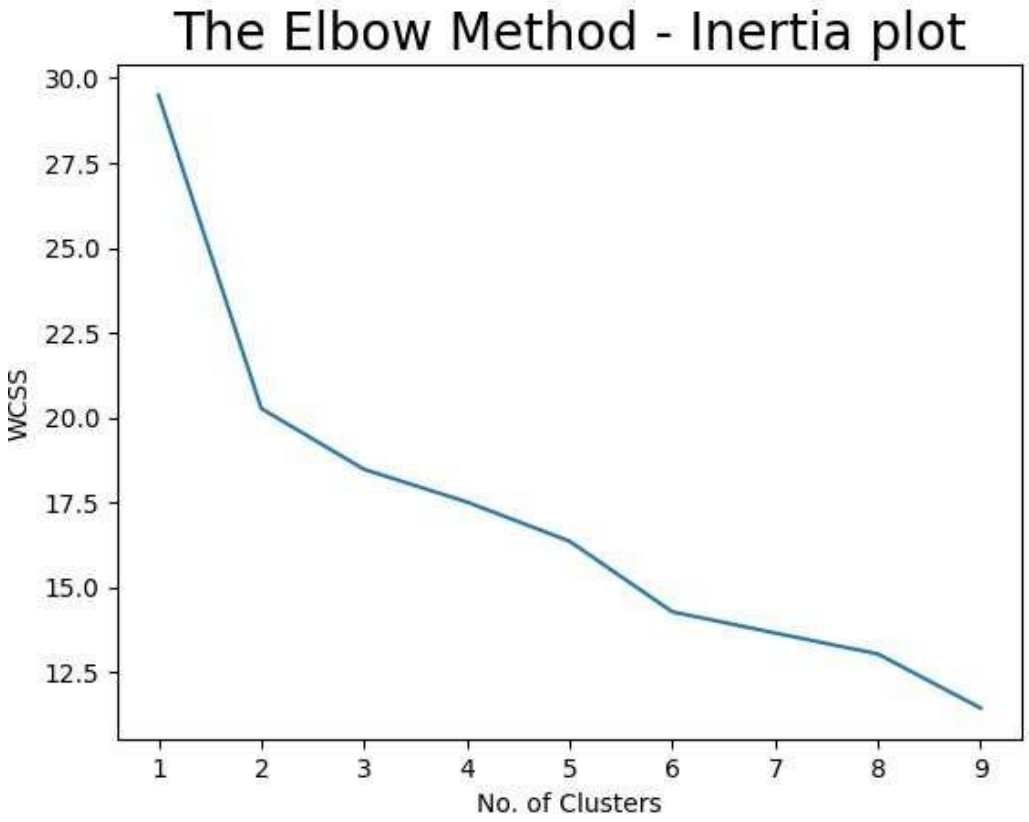
Tabel 4.25 merupakan *source code* untuk tahap seleksi fitur pada proses modelling dengan menggunakan dataset Covid-19 di Indonesia. Fitur-fitur yang dipilih untuk dilibatkan dalam model ini adalah **Mortality, Total Cases, Total Active Cases, Population Density, Population**, dan **Total Deaths**. Fitur-fitur tersebut dipilih karena mempengaruhi perkembangan kasus Covid-19 di Indonesia. Dalam *source code* tersebut, variabel X diisi dengan data dari fitur-fitur tersebut yang sudah dipilih. Data fitur-fitur tersebut digunakan sebagai input dalam model K-Medoids.

Tabel 4.26. Metode Elbow untuk penentuan jumlah klaster optimal pada K-Means

```
from sklearn_extra.cluster import KMedoids
import matplotlib.pyplot as plt

#Elbow Method - Inertia plot
inertia = []
#looping the inertia calculation for each k
for k in range(1, 10):
    #Assign KMeans as cluster_model
    cluster_model    =    KMedoids(n_clusters    =    k,
random_state = 24)
    #Fit cluster_model to X
    cluster_model.fit(X)
    #Get the inertia value
    inertia_value = cluster_model.inertia_
    #Append the inertia_value to inertia list
    inertia.append(inertia_value)
##Inertia plot
```

```
plt.plot(range(1, 10), inertia)
plt.title('The Elbow Method - Inertia plot', fontsize
= 20)
plt.xlabel('No. of Clusters')
plt.ylabel('WCSS')
plt.show()
```

Tabel 4.26 merupakan implementasi dari metode elbow untuk menentukan jumlah klaster yang optimal pada data sebaran Covid-19. Elbow method adalah merupakan teknik untuk menentukan jumlah klaster optimal saat pengelompokkan data dengan algoritma K-Medoids. Pada *source code* tersebut, dilakukan looping dari k=1 hingga k=9, dimana pada setiap iterasi dibuat objek KMedoids dengan jumlah klaster k yang bersesuaian. Kemudian objek tersebut dilatih dengan fitur X dan dihitung nilai inertia (*Within Cluster Sum of Squares*/WCSS) nya. Nilai inertia tersebut kemudian di-append ke dalam list inertia. Setelah proses loop selesai, nilai inertia diplot pada grafik Inertia plot dengan sumbu-x merepresentasikan jumlah klaster, dan sumbu-y merepresentasikan nilai inertia. Plot ini bertujuan untuk menentukan nilai k yang optimal, yaitu ketika penurunan nilai inertia mulai mengalami perubahan yang lebih lambat (bentuk seperti siku), sehingga sering disebut juga sebagai "elbow point" yang ditampilkan pada Gambar 4.14.



Gambar 4.14. Grafik hasil metode Elbow untuk penentuan jumlah klaster optimal pada K-Medoids

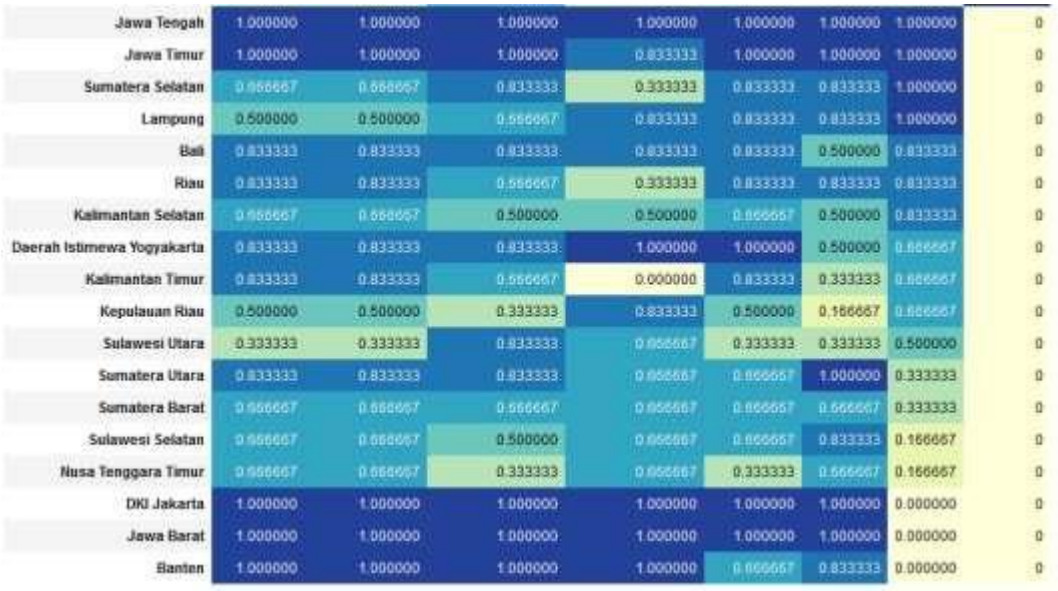
Berdasarkan Gambar 4.14, penentuan nilai k yang optimal didapatkan ketika penurunan nilai inertia mulai mengalami perubahan yang lebih lambat, sehingga k yang optimal didapatkan pada nilai k = 2. Nilai k tersebut digunakan dalam proses selanjutnya, yakni modelling algoritma K-Medoids.

Tabel 4.27. Klastering menggunakan K-Medoids

```
kmedoids = KMedoids(n_clusters=2)
pred = kmedoids.fit_predict(d[d.columns])
t['K-medoids'], d['K-medoids'] = [pred, pred]
d[d.columns].sort_values(['K-medoids',
ColumnData.mortality,                ColumnData.cases,
ColumnData.actives_cases,          ColumnData.density],
ascending=False).style.background_gradient(
    cmap='YlGnBu', low=0, high=0.2)
```

Tabel 4.27 adalah *source code* untuk melakukan *clustering* dengan menggunakan K-Means dengan jumlah cluster sebanyak 2 (n_clusters=2) pada data COVID-19 Indonesia yang telah diolah sebelumnya. Selain itu, terdapat proses untuk menambahkan kolom baru bernama **'K-medoids'** pada data **t** dan **d**, yang berisi hasil prediksi klaster dari K-Medoids. Luaran dari *source code* tersebut adalah Gambar 4.15 yang menampilkan data **d** yang diurutkan berdasarkan klaster **'K-means', 'Mortality', 'Total Cases', 'Total Active Cases'**, dan **'Population Density'**. Pada Gambar 4.15 semakin gelap warna pada sel data, semakin besar nilainya. Dari gambar tersebut, dapat dilihat bagaimana data terbagi menjadi dua kelompok (klaster) yang dihasilkan oleh K-Medoids.

Province	Total Cases	Total Recovered	Total Active Cases	Population Density	Total Deaths	Population	Mortality	K-medoids
Aceh	0.166667	0.166667	0.333333	0.333333	0.500000	0.666667	1.000000	1
Sulawesi Tengah	0.333333	0.333333	0.333333	0.166667	0.500000	0.333333	0.833333	1
Gorontalo	0.000000	0.000000	0.000000	0.500000	0.000000	0.000000	0.833333	1
Kalimantan Tengah	0.333333	0.333333	0.500000	0.000000	0.333333	0.166667	0.666667	1
Sulawesi Barat	0.000000	0.000000	0.000000	0.500000	0.000000	0.166667	0.666667	1
Kepulauan Bangka Belitung	0.500000	0.500000	0.166667	0.333333	0.500000	0.000000	0.500000	1
Jambi	0.166667	0.166667	0.166667	0.333333	0.166667	0.333333	0.500000	1
Nusa Tenggara Barat	0.166667	0.166667	0.000000	0.833333	0.333333	0.666667	0.500000	1
Kalimantan Utara	0.333333	0.333333	0.000000	0.000000	0.166667	0.000000	0.333333	1
Sulawesi Tenggara	0.000000	0.000000	0.166667	0.166667	0.166667	0.333333	0.333333	1
Maluku Utara	0.000000	0.000000	0.000000	0.166667	0.000000	0.000000	0.333333	1
Kalimantan Barat	0.500000	0.500000	0.333333	0.166667	0.333333	0.666667	0.166667	1
Bengkulu	0.166667	0.166667	0.166667	0.500000	0.166667	0.166667	0.166667	1
Maluku	0.000000	0.000000	0.166667	0.166667	0.000000	0.166667	0.166667	1
Papua	0.333333	0.333333	0.666667	0.000000	0.166667	0.500000	0.000000	1
Papua Barat	0.166667	0.166667	0.500000	0.000000	0.000000	0.000000	0.000000	1



Gambar 4.15. Hasil klasterisasi data sebaran Covid-19 di Indonesia menggunakan K-Medoids

Berdasarkan hasil klasterisasi data sebaran Covid-19 di Indonesia menggunakan K-Medoids, didapatkan bahwa sebaran Covid-19 di Indonesia terbagi menjadi 2 klaster. Dalam klasterisasi, provinsi di Indonesia masuk dalam klaster yang sama jika atribut-atribut yang digunakan untuk klasterisasi memiliki kemiripan yang cukup tinggi di antara provinsi-provinsi tersebut sebagaimana ditampilkan pada Gambar 4.15. Kemiripan ini diukur berdasarkan jarak euclidean antara titik data. Misalnya, jika menggunakan atribut-atribut seperti **Total Cases, Total Recovered, Total Active Cases, Population Density, Total Deaths, Population**, dan **Mortality** sebagai fitur untuk klasterisasi, maka provinsi-provinsi yang memiliki nilai-nilai serupa untuk atribut-atribut ini cenderung masuk ke dalam klaster yang sama. Sebagai contoh, terdapat dua provinsi yang memiliki jumlah kasus Covid-19, angka kematian, jumlah kasus aktif, dan kepadatan penduduk yang hampir sama seperti Provinsi Jawa Timur dan Provinsi Jawa Tengah, algoritma K-

Medoids menganggap kedua provinsi tersebut memiliki kemiripan yang tinggi dan akan menempatkannya dalam klaster yang sama

Source code pada Tabel 4.28 digunakan untuk membuat visualisasi *Treemap* menggunakan library *Plotly Express*. *Treemap* adalah jenis grafik yang menampilkan hierarki data dengan menggunakan persegi panjang dan subpersegi panjang untuk merepresentasikan setiap kategori. Visualisasi treemap menunjukkan jumlah kasus COVID-19 untuk setiap provinsi di Indonesia yang diklasifikasikan ke dalam dua cluster hasil dari algoritma K-medoids sebelumnya. Pada axis **horizontal (x)**, kita dapat melihat kedua klaster, yaitu 0 dan 1, sedangkan pada axis **vertical (y)**, kita dapat melihat nama-nama provinsi. Ukuran persegi panjang menunjukkan jumlah kasus COVID-19 di masing-masing provinsi. Luaran dari *source code* ini adalah sebuah visualisasi *treemap* yang menunjukkan dua klaster hasil dari algoritma K-medoids yang ditampilkan pada Gambar 4.16. Setiap persegi panjang pada *treemap* menunjukkan jumlah kasus COVID-19 untuk setiap provinsi di Indonesia.

Tabel 4.28. *Source code treemap* untuk visualisasi hasil klasterisasi menggunakan K-Medoids

```
vis_tmap = px.treemap(t.reset_index(), path=['K-
medoids', ColumnData.province],
values=ColumnData.cases)
vis_tmap.update_layout(title='K-medoids clusters')
vis_tmap.show()
```

Gambar 4.16. Visualisasi hasil klasterisasi K-Medoids menggunakan *treemap*

Tabel 4.29. *Source code treemap* untuk visualisasi *mortality* berdasarkan hasil klasterisasi dengan algoritma K-medoids

```
vis_tmap = px.treemap(t.reset_index(), path=['K-
medoids', ColumnData.province],
values=ColumnData.mortality)
vis_tmap.update_layout(title='K-medoids clusters')
vis_tmap.show()
```

Tabel 4.29 menunjukkan *source code* yang digunakan untuk membuat visualisasi treemap yang menunjukkan nilai kematian (*mortality*) pada setiap

provinsi di Indonesia berdasarkan hasil klasterisasi dengan algoritma K-means. Luarannya berupa treemap yang terdiri dari beberapa kotak berukuran berbeda, masing-masing mewakili satu provinsi di Indonesia, yang dikategorikan berdasarkan klaster K-medoids yang telah dihasilkan. Ukuran kotak menunjukkan nilai kematian (*mortality*) pada setiap provinsi, sehingga semakin besar kotaknya, semakin tinggi angka kematian di provinsi tersebut. Luaran ditunjukkan pada Gambar 4.17.

Gambar 4.17. Visualisasi *mortality* berdasarkan hasil klasterisasi dengan algoritma K-medoids

Tabel 4.30. *Source code* visualisasi untuk menampilkan jumlah kasus Covid-19 di setiap provinsi di Indonesia

```
c = t.sort_values(['K-medoids', ColumnData.cases],
ascending=False)
data = [go.Bar(x=c[(c['K-medoids'] == i)].index,
y=c[(c['K-medoids'] == i)][ColumnData.cases],
                text=c[(c['K-medoids'] ==
i)][ColumnData.cases], name=i) for i in range(0, 6)]

vis_bar = go.Figure(data=data)
vis_bar.update_layout(title='K-medoids Clustering:
number of cases by cluster',
                      xaxis_title='Indonesia State',
yaxis_title='Deaths per case')
vis_bar.show()
```

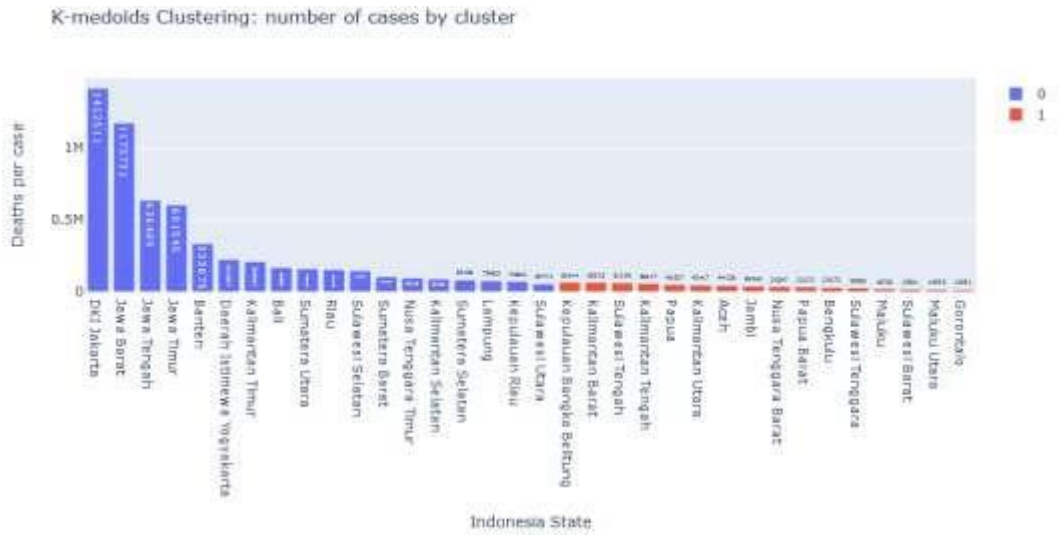
Source code pada Tabel 4.30 digunakan untuk membuat visualisasi untuk menampilkan jumlah kasus Covid-19 di setiap provinsi di Indonesia, yang diurutkan berdasarkan *cluster* hasil dari K-means *clustering*. Berikut adalah penjelasan baris per baris dari source code tersebut:

1. **c = t.sort_values(['K-medoids', ColumnData.cases], ascending=False):** Kode ini mengurutkan dataframe t berdasarkan kolom '**K-medoids**' dan '**cases**' secara menurun (*descending*) dan hasilnya disimpan dalam variabel **c**. Tujuan dari pengurutan ini adalah untuk mempersiapkan data agar dapat diplot dalam bentuk bar chart dengan urutan yang sesuai.
2. **data = [go.Bar(x=c[(c['K-medoids'] == i)].index, y=c[(c[' K-medoids'] == i)][ColumnData.cases], text=c[(c['K-medoids'] == i)][ColumnData.cases], name=i) for i in range(0, 6)]**: Kode ini melakukan iterasi dari 0 hingga 5 (6 angka), dan untuk setiap iterasi, data yang sesuai dengan nilai '**K-medoids**' diambil dari *dataframe* **c**. Kemudian, nilai indeks dari data tersebut digunakan sebagai nilai sumbu-x dalam bar chart, nilai '**cases**' digunakan sebagai nilai

sumbu-y, dan juga sebagai teks yang ditampilkan di atas setiap bar. Setiap iterasi menghasilkan sebuah objek go.Bar yang direpresentasikan dalam format list, dan keseluruhan list tersebut disimpan dalam variabel data.

3. **vis_bar = go.Figure(data=data)**: Kode ini membuat objek **go.Figure** yang akan digunakan untuk menyusun visualisasi data. Data yang digunakan dalam visualisasi diambil dari variabel data yang sudah dibuat sebelumnya.
4. **vis_bar.update_layout(title='K-medoids Clustering: number of cases by cluster', xaxis_title='Indonesia State', yaxis_title='Deaths per case'):** Kode ini mengatur layout (tata letak) dari visualisasi. Judul visualisasi adalah "**K-medoids Clustering: number of cases by cluster**", sumbu-x diberi label "**Indonesia State"**, dan sumbu-y diberi label "**Deaths per case**".
5. **vis_bar.show():** Kode ini menampilkan visualisasi yang sudah dibuat menggunakan objek **vis_bar**.

Pada hasil outputnya, terdapat *bar chart* yang menampilkan jumlah kasus Covid-19 di setiap provinsi di Indonesia, yang diurutkan berdasarkan cluster hasil dari K-medoids *clustering*. Setiap *cluster* ditampilkan dengan warna yang berbeda pada plot bar tersebut. Pada sumbu x terdapat daftar provinsi di Indonesia, sedangkan pada sumbu y terdapat jumlah kasus Covid-19. Terdapat juga nilai jumlah kasus yang ditampilkan pada setiap bar pada plot tersebut, sebagaimana ditampilkan pada Gambar 4.18.



Gambar 4.18. Visualisasi jumlah kasus Covid-19 di setiap provinsi di Indonesia menggunakan K-Medois

Tabel 4.31. *Source code* evaluasi DBI untuk hasil klasterisasi menggunakan K-Medoids

```
#Evaluasi DBI K-medoids
from sklearn.metrics import davies_bouldin_score
import matplotlib.pyplot as plt

db_index = davies_bouldin_score(X, pred)
print(db_index)
```

```
#Evaluasi DBI
from sklearn.metrics import davies_bouldin_score
import matplotlib.pyplot as plt

db_index = davies_bouldin_score(X, pred)
print(db_index)
```

```
0.9809235412405508
```

Gambar 4.19. Hasil evaluasi DBI untuk hasil klasterisasi menggunakan K-Medoids

Evaluasi model K-Medoids dengan menggunakan metode Davies-Bouldin Index (DBI) sebagaimana *source code* pada Tabel 4.31. DBI digunakan untuk mengukur kualitas pengelompokan data pada metode *clustering*. Semakin kecil nilai DBI, semakin baik pengelompokan data yang terjadi. Pada hasil output, didapatkan nilai DBI sebesar 0.9809235412405508 seperti ditunjukkan pada Gambar 4.19. Nilai ini menunjukkan kualitas pengelompokan data yang baik karena mendekati 0. Semakin mendekati 0, semakin baik hasil pengelompokan data. Dapat dikatakan bahwa model *clustering* yang digunakan pada *source code* tersebut telah memberikan hasil yang baik dalam melakukan klasterisasi data sebaran Covid-19 di Indonesia.

4.7 Komparasi Kinerja K-Means dan K-Medoids

DBI (Davies-Bouldin Index) adalah metrik evaluasi yang digunakan untuk mengukur kualitas partisi dalam *clustering*. Semakin rendah nilai DBI, semakin baik partisi yang dihasilkan oleh model. Dalam penelitian ini, K-Means dan K-Medoids adalah dua metode *clustering* yang digunakan untuk mempartisi data sebaran Covid-19 di Indonesia. Setelah dilakukan evaluasi DBI, K-Means mendapatkan nilai 0.9762331449809145, sedangkan K-Medoids mendapatkan nilai 0.9809235412405508. Karena K-Means memiliki nilai DBI yang lebih rendah dibandingkan K-medoids, maka dapat dikatakan K-Means menghasilkan klasterisasi yang lebih baik dalam klasterisasi data sebaran Covid-19 di Indonesia.

4.8. Konsep Clustering Perspektif al-Qur’an

Manusia adalah salah satu dari sekian banyak makhluk ciptaan Allah. Beberapa peneliti mengklaim bahwa ada kesamaan antara manusia dan hewan. Namun yang membedakan seseorang adalah kemampuan bernalar, yang seharusnya digunakan sesuai dengan kemampuan seseorang. Manusia memiliki kearifan yang bersumber dari hati nurani manusia dan perilaku yang saling menghormati dan berpegang pada standar kesusilaan manusia. Oleh karena itu, manusia sangat berbeda dengan hewan dan makhluk ciptaan Tuhan lainnya.

Allah membagi konsep manusia ke dalam beberapa kelompok, baik secara tersurat maupun tersirat. Terdapat tiga kata yang umum digunakan dalam Al-Qur'an untuk merujuk pada manusia dan segala bentuknya, yaitu al-insan, al-basyar, dan bani Adam atau zurriyat Adam.

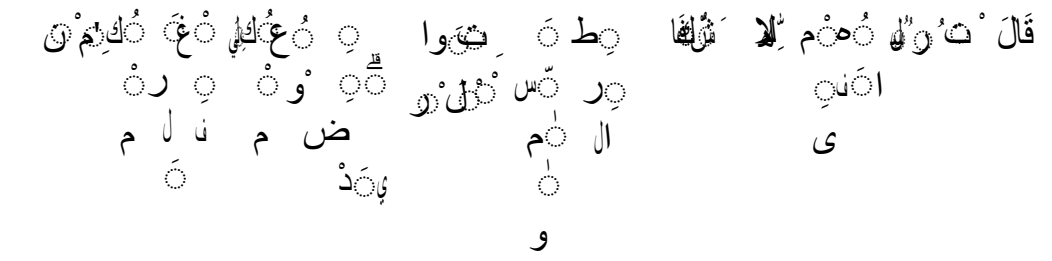
1. Konsep Al Insan

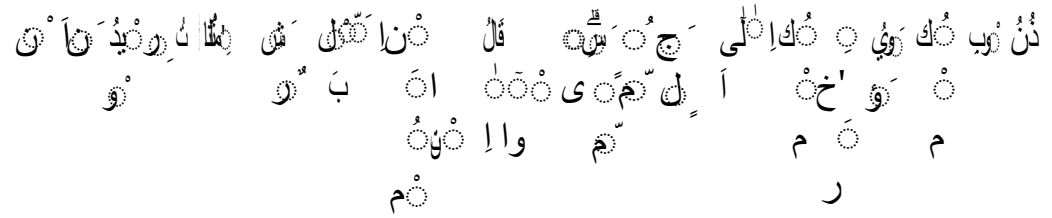
Dalam Al-Qur'an, kata "al-insan" berasal dari kata "al-uns" dan disebutkan sebanyak 73 kali yang tersebar dalam 43 surah. Secara etimologi, kata "al-insan" dapat diartikan sebagai lemah lembut, harmonis, tampak, dan pelupa. Kata tersebut berasal dari akar kata "naus" yang memiliki arti pergerakan atau dinamis. Dari asal katanya, dapat dipahami bahwa manusia pada dasarnya memiliki potensi positif untuk tumbuh dan berkembang baik secara fisik maupun mental-spiritual. Selain itu, manusia juga diberi potensi lain oleh Allah yang dapat mendorongnya ke arah tindakan, sikap, dan perilaku negatif yang merugikan. Penggunaan kata "al-insan" oleh Al-Qur'an menunjukkan bahwa manusia adalah makhluk yang telah sempurna baik secara jasmani maupun rohani. Hubungan antara aspek jasmani dan rohani ini, bersama

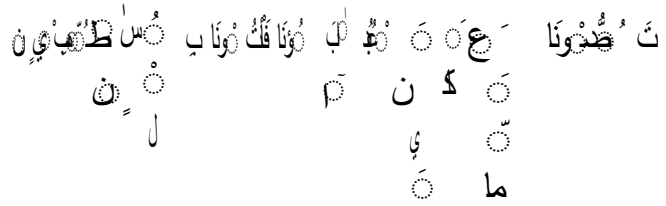
dengan potensi yang dimiliki, menjadikan manusia sebagai ciptaan Allah yang unik dan berbeda antara satu dengan yang lain, istimewa dan sempurna sebagai individu dan makhluk sosial. Sebagai manusia, kita mampu menerima tanggung jawab sebagai khalifah di bumi.

2. Konsep Al Basyar

Dalam Al-Qur'an, kata "al-basyar" disebutkan sebanyak 36 kali dalam 26 surat. Secara etimologi, “al-basyar” berarti “mulamasah”, yang mengacu pada kontak kulit antara laki-laki dan perempuan. Hal ini menunjukkan bahwa manusia adalah makhluk hidup dengan kebutuhan manusia yang terbatas, seperti makan dan minum, berhubungan seks, rasa aman, bahagia, dan lain-lain. Penggunaan kata “al-basyar” dalam Al-Qur’an merujuk kepada semua orang tanpa kecuali, baik yang beriman maupun tidak, termasuk Rasul- Nya. Dalam konsep “Al-Basyar” manusia tidak berbeda dengan makhluk biologis lainnya. Artinya dalam hidup manusia terikat oleh prinsip-prinsip kehidupan biologis, seperti keinginan untuk bereproduksi, masa tumbuh kembang dari anak hingga dewasa, serta perkembangan dan kematangan berpikir logis. Konsep manusia sebagai “al-basyar” banyak dijelaskan dalam Al-Qur’an, termasuk dalam QS. Ibrahim ayat 10.







Artinya: *Berkata rasul-rasul mereka "Apakah ada keragu-raguan terhadap Allah, Pencipta langit dan bumi? Dia menyeru kamu untuk memberi ampunan kepadamu dari dosa-dosamu dan menangguhkan (siksaan)mu sampai masa yang ditentukan?" Mereka berkata: "Kamu tidak lain hanyalah manusia seperti kami juga. Kamu menghendaki untuk menghalang-halangi (membelokkan) kami dari apa yang selalu disembah nenek moyang kami, karena itu datangkanlah kepada kami, bukti yang nyata*"

3. Konsep Bani Adam

Penggunaan istilah "Bani Adam" di dalam Al-Quran lebih menekankan pada peringatan kepada umat manusia agar selalu bersyukur atas segala nikmat yang diberikan oleh Allah. Ini termasuk nikmat pemberian kehidupan yang mulia, hasil alam yang baik di darat dan di laut, pemberian rezeki, dan kedudukan yang tinggi di antara makhluk ciptaan Allah lainnya. Hal ini dinyatakan dalam QS. Al-Isra' ayat 70.

Artinya: *"Dan sungguh, Kami telah memuliakan anak cucu Adam, dan Kami angkut mereka di darat dan di laut, dan Kami beri mereka rezeki dari yang baik-baik dan Kami lebihkan mereka di atas banyak makhluk yang Kami ciptakan dengan kelebihan yang sempurna."*

Menurut Thabathaba'i dalam kitab Samsul Nizar, penggunaan kata "Bani Adam" yang merujuk pada manusia secara umum memiliki tiga aspek yang perlu dipahami, yaitu:

a. Anjuran untuk berbudaya sesuai dengan ketentuan Allah, misalnya dengan berpakaian sopan untuk menutupi aurat. Hal ini menunjukkan pentingnya manusia menjalankan budaya yang sesuai dengan ajaran Allah.
b. Pengingat kepada seluruh keturunan Nabi Adam agar tidak terjerumus dalam godaan setan yang mengajak kepada keingkaran dan penolakan terhadap Allah. Hal ini bertujuan untuk menjaga umat manusia agar tidak tersesat dan tetap berpegang pada iman yang benar.
c. Manusia harus menggunakan segala sesuatu di alam semesta ini untuk menyembah dan menyembah Allah SWT. Ini mengingatkan orang untuk menggunakan semua sumber daya dan karunia dunia ini untuk mendekatkan diri kepada Tuhan dan beribadah.

Dalam keseluruhan konteks, penggunaan kata "Bani Adam" dalam Al-Qur'an memiliki makna yang mencakup aspek budaya, peringatan akan godaan setan, serta pentingnya memanfaatkan sumber daya alam untuk beribadah kepada Allah.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penelitian ini memiliki kesimpulan yang menjawab rumusan masalah. Penelitian ini membandingkan evaluasi algoritma K-Means dan K-Medoids dalam menganalisis sebaran Covid-19 di Indonesia menggunakan metode evalusi Davies-Bouldin Index. Langkah penelitian dimulai dengan melakukan *pre-processing* berupa proses binning data hingga normalisasi data. Selanjutnya, menampilkan visualisasi data sebaran Covid-19. Dalam melakukan modeling Algoritma K-Means dan K-Medoids, penentuan nilai k sebagai jumlah klaster yang optimal didapatkan menggunakan Metode Elbow. Hal ini ditandai ketika penurunan nilai inertia mulai mengalami perubahan yang lebih lambat, sehingga k yang optimal didapatkan pada nilai k = 2. Nilai k tersebut digunakan dalam proses selanjutnya, yakni modelling algoritma K-Means dan K-Medoids. Dalam klasterisasi sebaran Covid-19 menggunakan algoritma K-Means dan K-Medoids, provinsi di Indonesia masuk dalam klaster yang sama jika atribut-atribut yang digunakan untuk klasterisasi memiliki kemiripan yang cukup tinggi di antara provinsi-provinsi tersebut. Hasilnya, semakin rendah nilai DBI, semakin baik partisi yang dihasilkan oleh model. Dalam penelitian ini, K-Means dan K-Medoids adalah dua metode *clustering* yang digunakan untuk mempartisi data sebaran Covid-19 di Indonesia. Setelah dilakukan evaluasi DBI, K-Means mendapatkan nilai 0.9762331449809145, sedangkan K-Medoids mendapatkan nilai 0.9809235412405508. Karena K-Means memiliki nilai DBI yang lebih rendah

dibandingkan K-medoids, maka dapat disimpulkan bahwa K-Means menghasilkan klasterisasi yang lebih baik dalam klasterisasi data sebaran Covid-19 di Indonesia.

5.2 Saran

Terdapat beberapa saran penelitian yang dapat berikan oleh peneliti untuk penelitian berikutnya, antara lain:

1. Peneliti menyarankan untuk melakukan *update* data ketika melanjutkan penelilian ini. Hal ini disebabkan karena data yang digunakan adalah data *time-series* yang dapat diakses di sumber data yang sama sebagaimana sumber penelitian yang dijelaskan dalam penelitian ini.
2. Penyajian data dan visualisasi data dapat memberikan kontribusi luas bagi masyarakat jika *platform* yang digunakan berbasiskan web yang terintegrasi secara *real-time* dengan sumber data.

DAFTAR PUSTAKA

Abdullah, D., Susilo, S., Ahmar, A. S., Rusli, R., & Hidayat, R. (2022). The application of K-means *clustering* for province *clustering* in Indonesia of the risk of the COVID-19 pandemic based on COVID-19 data. *Quality and Quantity*, *56*(3), 1283–1291. https://doi.org/10.1007/s11135-021-01176-w

Adha, R., Nurhaliza, N., Sholeha, U., & Mustakim, M. (2021). Perbandingan Algoritma DBSCAN dan K-Means *Clustering* untuk Pengelompokan Kasus Covid-19 di Dunia. *SITEKIN: Jurnal Sains, Teknologi Dan Industri*, *18*(2), 206–211.

Andini, A. D., & Arifin, T. (2020). Implementasi Algoritma K-Medoids Untuk Klasterisasi Data Penyakit Pasien Di Rsud Kota Bandung. *JURNAL RESPONSIF: Riset Sains & …*, *2*(2), 128–138.

Annur, C. M. (2022). *Tingkat Kematian Akibat Covid-19 di Indonesia Capai 2,58%, Peringkat Berapa di ASEAN?* Katadata.Com.

Deswiaqsa, K., Darmawan, E., & Sugiyarto, S. (2022). Application of K-Means for *Clustering* Based on the Severity of COVID-19 in Indonesian Private Hospitals. *EKSAKTA: Journal of Sciences and Data Analysis*, *3*(2), 95–102. https://doi.org/10.20885/eksakta.vol3.iss2.art5

Dewi, D. A. I. C., & Pramita, D. A. K. (2019). Analisis Perbandingan Metode Elbow dan Silhouette pada Algoritma *Clustering* K-Medoids dalam Pengelompokan Produksi Kerajinan Bali. *Matrix : Jurnal Manajemen Teknologi Dan Informatika*, *9*(3), 102–109.

https://doi.org/10.31940/matrix.v9i3.1662

Fira, A., Rozikin, C., & Garno, G. (2021). Komparasi Algoritma K-Means dan K-Medoids Untuk Pengelompokkan Penyebaran Covid-19 di Indonesia. *Journal of Applied Informatics and Computing*, *5*(2), 133–138. https://doi.org/10.30871/jaic.v5i2.3286

Nahdliyah, M. A., Widiharih, T., & Prahutama, A. (2019). Metode K-Medoids *Clustering* dengan Validasi Silhouette Index dan C-Index. *Jurnal Gaussian*, *8*(2), 161–170.

Nurhayati, Sinatrya, N. S., Wardhani, L. K., & Busman. (2019). Analysis of K-Means and K-Medoids's Performance Using Big Data Technology. *2018 6th International Conference on Cyber and IT Service Management, CITSM 2018*, *Citsm*, 1–5. https://doi.org/10.1109/CITSM.2018.8674251

Paembonan, S., & Abduh, H. (2021). Penerapan Metode Silhouette Coefficient untuk Evaluasi *Clustering* Obat. *PENA TEKNIK: Jurnal Ilmiah Ilmu-Ilmu Teknik*, *6*(2), 48. https://doi.org/10.51557/pt_jiit.v6i2.659

Putra, P. M. A., & Kadyanan, I. G. A. G. A. (2021). Implementation of K-Means *Clustering* Algorithm in Determining Classification of the Spread of the COVID-19 Virus in Bali. *JELIKU (Jurnal Elektronik Ilmu Komputer Udayana)*, *10*(1), 11. https://doi.org/10.24843/jlk.2021.v10.i01.p03

Qomariyah, & Siregar, M. U. (2022). Comparative Study of K-Means *Clustering* Algorithm and K-Medoids *Clustering* in Student Data *Clustering*. *JISKA (Jurnal Informatika Sunan Kalijaga)*, *7*(2), 91–99. https://doi.org/10.14421/jiska.2022.7.2.91-99

Rahayu, A. E., Hikmah, K., Yustia, N., & Fauzan, A. C. (2019). Penerapan K-Means *Clustering* Untuk Penentuan Klasterisasi Beasiswa Bidikmisi Mahasiswa. *ILKOMNIKA: Journal of Computer Science and Applied Informatics*, *1*(2), 82–86. https://doi.org/10.28926/ilkomnika.v1i2.23

Rodriguez, M. Z., Comin, C. H., Casanova, D., Bruno, O. M., Amancio, D. R., Costa, L. da F., & Rodrigues, F. A. (2019). *Clustering* algorithms: A comparative approach. In *PLoS ONE* (Vol. 14, Issue 1). https://doi.org/10.1371/journal.pone.0210236

Septiani, I. W., Fauzan, A. C., & Huda, M. M. (2022). Implementasi Algoritma K-Medoids Dengan Evaluasi Davies-Bouldin- Index Untuk Klasterisasi Harapan Hidup Pasca Operasi Pada Pasien Penderita Kanker Paru-Paru. *Jurnal Sistem Komputer Dan Informatika (JSON)*, *3*(4), 556–566. https://doi.org/10.30865/json.v3i4.4055

Silvi, R. (2018). Analisis Cluster dengan Data Outlier Menggunakan Centroid Linkage dan K-Means *Clustering* untuk Pengelompokkan Indikator HIV/AIDS di Indonesia. *Jurnal Matematika "MANTIK," 4*(1), 22–31. https://doi.org/10.15642/mantik.2018.4.1.22-31

Sindi, S., Ningse, W. R. O., Sihombing, I. A., R.H.Zer, F. I., & Hartama, D. (2020). Analisis Algoritma K-Medoids *Clustering* Dalam Pengelompokan Penyebaran Covid-19 Di Indonesia. *Jurnal Teknologi Informasi*, *4*(1), 166–173. https://doi.org/10.36294/jurti.v4i1.1296

Singh, A. K., Mittal, S., Malhotra, P., & Srivastava, Y. V. (2020). *Clustering* Evaluation by Davies-Bouldin Index(DBI) in Cereal data using K-Means.

Proceedings of the 4th International Conference on Computing Methodologies and Communication, ICCMC 2020, *Iccmc*, 306–310. https://doi.org/10.1109/ICCMC48092.2020.ICCMC-00057

Suarna, N., Wijaya, Y. A., Mulyawan, Hartati, T., & Suprapti, T. (2021). Comparison K-Medoids Algorithm and K-Means Algorithm for *Clustering* Fish Cooking Menu from Fish Dataset. *IOP Conference Series: Materials Science and Engineering*, *1088*(1), 012034. https://doi.org/10.1088/1757-899x/1088/1/012034

Supriyadi, A., Triayudi, A., & Sholihati, I. D. (2021). Perbandingan Algoritma K-Means Dengan K-Medoids Pada Pengelompokan Armada Kendaraan Truk Berdasarkan Produktivitas. *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *6*(2), 229–240. https://doi.org/10.29100/jipi.v6i2.2008

Suyanto. (2017). *Data Mining untuk klasifikasi dan klasterisasi data*. Informatika.

Virgantari, F., & Faridhan, Y. E. (2020). K-Means *Clustering* of COVID-19 Cases in Indonesia's Provinces. *ADRI International Journal of Engineering and Natural Science*, *5*(2), 34–39. https://doi.org/10.29138/aijens.v5i2.15

Wang, R., Fung, B. C. M., & Zhu, Y. (2020). Heterogeneous data release for cluster analysis with differential privacy. *Knowledge-Based Systems*, *201–202*, 106047. https://doi.org/10.1016/j.knosys.2020.106047