Get Press Indonesia

ISBN 978-634-255-235-3

BIG DATA ANALYTICS

Rudiansyah | Ayu Meida | Nyimas Sriwihajriyah | Pita Rosemari | Theo Vhaldino | Arpa Pauziah | Fitri Yani | Arif Fadillah

BIG DATA ANALYTICS

Rudiansyah
Ayu Meida
Nyimas Sriwihajriyah
Pita Rosemari
Theo Vhaldino
Arpa Pauziah
Fitri Yani
Arif Fadillah

GETPRESS INDONESIA

BIG DATA ANALYTICS

Penulis :
Rudiansyah
Ayu Meida
Nyimas Sriwihajriyah
Pita Rosemari
Theo Vhaldino
Arpa Pauziah
Fitri Yani
Arif Fadillah

ISBN : 978-634-255-235-3

Editor : Mila Sari, M. Si.
Desain Sampul dan Tata Letak : Atyka Trianisa, S.Pd.
PENERBIT : GET PRESS INDONESIA
Anggota IKAPI No. 033/SBA/2022
Jl. Palarik RT 01 RW 06 Kelurahan Air Pacah
Kecamatan Koto Tangah Padang Sumatera Barat
website: www.getpress.co.id
email: adm.getpress@gmail.com

Cetakan pertama, Desember 2025

KATA PENGANTAR

Puji dan syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa, yang telah memberikan rahmat dan karunia-Nya sehingga buku berjudul *Big Data Analytics* ini dapat diselesaikan. Buku ini hadir sebagai bentuk respon terhadap perkembangan cara manusia mengumpulkan, menyimpan, dan menganalisis data. Isi buku ini mencakup pembahasan tentang konsep dasar *big data analytics, data analytics lifecycle*, metode analitik dalam *big data, cluster analysis* dan *association rules, big data tools* dan infrastruktur pendukung, *data ingestion* dan *data storage*, aplikasi & studi kasus analisis *big data*, dan visualisasi, interpretasi, dan kolaborasi proyek *big data.* Akhir kata, semoga buku ini dapat menjadi referensi dan dapat memberikan manfaat serta inspirasi bagi semua pembaca.

Padang, September 2025

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

KONSEP DASAR *BIG DATA ANALYTICS*

1.1 Pendahuluan

Perkembangan teknologi informasi dan komunikasi yang pesat telah membawa perubahan besar terhadap cara manusia mengumpulkan, menyimpan, dan menganalisis data. Setiap detik, jutaan perangkat digital menghasilkan data dalam jumlah sangat besar mulai dari transaksi keuangan, unggahan media sosial, rekaman sensor industri, hingga data dari sistem kesehatan. Kondisi ini melahirkan fenomena yang dikenal sebagai *Big Data*, yaitu kumpulan data dalam volume sangat besar, kecepatan tinggi, dan variasi yang beragam sehingga tidak dapat diolah secara efektif menggunakan sistem pengelolaan data konvensional (Mayer-Schönberger & Cukier, 2013).

Istilah *Big Data* pertama kali populer pada awal tahun 2000-an ketika perusahaan teknologi mulai menghadapi tantangan dalam menangani data berskala masif dari aktivitas daring. Seiring dengan meningkatnya kapasitas penyimpanan dan kemampuan komputasi, muncul kebutuhan untuk tidak hanya menyimpan data tersebut, tetapi juga menganalisisnya guna menghasilkan informasi yang bermakna. Dari sinilah konsep *Big Data Analytics* berkembang yakni proses penerapan metode analisis

statistik, algoritma, dan teknologi komputasi untuk mengekstraksi pola, tren, dan wawasan dari data dalam skala besar.

Dalam konteks global, *Big Data Analytics* telah menjadi salah satu fondasi utama dalam pengambilan keputusan berbasis bukti (*evidence-based decision making*). Perusahaan, lembaga pemerintah, maupun organisasi riset kini memanfaatkan analitik data besar untuk memprediksi tren ekonomi, memantau kesehatan masyarakat, mengoptimalkan rantai pasok, hingga mendukung inovasi produk. Analisis data besar juga memberikan peluang bagi peningkatan efisiensi dan efektivitas di berbagai sektor dengan cara mengubah data mentah menjadi pengetahuan yang aplikatif (Kitchin, 2014).

Bagi dunia akademik dan industri di Indonesia, pemahaman mengenai *Big Data Analytics* menjadi sangat penting karena mampu menjembatani kesenjangan antara data yang melimpah dengan kemampuan untuk menginterpretasikannya. Melalui pemanfaatan analisis data besar, lembaga pendidikan, pemerintah, dan sektor swasta dapat memperkuat proses pengambilan keputusan strategis berbasis data aktual. Oleh karena itu, memahami konsep dasar ***Big Data Analytics*** bukan hanya relevan bagi ilmuwan komputer, tetapi juga bagi para profesional di bidang ekonomi, pertanian, kesehatan, dan sosial.

1.2 Pengertian Big Data

Secara terminologis, **Big Data** mengacu pada kumpulan data yang memiliki ukuran, kompleksitas, dan kecepatan pertumbuhan yang melampaui kemampuan sistem pengelolaan data tradisional untuk memprosesnya secara efisien. Data tersebut tidak hanya berjumlah besar (*volume*), tetapi juga dihasilkan dengan kecepatan tinggi (*velocity*) dan berasal dari berbagai sumber dengan format yang beragam

(*variety*). Seiring waktu, konsep ini diperluas dengan dua dimensi tambahan, yaitu *veracity* (tingkat keakuratan dan keandalan data) serta *value* (nilai atau manfaat yang dapat diekstraksi dari data). Kelima aspek ini dikenal sebagai karakteristik 5V Big Data (Marr, 2016).

Big Data tidak hanya mencakup data terstruktur seperti tabel transaksi atau catatan log, tetapi juga data semi-terstruktur dan tidak terstruktur seperti teks, citra, video, dan sinyal sensor. Oleh karena itu, sistem pengelolaan Big Data menuntut pendekatan baru yang melibatkan arsitektur penyimpanan terdistribusi, pemrosesan paralel, serta algoritma canggih yang mampu mengekstrak pola dari data dalam jumlah masif.

Tabel 1. 1 Karakteristik Utama *Big Data* (5V)

Karakteristik	Deskripsi	Contoh
Volume	Jumlah data yang dihasilkan sangat besar, mencapai terabyte hingga petabyte.	Transaksi e-commerce, sensor IoT, rekaman video streaming.
Velocity	Kecepatan data masuk dan diproses sangat tinggi secara real-time.	Update media sosial, sistem perbankan online.
Variety	Data memiliki format beragam: teks, gambar, audio, video, log, dan data sensor.	Tweet, hasil CT-scan, data GPS.
Veracity	Menunjukkan kualitas,	Data hoaks vs data terverifikasi.

Karakteristik	Deskripsi	Contoh
	keandalan, dan tingkat kebersihan data.	
Value	Nilai informasi yang dapat digunakan untuk pengambilan keputusan.	Prediksi pasar, diagnosis penyakit, rekomendasi produk.

Sumber: Zikopoulos & Eaton, 2011

Selain lima karakteristik tersebut, beberapa literatur menambahkan aspek lain seperti *Variability* (perubahan pola data dari waktu ke waktu) dan *Visualization* (kemampuan menampilkan data dalam bentuk visual yang mudah dipahami). Dengan demikian, Big Data tidak hanya menyoroti besarnya ukuran data, tetapi juga kompleksitas dan nilai yang dikandung di dalamnya.

1.3 Arsitektur dan Komponen *Big Data*

Arsitektur Big Data merupakan kerangka kerja yang menjelaskan bagaimana data dalam skala besar dikumpulkan, disimpan, diproses, dan dianalisis secara efisien. Sistem ini dirancang untuk menangani data dengan karakteristik *volume*, *velocity*, dan *variety* yang tinggi melalui pendekatan terdistribusi dan paralel. Tujuannya adalah agar data dari berbagai sumber dapat diintegrasikan dan diolah secara cepat guna menghasilkan informasi yang bernilai strategis (White, 2015). Secara umum, arsitektur Big Data terdiri atas empat lapisan utama, yaitu (1) *Data Source Layer*, (2) *Data Storage Layer*, (3) *Data Processing Layer*, dan (4) *Data Analytics Layer*. Keempat komponen ini bekerja secara

berurutan untuk memastikan aliran data yang terstruktur dari tahap pengumpulan hingga tahap analisis akhir.

1.3.1 *Data Source Layer* (Lapisan Sumber Data)

Lapisan ini mencakup berbagai sumber yang menghasilkan data dalam jumlah besar, baik yang terstruktur maupun tidak terstruktur. Contohnya meliputi aplikasi transaksi (*transactional systems*), sensor *Internet of Things* (IoT), media sosial, sistem log web, serta data publik dari lembaga pemerintah. Data yang dihasilkan bersifat heterogen sehingga memerlukan mekanisme pengumpulan yang adaptif dan *real-time*, seperti melalui *streaming data ingestion* menggunakan Apache Kafka atau Flume.

1.3.2 *Data Storage Layer* (Lapisan Penyimpanan Data)

Lapisan ini berfungsi untuk menyimpan data dalam format terdistribusi agar dapat diakses dan diproses secara paralel. Salah satu teknologi utama yang digunakan adalah *Hadoop Distributed File System* (HDFS), yang mampu membagi dan menyebarkan potongan data ke beberapa node dalam suatu cluster. Selain HDFS, teknologi lain yang banyak digunakan adalah NoSQL Database seperti MongoDB, Cassandra, atau HBase yang memungkinkan penyimpanan data tidak terstruktur dengan skalabilitas tinggi.

1.3.3 *Data Processing Layer* (Lapisan Pemrosesan Data)

Lapisan pemrosesan bertugas menjalankan analisis atau transformasi data dengan menggunakan teknik *batch processing* maupun *stream processing*.

- *Batch processing* digunakan untuk memproses data dalam jumlah besar secara periodik menggunakan platform seperti MapReduce atau Apache Spark.

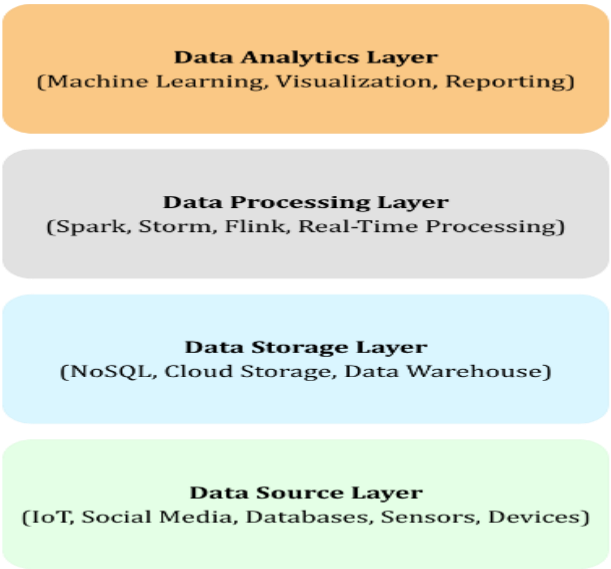
- *Stream processing* digunakan untuk menganalisis data yang terus-menerus mengalir secara real-time melalui sistem seperti Apache Storm, Samza, atau Flink.

Kombinasi kedua metode ini dikenal sebagai *lambda architecture,* yaitu pendekatan yang memungkinkan integrasi antara pemrosesan historis dan pemrosesan *real-time* untuk menghasilkan hasil analisis yang cepat dan akurat.

1.3.4 *Data Analytics Layer* (Lapisan Analitik Data)

Pada tahap ini, data yang telah melalui proses *cleaning*, *integration*, dan *transformation* kemudian dianalisis menggunakan algoritma statistik, *machine learning*, atau *data mining* untuk menemukan pola dan pengetahuan baru. Hasil analisis biasanya divisualisasikan melalui dashboard atau sistem pelaporan interaktif agar dapat mendukung pengambilan keputusan berbasis data (*data-driven decision making*). Selain empat lapisan utama tersebut, sistem Big Data modern juga menambahkan dua lapisan pendukung, yaitu:

- *Security and Governance Layer*, yang mengatur kontrol akses, privasi, dan audit data.
- *Metadata Management Layer*, yang menyimpan informasi deskriptif mengenai struktur dan sumber data.



Gambar 1. 1 Arsitektur Umum Big Data
(Sumber : Adaptasi White, 2015)

1.4 Konsep Dasar *Analytics*

Analitik, dalam konteks Big Data merujuk pada serangkaian teknik yang digunakan untuk menggali, mengolah, dan menganalisis data besar guna menghasilkan wawasan yang dapat digunakan untuk pengambilan keputusan. Tujuan utama dari analitik adalah untuk menemukan pola tersembunyi dalam data, memberikan prediksi mengenai kejadian-kejadian di masa depan, serta memberi rekomendasi yang dapat mengarahkan tindakan strategis.

Ada empat jenis utama dalam *Big Data Analytics,* yang masing-masing berfokus pada tujuan dan tahap analisis yang berbeda. Berikut adalah penjelasan setiap jenis analitik beserta penerapannya dalam berbagai bidang:

1.4.1 *Descriptive Analytics* (Analitik Deskriptif)

Analitik deskriptif adalah jenis analitik yang paling dasar, berfokus pada **pemahaman data historis** dengan memberikan gambaran atau ringkasan tentang data yang telah terjadi. Jenis analitik ini sering digunakan untuk laporan dan pelaporan status terkini dari data yang ada. Contoh:

- Laporan keuangan tahunan yang menunjukkan tren pendapatan dan biaya selama beberapa tahun terakhir.
- Statistik penjualan yang menggambarkan produk mana yang terlaris selama periode tertentu.

1.4.2 *Diagnostic Analytics* (Analitik Diagnostik)

Analitik diagnostik berfokus pada menjawab pertanyaan **"Mengapa ini terjadi?"** dengan menggali lebih dalam untuk mencari akar penyebab dari suatu fenomena. Analitik ini sering menggunakan teknik analisis data lebih lanjut seperti perbandingan antar periode atau analisis korelasi antar variabel. Contoh:

- Mengapa ada penurunan penjualan pada bulan tertentu?
- Apa yang menyebabkan ketidaksesuaian antara hasil yang diprediksi dengan hasil yang sebenarnya?

1.4.3 *Predictive Analytics* (Analitik Prediktif)

Analitik prediktif digunakan untuk **memprediksi kejadian atau tren masa depan** berdasarkan data historis. Teknik-teknik yang digunakan dalam analitik prediktif meliputi *machine learning*, *regression analysis*, dan *time series forecasting*. Contoh:

- Model prediksi untuk meramalkan penjualan produk di bulan depan.
- Prediksi kecelakaan lalu lintas berdasarkan pola perjalanan dan kondisi cuaca.

1.4.4 *Prescriptive Analytics* (Analitik Preskriptif)

Analitik preskriptif adalah jenis analitik yang paling kompleks dan bertujuan memberikan rekomendasi **"Apa yang harus dilakukan?"** untuk menghadapi situasi tertentu. Analitik ini menggabungkan prediksi dengan simulasi untuk memberikan tindakan atau keputusan yang terbaik dalam suatu kondisi yang telah dianalisis. Contoh:

- Algoritma yang menyarankan rute tercepat pengiriman barang berdasarkan kondisi lalu lintas dan cuaca.
- Sistem rekomendasi yang memberi tahu pelanggan produk yang perlu dibeli berikutnya berdasarkan riwayat pembelian mereka.

1.5 Teknologi Pendukung *Big Data Analytics*

Seiring dengan berkembangnya kebutuhan untuk mengelola dan menganalisis data besar, berbagai teknologi telah muncul untuk mendukung *Big Data Analytics*. Teknologi-teknologi ini mencakup berbagai lapisan, mulai dari sistem penyimpanan data terdistribusi, pemrosesan paralel, hingga penggunaan algoritma canggih dalam *machine learning* dan *data mining*. Berikut adalah beberapa teknologi utama yang mendukung ekosistem *Big Data Analytics.*

1.5.1 *Hadoop Ecosystem*

Hadoop adalah kerangka kerja *open-source* yang memungkinkan pemrosesan data besar secara terdistribusi di kluster komputasi. Hadoop menyediakan sistem penyimpanan data yang dapat diskalakan (HDFS) dan sistem pemrosesan paralel menggunakan ***MapReduce***, yang membagi tugas pemrosesan data menjadi potongan-potongan kecil yang kemudian diproses di banyak node secara bersamaan.

1.5.2 *Apache Spark*

Apache Spark adalah sistem pemrosesan data yang lebih cepat daripada Hadoop MapReduce, yang dapat menangani data dalam memori (*in-memory processing*), sehingga mempercepat analisis. Spark mendukung pemrosesan batch dan streaming data secara *real-time*, menjadikannya pilihan populer untuk aplikasi yang membutuhkan kecepatan pemrosesan tinggi, seperti analitik web dan pemrosesan data IoT. Spark memiliki berbagai modul untuk mendukung berbagai jenis analitik:

- Spark SQL: Untuk query data terstruktur.
- MLlib: Untuk analitik machine learning dan algoritma data mining.
- GraphX: Untuk pemrosesan data graf.
- Spark Streaming: Untuk pemrosesan data real-time.

1.5.3 *NoSQL Databases*

NoSQL adalah jenis basis data yang tidak menggunakan model relasional untuk menyimpan data. Basis data NoSQL dirancang untuk mengelola data yang sangat besar dan terdistribusi, serta memungkinkan

skalabilitas horizontal. Beberapa jenis database NoSQL yang populer adalah:

- MongoDB: Database NoSQL berbasis dokumen yang menyimpan data dalam format JSON.
- Cassandra: Database NoSQL yang sangat terdistribusi dan dapat menangani data dalam jumlah besar dengan latensi rendah.
- HBase: Database NoSQL untuk penyimpanan data dalam bentuk tabel yang terdistribusi, biasanya digunakan dengan Hadoop.

1.5.4 *Machine Learning Tools*

Dalam konteks Big Data Analytics, *Machine Learning* (ML) digunakan untuk menganalisis data dan menghasilkan model yang dapat memprediksi atau mengklasifikasikan data berdasarkan pola yang ditemukan. Beberapa alat *machine learning* yang populer dalam analitik Big Data meliputi:

- *TensorFlow: Framework open-source* untuk pengembangan dan pelatihan model *machine learning* dan *deep learning.*
- *Scikit-learn: Library Python* yang digunakan untuk pengolahan data dan penerapan algoritma machine learning seperti regresi, klasifikasi, dan *clustering*.
- *Apache Mahout: Proyek open-source* yang digunakan untuk membangun aplikasi *machine learning* terdistribusi yang bekerja dengan Hadoop.

1.5.5 *Cloud Computing for Big Data*

Selain menggunakan infrastruktur lokal, banyak organisasi yang beralih ke *cloud computing* untuk mengelola dan menganalisis data besar. Penyedia layanan cloud seperti *Amazon Web Services* (AWS), *Microsoft Azure,* dan *Google Cloud Platform* (GCP)

menyediakan layanan penyimpanan data, pemrosesan paralel, dan *machine learning* yang dapat diskalakan sesuai dengan kebutuhan.

- AWS EMR (*Elastic MapReduce*): Layanan untuk menjalankan aplikasi big data seperti Hadoop dan Spark di cloud.
- Google BigQuery: Platform untuk analitik data besar dengan kemampuan pemrosesan real-time dan skalabilitas.
- Azure HDInsight: Layanan *cloud* untuk menjalankan kluster Hadoop dan Spark.

1.6 Tantangan dan Isu dalam *Big Data Analytics*

Meskipun *Big Data Analytics* menawarkan banyak peluang, tantangan besar tetap ada dalam implementasinya. Salah satu tantangan utama adalah keamanan data, karena data sensitif yang dikumpulkan dalam jumlah besar bisa rentan terhadap pencurian atau penyalahgunaan. Untuk itu, penting bagi organisasi untuk memastikan bahwa data dilindungi dengan baik melalui enkripsi, kontrol akses yang ketat, dan pemantauan keamanan secara teratur.

Isu privasi juga menjadi perhatian utama dalam analitik Big Data. Pengumpulan data pribadi harus mematuhi regulasi yang ketat, seperti GDPR, untuk memastikan bahwa hak privasi individu tidak dilanggar. Oleh karena itu, perusahaan perlu memperhatikan prinsip *privacy by design* dan memberikan kontrol lebih kepada pengguna atas data yang mereka miliki. Selain itu, kualitas data menjadi tantangan besar lainnya. Karena data yang digunakan dalam Big Data berasal dari berbagai sumber yang berbeda dan sering kali tidak terstruktur, memastikan bahwa data tersebut konsisten dan akurat adalah hal yang sangat penting. Untuk itu, organisasi perlu menerapkan proses data cleaning yang

efektif dan menggunakan teknologi untuk mendeteksi kesalahan dalam data.

Integrasi data juga menjadi isu karena Big Data melibatkan banyak jenis data dari berbagai sumber yang memiliki format berbeda. Hal ini membuat proses penggabungan data menjadi lebih sulit dan memerlukan alat seperti ETL (*Extract, Transform, Load*) untuk menyatukan berbagai data tersebut agar dapat dianalisis dengan baik.

Seiring dengan pertumbuhan data yang terus meningkat, masalah skalabilitas pun muncul. Sistem yang digunakan harus mampu menangani volume data yang semakin besar tanpa menurunkan kinerja. Penyimpanan terdistribusi dan teknologi *cloud computing* menjadi solusi penting untuk mengatasi masalah ini, karena dapat menyediakan kapasitas penyimpanan dan pemrosesan yang dapat diperluas sesuai dengan kebutuhan.

Terakhir, ada juga isu etika dalam penggunaan Big Data, terutama terkait dengan bagaimana data dikumpulkan, digunakan, dan disebarkan. Penggunaan algoritma yang tidak transparan atau bias bisa menghasilkan keputusan yang merugikan beberapa kelompok tertentu. Oleh karena itu, penting untuk memastikan bahwa algoritma yang digunakan dalam analitik Big Data dapat dipertanggungjawabkan dan adil, serta melakukan audit secara berkala untuk meminimalkan bias. Dengan menghadapi tantangan-tantangan ini, organisasi dapat memastikan bahwa mereka dapat memanfaatkan *Big Data Analytics* secara optimal, sambil menjaga keamanan, privasi, dan keadilan dalam proses analisis data.

1.7 Aplikasi *Big Data Analytics*

Big Data Analytics telah diterapkan secara luas di berbagai sektor untuk meningkatkan efisiensi, inovasi, dan pengambilan keputusan berbasis data. Beberapa sektor yang paling diuntungkan dari penerapan teknologi ini antara lain bisnis, kesehatan, pemerintahan, dan pendidikan.

1.7.1 Bisnis

Data digunakan untuk memahami perilaku konsumen, mengoptimalkan pemasaran, dan memprediksi tren pasar. Perusahaan seperti Amazon dan Netflix, misalnya, menggunakan algoritma rekomendasi berbasis Big Data untuk menyarankan produk atau film kepada pengguna berdasarkan riwayat pembelian atau tontonan mereka. Selain itu, analitik prediktif memungkinkan perusahaan untuk meramalkan permintaan produk, mengelola rantai pasokan dengan lebih efisien, dan meningkatkan pengalaman pelanggan.

1.7.2 Kesehatan

Big Data digunakan untuk menganalisis data medis dalam jumlah besar untuk menemukan pola-pola yang dapat membantu dalam diagnosis penyakit atau pengembangan obat. Rumah sakit dan lembaga kesehatan memanfaatkan data dari catatan medis elektronik, sensor kesehatan, dan studi klinis untuk memberikan perawatan yang lebih personal dan tepat sasaran. Sebagai contoh, analitik prediktif digunakan untuk memantau kondisi pasien dan memprediksi kemungkinan komplikasi kesehatan berdasarkan data riwayat medis mereka.

1.7.3 Pemerintahan

Big Data memberikan wawasan yang berguna dalam merumuskan kebijakan publik dan meningkatkan pelayanan kepada masyarakat. Pemerintah menggunakan analitik data besar untuk memantau dan menganalisis tren sosial-ekonomi, mendeteksi penipuan, serta meningkatkan pengelolaan sumber daya alam dan energi. Sebagai contoh, analitik Big Data digunakan untuk memantau kemacetan lalu lintas, merencanakan pembangunan infrastruktur, serta mengoptimalkan alokasi anggaran publik.

1.7.4 Pendidikan

Big Data digunakan untuk meningkatkan pengalaman belajar dan kinerja akademik siswa. Institusi pendidikan menggunakan analitik untuk mengidentifikasi pola belajar siswa, mengevaluasi efektivitas kurikulum, dan memberikan umpan balik yang lebih cepat kepada siswa. Dengan menganalisis data dari platform pembelajaran online, misalnya, dapat diperoleh wawasan mengenai cara-cara terbaik untuk mendukung perkembangan akademik siswa dan merancang pendekatan yang lebih individual.

Dengan terus berkembangnya teknologi dan peningkatan kapasitas komputasi, Big Data Analytics memiliki potensi untuk mengubah cara organisasi dan individu dalam memahami dan memanfaatkan data untuk berbagai tujuan, baik itu untuk inovasi produk, peningkatan layanan, maupun pengambilan keputusan yang lebih cerdas dan tepat waktu.

DAFTAR PUSTAKA

Mayer-Schönberger, V., & Cukier, K. (2013). Big Data: A Revolution That Will Transform How We Live, Work, and Think. Houghton Mifflin Harcourt.

Kitchin, R. (2014). The Data Revolution: Big Data, Open Data, Data Infrastructures and Their Consequences. SAGE Publications.

Marr, B. (2016). Big Data in Practice. John Wiley & Sons.

Zikopoulos, P., & Eaton, C (2011). Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. McGraw Hill Professional.

White, T. (2015). Hadoop: The Definitive Guide. O'Reilly Media.

Marz, N., & Warren, J. (2015). Big Data: Principles and Best Practices of Scalable Real-Time Data Systems. Manning Publications.

BAB 2
DATA ANALYTICS LIFECYCLE

2.1 Pendahuluan

Dalam era transformasi digital dan ledakan data ("big data"), organisasi di berbagai bidang, mulai dari kesehatan, manufaktur, pemerintahan, hingga bisnis semakin mengandalkan data sebagai sumber utama insight dan keputusan strategis. Namun, penggunaan data yang efektif tidak hanya sekadar mengumpulkan data. Diperlukan serangkaian tahapan sistematis agar analitik data dapat dijalankan dengan benar, menghasilkan model yang andal, dan memberikan manfaat nyata. Kerangka kerja inilah yang biasa disebut *Data Analytics Lifecycle*.

Data Analytics Lifecycle merujuk kepada keseluruhan proses dari pemahaman masalah bisnis, pengumpulan data, persiapan data, eksplorasi data, pemodelan (*modeling*), hingga evaluasi, dan akhirnya penempatan model (*deployment*) serta monitoring / pemeliharaan. Setiap tahap memegang peranan penting dalam menjamin kualitas, reliabilitas, dan keberlanjutan hasil analisis.

Seiring perkembangan teknologi seperti *machine learning, Internet of Things* (IoT), *artificial intelligence* (AI), dan kebutuhan operasional *real-time,* konsep *lifecycle* ini juga mengalami perkembangan: penekanan lebih besar pada automasi, pengelolaan kualitas data yang terus menerus, pemeliharaan model terhadap perubahan kondisi (*drift*), dan

integrasi aspek governance, keamanan, etika, serta keberlanjutan (*sustainability*).

Penelitian-terkini memberikan berbagai insight tentang tantangan dan praktik terbaik dalam *lifecycle* ini. Misalnya:

- An, Lim, dan Lee (2025) dalam artikel *Challenges for Data Quality in the Clinical Data Life Cycle: Systematic Review* menunjukkan bahwa dalam konteks data kesehatan/rekam medis elektronik (EHR), pengelolaan kualitas data sepanjang lifecycle, mulai dari perencanaan, konstruksi, operasional hingga pemanfaatan sangat krusial, terutama karena variasi dalam metode pengumpulan dan format data antar institusi dapat mempengaruhi integrasi, reliabilitas, dan penggunaan data (An, Lim and Lee, 2025).
- Langer (2025) dalam *Understanding Data & Analytics Maturity: A Systematic Review of Maturity Model Composition* membahas berbagai model kematangan (maturity models) data & analitik yang mengandung aspek-teknis dan aspek organisasi. Model-model ini membantu organisasi menilai di mana posisi mereka dalam lifecycle, serta langkah-langkah yang perlu dilakukan untuk meningkatkan kapabilitas analitik secara keseluruhan (Langer, 2025).
- El-Sayed, Abougabal, dan Lazem (2025) dalam *Practical big data techniques for end-to-end machine learning deployment: a comprehensive review* mengulas teknik-teknik modern yang mendukung tahapan full ML lifecycle, terutama di domain big data, seperti preprocessing, teknik deployment, monitoring, dan manajemen siklus hidup model (El-Sayed, Abougabal and Lazem, 2025).

2.1.1 Beberapa Definisi & Komponen Utama

Berdasarkan literatur terkini, beberapa komponen mendasar dalam Data Analytics Lifecycle adalah:

1. ***Business & Problem Understanding***

   Memahami konteks bisnis, tujuan (*goals*), KPI, kebutuhan pemangku kepentingan agar semua aktivitas analitik menjadi relevan dan memiliki dampak nyata.

2. ***Data Collection / Acquisition***

   Mengumpulkan data dari berbagai sumber : internal, eksternal, sensor, log, API, dan lain-lain. Penting juga memperhatikan aspek format, frekuensi, izin akses dan metadata.

3. ***Data Preparation / Cleaning***

   Meliputi pembersihan data (*missing values, duplikasi, outlier*), transformasi, normalisasi, penggabungan antar data, encoding fitur, dan manajemen data yang tidak konsisten.

4. ***Exploratory Data Analysis* (EDA)**

   Pengujian awal terhadap data: distribusi, korelasi, visualisasi, pengecekan asumsi, identifikasi pola dan anomali.

5. ***Modeling / Machine Learning / Statistical Modeling***

   Pemilihan algoritma, pelatihan model, *parameter tuning, cross-validation,* dan model *interpretability.*

6. **Evaluation**

   Penilaian model berdasarkan metrik (akurasi, precision, recall, F1, AUC, dsb.), serta evaluasi dampak bisnis, validasi data luar (*out-of-sample*).

7. ***Deployment, Monitoring, Maintenance***

   Model diproduksi dan diintegrasikan ke dalam sistem operasional, kemudian dimonitor performanya; perlu retraining jika kondisi data berubah.

2.1.2 Tantangan & Isu Kontemporer

Beberapa tantangan yang sering muncul dalam penelitian terkini:

- **Kualitas Data**: Variasi dalam cara pengumpulan dan penyimpanan data, serta format yang berbeda-beda antar lembaga. Ini dapat mempengaruhi integrasi dan penggunaan data nanti (An, Lim and Lee, 2025).
- ***Data Drift* / Perubahan Lingkungan**: Model yang awalnya baik bisa menjadi kurang akurat ketika data baru atau kondisi berubah. Perlu mekanisme monitoring dan deteksi mismatch antara data baru/timely/batch dengan data latih (El-Sayed, Abougabal and Lazem, 2025).
- ***Maturity* & Organisasi** : Banyak organisasi belum mencapai tahap kematangan analitik & data yang tinggi, baik dari sisi teknologi, proses, maupun budaya organisasi (Langer, 2025).
- ***Automasi & Pipeline**:* Kebutuhan untuk automasi pada tahap-tahap seperti pengumpulan, cleaning, penyediaan data, *deployment* agar *lifecycle* bisa berulang, efisien, dan lebih cepat.

2.1.3 Relevansi dan Manfaat

Menggunakan pendekatan lifecycle memungkinkan organisasi:

- Meminimalkan kesalahan dan bias di awal (g*arbage in, garbage out*).
- Meningkatkan efisiensi waktu dan sumber daya karena langkah-langkahnya terdokumentasi dan sistematis.
- Menjamin model tetap relevan dan valid melalui monitoring dan pemeliharaan.
- Lebih mudah untuk menggabungkan praktik terbaik, *governance*, keamanan, dan etika ke dalam setiap tahap.

2.2 Tahapan dalam *Data Analytics Lifecycle*

Data Analytics Lifecycle terdiri dari beberapa tahapan utama yang saling berhubungan. Berdasarkan literatur terkini, berikut adalah deskripsi tiap tahapan dan contoh dari penelitian terbaru :

2.2.1 Tahapan Utama

1. ***Business / Problem Understanding***

   Tahap awal untuk memahami konteks masalah, tujuan bisnis, pemangku kepentingan, serta metrik keberhasilan (KPI). Penting agar penelitian atau proyek data analytics berorientasi kepada nilai yang akan dihasilkan.

2. ***Data Collection / Acquisition***

   Pengumpulan data dari sumber yang relevan (internal, eksternal, sensor, log, API, eksperimen).

Memastikan format, volume, frekuensi, dan kualitas data.

3. ***Data Preparation & Cleaning***

   Meliputi pembersihan data (mengatasi missing values, duplikasi, outlier), transformasi, integrasi antar dataset, encoding variabel, normalisasi, feature engineering.

4. ***Exploratory Data Analysis (EDA) / Data Understanding***

   Analisis eksploratif untuk memahami distribusi data, korelasi antar variabel, identifikasi pola, visualisasi, deteksi anomali. Tahap ini sering iteratif dan mendasari keputusan modeling.

5. ***Modeling / Algoritma / Machine Learning Modeling***

   Pemilihan model atau algoritma sesuai jenis permasalahan (klasifikasi, regresi, clustering, dsb.), pelatihan model, tuning hyperparameter, validasi (cross-validation, k-fold), serta pemilihan fitur yang efektif.

6. ***Evaluation***

   Mengukur performa model menggunakan metrik-metrik yang sesuai seperti akurasi, presisi, recall, F1-score, AUC, error metrics, dsb. Evaluasi juga mencakup pengujian terhadap data luar (*out-of-sample*) dan analisa aspek bisnis seperti dampak ekonomis atau operasional.

7. ***Deployment & Operationalization***

   Mengimplementasikan model ke lingkungan produksi, integrasi ke sistem operasional, penggunaan pipeline deployment, otomatisasi

(CI/CD, MLOps), dan dokumentasi serta versioning model.

8. ***Monitoring, Maintenance, Feedback Loop***

   Pemantauan performa model setelah deployment, deteksi drift data/performa, retraining model bila dibutuhkan, serta *feedback loop* dari pemangku kepentingan agar model tetap relevan dan berguna.

2.2.2 Contoh dari Penelitian Terkini

Berikut penelitian-yang membahas bagian-bagian *lifecycle* tersebut:

- *Enhancing Machine Learning Life Cycle through Advanced Data Engineering* (Jayabalan and Indra, 2024)

  Penelitian ini membahas bagaimana teknik data engineering dapat memperbaiki efisiensi lifecycle ML, mulai dari data preparation & feature engineering**,** model training & tuning**,** governance & offline evaluation**,** hingga deployment, serta eksperimen online.

- *An Analysis of Data Quality Requirements for Machine Learning Development Pipelines Frameworks* (Rangineni, 2023)

  Fokus penelitian ini adalah persyaratan kualitas data yang mengaruh terhadap berbagai tahap pipeline ML: data collection, preprocessing, model training, dan validation. Menegaskan bahwa jika kualitas data rendah, performa model buruk.

- *An Exploratory Study on Machine Learning Model Management* (Latendresse *et al.*, 2024)

  Penelitian ini meneliti bagaimana aktivitas pengelolaan model (*model management*) dilakukan:

versioning, pemantauan performa, pemeliharaan model dalam kondisi dinamis. Ini terkait dengan tahap *deployment* dan *monitoring & maintenance* dalam *lifecycle.*

2.2.3 Integrasi Literature ke dalam Tahapan

Berdasarkan penelitian di atas, kita bisa menarik poin-integrasi antara literatur modern dengan tiap tahap lifecycle:

Tabel 2. 1 Tahapan *Lifecyle*

Tahapan	Temuan dari Penelitian Terbaru
Data Preparation & Cleaning	Penekanan pada data engineering untuk mempercepat dan meningkatkan kualitas tahapan persiapan data (Jayabalan and Indra, 2024)
Data Quality	Kualitas data sangat mempengaruhi semua tahap berikutnya: model training, evaluasi (Rangineni, 2023).
Model Management (Deployment & Monitoring)	Pentingnya versioning, pengelolaan model, tracking performa di produksi; literatur modern menuntut mekanisme manajemen model yang sistematis (Latendresse *et al.*, 2024).

2.3 Perkembangan & Isu Terkini (2023-2025)

Berikut beberapa hasil penelitian terbaru yang terkait *lifecycle*, terutama tantangan yang muncul, model/ *framework* baru dan aplikasi praktis :

2.3.1 Framework dan Studi Konseptual

- *A systematic data characteristic understanding framework towards physical-sensor big data challenges* (Ma, Jørgensen and Ma, 2024).

  Paper ini membahas kerangka yang mengkuantifikasi karakteristik data (6Vs: *volume, variety, velocity, veracity, value, variability*) untuk data sensor fisik. Juga menyertakan pipeline untuk analisis karakteristik dan rekomendasi preprocessing.

- *VeML: An End-to-End Machine Learning Lifecycle for Large-scale and High-dimensional Data* (Le, Bui and Li, 2023). Memperkenalkan sistem version management untuk keseluruhan lifecycle ML, termasuk bagaimana mentransfer lifecycle dari dataset serupa dan mendeteksi mismatch antara data latih dan data uji.

- *Data-Centric Artificial Intelligence: A Survey* (Zha *et al.*, 2025). Fokus pada pentingnya data dalam AI, termasuk tahap-tahap *lifecycle* data: pengembangan data untuk training, inference, dan maintenance data.

- *Foundations and Scoping of Data Science* (Özsu, 2023). Memaparkan model *end-to-end* untuk ekosistem data science, termasuk komponen lifecycle pengembangan dan bagaimana bagian-bagian saling terkait.

2.3.2 Penelitian Aplikasi & Manajemen

- *Secure Data Management Life Cycle for Government Big-Data Ecosystem: Design and Development Perspective* (Zahid *et al.*, 2023). Membahas lifecycle pengelolaan data secara aman untuk big data di sektor pemerintahan.
- *Improving value chain data lifecycle management utilising design for Lean Six Sigma methods* (Trubetskaya *et al.*, 2024). Studi penerapan Lean Six Sigma untuk optimasi lifecycle data dan reporting BI pada agribisnis, dari definisi, pengukuran, analisis, desain hingga verifikasi (*lifecycle monitoring*).
- *Data Science and Big Data Analytics: a systematic review of methodologies used in the supply chain and logistics research* (Jahani, Jain and Ivanov, 2023). Ulasan bagaimana metode-metode analitik dan data science digunakan dalam konteks supply chain, yang menyentuh aspek-aspek lifecycle: pengumpulan data, pemodelan, evaluasi.
- *Supply Chain Data Analytics for Digital Twins: A Comprehensive Framework* (Xiros *et al.*, 2025). Menggabungkan konsep digital twins dengan analytics lifecycle untuk supply chain: penggunaan data real-time, monitoring, dan feedback loop.

2.4 Tantangan dan Best Practices Terbaru

Dari literatur terkini muncul sejumlah tantangan dan solusi:

2.4.1 Tantangan

- ***Data drift*** dan ***environment change****:* model yang sudah diproduksi bisa kehilangan akurasi karena kondisi data berubah seiring waktu.
- **Karakteristik data sensor / IoT**: volume besar, frekuensi tinggi, kecepatan & variabilitas waktu, noise, missing data. (Ma *et al*.)
- **Keamanan, privasi, dan *governance****:* terutama di sektor publik & pemerintahan karena isu regulasi dan kepercayaan publik. (Zahid *et al*.)
- **Keterbatasan sumber daya & teknologi**: komputasi, penyimpanan, skill analis; serta kebutuhan integrasi sistem lama vs sistem baru.

2.4.2 *Best Practices*

- **Automasi dalam *pipeline***: mulai dari pengolahan data, *feature engineering, model deployment* & monitoring. Ini muncul di beberapa framework dan studi kasus modern (Le, Bui and Li, 2023).
- **Penggunaan metrik kualitas data & karakteristik eksploratif awal**: seperti model 6Vs. Memahami data sebelum modeling untuk mendeteksi masalah awal(Ma, Jørgensen and Ma, 2024) .
- **Versioning dan *monitoring*** lifecycle: tracking versi data dan model, monitoring performa, evaluasi ulang model secara periodik. (VeML) (Le, Bui and Li, 2023).
- **Integrasi *governance* & *security* sejak awal**: kebijakan, akses, audit, kepatuhan terhadap regulasi privasi (Zahid *et al.*, 2023).

- **Align organisasi dan tujuan bisnis**: definisi KPI & target bisnis harus jadi landasan tiap tahap *lifecycle*. (*Guiding the integration of analytics in business operations through a maturity framework*)(Menukhin *et al.*, 2025)

2.5 Model / *Framework Lifecycle* Terkait

Berikut contoh framework lifecycle atau maturity level dari penelitian terkini:

- **VeML**: sistem yang mendukung lifecycle ML *end-to-end,* termasuk transfer lifecycle dan deteksi mismatch data (Le, Bui and Li, 2023).
- **6V *Characteristic Framework*** (*Volume, Variety, Velocity, Veracity, Value, Variability*) sebagai bagian dari tahap eksploratif & persiapan data (Le, Bui and Li, 2023).
- ***Maturity Framework* untuk *Analytics-Business Alignment*** (Annals of Operations Research, 2023): aspek data, proses, teknologi, *people, governance* (Menukhin *et al.*, 2025).
- ***Framework* untuk *Digital Twins & Data Analytics*** untuk *supply chain: feedback loops, real-time monitoring,* penggunaan data *lifecycle* yang terus diperbaharui (Xiros *et al.*, 2025).

2.6 Contoh Kasus / Aplikasi Pada Industri

- Kasus agribisnis di mana lifecycle data dan reporting BI dioptimasi menggunakan Lean Six Sigma (Trubetskaya *et al.*, 2024).
- Sektor pemerintahan: pengelolaan big data dengan lifecycle yang aman & governance ketat (Zahid *et al.*, 2023).

- *Supply chai*n & digital twins: penerapan data analytics secara *real-time, monitoring*, umpan balik untuk adaptasi model (Xiros *et al.*, 2025).

2.7 Rekomendasi untuk Praktisi & Peneliti

- Terapkan **MLOps** agar deployment dan pemeliharaan model menjadi lebih reliabel dan otomatis.
- Gunakan *framework* karakteristik data (6Vs atau lain) sebagai bagian dari tahap eksploratif.
- Integrasikan *security & governance* sejak tahap awal: pengumpulan data hingga deployment.
- Pilih KPI bisnis yang jelas dan evaluasi dampak nyata (bukan hanya metrik teknis).
- Penelitian lebih lanjut bisa fokus pada: *data drift, explainability, fairness, interpretability, integrasi data real-time* dari IoT, automatisasi *pipeline*.

2.8 Kesimpulan

- Data analytics lifecycle tetap sangat relevan di era big data, AI, IoT.
- Perkembangan terkini menekankan aspek karakteristik data, otomatisasi, governance, monitoring, dan model yang adaptif terhadap perubahan lingkungan.
- *Framework* dan studi terbaru menawarkan insights dan tools untuk memperkuat tiap tahap *lifecycle* agar lebih efisien, akurat, dan aman.

DAFTAR PUSTAKA

An, D., Lim, M. and Lee, S. (2025) "Challenges for data quality in the clinical data life cycle: systematic review," *Journal of Medical Internet Research*, 27, p. e60709.

El-Sayed, A., Abougabal, M. and Lazem, S. (2025) "Practical big data techniques for end-to-end machine learning deployment: a comprehensive review," *Discover Data*, 3(1), p. 11.

Jahani, H., Jain, R. and Ivanov, D. (2023) "Data science and big data analytics: a systematic review of methodologies used in the supply chain and logistics research," *Annals of Operations Research*, pp. 1–58.

Jayabalan, D. and Indra, S. (2024) "Enhancing Machine Learning Life Cycle through Advanced Data Engineering," *International Journal of Computer Trends and Technology*, 7(4), pp. 136–139.

Langer, B. (2025) "Understanding Data & Analytics Maturity: A Systematic Review of Maturity Model Composition," *Schmalenbach Journal of Business Research*, pp. 1–23.

Latendresse, J. *et al.* (2024) "An exploratory study on machine learning model management," *ACM Transactions on Software Engineering and Methodology*, 34(1), pp. 1–31.

Le, V.-D., Bui, C.-T. and Li, W.-S. (2023) "Veml: An end-to-end machine learning lifecycle for large-scale and high-dimensional data," *arXiv preprint arXiv:2304.13037* [Preprint].

Ma, Z., Jørgensen, B.N. and Ma, Z.G. (2024) "A systematic data characteristic understanding framework towards physical-sensor big data challenges," *Journal of Big Data*, 11(1), p. 84.

Mansyur, A.R. (2020) "Dampak COVID-19 Terhadap Dinamika Pembelajaran Di Indonesia," *Education and Learning Journal*, Vol. 1, No, pp. 113–123.

Menukhin, O. *et al.* (2025) "Guiding the integration of analytics in business operations through a maturity framework," *Annals of Operations Research*, 348(3), pp. 2017–2047.

Özsu, M.T. (2023) "Foundations and scoping of data science," *arXiv preprint arXiv:2301.13761* [Preprint].

Rangineni, S. (2023) "An analysis of data quality requirements for machine learning development pipelines frameworks," *International Journal of Computer Trends and Technology*, 71(9), pp. 16–27.

Trubetskaya, A. *et al.* (2024) "Improving value chain data lifecycle management utilising design for Lean Six Sigma methods," *The TQM Journal*, 36(9), pp. 136–154.

Xiros, V. *et al.* (2025) "Supply Chain Data Analytics for Digital Twins: A Comprehensive Framework," *Applied Sciences* [Preprint].

Zahid, R. *et al.* (2023) "Secure data management life cycle for government big-data ecosystem: Design and development perspective," *Systems*, 11(8), p. 380.

Zha, D. *et al.* (2025) "Data-centric artificial intelligence: A survey," *ACM Computing Surveys*, 57(5), pp. 1–42.

BAB 3

METODE ANALITIK DALAM *BIG DATA*

3.1 Pendahuluan

Pada era digital saat ini, volume, kecepatan, dan ragam (*volume, velocity, variety*) data meningkat secara eksponensial fenomena yang dikenal sebagai "big data". Analitik pada big data bukan saja sekedar menggunakan teknik statistik tradisional, namun memerlukan metode analitik yang lebih maju, terdistribusi, dan sering kali real-time. Buku ini menggarisbawahi berbagai metode analitik yang digunakan dalam konteks big data, tantangan yang muncul, dan bagaimana metode-metode tersebut diterapkan dalam berbagai domain. Penekanan bab ini adalah pada pendekatan analitik (deskriptif, diagnostik, prediktif, preskriptif) serta teknik-metodologi (*machine learning, data mining, streaming analytics, real-time analytics, visualisasi, analitik kualitatif*) yang relevan dengan big data.

3.2 Karakteristik Big Data dan Implikasi untuk Analitik

Sebelum masuk ke metode analitik, penting memahami karakteristik khas big data dan bagaimana karakteristik ini mempengaruhi pemilihan metode analitik.

3.2.1 Karakteristik *Big Data*

- Volume: Ukuran dataset yang sangat besar (terabyte hingga petabyte).
- *Velocity*: Data datang dan diproses dengan kecepatan tinggi (*real-time* atau hampir *real-time*).
- *Variety*: Heterogenitas format data (terstruktur, semi-terstruktur, tidak terstruktur).
- *Veracity & Value*: Kualitas data serta nilai yang dapat diekstrak.
  Karakteristik ini memaksa pengembangan metode analitik yang skalabel, paralel, dan terkadang terdistribusi.

3.2.2 Implikasi Bagi Metode Analitik

Metode-metode tradisional statistik atau analitik perangkat tunggal kadang tidak memadai untuk big data. Diperlukan sistem terdistribusi (misalnya Hadoop, Spark), algoritma yang tahan skala besar, dan pipeline analitik yang mencakup preprocessing, integrasi data, transformasi, dan visualisasi secara efisien. Sebagai contoh, sebuah studi menunjukkan bahwa analitik big data dalam komputasi awan (*cloud computing*) mengusulkan integrasi teknologi mining dan *big data analytics* (Ren and Wang, 2023).

Selain itu, metodologi *review-systematic* menunjukkan bahwa dalam rantai pasok/logistik, metode data science dan big data analytics semakin mapan (Jahani, Jain and Ivanov, 2023).

3.3 Klasifikasi Utama Metode Analitik dalam Big Data

Dalam praktik, analitik pada big data biasanya diklasifikasikan menjadi empat level utama: deskriptif,

diagnostik, prediktif, dan preskriptif. Selain itu, muncul juga analitik real-time, streaming analytics, serta analitik kualitatif pada big data.

3.3.1 Analitik Deskriptif

Menjawab pertanyaan "Apa yang telah terjadi?". Teknik-umum: agregasi, visualisasi, statistik ringkasan, clustering sederhana, dashboard. Menurut literatur, jenis analitik ini masih menjadi tahap awal dalam implementasi big data (Data and Analytics, 2019).

3.3.2 Analitik Diagnostik

Menjawab "Mengapa itu terjadi?". Teknik-umum: *drill-down, korelasi, root-cause analysis, data mining*. Pada big data, metode ini sering membutuhkan mining pada data heterogen dan penggabungan data historis dengan metadata.

3.3.3 Analitik Prediktif

Menjawab "Apa yang mungkin akan terjadi?". Teknik-umum: regresi, klasifikasi, pohon keputusan, random forest, machine learning, deep learning. Contoh terbaru: studi sistematis menemukan bahwa metode prediktif dengan big data mencakup machine learning, data mining, AI (Jamarani *et al.*, 2024). Studi lain dalam domain pendidikan: "*Designing and evaluating a big data analytics approach for predicting students' success factors*" (Fahd and Miah, 2023). Untuk big data kualitatif, analitik prediktif mulai memasuki ranah analisis tekstual/unstructured data.

3.3.4 Analitik Preskriptif

Menjawab "Apa yang harus dilakukan?". Teknik-umum: optimisasi, simulasi, rekomendasi, *reinforcement*

learning. Dalam big data, preskriptif sering diaplikasikan dalam skenario real-time decision making dan sistem otonom yang memanfaatkan data aliran (*streaming*). Sebagai contohnya, studi "*Enhancing Business Intelligence Through AI and Big Data*" memfokuskan pada integrasi AI + Big Data untuk analitik real-time (Wang, 2025).

3.3.5 Analitik *Real-Time / Streaming Analytics*

Karena big data banyak datang dalam aliran (*stream*), metode analitik streaming (misalnya Apache Kafka, *Spark Streaming*) menjadi penting. Sebuah paper meninjau teknik pengolahan dan analisis data besar melalui batch dan stream processing. Metode real-time menuntut arsitektur pemrosesan dan analitik yang rendah latensi (Pillai, 2024).

3.3.6 Analitik Kualitatif dalam Big Data

Analitik tidak selalu kuantitatif; dataset besar yang bersifat kualitatif (seperti teks, wawancara, media sosial) membutuhkan metode analitik khusus. Sebuah living systematic review baru-baru ini menyajikan analisis metode untuk dataset kualitatif besar (Chandrasekar *et al.*, 2024). Metode-metode seperti thematic analysis, NLP, analisis jaringan sosial, digunakan.

3.4 Teknik & Algoritma Umum untuk Big Data Analitik

Di bagian ini kita bahas teknik/algoritma populer dan cocok untuk big data. Setiap teknik diadaptasi agar skalabel dan efisien.

3.4.1 *Data Pre-processing* & Integrasi

- Pembersihan data (*cleaning*) – menghapus duplikasi, menangani missing data, outliers.
- Integrasi data dari berbagai sumber dan format (terstruktur, semi, tidak terstruktur).
- Transformasi/normalisasi, reduksi dimensi (PCA, t-SNE, UMAP) untuk mengatasi *high-dimensionality.*
- Pipeline ETL/ELT untuk big data, sering dalam arsitektur terdistribusi.
  Contoh: studi "*Methods and Applications of Data Management and Analytics*" membahas manajemen data dan analitik dalam konteks besar (Zhang and Yang, 2025).

3.4.2 *Clustering* & Segmentasi

Metode *unsupervised* seperti K-means, hierarchical clustering, DBSCAN, digunakan untuk menemukan pola dan segmentasi dalam dataset besar. Karena volume besar, algoritma harus scalable dan sering di-parallel (misalnya menggunakan Spark MLlib). Studi "*Comparative Study Analysis of Big Data Analytics and techniques in IT*" menyentuh analisis metode dengan Hadoop (Maahanwal, 2024).

3.4.3 Klasifikasi & Regresi

Algoritma supervised seperti logistic regression, decision tree, random forest, SVM, neural network digunakan untuk memprediksi kategori/outcome. Untuk big data, versi paralel/distribusi (misalnya Spark ML, XGBoost) sering diterapkan. Studi "Designing and evaluating predicting students' success factors" menggunakan machine learning untuk analitik big data (Fahd and Miah, 2023).

3.4.4 Asosiasi & Rule Discovery

Algoritma seperti Apriori, FP-Growth, digunakan untuk menemukan aturan asosiasi dalam dataset besar (misalnya transaksi ritel, big log data). Part dari literatur umum big data analytics menyebut fungsi mining semacam ini (Maahanwal, 2024).

3.4.5 *Deep Learning & Neural Networks*

Untuk dataset besar dan kompleks (gambar, video, teks, sensor), *deep learning* menjadi metode utama: CNN, RNN, LSTM, Transformer, GNN. Kombinasi big data dan DL memungkinkan ekstraksi fitur otomatis dan prediksi tingkat lanjut. Bahkan review arXiv memaparkan transformasi big data analytics dengan deep learning (Hsieh *et al.*, 2024).

3.4.6 *Streaming / Real-Time Analytics*

Metode seperti *windowing, event-stream processing, real-time anomaly detection, online learning.* Arsitektur sering menggunakan Spark Streaming, Kafka, Flink. Digambarkan dalam studi "Techniques for Processing and Analyzing Large Data Sets Using Big Data Analytics" (Pillai, 2024).

3.4.7 Visualisasi dan *Dashboards*

Visualisasi dalam big data menuntut teknik yang bisa mem-render dan menyaring data besar secara interaktif: heatmap, network graphs, dashboards real-time. Visualisasi menjadi jembatan antara hasil analitik dan keputusan manajemen.

3.4.8 Analitik Kualitatif & Teks

Data tidak terstruktur seperti teks, wawancara, media sosial memerlukan analitik khusus: NLP,

sentiment analysis, topic modelling (LDA), jaringan sosial (SNA), visualisasi teks. Studi "Making the most of big qualitative datasets" membahas hal ini (Chandrasekar *et al.*, 2024).

3.5 Arsitektur & Pipeline Analitik Big Data

Agar metode-metode di atas berjalan optimal dalam big data, diperlukan arsitektur/pipeline yang sesuai. Bagian ini menguraikan unsur-unsurnya.

3.5.1 Ingesti & Penyimpanan

- *Data ingestion: batch vs streaming.*
- *Data lake, data warehouse, data lakehouse.*

  Studi "*Big Data Analytics Technology and Applications in Cloud Computing Perspective*" menekankan integrasi big data dan cloud computing (Ren and Wang, 2023).
- Storage terdistribusi: HDFS, NoSQL (Cassandra, MongoDB), cloud storage.

3.5.2 Pemrosesan & Analisis

- *Batch processing: MapReduce, Spark.*
- *Real-time/stream: Spark Streaming, Flink, Kafka. Studi "Comparative Study …* Big Data Analytics and techniques in IT" membahas komponen Hadoop (Maahanwal, 2024).
- *Interactive/Ad-hoc processing: data warehousing, OLAP on big data.*

3.5.3 Analitik & Model

- *Pipeline machine learning/deep learning* yang terintegrasi.
- Model *deployment* dan *scoring* di produksi (*online/offline*).
- Visualisasi dan *dashboarding*.

3.5.4 *Governance*, Keamanan & Etika

- *Data governance, data quality, privacy, compliance* (GDPR, HIPAA).
- Tantangan kualitas dan kepercayaan data sangat penting. Sebagai contoh, review "15 years of Big Data" mengidentifikasi banyak tantangan open issues di big data analytics (Tosi, Kokaj and Roccetti, 2024).

3.6 Tantangan dan Rintangan dalam Metode Analitik Big Data

Walaupun banyak keuntungan, implementasi metode analitik dalam big data memiliki sejumlah tantangan:

- Data silos / integrasi data yang buruk.
- Kualitas data buruk (missing, noise, bias).
- Skalabilitas algoritma – metode yang bagus secara teori belum tentu bisa skala ke petabyte.
- Latensi dan kebutuhan real-time.
- Interpretabilitas model (khususnya deep learning).
- Privasi dan keamanan data.

Kekurangan tenaga ahli dengan kompetensi big data analytics. Sebagai contoh, studi "*Big Data Analytics in*

Financial Statement Analysis" menyebutkan tantangan integrasi dan interpretasi data tidak terstruktur (Simatupang, 2024). Lebih jauh, studi "*Big Data Analytics Capability Effect on Indonesia Firm Performance*" membahas kebutuhan agility proses bisnis sebagai mediator (Kusbianto and Darmawan, 2024).

3.7 Aplikasi dan Studi Kasus Terbaru

Di sini diberikan ringkasan beberapa aplikasi dan studi kasus dari literatur terkini:

- **Rantai pasok & logistik**: studi sistematis metode DS & BDA dalam supply chain/logistic (Jahani, Jain and Ivanov, 2023).
- **Pendidikan**: prediksi faktor keberhasilan mahasiswa menggunakan big data analytics (Fahd and Miah, 2023).
- **Manajemen akuntansi & bisnis**: penggunaan big data & business intelligence dalam akuntansi manajemen (Narulita, Baderi and Hidayati, 2025).
- **Healthcare & kualitatif**: analisis dataset kualitatif besar di bidang kesehatan (Chandrasekar *et al.*, 2024).
- **Persaingan bisnis & keunggulan kompetitif**: aplikasi big data dan analytics untuk competitive advantage (Adi Sahputra and Nendi, 2024).

3.8 Panduan Pemilihan Metode Analitik

Untuk praktisi atau peneliti, berikut langkah panduan memilih metode analitik yang tepat:

1. **Definisikan tujuan analitik**:

   Apakah ingin deskriptif, diagnostik, prediktif atau preskriptif?

2. **Pahami karakteristik data**:

   Volume, kecepatan, ragam, struktur.

3. **Pilih teknik sesuai**:

   - Untuk segmentasi → clustering.
   - Untuk prediksi → klasifikasi/regresi.
   - Untuk rekomendasi → preskriptif/optimasi.
   - Untuk data streaming → streaming analytics.

4. **Pertimbangkan skalabilitas dan arsitektur**:

   Apakah platform mendukung data besar?

5. **Uji dan evaluasi**:

   Metrik seperti akurasi, latensi, interpretabilitas, biaya komputasi. Review "Big data and predictive analytics…" menegaskan bahwa evaluasi metrik seperti timeliness, accuracy, scalability sangat penting (Jamarani *et al.*, 2024).

6. **Iterasi dan penerapan**:

   Tentukan model produksi, monitoring, perbaikan.

3.9 Tren Metode Analitik Big Data ke Depan

Beberapa tren yang muncul dalam beberapa tahun terakhir dan ke depan:

- Kombinasi *machine learning/deep learning* dengan r*eal-time streaming analytics.*
- *Augmented analytics* (AI-*driven analytics)* dan automasi pipeline analitik.
- Analitik kualitatif besar dan Mixed-methods analytics.
- *Edge analytics* (analitik di edge devices) dan IoT.

- Tingkat interpretabilitas dan fairness dalam model big data.
  Studi "*Exploring the Evolution of Big Data Technologies*" (2025) memuat tren dan tantangan ke depan (Hakami, Alginahi and Sabri, 2025).
- Studi bibliometrik "Big Data Analytics (BDA) in the Research Landscape" (2025) menunjukkan penggunaan Python + VOSviewer untuk visualisasi tren riset (Arifin *et al.*, 2025).

3.10 Ringkasan

Bab ini telah menguraikan kerangka metode analitik yang digunakan dalam big data: dari klasifikasi metode (deskriptif hingga preskriptif), teknik/algoritma utama, arsitektur pipeline, tantangan implementasi, hingga panduan pemilihan dan tren ke depan. Dengan pemahaman ini, pembaca diharapkan dapat mengidentifikasi dan merancang pendekatan analitik yang sesuai untuk konteks big data masing-masing.

DAFTAR PUSTAKA

Adi Sahputra, E.S. and Nendi, I. (2024) "Application of Big Data and Analytics to Increase Competitive Advantage.," *Jurnal Indonesia Sosial Teknologi*, 5(5).

Arifin, S. *et al.* (2025) "Big Data Analytics (BDA) in the Research Landscape: Using Python and VOSviewer for Advanced Bibliometric Analysis," *J. Comput. Sci*, 21(2).

Chandrasekar, A. *et al.* (2024) "Making the most of big qualitative datasets: a living systematic review of analysis methods," *Frontiers in big Data*, 7, p. 1455399.

Data, B. and Analytics, B.D. (2019) "Big Data Analytics : Tools and Approaches," *Acm Isbn 978- 1-4503-7284-8/19/12…$15.00.*, 12(2019), pp. 22–25.

Fahd, K. and Miah, S.J. (2023) "Designing and evaluating a big data analytics approach for predicting students' success factors," *Journal of Big Data*, 10(1), p. 159.

Hakami, T.A., Alginahi, Y.M. and Sabri, O. (2025) "Exploring the Evolution of Big Data Technologies: A Systematic Literature Review of Trends, Challenges, and Future Directions," *Future Internet* [Preprint].

Hsieh, W. *et al.* (2024) "Deep Learning, Machine Learning, Advancing Big Data Analytics and Management," *arXiv preprint arXiv:2412.02187* [Preprint].

Jahani, H., Jain, R. and Ivanov, D. (2023) "Data science and big data analytics: a systematic review of methodologies used in the supply chain and logistics research," *Annals of Operations Research*, pp. 1–58.

Jamarani, A. *et al.* (2024) "Big data and predictive analytics: A systematic review of applications," *Artificial Intelligence Review*, 57(7), p. 176.

Kusbianto, N. and Darmawan, A. (2024) "Big data Analytics Capability Effect on Indonesia Firm Performance: The Mediating Role of Business Process Agility," *The International Journal of Accounting and Business Society*, 32(2), pp. 224–248.

Maahanwal, D. (2024) "Comparative Study Analysis of Big Data Analytics and techniques in IT," (9), pp. 1–6.

Narulita, F.D., Baderi, R.N. and Hidayati, C. (2025) "The Use of Big Data and Business Intelligence in Management Accounting Decision Making," *Journal of Advances in Accounting, Economics, and Management*, 2(4), p. 18.

Pillai, V. (2024) "Techniques for Processing and Analyzing Large Data Sets Using Big Data Analytics," *Available at SSRN 4991707* [Preprint].

Ren, X. and Wang, Z. (2023) "Big Data Analytics Technology and Applications in Cloud Computing Perspective," *Applied Mathematics and Nonlinear Sciences*, 8(2), pp. 1415–1432.

Simatupang, O. (2024) "Big Data Analytics in Financial Statement Analysis : A Systematic Review of Challenges , Techniques , and Future Directions."

Tosi, D., Kokaj, R. and Roccetti, M. (2024) "15 years of Big Data: a systematic literature review," *Journal of Big Data*, 11(1), p. 73.

Wang, Z. (2025) "Enhancing Business Intelligence Through AI and Big Data: A Focus on Precision Mining and Real-Time Analysis," *Applied and Computational Engineering*, 119, pp. 61–66.

Zhang, W. and Yang, Z. (2025) "Methods and Applications of Data Management and Analytics." MDPI-Multidisciplinary Digital Publishing Institute.

BAB 4

CLUSTER ANALYSIS DAN *ASSOCIATION RULES*

Dua teknik penting dalam data mining untuk big data adalah *Cluster Analysis* dan *Association Rule Mining.* Keduanya digunakan untuk menemukan pola tersembunyi, segmentasi, dan hubungan antar item dalam dataset besar.

4.1 *Cluster Analysis*

Cluster Analysis adalah teknik analisis data yang mengelompokkan objek atau data point ke dalam kelompok (*cluster*) sehingga objek dalam satu cluster memiliki kemiripan yang tinggi dibandingkan dengan objek di cluster lain (Purnamasari, 2010). Metode ini termasuk dalam unsupervised learning karena tidak memerlukan label data sebelumnya.

4.2 Jenis-Jenis *Cluster Analysis*

4.2.1 *Partitioning Methods*

Partitioning Methods dalam *clustering analysis* adalah pendekatan yang membagi data ke dalam sejumlah klaster yang telah ditentukan sebelumnya, berdasarkan kesamaan atau kedekatan antar data.

Metode ini mencoba menemukan partisi optimal dari data sehingga setiap data hanya termasuk dalam satu klaster dan klaster-klaster tersebut saling eksklusif (Abdurrahman *et al*., 2021).

Tahapan dalam Partioning Methods :

1. Menentukan jumlah klaster **k** terlebih dahulu.
2. Mengalokasikan setiap data ke salah satu dari **k** klaster berdasarkan kriteria tertentu (misalnya jarak Euclidean).
3. Mengoptimalkan partisi dengan meminimalkan variasi dalam klaster dan memaksimalkan variasi antar klaster.

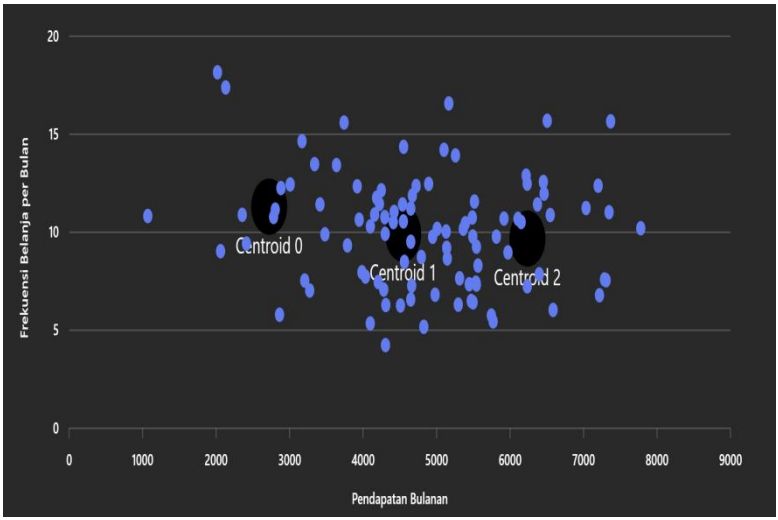
Contoh Penerapan Partioning Methods dengan menggunakan K-Means Clustering. Asumsikan kita memiliki data pelanggan yaitu Pendapatan Bulanan dan Frekuensi Belanja Per-Bulan, dengan menggunakan K-Means Clustering kita mencoba mengelompokkan pelanggan menjadi 3 segmen berdasarkan prilaku belanja mereka.

```
import numpy as np
import pandas as pd
import plotly.express as px
from sklearn.cluster import KMeans
# Membuat data simulasi pelanggan
np.random.seed(42)
pendapatan = np.random.normal(5000, 1500, 100)  # rata-rata pendapatan 5000
frekuensi_belanja = np.random.normal(10, 3, 100)  # rata-rata belanja 10 kali


# Membuat DataFrame
```

```
data = pd.DataFrame({
'Pendapatan': pendapatan,
'Frekuensi Belanja': frekuensi_belanja
})

# Menerapkan K-Means dengan 3 klaster
kmeans = KMeans(n_clusters=3, random_state=42)
data['Klaster'] = kmeans.fit_predict(data[['Pendapatan', 'Frekuensi Belanja']])

# Visualisasi hasil klaster
fig = px.scatter(
data,
 x='Pendapatan',
 y='Frekuensi Belanja',
 color='Klaster',
title='Hasil Klaster Pelanggan dengan K-Means',
 labels={'Pendapatan': 'Pendapatan Bulanan', 'Frekuensi Belanja': 'Frekuensi Belanja per Bulan'}
)

fig.write_image("kmeans_clustering.png")
fig.write_json("kmeans_clustering.json")

# Menampilkan jumlah anggota tiap klaster
```

```
    cluster_counts                                  =
data['Klaster'].value_counts().to_dict()
    print("Jumlah    anggota    tiap    klaster:",
cluster_counts)
```

maka didapat hasil sebagai berikut:



Gambar 4. 1 *Hasil Partitioning Metod*

Sumber: hasil pengolahan data

Tabel 4. 1 *Klustering Partitioning Method*

Klaster	Pendapatan	Frekuensi Belanja	Anggota Klaster
Centroid 0	2721,73	11,30	17
Centroid 1	4545,74	9,92	47
Centroid 2	6236,22	9,68	36

Sumber: hasil olahan data

4.2.2 *Hierarchical Clustering*

Hierarchical Clustering adalah salah satu metode unsupervised learning yang digunakan untuk mengelompokkan data ke dalam cluster berdasarkan tingkat kemiripan (*similarity*). Berbeda dengan metode seperti K-Means yang membutuhkan jumlah cluster di awal, hierarchical clustering membentuk hierarki (struktur pohon) yang disebut dendrogram. Ada dua pendekatan utama yaitu ***Agglomerative (Bottom-Up)***, Mulai dari setiap data sebagai cluster terpisah, kemudian menggabungkan cluster yang paling mirip secara bertahap hingga menjadi satu cluster besar. dan ***Divisive (Top-Down)***, Mulai dari satu cluster besar, lalu memecahnya menjadi cluster yang lebih kecil (Everitt *et al*., 2011).

Keunggulann daru metode ini Tidak perlu menentukan jumlah cluster di awal (bisa dilihat dari dendrogram) dan Memberikan visualisasi hubungan antar data. Misalkan kita memiliki data tentang **tingkat pendapatan dan pengeluaran** dari 10 orang, dan kita ingin mengelompokkan mereka berdasarkan pola ekonomi.

Langkah Analisis yang dilakukan :

1. Siapkan data (misal pendapatan vs pengeluaran).
2. Hitung jarak antar data (misal Euclidean).
3. Terapkan hierarchical clustering (*Agglomerative*).
4. Visualisasikan dendrogram.
5. Tentukan jumlah cluster (misal dengan memotong dendrogram).

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
from scipy.cluster.hierarchy import linkage,
dendrogram, fcluster


# 1. Buat data contoh (Pendapatan vs
Pengeluaran)
data = {
    'Pendapatan': [45, 50, 55, 60, 65, 70,
80, 85, 90, 100],
    'Pengeluaran': [40, 42, 43, 48, 50, 52,
60, 62, 65, 70]
}
df = pd.DataFrame(data)


# 2. Hitung linkage (metode Ward)
Z = linkage(df, method='ward')


# 3. Visualisasi dendrogram
plt.figure(figsize=(10, 6))
dendrogram(Z,      labels=df.index.tolist(),
leaf_rotation=90)
plt.title('Dendrogram          Hierarchical
Clustering')
plt.xlabel('Individu')
plt.ylabel('Jarak')
plt.show()


# 4. Tentukan cluster (misal 3 cluster)
clusters      =      fcluster(Z,      t=3,
criterion='maxclust')
df['Cluster'] = clusters
```

4.2.3 *Density-Based Clustering*

Density-Based Clustering adalah metode dalam analisis klaster yang mengelompokkan data berdasarkan kepadatan titik-titik data di ruang fitur. Salah satu algoritma paling terkenal dalam kategori ini adalah **DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*)** (Yewang *et al.*, 2020).

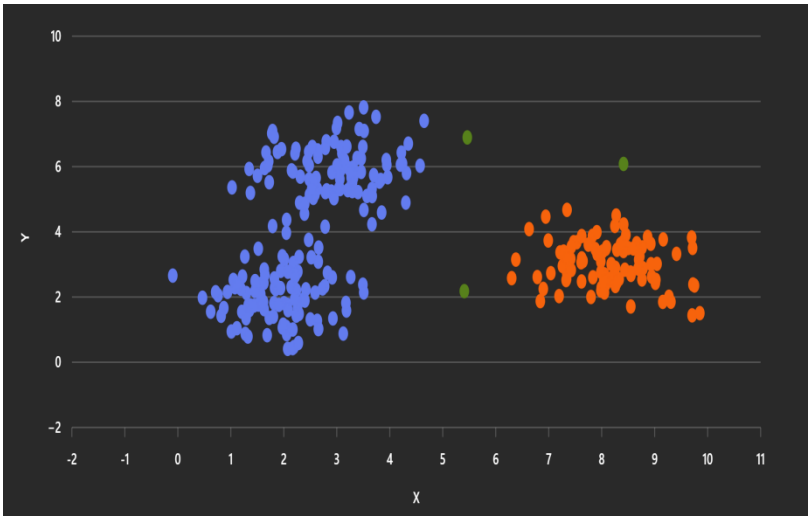
DBSCAN mengelompokkan titik-titik data yang saling berdekatan (berdasarkan jarak tertentu) dan menandai titik-titik yang berada di area dengan kepadatan rendah sebagai ***noise*** atau ***outlier***. DBSCAN membutuhkan dua parameter utama **ε** (epsilon): radius maksimum dari titik pusat untuk mencari tetangga, dan MinPts: jumlah minimum titik dalam radius ε agar dapat membentuk klaster

Kita akan menggunakan data sintetis yang merepresentasikan koordinat lokasi pelanggan, lalu menerapkan DBSCAN untuk menemukan klaster pelanggan, lalu menerapkan coding sebagai berikut:

```
# Import library yang diperlukan
import numpy as np
import pandas as pd
import plotly.express as px
from sklearn.cluster import DBSCAN
from sklearn.datasets import make_blobs


# 1. Membuat data sintetis: koordinat lokasi
pelanggan
X, _ = make_blobs(
    n_samples=300,
    centers=[[2, 2], [8, 3], [3, 6]],
```

```
    cluster_std=0.8,
    random_state=42
)

# 2. Terapkan algoritma DBSCAN
dbscan = DBSCAN(eps=0.9, min_samples=5)
labels = dbscan.fit_predict(X)

# 3. Buat DataFrame untuk visualisasi
df = pd.DataFrame(X, columns=['X', 'Y'])
df['Cluster'] = labels.astype(str)

# 4. Tandai outlier dengan label khusus
df['Cluster'] = df['Cluster'].replace({'-1':
'Outlier'})

# 5. Visualisasi hasil clustering
fig = px.scatter(
    df, x='X', y='Y', color='Cluster',
    title='Hasil Clustering DBSCAN pada
Lokasi Pelanggan',
    labels={'X': 'Koordinat X', 'Y':
'Koordinat Y'}
)
fig.show()

# 6. (Opsional) Simpan hasil ke file CSV
df.to_csv('hasil_clustering_dbscan.csv',
index=False)
```

Hasilnya sebagai berikut



Gambar 4. 2 Gambar *Klastering* menggunakan *Density Based Clustering*

Sumber: Hasil Olahan Data

4.2.4 Model-*Based Clustering*

Model-*Based Clustering* adalah pendekatan clustering yang mengasumsikan bahwa data berasal dari campuran (*mixture*) beberapa distribusi probabilistik, biasanya Gaussian (Normal). Setiap cluster direpresentasikan oleh sebuah komponen distribusi, dan proses clustering dilakukan dengan mengestimasi parameter distribusi menggunakan metode seperti *Expectation-Maximization* (EM).

Keunggulan Metode ini:

1. Menggunakan probabilitas untuk menentukan keanggotaan cluster.
2. Cocok untuk data dengan bentuk cluster yang elips atau tidak simetris.

3. Lebih fleksibel dibanding metode seperti K-Means karena mempertimbangkan variansi dan kovarians antar fitur.

Berikut contoh penerapan metode ini dengan menggunakan data Pendapatan Tahunan dan Pengeluaran Tahunan.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.patches import Ellipse
from sklearn.mixture import GaussianMixture

#
==========================================
==================
# 0) Parameter Global
#
==========================================
==================
SEED = 42
N = 120   # jumlah titik per cluster pada
simulasi
N_COMPONENTS = 3  # jumlah komponen GMM
# Kuantil chi-square untuk df = 2 (2D)
CHI2_LEVELS = {
    '68%': 2.279,
    '95%': 5.991,
    '99%': 9.210,
}
```

```
OUT_FIG = 'gmm_ellipses_detailed.png'
OUT_CSV = 'ringkasan_elips_gaussian.csv'
OUT_XLSX = 'ringkasan_elips_gaussian.xlsx'
#
==========================================
==================
# 1) Buat Data Simulasi: Pendapatan Tahunan
vs Skor Pengeluaran
#
==========================================
==================
np.random.seed(SEED)


# Cluster 1: pendapatan sedang, pengeluaran
rendah-menengah
income1   = np.random.normal(40, 5, N)
spending1 = np.random.normal(30, 5, N)


# Cluster 2: pendapatan tinggi, pengeluaran
sedang
income2   = np.random.normal(70, 8, N)
spending2 = np.random.normal(60, 8, N)


# Cluster 3: pendapatan menengah-tinggi,
pengeluaran sangat tinggi
income3   = np.random.normal(55, 6, N)
spending3 = np.random.normal(80, 6, N)


X = np.column_stack([
```

```
    np.concatenate([income1,        income2,
income3]),
    np.concatenate([spending1,    spending2,
spending3])
])
# X.shape -> (3*N, 2), berisi [Pendapatan
Tahunan, Skor Pengeluaran]
#
==========================================
=================
# 2) Fit Gaussian Mixture Model
#
==========================================
=================
gmm = GaussianMixture(
    n_components=N_COMPONENTS,
    covariance_type='full',
    random_state=SEED
)
gmm.fit(X)
labels = gmm.predict(X)         # label keras
(hard assignment)
probas    =    gmm.predict_proba(X)          #
probabilitas keanggotaan per komponen


means = gmm.means_                # (K, 2)
covariances = gmm.covariances_  # (K, 2, 2)
#
==========================================
=================
```

```
# 3) Fungsi bantu untuk gambar elips Gaussian
per komponen
#     Elips: (x-μ)^T Σ^{-1} (x-μ) = q, q =
kuantil chi-square (df=2)
#    Semi-sumbu: a = sqrt(q * λ1), b = sqrt(q
* λ2), width=2a, height=2b
#     Sudut elips = arah eigenvector utama
(derajat)
#
==========================================
==================
def ellipse_params(mean, cov, q):
    """Kembalikan width, height, angle_deg
untuk elips confidence level q."""
    eigvals, eigvecs = np.linalg.eigh(cov)
    order = np.argsort(eigvals)[::-1]     #
urutkan dari besar -> kecil
    eigvals = eigvals[order]
    eigvecs = eigvecs[:, order]

    # sudut elips (derajat) mengikuti
eigenvector utama
    angle_deg                                =
np.degrees(np.arctan2(eigvecs[1,         0],
eigvecs[0, 0]))

    # semi-sumbu
    a = np.sqrt(q * eigvals[0])
    b = np.sqrt(q * eigvals[1])

    width = 2.0 * a
```

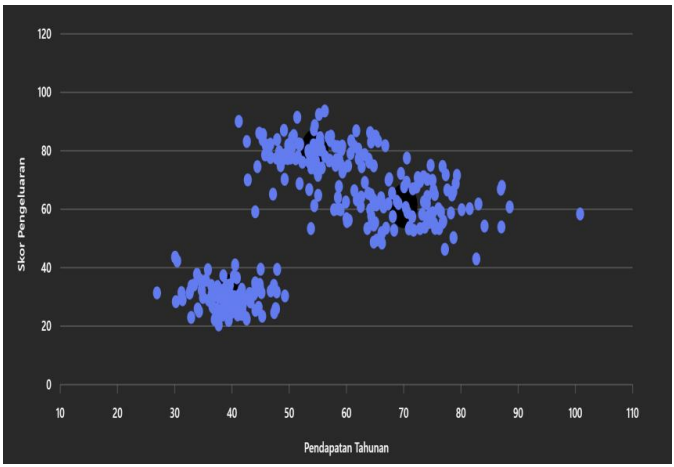
```
    height = 2.0 * b
    area = np.pi * a * b  # luas elips pada
level q

    return width, height, angle_deg, a, b,
eigvals
#
==========================================
==================
# 4) Visualisasi: Scatter + Elips Gaussian
(68%, 95%, 99%)
#
==========================================
==================
cmap = plt.get_cmap('tab10')
colors = [cmap(i) for i in
range(N_COMPONENTS)]
fig, ax = plt.subplots(figsize=(9, 7))

# Scatter seluruh titik, diwarnai sesuai
label cluster
sc = ax.scatter(
    X[:, 0], X[:, 1],
    c=labels, cmap='tab10', s=22,
alpha=0.75, edgecolors='none'
)
# Gambar elips untuk tiap komponen, semua
level q
summary_rows = []
for k in range(N_COMPONENTS):
    mean = means[k]
```

```
    cov = covariances[k]
    base_color = colors[k]

    # pusat komponen
    ax.scatter(mean[0], mean[1], marker='x',
c='black', s=90, linewidths=2.5)
    ax.text(mean[0], mean[1], f"  μ{k+1}",
fontsize=10, color='black', va='center')

    for level_name, q in
CHI2_LEVELS.items():
        width, height, angle_deg, a, b,
eigvals = ellipse_params(mean, cov, q)

        # gaya garis per level
        if level_name == '68%':
            lw, alpha_fill, ls = 1.6, 0.10,
'-'
        elif level_name == '95%':
            lw, alpha_fill, ls = 2.0, 0.12,
'--'
        else:  # '99%'
            lw, alpha_fill, ls = 2.0, 0.08,
':'
        ell = Ellipse(
xy=mean, width=width, height=height,
angle=angle_deg,
edgecolor=base_color, facecolor=base_color,
            lw=lw, ls=ls, alpha=alpha_fill
        )
```

```
            ax.add_patch(ell)

            # simpan ringkasan numerik elips
            area = np.pi * a * b
            summary_rows.append({
                'Cluster': k + 1,
                'Level': level_name,
                'Mu_X': mean[0],
                'Mu_Y': mean[1],
                'Angle_deg': angle_deg,
                'Eigval1': eigvals[0],
                'Eigval2': eigvals[1],
                'SemiAxis_a': a,
                'SemiAxis_b': b,
                'Width': width,
                'Height': height,
                'Area': area,
            })

    # Judul & label sumbu (Bahasa Indonesia)
    ax.set_title('GMM: Elips Gaussian Detail
    (68%, 95%, 99%)', fontsize=14)
    ax.set_xlabel('Pendapatan Tahunan')
    ax.set_ylabel('Skor Pengeluaran')

    # Legenda: cluster + level kontur
    from matplotlib.lines import Line2D
    legend_elements = [
```

```
    Line2D([0], [0], color='gray', lw=1.6,
linestyle='-', label='Kontur 68%'),
    Line2D([0], [0], color='gray', lw=2.0,
linestyle='--', label='Kontur 95%'),
    Line2D([0], [0], color='gray', lw=2.0,
linestyle=':', label='Kontur 99%'),
]
handles, labels_sc = sc.legend_elements()
cluster_legend = [handles[i] for i in
range(N_COMPONENTS)]
cluster_labels = [f"Cluster {i+1}" for i in
range(N_COMPONENTS)]
first_legend = ax.legend(cluster_legend,
cluster_labels, title='Cluster', loc='upper
left')
ax.add_artist(first_legend)
ax.legend(handles=legend_elements,
loc='lower right', title='Level Kontur')
ax.grid(True, alpha=0.25)
plt.tight_layout()

# Simpan grafik
plt.savefig(OUT_FIG, dpi=160)
print(f"Grafik disimpan: {OUT_FIG}")

#
==========================================
==================
# 5) Simpan Ringkasan Elips ke CSV & Excel
```

```
#
==========================================
==================

summary_df = pd.DataFrame(summary_rows)

summary_df = summary_df[

    ['Cluster',  'Level',  'Mu_X',  'Mu_Y',
'Angle_deg',

     'Eigval1',  'Eigval2',  'SemiAxis_a',
'SemiAxis_b', 'Width', 'Height', 'Area']

]

summary_df.to_csv(OUT_CSV, index=False)

print(f"Ringkasan  elips  disimpan  (CSV):
{OUT_CSV}")


# Simpan ke Excel (engine=openpyxl)

summary
```



Gambar 4. 3 Klastering menggunakan Model *Based Clustering*

Sumber: Hasil Olahan Data

4.2.5 *Fuzzy Clustering*

Fuzzy Clustering adalah metode pengelompokan data di mana setiap data dapat menjadi anggota lebih dari satu cluster dengan derajat keanggotaan tertentu. Berbeda dengan metode clustering konvensional seperti K-Means yang bersifat hard clustering (setiap data hanya masuk satu cluster), fuzzy clustering bersifat soft clustering. Metode yang paling umum digunakan adalah Fuzzy C-Means (FCM).(Everitt *et al*., 2011)

Sebagai contoh sebuah toko ingin mengelompokkan pelanggannya berdasarkan dua fitur:

- **Frekuensi kunjungan**
- **Jumlah pembelian**

Namun, pelanggan bisa memiliki karakteristik campuran, misalnya:

- Sering berkunjung tapi belanja sedikit
- Jarang berkunjung tapi belanja banyak

Dengan Fuzzy C-Means, kita bisa mengetahui **derajat keanggotaan** setiap pelanggan terhadap beberapa segmen, misalnya:

- Cluster 1: Pelanggan loyal
- Cluster 2: Pelanggan hemat
- Cluster 3: Pelanggan impulsif

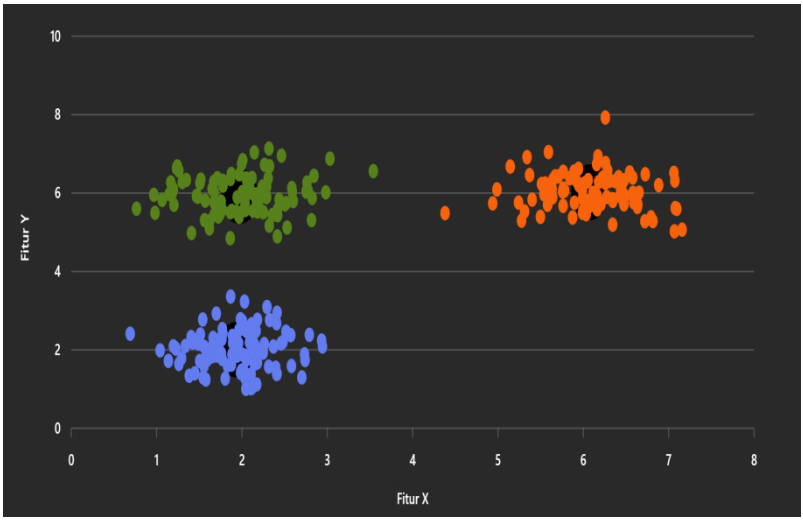
```
import numpy as np
import plotly.express as px
import plotly.graph_objects as go

# 1. Membuat data sintetis
```

```
np.random.seed(42)
data1    =    np.random.normal(loc=[2,    2],
scale=0.5, size=(100, 2))
data2    =    np.random.normal(loc=[6,    6],
scale=0.5, size=(100, 2))
data3    =    np.random.normal(loc=[2,    6],
scale=0.5, size=(100, 2))
data = np.vstack((data1, data2, data3))


# 2. Parameter Fuzzy C-Means
n_clusters = 3
m = 2.0  # Fuzziness parameter
max_iter = 100
error = 1e-5


# 3. Inisialisasi keanggotaan secara acak
n_samples = data.shape[0]
u = np.random.dirichlet(np.ones(n_clusters),
size=n_samples)


# 4. Iterasi Fuzzy C-Means
for iteration in range(max_iter):
    # Hitung pusat cluster
    um = u ** m
    centers  =  um.T  @  data  /  np.sum(um.T,
axis=1)[:, None]


    # Hitung jarak ke pusat cluster
```

```
    dist = np.linalg.norm(data[:, None] -
centers[None, :], axis=2)
    dist           =           np.fmax(dist,
np.finfo(np.float64).eps)       #    Hindari
pembagian dengan nol


    # Update keanggotaan
    u_new = 1.0 / np.sum((dist[:, :, None] /
dist[:, None, :]) ** (2 / (m - 1)), axis=2)


    # Cek konvergensi
    if np.linalg.norm(u_new - u) < error:
        break
    u = u_new


# 5. Tentukan cluster berdasarkan
keanggotaan tertinggi
labels = np.argmax(u, axis=1)


# 6. Visualisasi hasil clustering
fig = px.scatter(x=data[:, 0], y=data[:, 1],
color=labels.astype(str),
                 labels={'x': 'Fitur X',
'y': 'Fitur Y', 'color': 'Cluster'},
                 title='Hasil Clustering
dengan Metode Fuzzy C-Means')


# Tambahkan pusat cluster ke grafik
fig.add_trace(go.Scatter(x=centers[:, 0],
y=centers[:, 1],
```

```
                              mode='markers',
marker=dict(color='black',          size=12,
symbol='x'),
                              name='Pusat
Cluster'))


fig.show()
```

didapat hasil kluster sebagai berikut



Gambar 4. 4 Klastering menggunakan Fuzzy Clustering

Sumber: Hasil Olahan Data

4.3 *Association Rules*

Association Rule Mining adalah teknik untuk menemukan hubungan atau pola *if-then* antar item dalam dataset besar. Aturan ini biasanya dievaluasi menggunakan metrik seperti support, *confidence*, dan lift (Han *et al.*, 2012). Contoh aturan: {Roti, Susu} → {Mentega} berarti jika seseorang membeli roti dan susu, kemungkinan besar juga membeli mentega

4.4 Jenis-Jenis *Association Rules*

4.4.1 *General Association Rules*

Berikut contoh nyata penerapan *General Association Rules* (aturan asosiasi umum) lengkap dengan coding Python yang dapat langsung dijalankan. Dengan menggunakan skenario market basket analysis di minimarket (mis. melihat pola belanja: *Indomie*, *Telur*, *Minyak Goreng*, *Gula*, dll.)

```
# Implementasi Apriori sederhana + pembentukan Association Rules
from itertools import combinations
import pandas as pd
import math

# ---- 1) Data transaksi (contoh minimarket Indonesia) ----
transactions = [
    ['Indomie','Telur','Minyak Goreng','Saus'],
    ['Indomie','Telur','Aqua','Gula'],
    ['Kopi','Gula','Susu'],
    ['Rokok','Aqua'],
    ['Indomie','Minyak Goreng','Gula'],
    ['Teh','Gula','Susu'],
    ['Kopi','Rokok','Aqua'],
    ['Indomie','Telur','Minyak Goreng'],
    ['Aqua','Susu'],
    ['Indomie','Saus','Aqua'],
```

```
    ['Telur','Minyak Goreng','Gula'],
    ['Indomie','Telur','Gula','Susu'],
    ['Kopi','Gula'],
    ['Teh','Gula','Rokok'],
    ['Indomie','Minyak
Goreng','Telur','Saus'],
    ['Shampo','Sabun','Pasta Gigi'],
    ['Sabun','Shampo','Pasta Gigi'],
    ['Indomie','Aqua','Gula'],
    ['Rokok','Korek'],
    ['Kopi','Teh','Gula'],
]

n_tx = len(transactions)

# ---- 2) Apriori ----
from collections import defaultdict
def apriori(transactions, min_support=0.20):
    n_tx = len(transactions)
    # Hitung 1-itemset
    item_counts = defaultdict(int)
    for tx in transactions:
        for item in set(tx):
            item_counts[tuple([item])] += 1

    # L1
    L = {k for k,v in item_counts.items() if
v / n_tx >= min_support}
```

```
    freq_counts = {k: item_counts[k] for k
in L}
    k = 2
    current_L = L

    while current_L:
        # Candidate generation dari frequent
(k-1)-itemsets
        current_list = [set(x) for x in
current_L]
        Ck = set()
        for i in range(len(current_list)):
            for j in range(i+1,
len(current_list)):
                union = current_list[i] |
current_list[j]
                if len(union) == k:
                    # pruning: semua subset
(k-1) harus frequent
                    all_subsets_frequent =
all(tuple(sorted(s)) in freq_counts

for s in combinations(union, k-1))
                    if
all_subsets_frequent:

Ck.add(tuple(sorted(union)))

        # Hitung support kandidat
        counts_k = defaultdict(int)
        for tx in transactions:
```

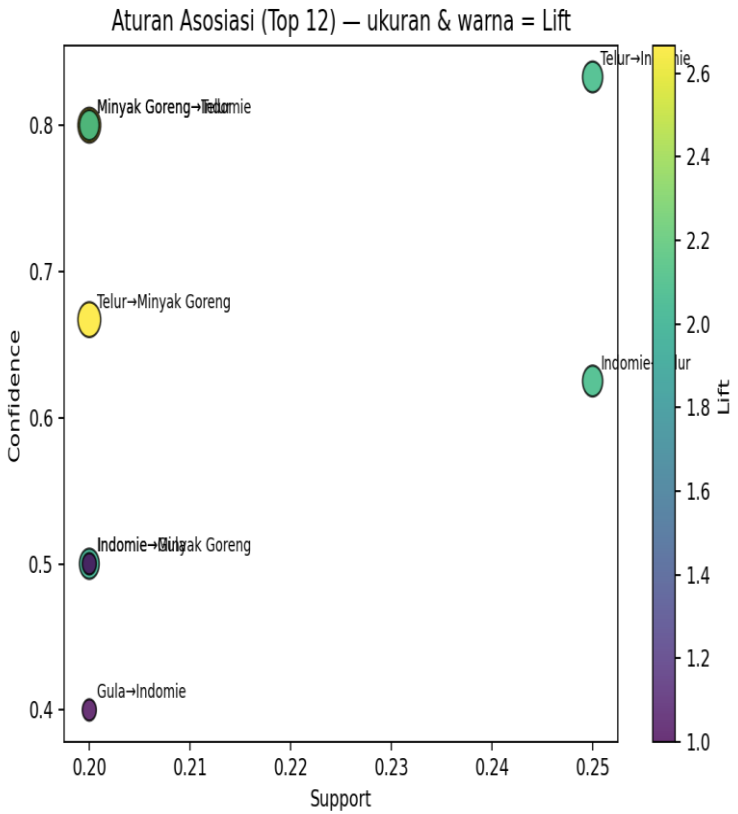
```
            s_tx = set(tx)
            for cand in Ck:
                if
set(cand).issubset(s_tx):
                    counts_k[cand] += 1

        # Lk
        Lk    =    {c    for    c,count    in
counts_k.items()   if   count   /   n_tx   >=
min_support}
        for c in Lk:
            freq_counts[c] = counts_k[c]

        current_L = Lk
        k += 1

    return freq_counts, n_tx

def get_support(itemset, counts, n_tx):
    return
counts.get(tuple(sorted(itemset)), 0) / n_tx

def build_rules(freq_counts, n_tx):
    rows = []
    for       items,       count_xy       in
freq_counts.items():
        if len(items) < 2:
            continue
        sup_xy = count_xy / n_tx
```

```
        items_set = set(items)
        # semua subset non-kosong dan bukan keseluruhan
        for r in range(1, len(items)):
            for antecedent in combinations(items, r):
                antecedent = set(antecedent)
                consequent = items_set - antecedent
                sup_x = get_support(antecedent, freq_counts, n_tx)
                sup_y = get_support(consequent, freq_counts, n_tx)
                if sup_x == 0 or sup_y == 0:
                    continue
                confidence = sup_xy / sup_x
                lift = confidence / sup_y
                leverage = sup_xy - (sup_x * sup_y)
                conviction = (1 - sup_y) / (1 - confidence) if confidence < 1 else float('inf')
                rows.append({
                    'antecedent': tuple(sorted(antecedent)),
                    'consequent': tuple(sorted(consequent)),
                    'support': sup_xy,
                    'confidence': confidence,
                    'lift': lift,
```

```
                    'leverage': leverage,
                    'conviction':
conviction if math.isfinite(conviction) else
float('inf'),
                })
    df = pd.DataFrame(rows)
    if not df.empty:
        df                                  =
df.sort_values(['lift','confidence','suppor
t'],
ascending=False).reset_index(drop=True)
    return df


# Jalankan
freq_counts,   n   =   apriori(transactions,
min_support=0.20)
rules_df = build_rules(freq_counts, n)


# Tampilkan 10 aturan teratas
print(rules_df.head(10).round(3).to_string(
index=False))


# (Opsional) Visualisasi cepat
import matplotlib.pyplot as plt
top = rules_df.head(12).copy()
plt.figure(figsize=(7,5))
sc      =      plt.scatter(top['support'],
top['confidence'],
                 s=(top['lift']*80),
c=top['lift'],
```

```
                cmap='viridis', alpha=0.8,
edgecolors='k')

for _,row in top.iterrows():

    label                                    =
f\"{','.join(row['antecedent'])}→{','.join(
row['consequent'])}\"

    plt.annotate(label,     (row['support'],
row['confidence']),

                xytext=(5,5),
textcoords='offset points', fontsize=8)

plt.title('Aturan Asosiasi (Top 12) — ukuran
& warna = Lift')

plt.xlabel('Support');
plt.ylabel('Confidence')

cb = plt.colorbar(sc); cb.set_label('Lift')

plt.tight_layout(); plt.show()
``
```

Hasilnya sebagai berikut.



Gambar 4. 5 ***General Association Rules***

Sumber: Hasil Olahan Data

4.4.2 *Quantitative Association Rules*

Berbeda dari aturan asosiasi klasik (Apriori/Market Basket) yang umumnya berbasis biner (ada/tidak ada item), QAR bekerja dengan atribut numerik: jumlah unit, nilai belanja (Rp), berat, durasi, dll

Contoh Aturan QAR

Jika Milk_qty ≥ 3 **dan** Bread_qty ≥ 2 **maka** Total_spend ≥ Rp150.000

dengan **confidence** 75% dan **lift** 2,7 (artinya 2,7× lebih mungkin dibanding baseline).

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from itertools import combinations

np.random.seed(42)

# --- 1) Bangun dataset transaksi ritel sintetis ---
N = 1000
PRICES = {
    'Milk_qty': 20000,    # harga susu per unit (IDR)
    'Bread_qty': 15000,  # roti per unit
    'Eggs_qty': 2000,    # telur per butir
    'Apple_qty': 5000    # apel per buah
}

rows = []
for i in range(N):
    # Kuantitas dasar (Poisson) agar realistis
    milk   = np.random.poisson(1.5)
    bread  = np.random.poisson(1.2)
    eggs   = np.random.poisson(8)
    apples = np.random.poisson(2.5)
```

```
    # Pola tertanam: pelanggan "keluarga"
cenderung beli susu & roti lebih banyak
    if np.random.rand() < 0.32:
        milk   += np.random.randint(2, 5)  #
dorong ke >=3
        bread  += np.random.randint(1, 3)  #
dorong ke >=2
        eggs   += np.random.randint(2, 6)
        apples += np.random.randint(0, 3)


    # Pola lain: telur banyak (>=12) sering
berkorelasi dengan belanja besar
    if eggs >= 12:
        milk   += np.random.randint(0, 2)
        bread  += np.random.randint(0, 2)
        apples += np.random.randint(0, 2)


    # Hindari negatif
    milk   = max(milk, 0)
    bread  = max(bread, 0)
    eggs   = max(eggs, 0)
    apples = max(apples, 0)


    total   =   (milk*PRICES['Milk_qty']   +
bread*PRICES['Bread_qty'] +
             eggs*PRICES['Eggs_qty']       +
apples*PRICES['Apple_qty'])
    total     =     max(0,     total     +
np.random.normal(0, 5000))  # noise harga
```

```
    rows.append({
        'Milk_qty': milk,
        'Bread_qty': bread,
        'Eggs_qty': eggs,
        'Apple_qty': apples,
        'Total_spend': total
    })

df = pd.DataFrame(rows)

# --- 2) Definisikan ambang kuantitatif ->
fitur boolean ---
thresholds = {
    'Milk_qty':    [1, 3],
    'Bread_qty':   [1, 2],
    'Eggs_qty':    [6, 12],
    'Apple_qty':   [3, 5],
    'Total_spend': [50000, 100000, 150000]
# dalam Rupiah
}

bool_feats = {}
for col, cuts in thresholds.items():
    for c in cuts:
        name = f"{col}>={c}"
        bool_feats[name] = (df[col] >= c)
```

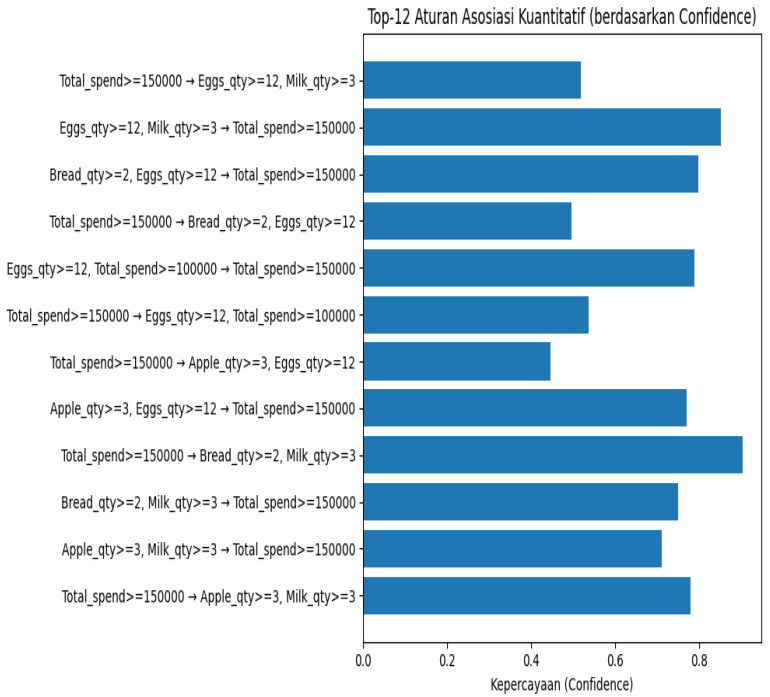
```
T = pd.DataFrame(bool_feats)   # transaksi
biner hasil kuantisasi


# --- 3) Apriori sederhana untuk frequent
itemsets ---
min_support = 0.12  # syarat dukungan minimal
(12%)


supports_cache = {}
def support(itemset):
    key = tuple(sorted(itemset))
    if key in supports_cache:
        return supports_cache[key]
    s               =               T.loc[:,
list(key)].all(axis=1).mean()
    supports_cache[key] = s
    return s


# 1-itemsets
items = list(T.columns)
freq1 = []
for it in items:
    s = support([it])
    if s >= min_support:
        freq1.append(([it], s))


# 2-itemsets (kombinasi dari freq1)
freq2 = []
```

```
for a, b in combinations([i[0][0] for i in
freq1], 2):
    s = support([a, b])
    if s >= min_support:
        freq2.append(([a, b], s))


# 3-itemsets (opsional, agar contoh kaya)
freq3 = []
base_items = sorted(set([i[0][0] for i in
freq1]))
for combo in combinations(base_items, 3):
    s = support(list(combo))
    if s >= min_support:
        freq3.append((list(combo), s))


# --- 4) Bangun aturan asosiasi (Antecedent
-> Consequent) ---
def all_proper_subsets(itemset):
    subs = []
    for r in range(1, len(itemset)):
        for s in combinations(itemset, r):
            subs.append(list(s))
    return subs


all_freq = freq1 + freq2 + freq3
freq_sets = [fs for fs, s in all_freq if
len(fs) >= 2]


rules = []
```

```
for fs in freq_sets:
    s_fs = support(fs)
    for A in all_proper_subsets(fs):
        B = sorted(list(set(fs) - set(A)))
        if not B:
            continue
        s_A = support(A)
        s_B = support(B)
        conf = s_fs / s_A if s_A > 0 else 0.0
        lift = conf / s_B if s_B > 0 else
np.nan
        rules.append({
            'Antecedent': A,
            'Consequent': B,
            'Support': s_fs,
            'Confidence': conf,
            'Lift': lift
        })

rules_df = pd.DataFrame(rules)
rules_top = rules_df.sort_values(['Lift',
'Confidence',                    'Support'],
ascending=False).head(12)

# --- 5) Visualisasi Top-12 aturan
berdasarkan Confidence ---
plt.figure(figsize=(10, 6))
```

```
labels = [f"{', '.join(a)} → {', '.join(c)}" for a, c in zip(rules_top['Antecedent'], rules_top['Consequent'])]

conf_vals = rules_top['Confidence'].values

plt.barh(range(len(labels)), conf_vals, color='#1f77b4')

plt.yticks(range(len(labels)), labels)

plt.gca().invert_yaxis()

plt.xlabel('Kepercayaan (Confidence)')

plt.title('Top-12 Aturan Asosiasi Kuantitatif (berdasarkan Confidence)')

plt.tight_layout()

plt.savefig('top_rules_confidence.png')


# Simpan tabel hasil ke CSV

rules_top[['Antecedent','Consequent','Support','Confidence','Lift']].to_csv('top_rules.csv', index=False)

print(rules_top[['Antecedent','Consequent','Support','Confidence','Lift']].to_string(index=False))
```

Hasilnya seperti ini:



Gambar 4. 6 *Quatitative Association Rules*

Sumber: Hasil Olahan Data

4.4.3 *Multi-Relational Association Rules*

Inti MRAR: berbeda dari market basket biasa (satu tabel transaksi), MRAR mengekstraksi aturan yang dapat memadukan atribut pelanggan, atribut pesanan, dan item—misalnya: Jika `SEGMENT=Family` dan `CITY=Yogyakarta` maka cenderung membeli `BOUGHT:Groceries` dan `BOUGHT: Baby` (dukungan, kepercayaan, lift, dsb).

Tahapan

1. Bangun data relasional kecil (pelanggan, produk, pesanan, item pesanan).

2. Propositionalization: ubah tiap pesanan menjadi *itemset* yang berisi:
   - Atribut pelanggan (CITY=…, SEGMENT=…, AGE_GRP=…, GENDER=…)
   - Atribut pesanan (WEEKEND=Yes/No, HIGH_VALUE=Yes/No)
   - Kategori produk yang benar-benar dibeli (BOUGHT:Electronics, BOUGHT:Groceries, dst.)
3. Jalankan Apriori sederhana untuk mendapat frequent itemsets.
4. Bentuk association rules lintas relasi + hitung support / confidence / lift / conviction.

```
# -*- coding: utf-8 -*-
"""
Contoh penerapan Multi-Relational Association Rules (MRAR) di Python
-------------------------------------------------------------------
Kita membangun dataset relasional kecil (pelanggan, pesanan, item pesanan, produk),
melakukan *propositionalization* (mengubah relasi menjadi fitur per transaksi),
kemudian menambang association rules yang melibatkan atribut lintas-tabel
(misal: kota pelanggan, segmen, hari pesanan, kategori produk yang dibeli, dsb.).
```

```
Catatan: Implementasi Apriori di bawah
ini dibuat ringan agar mandiri tanpa
ketergantungan paket eksternal seperti
mlxtend.
"""

import random
from itertools import combinations
import pandas as pd
import numpy as np

random.seed(42)
np.random.seed(42)

# ----------------------------
# 1) Bangun data relasional
# ----------------------------
customers = pd.DataFrame([
    {"customer_id": f"C{i}", "gender":
g, "age": a, "city": c, "segment": s}
    for i, (g, a, c, s) in enumerate([
        ("F",       32,       "Semarang",
"Family"),
        ("M",       23,       "Semarang",
"Single"),
        ("F",      28,      "Yogyakarta",
"Family"),
        ("M",       26,       "Jakarta",
"Single"),
```

```
        ("F",       41,       "Semarang",
"Family"),
        ("M",      21,      "Yogyakarta",
"Single"),
        ("F",       36,       "Jakarta",
"Family"),
        ("M",       24,       "Semarang",
"Single"),
        ("F",      30,      "Yogyakarta",
"Family"),
        ("M",       27,       "Semarang",
"Single"),
        ("F",       45,       "Jakarta",
"Family"),
        ("M",      22,      "Yogyakarta",
"Single"),
    ], start=1)
])

# Kelompok usia
bins = [0, 25, 35, 50, 100]
labels  =  ["18-25",  "26-35",  "36-50",
">50"]
customers["age_group"]                  =
pd.cut(customers["age"],      bins=bins,
labels=labels, right=True)

products = pd.DataFrame([
    ("P1", "Electronics", 500),
    ("P2", "Accessories", 50),
    ("P3", "Baby", 80),
```

```
    ("P4", "Groceries", 30),
    ("P5", "Beverages", 10),
    ("P6", "Snacks", 8),
    ("P7", "Books", 20),
    ("P8", "Home", 60),
    ("P9", "PersonalCare", 15),
], columns=["product_id", "category",
"price"])


# Pemetaan kategori -> product_id dan
harga (satu produk per kategori)
cat_to_pid                              =
products.groupby("category").first()["p
roduct_id"].to_dict()
cat_to_price                            =
products.groupby("category").first()["p
rice"].to_dict()


# Buat pesanan dengan pola yang disengaja
agar ada rules yang menarik
orders = []
order_items = []
order_id_counter = 1


# Helper: pilih kategori dengan
probabilitas (berdasarkan atribut
pelanggan & akhir pekan)
def pick_categories(cust_row,
is_weekend):
    cats = []
```

```
    # Pola 1: Family di Semarang/Yogyakarta cenderung beli Baby + Groceries + Beverages
    if cust_row.segment == "Family" and cust_row.city in ["Semarang", "Yogyakarta"]:
        if random.random() < 0.8:
            cats += ["Baby", "Groceries"]
        if random.random() < 0.7:
            cats += ["Beverages"]
    # Pola 2: Laki-laki 18-25 cenderung Electronics + Accessories + Beverages
    if cust_row.gender == "M" and cust_row.age_group == "18-25":
        if random.random() < 0.75:
            cats += ["Electronics", "Accessories"]
        if random.random() < 0.6:
            cats += ["Beverages"]
    # Pola 3: Single di Yogyakarta suka Books + Snacks
    if cust_row.segment == "Single" and cust_row.city == "Yogyakarta":
        if random.random() < 0.7:
            cats += ["Books", "Snacks"]
    # Pola 4: Akhir pekan sering ada Snacks + Beverages
    if is_weekend:
        if random.random() < 0.7:
```

```
            cats      +=      ["Snacks",
"Beverages"]
    # Tambahan variasi umum
    if random.random() < 0.3:
        cats += ["PersonalCare"]
    if random.random() < 0.25:
        cats += ["Home"]
    # Pastikan unik
    return sorted(list(set(cats)))


# Generate ~80 orders tersebar di 12
pelanggan, sebagian di akhir pekan
for _ in range(80):
    cust      =      customers.sample(1,
random_state=random.randint(0,
10)).iloc[0]
    is_weekend = random.random() < 0.4
    cats     =     pick_categories(cust,
is_weekend)
    if not cats:
        # fallback minimal
        cats                              =
[random.choice(list(cat_to_pid.keys()))
]
    oid = f"O{order_id_counter}"
    orders.append({
        "order_id": oid,
        "customer_id": cust.customer_id,
        "is_weekend": is_weekend,
    })
```

```
    for cat in cats:
        order_items.append({
            "order_id": oid,
            "product_id":
cat_to_pid[cat],
            "category": cat,
            "price": cat_to_price[cat],
            "qty": 1,
        })
    order_id_counter += 1


orders = pd.DataFrame(orders)
order_items = pd.DataFrame(order_items)


# ----------------------------
# 2) Propositionalization (ubah relasi →
itemset transaksi)
# ----------------------------
# Gabungkan atribut pelanggan ke setiap
order
orders_full  =  orders.merge(customers,
on="customer_id", how="left")


# Agregasi items per order
cats_per_order                          =
order_items.groupby("order_id")["catego
ry"].apply(list).reset_index()
value_per_order = (order_items
```

```
.assign(line_total=lambda d: d["price"]
* d["qty"])

.groupby("order_id")["line_total"].sum(
)

.reset_index(name="order_value"))
orders_full = (orders_full
               .merge(cats_per_order,
on="order_id", how="left")
               .merge(value_per_order,
on="order_id", how="left"))


# Bentuk itemset per order (lintas tabel)
def order_to_items(row):
    items = set()
    # Atribut pelanggan
    items.add(f"CITY={row.city}")
    items.add(f"SEGMENT={row.segment}")
    items.add(f"GENDER={row.gender}")

items.add(f"AGE_GRP={row.age_group}")
    # Atribut order
    items.add("WEEKEND=Yes"           if
row.is_weekend else "WEEKEND=No")
    items.add("HIGH_VALUE=Yes"        if
row.order_value     >=     200     else
"HIGH_VALUE=No")
    # Kategori produk yang hadir dalam
order tsb
```

```
    for c in set(row.category):
        items.add(f"BOUGHT:{c}")
    return items


orders_full["itemset"] =
orders_full.apply(order_to_items,
axis=1)


transactions = orders_full[["order_id",
"itemset"]]
num_tx = len(transactions)


# ----------------------------
# 3) Implementasi Apriori sederhana
# ----------------------------
def get_support_counts(transactions,
candidates):
    counts = {cand: 0 for cand in
candidates}
    for s in transactions["itemset"]:
        for cand in candidates:
            if cand.issubset(s):
                counts[cand] += 1
    return counts


# Inisialisasi L1 (frekuen 1-itemset)
item_counts = {}
for s in transactions["itemset"]:
    for it in s:
```

```
        item_counts[it]                 =
item_counts.get(it, 0) + 1


min_support = 0.2  # 20%

min_conf = 0.6     # 60%


L = []  # daftar dict: {frozenset(items):
support}

L1 = {frozenset([k]): v/num_tx for k, v
in  item_counts.items()  if  v/num_tx  >=
min_support}

L.append(L1)

k = 2

current_L = L1

while current_L:

    prev_itemsets                       =
list(current_L.keys())

    # Join step: buat kandidat k-itemset

    Ck = set()

    for i in range(len(prev_itemsets)):

        for     j     in     range(i+1,
len(prev_itemsets)):

            union  =  prev_itemsets[i]  |
prev_itemsets[j]

            if len(union) == k:

                # Prune: semua subset
(k-1) harus frekuen

                all_subsets_freq       =
all((frozenset(ss) in current_L) for ss
in combinations(union, k-1))

                if all_subsets_freq:
```

```
                    Ck.add(union)
    if not Ck:
        break
    # Hitung support kandidat
    counts                              =
get_support_counts(transactions, Ck)
    Lk = {cand: cnt/num_tx for cand, cnt
in  counts.items()  if  cnt/num_tx  >=
min_support}
    if Lk:
        L.append(Lk)
        current_L = Lk
        k += 1
    else:
        break

# Gabungkan semua frequent itemsets
freq_itemsets = {}
for level in L:
    freq_itemsets.update(level)

# ----------------------------
#  4)  Bangun  association  rules  dari
frequent itemsets
# ----------------------------
# Peta support untuk akses cepat
support_map = freq_itemsets.copy()

rules = []
```

```
for itemset, supp_xy in freq_itemsets.items():
    if len(itemset) < 2:
        continue
    items = list(itemset)
    # Semua subset non-kosong sebagai antecedent
    for r in range(1, len(items)):
        for A in combinations(items, r):
            A = frozenset(A)
            B = itemset - A
            supp_x = support_map.get(A, None)
            supp_y = support_map.get(B, None)
            if supp_x is None or supp_y is None:
                # Safety: hitung manual jika subset tidak ada di peta (harusnya ada karena apriori)
                supp_x = get_support_counts(transactions, [A])[A] / num_tx
                supp_y = get_support_counts(transactions, [B])[B] / num_tx
            conf = supp_xy / supp_x if supp_x > 0 else 0
            if conf >= min_conf:
                lift = conf / supp_y if supp_y > 0 else np.inf
```

```
                conviction = (1 -
supp_y) / (1 - conf) if conf < 1 else
np.inf
                rules.append({
                    "antecedent":
tuple(sorted(A)),
                    "consequent":
tuple(sorted(B)),
                    "support":
round(supp_xy, 3),
                    "confidence":
round(conf, 3),
                    "lift": round(lift,
3),
                    "conviction":
round(conviction,       3)       if
np.isfinite(conviction) else np.inf,
                    "len": len(itemset)
                })

rules_df                                =
pd.DataFrame(rules).sort_values(["lift"
,       "confidence",       "support"],
ascending=False).reset_index(drop=True)

# (Opsional) Lihat ringkasan & 15 rules
teratas
summary = {
    "jumlah_pelanggan": len(customers),
    "jumlah_orders": len(orders),
```

```
    "rata_item_per_order":
round(order_items.groupby("order_id").s
ize().mean(), 2),
    "min_support":          min_support,
"min_confidence": min_conf
}


print("Ringkasan:", summary)
print("\nContoh struktur order (gabungan
lintas tabel):")
print(orders_full.head(2)[["order_id","
customer_id","city","segment","gender",
"age_group","is_weekend","order_value",
"category"]])


print("\nTop   15   Association   Rules
(lintas relasi):")
print(rules_df.head(15))
```

4.4.4 *Generalized Association Rules*

GAR menambang aturan asosiasi dengan memanfaatkan **taksonomi/hirarki** barang—bukan hanya di level item, tetapi juga di level kategori (mis: *Produk Susu → Roti & Biskuit*). Ini berguna untuk:

- Mengurangi sparsititas (item sangat spesifik), sehingga pola tetap terlihat di level kategori.
- Menangkap substitusi antar merek (mis: Indomie vs Mie Sedaap) karena keduanya naik ke kategori Mie Instan.
- Membantu perencanaan promosi bundling, planogram, dan cross-selling di toko.

Sebagai contoh

Transaksi: 10 keranjang belanja realistis (disintesis) berisi produk populer.

Taksonomi: Dua level kategori (L1 dan L2), misalnya:

- *Indomie Goreng* → Mie Instan (L1) → Makanan Kering (L2)
- *Susu UHT* → Produk Susu (L1) → Dairy (L2)
- *Roti Tawar* → Roti & Biskuit (L1) → Bakery & Snack (L2), dst.

```
# Contoh penerapan Generalized Association Rules (GAR)

# Dataset transaksi minimarket (disintesis) + taksonomi produk dua level


from itertools import combinations

from collections import defaultdict

# 1) Data transaksi (disintesis agar realistis untuk minimarket)

transactions = [

    ['Indomie Goreng', 'Teh Botol Sosro', 'Roti Tawar', 'Telur Ayam'],

    ['Mie Sedaap Kari', 'Air Mineral 600ml', 'Susu UHT', 'Roti Tawar'],

    ['Teh Celup Sariwangi', 'Gula Pasir', 'Kopi Kapal Api'],

    ['Susu Kental Manis', 'Kopi Kapal Api', 'Roti Sobek'],

    ['Indomie Goreng', 'Saus Sambal', 'Telur Ayam', 'Minyak Goreng'],
```

```
    ['Mie Sedaap Ayam Bawang', 'Teh Botol
Sosro', 'Air Mineral 600ml'],

    ['Yogurt Cimory', 'Buah Pisang', 'Roti
Tawar'],

    ['Beras 5kg', 'Minyak Goreng', 'Gula
Pasir'],

    ['Susu UHT', 'Biskuit Roma Kelapa', 'Teh
Celup Sariwangi'],

    ['Indomie Kuah', 'Telur Ayam', 'Sawi
Hijau', 'Cabe Rawit'],

]


# 2) Taksonomi: item -> kategori level-1 ->
kategori level-2

taxonomy = {

    'Indomie Goreng': ['Mie Instan',
'Makanan Kering'],

    'Mie Sedaap Kari': ['Mie Instan',
'Makanan Kering'],

    'Mie Sedaap Ayam Bawang': ['Mie Instan',
'Makanan Kering'],

    'Indomie Kuah': ['Mie Instan', 'Makanan
Kering'],

    'Teh Botol Sosro': ['Minuman Teh',
'Minuman'],

    'Teh Celup Sariwangi': ['Minuman Teh',
'Minuman'],

    'Air Mineral 600ml': ['Air Mineral',
'Minuman'],

    'Roti Tawar': ['Roti & Biskuit', 'Bakery
& Snack'],
```

```
    'Roti Sobek': ['Roti & Biskuit', 'Bakery
& Snack'],
    'Biskuit Roma Kelapa': ['Roti &
Biskuit', 'Bakery & Snack'],
    'Telur Ayam': ['Telur', 'Bahan Segar'],
    'Susu UHT': ['Produk Susu', 'Dairy'],
    'Yogurt Cimory': ['Produk Susu',
'Dairy'],
    'Susu Kental Manis': ['Produk Susu',
'Dairy'],
    'Gula Pasir': ['Bahan Pokok',
'Groceries'],
    'Kopi Kapal Api': ['Kopi', 'Minuman'],
    'Saus Sambal': ['Bumbu & Minyak', 'Bahan
Dapur'],
    'Minyak Goreng': ['Bumbu & Minyak',
'Bahan Dapur'],
    'Beras 5kg': ['Bahan Pokok',
'Groceries'],
    'Buah Pisang': ['Buah', 'Bahan Segar'],
    'Sawi Hijau': ['Sayur', 'Bahan Segar'],
    'Cabe Rawit': ['Sayur', 'Bahan Segar'],
}


# Level setiap label (0=item, 1=kategori L1,
2=kategori L2)
label_level = defaultdict(lambda: 0)
for item, anc in taxonomy.items():
    label_level[item] = 0
    label_level[anc[0]] = 1
```

```
    label_level[anc[1]] = 2


# Fungsi untuk mengambil seluruh "ancestor"
(kategori) dari sebuah item
def ancestors(item):
    return taxonomy.get(item, [])


#  3)  Generalisasi  transaksi:  tambahkan
kategori L1 & L2 untuk setiap item
generalized_transactions = []
for t in transactions:
    expanded = set()
    for it in t:
        expanded.add(it)  # item asli
        for anc in ancestors(it):
            expanded.add(anc)  # kategori L1
dan L2

generalized_transactions.append(sorted(expa
nded))


n_trans = len(generalized_transactions)


# 4) Apriori sederhana (frequent itemsets)
def support_count(itemset, trans_list):
    cnt = 0
    for t in trans_list:
        if all(x in t for x in itemset):
            cnt += 1
```

```
    return cnt


def     apriori(trans_list,     min_support,
k_max=3):
    min_sup_count    =    int(min_support    *
len(trans_list) + 1e-9)
    items = sorted({x for t in trans_list for
x in t})
    L = []
    C1  =  [(tuple([i]),  support_count([i],
trans_list)) for i in items]
    L1 = {i: c for (i, c) in C1 if c >=
min_sup_count}
    if L1:
        L.append(L1)
    k = 2
    while k <= k_max and L and L[-1]:
        prev_Lk_1 = list(L[-1].keys())
        # Kandidat via self-join
        candidates = set()
        for a, b in combinations(prev_Lk_1,
2):
            sa, sb = list(a), list(b)
            sa.sort(); sb.sort()
            if sa[:-1] == sb[:-1]:
                cand = tuple(sorted(set(a) |
set(b)))
                if len(cand) == k:
                    candidates.add(cand)
        # Pruning subset
```

```
        pruned_candidates = []
        prev_sets = set(prev_Lk_1)
        for cand in candidates:
            all_subsets_frequent = True
            for sub in combinations(cand, k-
1):
                if tuple(sorted(sub)) not in
prev_sets:
                    all_subsets_frequent   =
False
                    break
            if all_subsets_frequent:

pruned_candidates.append(cand)
        # Hitung support
        Lk = {}
        for cand in pruned_candidates:
            c     =     support_count(cand,
trans_list)
            if c >= min_sup_count:
                Lk[cand] = c
        if Lk:
            L.append(Lk)
        k += 1
    # Gabungkan semua frequent itemsets
    all_fis = {}
    for level_dict in L:
        for kset, c in level_dict.items():
            all_fis[tuple(sorted(kset))] = c
```

```
    return all_fis


# 5) Generate association rules (X -> Y)
def        generate_rules(frequent_itemsets,
trans_list, min_conf=0.6):
    rules = []
    for       itemset,       sup_xy       in
frequent_itemsets.items():
        if len(itemset) < 2:
            continue
        for r in range(1, len(itemset)):
            for  X  in  combinations(itemset,
r):
                Y                          =
tuple(sorted(set(itemset) - set(X)))
                sup_x   =   support_count(X,
trans_list)
                conf  =  sup_xy  /  sup_x  if
sup_x > 0 else 0.0
                sup      =      sup_xy      /
len(trans_list)
                if conf >= min_conf:
                    rules.append({
                        'X':
tuple(sorted(X)),
                        'Y': Y,
                        'support': sup,
                        'confidence': conf,
                        'support_count':
sup_xy,
```

```
                        'antecedent_count':
sup_x,
                    })
    rules.sort(key=lambda                r:
(r['confidence'],            r['support']),
reverse=True)
    return rules


# 6) Jalankan Apriori + Rules
min_support  =  0.3    #  30%  (>=3  dari  10
transaksi)
min_conf = 0.6     # 60%
frequent_itemsets                            =
apriori(generalized_transactions,
min_support=min_support, k_max=3)
rules  =  generate_rules(frequent_itemsets,
generalized_transactions,
min_conf=min_conf)


# 7) (Opsional) Filter aturan struktural
(ancestor-descendant) agar lebih actionable
# Bangun peta parent untuk semua label
parent = {}
for item, (l1, l2) in taxonomy.items():
    parent[item] = l1
    parent[l1] = l2
    parent.setdefault(l2, None)


def is_ancestor(a, b):
    cur = b
```

```
    visited = set()
    while cur is not None and cur not in
visited:
        visited.add(cur)
        if cur == a:
            return True
        cur = parent.get(cur, None)
    return False


non_structural_rules = []
for r in rules:
    drop = False
    for x in r['X']:
        for y in r['Y']:
            if    is_ancestor(x,    y)    or
is_ancestor(y, x):
                drop = True
                break
        if drop:
            break
    if not drop:
        non_structural_rules.append(r)


# Urutkan aturan non-struktural
non_structural_rules.sort(key=lambda      r:
(r['confidence'],            r['support']),
reverse=True)


# Cetak contoh aturan non-struktural teratas
```

```
for i, r in enumerate(non_structural_rules[:10], 1):
    print(f"{i:02d}. {{ {', '.join(r['X'])} }} -> {{ {', '.join(r['Y'])} }} "
          f"support={r['support']:.2f}, confidence={r['confidence']:.2f}"
```

DAFTAR PUSTAKA

Abdurrahman, D.D., Agus, F., Putra, G.M., 2021. Implementasi Algoritma Partitioning Around Medoids (PAM) untuk Mengelompokkan Hasil Produksi Komoditi Perkebunan (Studi Kasus : Dinas Perkebunan Provinsi Kalimantan Timur) 16.

Everitt, B.S., Landau, S., Leese, M., Stahl, D., 2011. Cluster Analysis, Wiley Series in Probability and Statistics. Wiley. https://doi.org/10.1002/9780470977811

Han, J., Pei, J., Kamber, M., 2012. Data Mining: Concept and Techniques. Morga Kauffman.

Purnamasari, S.M., 2010. Analisis Kelompok. Bandung.

Yewang, C., Lianlian, S., Caiming, Z., Tian, W., Yi, C., Jixiang, D., 2020. Survey on Density Peak Clustering Algorithm. Journal of Computer Research and Development 57, 378–394.

BAB 5

BIG DATA TOOLS DAN INFRASTRUKTUR PENDUKUNG

5.1 *Big Data*

Perkembangan dunia digital menyebabkan ledakan data yang sangat besar dalam hal volume, kecepatan, dan variasi (Ware, 2025). Data dari media sosial, perangkat IoT, transaksi bisnis, dan sistem informasi menghasilkan jutaan gigabyte setiap hari. Fenomena ini menuntut sistem yang tidak hanya mampu menyimpan, tetapi juga memproses dan menganalisis data dalam skala masif.

Big Data didefinisikan sebagai kumpulan data yang sangat besar dan kompleks sehingga tidak dapat dikelola dengan metode tradisional (El-Afifi, 2024). Oleh karena itu, diperlukan ekosistem alat (*tools*) dan infrastruktur pendukung seperti penyimpanan terdistribusi, komputasi paralel, dan platform orkestrasi yang efisien. Big Data tidak hanya sekadar tentang data yang berukuran "besar". Ia didefinisikan melalui karakteristik multidimensional yang dikenal sebagai 5V:

1. Volume: Merujuk pada ukuran data yang sangat besar, mulai dari terabyte hingga zettabyte. Sumbernya bisa dari berbagai macam, seperti streaming data dari sensor atau arsip historis bertahun-tahun.

2. *Velocity*: Menunjukkan kecepatan di mana data dihasilkan, diproses, dan dianalisis. Data real-time dari GPS, transaksi saham, atau umpan media sosial membutuhkan pemrosesan dengan latensi yang sangat rendah.
3. *Variety*: Mencakup beragam tipe dan format data. Data tidak lagi terstruktur rapi dalam tabel (*structured*), tetapi juga berupa data semi-structured (JSON, XML, log) dan unstructured (gambar, video, email, dokumen teks).
4. *Veracity*: Menyangkut kualitas, konsistensi, dan keandalan data. Data dari sumber yang berbeda sering kali mengandung noise, inkonsistensi, dan ketidakpastian, sehingga memerlukan proses pembersihan dan validasi.
5. *Value*: Aspek yang paling penting. Big Data pada akhirnya harus dapat diekstraksi nilainya untuk menghasilkan wawasan (*insight*) yang mendukung pengambilan keputusan yang lebih baik.

5.2 Tujuan *Big Data Tools* dan Ruang Lingkup

Adapun tujuan dari *Big Data Tools* dan Infrastruktur pendukung, sebagai berikut :

1. Kategori utama alat Big Data dan fungsinya.
2. Komponen infrastruktur pendukung
3. Tren arsitektur (*cloud-native, hybrid/multi-cloud, edge*) dan tantangan operasional.
4. Rekomendasi pemilihan tool & infrastruktur berdasarkan kasus penggunaan umum *(batch analytics, real-time streaming*, ML/AI).

5.3 Kategori Utama Big Data Tools

Perkembangan ekosistem Big Data telah melahirkan berbagai kategori tools yang spesifik, masing-masing dirancang untuk menyelesaikan tantangan tertentu dalam siklus hidup data. Pemahaman terhadap kategori-kategori ini sangat krusial untuk membangun arsitektur data yang robust, skalabel, dan efisien. Kategori-kategori utama tersebut meliputi pemrosesan batch, pemrosesan streaming, manajemen penyimpanan (Data Lake & Lakehouse), orkestrasi, serta tata kelola dan kualitas data.

5.3.1 Pemrosesan *Batch*

Pemrosesan batch merupakan paradigma pemrosesan data di mana kumpulan data yang sangat besar (batch) dikumpulkan dan dianalisis secara periodik dalam jeda waktu tertentu (misalnya, setiap hari atau setiap jam). Pendekatan ini ideal untuk analitik yang tidak memerlukan hasil secara real-time, seperti pembuatan laporan harian, pelatihan model machine learning, atau transformasi data dalam volume masif. Pemrosesan batch merupakan proses analitik terhadap kumpulan data besar yang dijalankan secara periodik. *Apache Hadoop* dan *Apache Spark* adalah dua *framework* utama. (Journal of Big Data, 2024).

1. **Apache Hadoop**, dengan inti Hadoop *Distributed File System* (HDFS) dan model pemrograman MapReduce, selama bertahun-tahun menjadi tulang punggung pemrosesan batch. Namun, kelemahan utamanya adalah banyaknya operasi baca/tulis disk yang memperlambat kinerja.
2. **Apache Spark** hadir sebagai solusi dengan pendekatan *in-memory computation,* yang menyimpan data intermediate di dalam memori utama (RAM), sehingga mengurangi latensi secara

signifikan. Spark tidak hanya lebih cepat hingga 100x dibanding Hadoop MapReduce untuk aplikasi tertentu, tetapi juga menyediakan API yang terintegrasi untuk SQL, streaming, dan machine learning.

5.3.2 Pemrosesan Streaming / Real-Time

Berlawanan dengan batch, pemrosesan streaming memproses data secara terus-menerus segera setelah data tersebut dihasilkan, sehingga memungkinkan organisasi untuk mendapatkan wawasan dan melakukan tindakan dengan latensi sangat rendah (dalam hitungan milidetik atau detik). Aplikasinya meliputi deteksi penipuan, pemantauan kesehatan sistem, rekomendasi produk secara real-time, dan analisis tren media sosial.

Sistem seperti Apache Flink dan Kafka Streams memungkinkan pemrosesan data secara terus-menerus dengan mekanisme *low-latency* yang efisien. Arsitektur ini tidak hanya mendukung analitik real-time tetapi juga memfasilitasi deteksi anomali secara langsung dan respons otomatis terhadap peristiwa penting bisnis (Surur, 2025).

1. **Apache Flink** unggul dalam kategori ini karena arsitekturnya yang benar-benar *stream-first*, artinya ia memperlakukan batch sebagai kasus khusus dari streaming. Flink menyediakan jaminan pemrosesan yang tepat-sekali (*exactly-once semantics*) dan manajemen state yang kuat, yang sangat kritis untuk aplikasi bisnis.
2. **Kafka Streams** adalah library client-side yang ringan untuk membangun aplikasi real-time yang memanfaatkan Apache Kafka sebagai backbone streaming-nya.

5.3.3 Data Lake dan Lakehouse

Data Lake muncul sebagai solusi penyimpanan yang menampung data dalam bentuk mentah (*raw*), terstruktur, semi-terstruktur, dan tidak terstruktur, dalam skala yang sangat masif. Biasanya dibangun di atas penyimpanan objek yang murah (seperti Amazon S3 atau Azure Blob Storage). Tantangan utama data lake tradisional adalah kurangnya tata kelola, yang dapat mengubahnya menjadi "data swamp" (rawa data) yang sulit dikelola dan ditelusuri.

Konsep Lakehouse kemudian dikembangkan untuk menjawab tantangan ini. Lakehouse memadukan fleksibilitas dan skalabilitas ekonomi dari data lake dengan kemampuan manajemen data dan transaksional dari data warehouse tradisional. Platform seperti Databricks Lakehouse dan Apache Iceberg menerapkan lapisan tabel terbuka (*open table formats*) yang memungkinkan operasi ACID, versioning data, dan kueri yang cepat langsung di atas penyimpanan objek.

Konsep data lake berevolusi dengan menggabungkan penyimpanan objek yang skalabel dengan lapisan metadata yang terstruktur. Platform modern seperti Databricks Lakehouse secara efektif memadukan keunggulan data lake dan data warehouse, menghasilkan efisiensi biaya dan kinerja analitik yang unggul" (El-Afifi, 2024).

5.3.4 Orkestrasi dan Workflow

Seiring dengan kompleksitas pipeline data yang meningkat, mengelola ketergantungan (*dependencies*), penjadwalan, dan pemantauan berbagai tugas menjadi tantangan tersendiri. Di sinilah tools orkestrasi berperan. Tools ini memungkinkan para engineer data

untuk mendefinisikan, menjadwalkan, dan memantau alur kerja data yang kompleks sebagai kode (as-code). Airflow, Prefect, dan Argo Workflows telah menjadi pilar utama dalam menjadwalkan dan mengelola dependensi pipeline Big Data yang kompleks (DoK, 2024). Orkestrasi tidak hanya memungkinkan otomatisasi alur kerja data secara skalabel, tetapi juga meningkatkan keandalan, kemampuan pelacakan, dan pemulihan kesalahan secara otomatis. Apache Airflow adalah tool yang paling populer, yang memungkinkan pembuatan workflow sebagai Directed Acyclic Graphs (DAGs) menggunakan Python. Prefect hadir sebagai alternatif modern dengan arsitektur yang lebih dinamis dan pengalaman developer yang lebih baik. Sementara Argo Workflows adalah solusi native untuk ekosistem Kubernetes, yang memanfaatkan container untuk menjalankan setiap langkah alur kerja.

5.3.5 *Data Governance* dan *Quality*

Dalam era yang semakin data-driven, memiliki data yang berkualitas dan terpercaya bukan lagi sebuah kemewahan, melainkan sebuah keharusan. Data Governance mencakup kerangka kebijakan, standar, dan proses untuk memastikan ketersediaan, kegunaan, integritas, dan keamanan data dalam suatu organisasi.

Tools seperti *Great Expectations* memungkinkan tim untuk membuat, mendokumentasikan, dan memvalidasi "harapan" terhadap kualitas data (misalnya, nilai kolom tidak boleh null, atau harus dalam rentang tertentu). DataHub atau Amundsen berfokus pada katalog data dan *data discovery*, memetakan lineage (asal-usul) data, dan menunjukkan kepada pengguna dari mana data berasal dan bagaimana transformasinya. Implementasi Data Mesh paradigma desain arsitektur yang terdesentralisasi menjadikan *Data Governance*

yang dapat diotomasi sebagai sebuah prasyarat. Platform katalog data aktif (*active metadata*) seperti DataHub memfasilitasi desentralisasi ini dengan menyediakan lapisan observability yang memungkinkan domain-domain data untuk mengelola sendiri aset mereka sambil tetap mematuhi standar organisasi secara global (Surur, 2025).

5.4 Infrastruktur Pendukung: Komponen Inti

5.4.1 Komputasi Terdistribusi

Komputasi terdistribusi adalah prinsip fundamental dalam Big Data, di mana suatu tugas komputasi yang sangat besar dipecah menjadi bagian-bagian kecil yang kemudian didistribusikan ke across banyak node (server) untuk diselesaikan secara paralel. Pendekatan ini mengatasi keterbatasan fisik satu mesin tunggal (seperti memori dan CPU) dengan "membagi-bagi beban kerja".

Awalnya, framework seperti Apache Hadoop (dengan YARN sebagai pengelola sumber dayanya) mendominasi landscape ini. Namun, evolusi menuju arsitektur yang lebih fleksibel dan efisien memunculkan Kubernetes sebagai platform orkestrasi container yang dominan. Kubernetes memungkinkan workload data (seperti pod Spark atau Flink) di-deploy, di-scale, dan dikelola dengan cara yang sama seperti aplikasi mikro lainnya, memberikan konsistensi operasional antara tim pengembangan dan tim data.

Kubernetes kini banyak diadopsi sebagai platform orkestrasi standar untuk menjalankan workload data dalam container. Laporan State of Kubernetes menunjukkan bahwa 65% organisasi yang menjalankan workload data secara produktif telah menggunakan atau

sedang dalam tahap migrasi ke Kubernetes, yang dinilai memberikan fleksibilitas portabilitas cloud dan efisiensi resource yang lebih baik dibandingkan dengan sistem manajemen cluster tradisional (Data on Kubernetes, 2024).

5.4.2 Penyimpanan Data

Penyimpanan data terdistribusi dirancang untuk menyimpan volume data yang jauh melampaui kapasitas satu disk atau satu server, sambil menjaga ketahanan (melalui replikasi) dan keteraksesan yang tinggi.

Hadoop Distributed File System (HDFS) adalah perintis dalam kategori ini, yang menyimpan data dalam blok-blok yang disebar di banyak node di dalam sebuah cluster. Namun, tren modern bergeser ke Object Storage seperti Amazon S3, Google Cloud Storage, dan solusi open-source seperti MinIO. Object storage menawarkan daya tahan yang sangat tinggi, skalabilitas yang hampir tak terbatas, dan model biaya pay-as-you-go yang menarik. Penyimpanan terdistribusi seperti HDFS dan object storage (S3, MinIO) terus menjadi tulang punggung sistem Big Data. Sementara itu, adopsi format penyimpanan kolumnar seperti Parquet dan ORC telah terbukti meningkatkan efisiensi I/O secara signifikan, dengan kompresi yang lebih baik dan performa pembacaan yang lebih cepat untuk workload analitik (El-Afifi, 2024).

Di lapisan yang lebih atas, format file kolumnar seperti Apache Parquet dan ORC (*Optimized Row Columnar*) telah menjadi standar de facto. Dengan menyimpan data per kolom, bukan per baris, format ini sangat meningkatkan efisiensi I/O karena hanya kolom yang relevan dalam sebuah kueri yang perlu dibaca dari

disk. Hal ini secara drastis mengurangi latensi dan biaya untuk operasi analitik.

5.4.3 Jaringan dan Transfer Data

Dalam sistem terdistribusi, jaringan adalah "sistem peredaran darah" yang menghubungkan semua komponen. Kinerja keseluruhan sangat bergantung pada kecepatan dan latensi jaringan antar-node dalam sebuah cluster. Kemacetan jaringan dapat dengan mudah menjadi bottleneck yang menghambat paralelisme.

Selain optimisasi jaringan dalam data center (seperti penggunaan RDMA - *Remote Direct Memory Access*), pola arsitektur seperti *Edge Computing* dan *Content Delivery Network* (CDN) menjadi semakin relevan. Edge computing memproses data di dekat sumbernya (misalnya, perangkat IoT), yang sangat mengurangi volume data yang perlu ditransfer ke cloud pusat, sehingga menurunkan latensi dan biaya bandwidth. CDN digunakan untuk mendistribusikan data hasil olahan (seperti model ML atau dataset referensi) ke lokasi-lokasi geografis yang dekat dengan pengguna akhir. Kinerja sistem Big Data sangat bergantung pada kecepatan jaringan antar-node. Solusi seperti *edge caching* dan *content delivery network* (CDN) dapat mengurangi latensi (Akamai, 2025).

5.4.4 *Observability* dan *Monitoring*

Ketika sistem menjadi semakin terdistribusi dan kompleks, kemampuan untuk mengamati keadaan internal sistem dari luar yang dikenal sebagai observability menjadi kritis. *Observability* melampaui *monitoring* tradisional dengan tidak hanya melihat metrik yang telah diketahui, tetapi juga mampu menanyakan pertanyaan baru tentang sistem tanpa harus memprediksi sebelumnya apa yang akan salah.

Observability telah menjadi kebutuhan mutlak untuk sistem data yang kompleks. Toolstack yang terdiri dari Prometheus untuk metrik, Grafana untuk visualisasi, dan Jaeger untuk distributed tracing, membentuk pilar utama untuk memantau kesehatan, performa, dan dependensi antar komponen secara komprehensif (Data on Kubernetes, 2024).

Prometheus menjadi standar industri untuk pengumpulan dan penyimpanan metrik time-series. Grafana digunakan untuk memvisualisasikan metrik tersebut menjadi dashboard yang powerful. Sementara untuk melacak request yang melintasi banyak layanan mikro (misalnya, sebuah kueri API yang memicu serangkaian job Spark dan Flink), distributed tracing dengan tools seperti Jaeger atau Zipkin adalah satu-satunya cara untuk memahami alur dan mengidentifikasi titik latency.

5.5 Arsitektur Modern

Evolusi arsitektur Big Data tidak hanya tentang bagaimana memproses data, tetapi juga *di mana* dan dalam *bentuk apa* infrastruktur tersebut dijalankan. Tren modern didorong oleh kebutuhan akan kecepatan, fleksibilitas, kepatuhan, dan efisiensi biaya, yang melahirkan tiga paradigma utama: Cloud-Native, Hybrid/Multi-Cloud, dan Edge-to-Cloud.

5.5.1 Cloud-Native

Arsitektur cloud-native merupakan pendekatan untuk membangun dan menjalankan aplikasi yang sepenuhnya memanfaatkan model pengiriman cloud computing. Dalam konteks Big Data, ini berarti meninggalkan beban mengelola infrastruktur fisik atau

virtual yang rumit, dan beralih ke layanan terkelola (*managed services*) yang skalabel dan elastis.

Platform seperti Amazon EMR, Google Dataproc, dan Azure HDInsight menyediakan cluster Hadoop atau Spark yang dapat dihidupkan dalam hitungan menit dan dimatikan setelah pekerjaan selesai (ephemeral), yang mengoptimalkan biaya. Untuk pemrosesan analitik, layanan seperti Google BigQuery dan Amazon Redshift menawarkan kemampuan data warehouse tanpa perlu mengelola server, yang secara otomatis menangani penskalaan dan pemeliharaan. Databricks di atas AWS, Azure, atau GCP menyajikan platform analitik terpadu yang mengabstraksi seluruh kompleksitas infrastruktur.

Arsitektur *cloud-native* dengan menggunakan layanan terkelola seperti AWS EMR, Google BigQuery, dan platform Databricks telah menjadi standar baru. Pendekatan ini secara signifikan mempercepat waktu implementasi solusi Big Data karena organisasi dapat fokus pada logika bisnis dan analitik, tanpa dibebani oleh pengelolaan infrastruktur fisik yang rumit (Ware, 2025).

5.5.2 Hybrid dan Multi-Cloud

Arsitektur hybrid menggabungkan infrastruktur on-premises (atau *private cloud*) dengan public cloud. Pendorongnya beragam, termasuk kebutuhan untuk mematuhi regulasi data yang mensyaratkan penyimpanan lokal (*data sovereignty*), investasi yang sudah tenggelam (*sunk cost*) dalam infrastruktur on-premise, atau keinginan untuk menggunakan layanan analitik terbaik dari berbagai vendor cloud (*best-of-breed*). Platform data terbuka seperti Apache Iceberg dan Delta Lake muncul sebagai solusi kunci untuk tantangan *hybrid/multi-cloud.* Format tabel terbuka ini memungkinkan mesin komputasi yang

berbeda (contoh: Spark on-premise dan BigQuery di cloud) untuk membaca dan menulis ke kumpulan data yang sama secara konsisten, sehingga meminimalkan perpindahan data yang mahal dan memecahkan masalah isolasi data silo (Proceedings of the VLDB Endowment, 2025).

Tantangan terbesar dalam model ini adalah menciptakan pengalaman yang terpadu dan konsisten. Sinkronisasi metadata antar katalog data (misalnya, antara Hive Metastore on-premise dan AWS Glue Data Catalog) menjadi kritis untuk memastikan "satu sumber kebenaran". Selain itu, menjaga konsistensi data di seluruh lingkungan, terutama untuk data yang sering diperbarui, memerlukan tools replikasi data yang kuat dan mekanisme data governance yang terpusat.

5.5.3 *Edge-to-Cloud*

Arsitektur *Edge-to-Cloud* (atau Edge-to-Cloud Continuum) memindahkan pemrosesan data dan komputasi secara fisik lebih dekat ke sumber data, seperti pabrik, kota pintar, atau kendaraan, daripada mengandalkan pusat data cloud yang terpusat. Model ini merespons keterbatasan bandwidth, latensi, dan keandalan konektivitas dalam skenario seperti Internet of Things (IoT).

Di edge, perangkat atau gateway melakukan pemrosesan awal: filtering, aggregasi, dan inferensi model machine learning secara real-time. Hanya hasil ringkasan, data yang penting, atau model yang telah diperbarui yang kemudian dikirim ke cloud untuk penyimpanan jangka panjang, analitik yang lebih mendalam, dan pelatihan model ulang secara global. Pembagian kerja ini menciptakan sistem yang sangat

responsif dan efisien. Edge computing menjawab tantangan latensi dan bandwidth dengan memproses data lebih dekat ke sumbernya sebelum dikirim ke cloud untuk analitik yang lebih mendalam. Paradigma ini sangat transformatif untuk aplikasi yang memerlukan respon dalam milidetik" (Belcastro, 2024). "Laporan terbaru juga mengonfirmasi bahwa strategi edge-to-cloud dapat mengurangi volume data yang ditransmisikan ke cloud hingga 90%, yang secara langsung menerjemahkan penghematan biaya operasional yang signifikan (Akamai, 2025).

Edge computing menjawab tantangan latensi dan bandwidth dengan memproses data lebih dekat ke sumbernya sebelum dikirim ke cloud untuk analitik yang lebih mendalam. Paradigma ini sangat transformatif untuk aplikasi yang memerlukan respon dalam milidetik" (Belcastro, 2024). "Laporan terbaru juga mengonfirmasi bahwa strategi edge-to-cloud dapat mengurangi volume data yang ditransmisikan ke cloud hingga 90%, yang secara langsung menerjemahkan penghematan biaya operasional yang signifikan" (Akamai, 2025).

5.6 Pemilihan *Tools Big Data*

Pemilihan stack teknologi Big Data merupakan keputusan strategis yang sangat bergantung pada kebutuhan spesifik use case, karakteristik data, dan pertimbangan arsitektural organisasi. Tidak ada solusi "satu untuk semua". Pendekatan terbaik adalah memilih kombinasi tools yang saling melengkapi untuk membentuk pipeline data yang kohesif dan berkinerja tinggi. Berikut adalah analisis mendalam untuk beberapa tools kunci berdasarkan penelitian terkini.

5.6.1 Apache Spark

Apache Spark sangat bagus Untuk Pemrosesan Batch, Analitik Interaktif, dan Machine Learning yang Dipercepat.Apache Spark tetap menjadi pilihan utama untuk workload batch yang membutuhkan kinerja tinggi dan workload analitik yang kompleks. Keunggulan utamanya terletak pada arsitektur *in-memory computation*-nya yang meminimalkan operasi I/O disk.

5.6.2 Apache Flink

Apache Flink untuk Untuk Pemrosesan Streaming Real-Time dengan State yang Kompleks Ketika aplikasi membutuhkan latensi yang sangat rendah (milidetik) dan kemampuan manajemen state yang robust untuk perhitungan yang kompleks, Apache Flink adalah jawabannya. Arsitektur true-streaming-nya memproses event segera setelah event tersebut tiba.

5.6.3 Apache Kafka

Apache Kafka berperan sebagai tulang punggung (*central nervous system*) untuk arsitektur data modern. Ia berfungsi sebagai platform streaming yang dapat diandalkan, skalabel, dan tahan lama untuk mengangkut aliran data real-time antara aplikasi dan sistem data.

5.6.4 Databricks Lakehouse

Databricks Lakehouse Platform merepresentasikan konvergensi antara data lake yang skalabel dan data warehouse yang andal. Ini adalah pilihan ideal untuk organisasi yang ingin memecah silo antara tim data engineering, data science, dan analisis bisnis.

DAFTAR PUSTAKA

Akamai. (2025). The State of the Edge: Latency and Bandwidth Trends in Distributed Computing.

Belcastro, L. (2024). "Edge Data Analytics: A Survey of Models and Systems." IEEE Transactions on Knowledge and Data Engineering, 36(3), 1125-1144.

Data on Kubernetes (DoK) Community. (2024). Annual Report: The State of Data on Kubernetes.

El-Afifi, M. (2024). "Converging Data Lakes and Data Warehouses: An Empirical Evaluation of the Lakehouse Architecture." Journal of Big Data, 11(1), 25.

IEEE Transactions on Knowledge and Data Engineering. (2025). "Advancements in Micro-Batch Processing: A Comparative Study of Spark Structured Streaming and Apache Flink," *37*(2), 450-465.

Journal of Big Data. (2024). "Benchmarking Distributed Data Processing Frameworks for Large-Scale Machine Learning," *11*(1), 45.

Proceedings of the VLDB Endowment. (2025). "Delta Lake: High- Performance ACID Table Storage over Cloud Object Stores," *18*(5), 1200-1213.

Surur, A. (2025). Stream Processing in the Modern Data Stack: A Deep Dive into Apache Flink and its Ecosystem. O'Reilly Media.

Ware, J. (2025). "The Economic Impact of Cloud-Native Data Architectures." International Journal of Cloud Computing and Data Science, 5(2), 88-102.

BAB 6

DATA INGESTION DAN *DATA STORAGE*

6.1 *Data Ingestion*

Dalam era **transformasi digital dan big data**, jumlah data yang dihasilkan oleh organisasi, individu, maupun perangkat meningkat secara eksponensial. Data berasal dari berbagai sumber seperti aplikasi bisnis, media sosial, perangkat *Internet of Things* (IoT), log sistem, hingga sensor industri. Setiap detik, data baru terus muncul dalam berbagai format — mulai dari teks, angka, gambar, hingga video.

Data ingestion berperan penting sebagai **tahap awal dalam pipeline pengelolaan data**. Tanpa adanya proses ingestion yang baik, data yang tersedia di berbagai sumber tidak akan bisa digunakan secara efisien untuk analisis, pelaporan, maupun pengambilan keputusan strategis.

6.1.1 Pengertian *Data Ingestion*

Data *Ingestionz* adalah proses mentransfer data dari berbagai sumber kedalam sistem *Big Data* untuk kemudian disimpan, di proses dan dianalisis. Dalam Konteks *Big Data,* proses ini menjadi semakin kompleks karena data bisa datang dari sumber yang sangat

beragam dengan format, volume, dan kecepatan yang berbeda-beda.(Tribuana,2025).

Istilah ***Data Ingestion*** berasal dari dua kata bahasa Inggris:

- "*Data*" yang berarti kumpulan informasi atau fakta yang dapat diolah menjadi pengetahuan, dan
- "*Ingestion*" yang berarti proses memasukkan sesuatu ke dalam *sistem atau tubuh*.

Secara terminologi dalam bidang teknologi informasi, data ingestion berarti proses memasukkan atau mengimpor data dari berbagai sumber ke dalam suatu sistem untuk diolah lebih lanjut.

Oleh karena itu, istilah "data ingestion" sering disebut juga sebagai tahapan awal dalam pipeline data (data pipeline entry point). Tanpa proses ini, data dari berbagai sumber tidak dapat dikonsolidasikan untuk dianalisis.

6.1.2 Tujuan *Data Ingestion*

Tujuan *Data* Ingestion yaitu :

1. Mengumpulkan Data dari Berbagai Sumber. Tujuan utama data ingestion adalah mengambil data dari berbagai sumber (seperti sensor IoT, aplikasi, database, media sosial, atau log server) dan memindahkannya ke sistem penyimpanan pusat seperti data warehouse, data lake, atau cloud
2. Menyediakan Data untuk Analisis. Dengan data yang sudah terintegrasi di satu tempat, analisis data dapat dilakukan lebih mudah dan cepat, mendukung pengambilan keputusan berbasis data.

3. Menjamin Ketersediaan Data Secara Real-Time. Dalam sistem modern, ingestion sering bersifat real-time atau near real-time, agar data terbaru langsung tersedia untuk monitoring, dashboard, atau machine learning.
4. Menstandarkan Format dan Kualitas Data. Proses ingestion juga mencakup pembersihan (*cleaning*) dan transformasi (*transformation*) agar data dari berbagai sumber memiliki format yang seragam dan akurat.
5. Mendukung Otomatisasi dan Efisiensi Operasional. Dengan ingestion otomatis, organisasi tidak perlu memproses data manual. Ini meningkatkan efisiensi dan mengurangi risiko kesalahan manusia.

6.1.3 Manfaat *Data Ingestion*

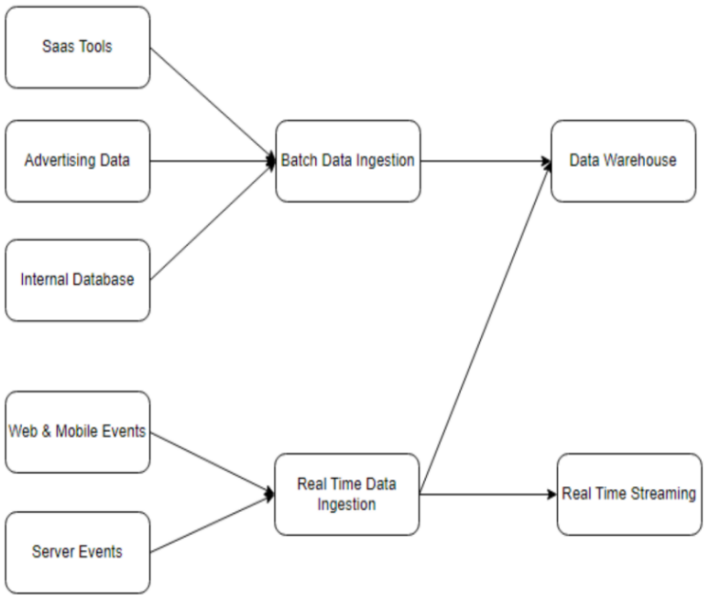
Manfaat dari data ingestion yaitu:

1. Mempercepat Pengambilan Keputusan. Data yang masuk secara cepat dan teratur membuat manajemen bisa mengambil keputusan berdasarkan data terbaru.
2. Meningkatkan Aksesibilitas Data. Semua tim (bisnis, analitik, IT) bisa mengakses data dari satu sumber terpusat tanpa harus mengambil langsung dari sistem asal.
3. Meningkatkan Kualitas Analisis dan Laporan. Data yang terintegrasi dan bersih menghasilkan laporan dan insight yang lebih akurat.
4. Memungkinkan Analisis Prediktif dan Machine Learning. Sistem AI dan ML memerlukan data dalam jumlah besar dan beragam ingestion memungkinkan pengumpulan data tersebut secara efisien.

5. Skalabilitas dan Fleksibilitas. Dengan sistem ingestion yang baik, perusahaan dapat meningkatkan volume data tanpa mengganggu proses analisis yang sudah berjalan.
6. Efisiensi Operasional. Mengurangi waktu dan biaya untuk integrasi manual antar sistem, sehingga tim bisa fokus pada analisis dan strategi bisnis.

6.1.4 Jenis-Jenis *Data Ingestion*

1. *Batch Data Ingestion*: Batch data ingestion adalah mentode pengambilan data dimana data dikumpulkan dan di proses dalam jumlah besar secara terjadwal atau berjenjang. (Iskandar : 2024).
2. *Real Time Data Ingestion:* Real time data ingestion adalah proses dimana data dikumpulkan dan diproses secara terus-menerus dan segera setelah data tersebut tersedia. Dalam skenario real time, data langsung dikirim ke sistem penyimpanan begitu data tersebut dihasilkan. (Iskandar:2024)
3. *Hybrid Ingestion :* Hybrid ingestion adalah metode pengambilan dan pemindahan data yang menggabungkan dua pendekatan utama, yaitu batch ingestion dan realtime ingestion. Dengan hybrid ingestion, sebagian data diproses secara real time (langsung saat data masuk), sementara data lain diproses secara berkala (misalnya setiap jam, harian, atau mingguan).



Gambar 6. 1 Contoh Sistem Sesuai dengan Jenis *Data Ingestion*
(Sumber : Iskandar:2024)

6.1.5 Komponen *Data Ingestion*

Komponen utama arsitektur Big Data dapat dijelaskan sebagai berikut:

1. Sumber Data (*Data Sources*). Proses arsitektur Big Data dimulai dari berbagai sumber data yang menghasilkan data dalam jumlah besar. Sumber ini dapat berupa basis data aplikasi, file log, perangkat IoT, media sosial, atau platform lain yang menghasilkan data secara real-time maupun batch.
2. Pengambilan Data (*Data Ingestion Layer*). Lapisan ini bertugas mengumpulkan dan menangkap data dari sumber yang sangat beragam, kemudian mengalirkannya ke penyimpanan data. Pada tahap

ini, data akan dikategorikan dan diprioritaskan agar proses selanjutnya menjadi lebih mudah dan terstruktur.

3. Pengumpulan dan Transmisi data (*Data Collector Layer*) pada lapisan ini, data yang telah masuk dari tahap pengambilan disalurkan sesuai kategori atau topik tertentu. Pengelolaan ini penting untuk memastikan data dapat diproses sesuai kebutuhan analitik. Sistem queue atau messaging seperti Pache Kafka sering digunakan dilapisan ini.

4. Pemprosesan data (*Data Processing Layer*). Lapisan ini mengolah data dengan teknik pemrosesan batch ataupun streaming. Data-data berat dikumpulkan, diproses, disaring, dan diubah menjadi format yang siap untuk dianalisis. Pemprosesan ini juga meliputi transformasi data dan integrasi data agar dapat digunakan secara efektif.

5. Penyimpanan Data (*Data Storage Layer*). Data hasil pengumpulan dan pemrosesan disimpan dalam sistem yang dapat menangani volume besar dan berbagai jenis format data, seperti data lake atau Hadoop Distributed File System. Penyimpanan ini harus scable dan mampu menampung data dalam berbagai bentuk, baik terstruktur maupun tidak terstruktur.

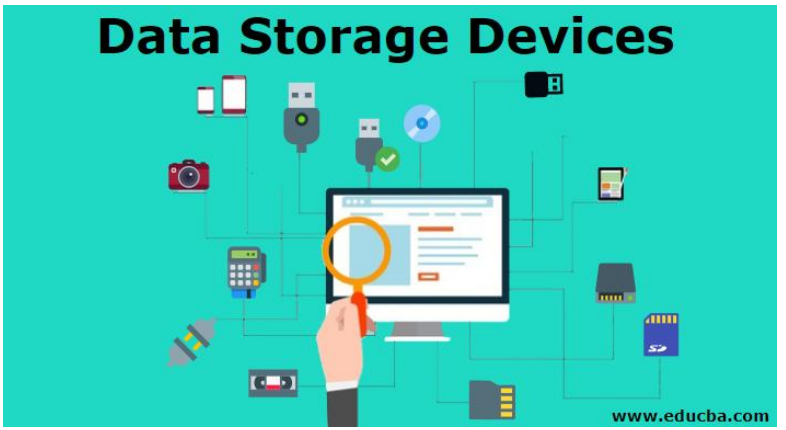
6. Analisis Data (*Data Analytics Layer*). Pada lapisan ini data yang sudah tersimpan dianalisis untuk dianalisis untuk mengeluarkan informasi penting yang menjadi dasar pengambilan keputusan bisnis. Proses analitik ini bisa berupa analisis statistik, pembelajaran mesin, atau teknik kecerdasan buatan lainya.

7. Visualisasi dan Pelaporan (*Data Visualization and Reporting Layer*). Hasil analisis disajikan dalam bentuk yang mudah dipahami, misalnya grafik, dashboard, dan laporan yang interaktif sehingga pengguna non-teknis dapat memahami insight dari data.(Saptadi:2025).

6.2 *Data Storage*

6.2.1 Pengertian *Data Storage*

Data storage adalah proses dan sistem yang digunakan untuk menyimpan, mengelola, serta menjaga data dalam bentuk digital agar dapat diakses dan digunakan kembali secara efisien kapan saja. Data storage berfungsi sebagai tempat penyimpanan utama bagi berbagai jenis data seperti teks, gambar, video, log sistem, hingga data sensor yang dihasilkan dari berbagai sumber. Dalam penerapannya, data storage dapat berbentuk perangkat fisik seperti server dan hard disk (*on-premise*) maupun berbasis cloud seperti Amazon S3, Google Cloud Storage, dan Azure. Tujuan utama data storage adalah untuk memastikan data tersimpan dengan aman, mudah diakses, memiliki keandalan tinggi, serta mampu menampung volume data yang terus meningkat. Dengan sistem penyimpanan yang baik, organisasi dapat memanfaatkan data tersebut untuk analisis, pengambilan keputusan, serta mendukung berbagai kebutuhan operasional dan bisnis secara efektif. (Iskandar:2024).



Gambar 6. 2 Berbagai Macam Jenis *Data Storage*
(Sumber : www.educba.com)

Jadi data storage adalah tempat atau media untuk menyimpan data agar dapat diakses, digunakan, dan dikelola kembali saat dibutuhkan.

Dengan kata lain, data storage berfungsi sebagai wadah penyimpanan informasi digital, baik secara sementara (seperti RAM) maupun permanen (seperti *hard disk* atau *cloud storage*).

6.2.2 Jenis-Jenis *Data Storage*

Jenis-jenis data storage (penyimpanan data) secara umum dapat dibagi menjadi beberapa kategori berdasarkan teknologi, fungsi, dan lokasi penyimpanan. Berikut penjelasannya :

1. ***Primary Storage* (Penyimpanan Utama)**
   Digunakan langsung oleh CPU untuk menyimpan data sementara saat proses berlangsung. Contoh:
   - RAM (*Random Access Memory*) → menyimpan data sementara, akan hilang jika listrik mati.

- *Cache Memory* → memori kecil dan sangat cepat untuk mempercepat akses CPU.
- Register → tempat penyimpanan paling cepat di dalam prosesor.

2. *Secondary Storage* (Penyimpanan Sekunder)

Menyimpan data secara permanen dan tidak mudah hilang.
Contoh:

- *Hard Disk Drive* (HDD)
- *Solid State Drive* (SSD)
- *Optical Disk* (CD/DVD/Blu-ray)
- F*lash Drive* / USB Drive

3. *Tertiary Storage* (Penyimpanan Tersier)

Digunakan untuk arsip atau backup jangka panjang.
Contoh:

- *Tape Drive (Magnetic Tape)*
- *Cloud Backup Storage* (Google Drive, AWS Glacier)

4. Cloud Storage

Data disimpan di server internet dan diakses secara online.
Contoh**:**

- Google Drive
- Dropbox
- OneDrive
- Amazon S3

5. ***Network Attached Storage*** **(NAS)**

   Sistem penyimpanan data yang terhubung ke jaringan lokal (LAN), digunakan bersama banyak pengguna.
   Contoh**:**

   - Synology NAS
   - QNAP NAS

6. ***Storage Area Network*** **(SAN)**

   Sistem penyimpanan berkecepatan tinggi yang terhubung ke beberapa server dalam jaringan besar (biasanya di data center).

6.2.3 Tujuan dan Manfaat data Storage

Data Storage mempunyai tujuan dan manfaat, yaitu:

1. Tujuan Data Storage:

   Tujuan utama data storage (penyimpanan data) adalah untuk menyimpan, mengelola, dan menjaga ketersediaan data agar dapat diakses dan digunakan kembali saat dibutuhkan.
   Secara lebih rinci tujuan data storage yaitu:

   1. Menyimpan data secara aman agar tidak hilang atau rusak.
   2. Memudahkan akses dan pengambilan data oleh pengguna atau sistem.
   3. Mendukung proses pengolahan data dalam berbagai aplikasi dan sistem informasi.
   4. Menjaga integritas dan konsistensi data agar tetap akurat.

5. Menyediakan cadangan (*backup*) untuk mencegah kehilangan data akibat kegagalan sistem atau serangan siber.

2. Manfaat Data Storage :

   Manfaat penggunaan sistem penyimpanan data antara lain:

   a. Efisiensi kerja meningkat — data mudah dicari, disimpan, dan dikelola.

   b. Keamanan data lebih terjamin — dengan enkripsi dan kontrol akses.

   c. Mendukung analisis dan pengambilan keputusan — data tersimpan rapi dan mudah diolah.

   d. Mendukung kolaborasi dan berbagi data antar pengguna atau departemen.

   e. Skalabilitas tinggi — kapasitas penyimpanan dapat ditambah sesuai kebutuhan (terutama pada cloud storage).

   f. Menghemat biaya dan ruang fisik dibanding penyimpanan manual atau berbasis kertas.

6.2.4 Tantangan dalam *Data Storage*

Berikut beberapa tantangan utama dalam *data storage* (penyimpanan data):

1. Pertumbuhan Volume Data Yang Cepat

   Setiap hari jumlah data yang dihasilkan meningkat sangat pesat (*big data*). Tantangannya adalah menyediakan kapasitas penyimpanan yang cukup besar dan mudah diakses tanpa mengorbankan kinerja.

2. Keamanan dan Privasi Data

   Data sensitif seperti data pribadi, keuangan, dan kesehatan rentan terhadap peretasan dan kebocoran data**.** Diperlukan enkripsi, kontrol akses, dan kebijakan keamanan yang ketat.

3. Kinerja dan Kecepatan Akses

   Sistem harus mampu membaca/menulis data dengan cepat**,** terutama untuk aplikasi real-time atau basis data besar. Penyimpanan yang lambat dapat menghambat performa aplikasi.

4. Manajemen dan Organisasi Data

   Semakin banyak data, semakin sulit untuk mengelola, mencari, dan mengkategorikannya**.** Diperlukan sistem manajemen data yang baik seperti *Data Management System* (DBMS) atau Data Lake**.**

5. Biaya Infrastruktur

   Menyediakan dan memelihara sistem penyimpanan besar (baik lokal maupun cloud) membutuhkan biaya tinggi. Tantangan bagi organisasi adalah menyeimbangkan kapasitas, kinerja, dan anggaran**.**

6. Migrasi dan Integrasi data

   Perpindahan data dari sistem lama ke sistem baru atau dari lokal ke cloud sering kali rumit dan berisiko kehilangan data. Diperlukan perencanaan dan teknologi migrasi yang baik.

7. Ketersediaan dan Backup
   Sistem penyimpanan harus tetap berjalan 24/7 dan memiliki backup otomatis agar tidak kehilangan data saat terjadi kerusakan atau bencana.

8. Kepatuhan terhadap Regulasi

   Ada berbagai aturan perlindungan data seperti GDPR (Eropa) atau PDP (Indonesia). Perusahaan harus memastikan penyimpanan data mematuhi hukum dan etika.

DAFTAR PUSTAKA

Tribuana, dkk. 2025. *Teknologi Big Data.* Payakumbuh: PT Serasi Media Teknologi.

Iskandar, dkk. 2024. *Teknologi Big Data.* Yogyakarta: PT Green Pustaka Indonesia.

Saptadi, dkk. 2025. *Analitik Big Data.* Batam: Yayasan Cendikia Mulya Mandiri.

Iskandar,dkk. 2024. *Teknologi Big Data.* Yogyakarta: PT Green Pustaka Indonesia.

Data Storage Devices | Quick Glance on Data Storage Devices

Educba. *Data Storage Devices.* Di akses 20 Oktober 2025.

https://id.wikipedia.org/wiki/ARPANET/

BAB 7

APLIKASI & STUDI KASUS ANALISIS *BIG DATA*

7.1 Pendahuluan

Perkembangan teknologi informasi dan komunikasi telah mendorong ledakan volume, kecepatan, dan variasi data dihampir semua bidang kehidupan dari layanan kesehatan, manufaktur, keuangan, sampai pendidikan. Analisis Big Data (*Big Data Analytics*) menjadi instrumen penting untuk mengekstrak wawasan dari data besar tersebut sehingga organisasi dapat mengambil keputusan yang lebih cepat, tepat, dan berbasis bukti. Kajian-kajian terbaru menunjukkan bahwa aplikasi Big Data telah berkembang luas mulai dari pemantauan kesehatan masyarakat dan pencegahan penyakit, optimasi operasi kawasan industri, hingga transformasi ekonomi berbasis prediksi dan ketahanan digital (Badshah *et al.*, 2024).

Pertumbuhan volume data yang dihasilkan oleh sensor, aplikasi mobile, transaksi digital, rekam medis elektronik, citra, dan media sosial menciptakan peluang baru untuk pengambilan keputusan yang lebih cepat dan akurat. Analisis big data (Big Data Analytics — BDA) memanfaatkan teknik statistik, pembelajaran mesin, komputasi terdistribusi, dan visualisasi untuk mengekstrak wawasan yang dapat ditindaklanjuti dari kumpulan data besar dan heterogen.

Dalam beberapa tahun terakhir, tinjauan sistematik menunjukkan percepatan riset dan adopsi BDA khususnya pada sektor industri dan kesehatan, dengan banyak tantangan tetap terbuka terkait kualitas data, tata kelola, dan kapabilitas SDM (Rahul, Banyal and Arora, 2023).

7.2 Konsep Big Data dan Analitik

7.2.1 Definisi dan Karakteristik

Big data secara umum mengacu pada kumpulan data yang sangat besar, sangat bervariasi, dan bergerak cepat (*volume, variety, velocity*). Analitik big data mencakup proses pengumpulan, penyimpanan, pemrosesan, analisis, dan interpretasi data untuk menghasilkan wawasan.

7.2.2 Kerangka Analitik dan Value Creation

Sebagai contoh, penelitian oleh *Big Data and Big Data Analytics in Value Creation and Innovation* (Sutriana, 2023) mengemukakan bahwa big data analytics mendukung penciptaan nilai dan inovasi dalam organisasi.

7.2.3 Proses dari Data ke Keputusan

Analitik big data sering mencakup langkah-langkah "*ask, prepare, process, analyse, share, act*". Sebuah studi kasus menunjukkan bagaimana siklus tersebut diterapkan dalam sebuah perusahaan sharing bike (Orji *et al.*, 2023).

7.3 Aplikasi *Big Data* di Berbagai Sektor

7.3.1 Kesehatan dan Kesehatan Masyarakat

- *Big Data Analytic* (Analisis Big Data) digunakan untuk pemodelan epidemi, pemantauan real-time kasus, analisis citra medis, dan optimasi rantai pasok obat. Tinjauan 2025 menegaskan pentingnya kombinasi model epidemiologi, time-series, dan ML untuk mitigasi pandemi dan deteksi dini, serta peran data alternatif (mobility, media sosial) dalam memperkaya sinyal epidemiologis. Namun isu privasi, bias, dan interoperabilitas data tetap menonjol (Nuha *et al.*, 2025).

7.3.2 Keuangan dan Fintech (*Finansial Technology*)

- *Big Data Analytic* (Analisis Big Data) di sektor keuangan meliputi deteksi *fraud real-time*, *credit scoring* yang memanfaatkan alternative data, *risk modelling*, dan analitik pasar. Literatur 2023–2024 menyoroti munculnya data alternatif (*satellite, web traffic, social sentiment*) yang mempengaruhi keputusan investasi dan peran tata kelola data dalam menjaga integritas model (Sharma, 2023).

7.3.3 Pendidikan (*Learning Analytics*)

- *Big Data Analytic* (Analisis Big Data) di dunia pendidikan digunakan untuk *early warning system* (prediksi drop-out), personalisasi pembelajaran, dan evaluasi efektivitas kurikulum. Tinjauan sistematik modern menunjukkan potensi besar serta tantangan implementasi (kualitas data, etika, adopsi institusi) (Stojanov and Daniel, 2024).

7.3.4 Industri dan Manufaktur

- Big Data Analytic (Analisis Big Data) di Dunia Industri dan Manufaktur dalam membuat sebuah Aplikasi termasuk *predictive maintenance, energy management*, dan pengawasan keselamatan. Studi terfokus pada kawasan industri (*industrial parks*) menampilkan pengurangan puncak beban energi, optimasi pengelolaan limbah, dan peningkatan keamanan melalui integrasi sensor IoT + BDA. Hasil empiris di beberapa kasus menunjukkan penurunan downtime dan efisiensi energi bila implementasi disertai proses bisnis yang matang (Sarker *et al.*, 2022).

7.3.5 *Smart City* dan *Governance* (Pemerintahan)

- *Big Data Analytic* (Analisis Big Data) di pemerintahan berupa manajemen lalu lintas, pemantauan kualitas udara, layanan darurat, dan pengelolaan sampah. Integrasi multi-sumber (sensor lalu lintas, CCTV, layanan publik) memungkinkan pengambilan keputusan operasional yang adaptif—tetapi interoperabilitas dan sinkronisasi kebijakan antarlembaga menjadi penghambat utama (Badshah *et al.*, 2024).

7.4 Studi Kasus Implementasi *Big Data*

Analisis Big Data (BDA) mengacu pada rangkaian teknik, infrastruktur, dan proses yang dipakai untuk mengumpulkan, menyimpan, memproses, dan mengekstrak wawasan dari dataset yang sangat besar, heterogen, dan (seringkali) real-time. Di tingkat aplikasi, BDA mengintegrasikan IoT, log operasional, citra, teks, dan sumber eksternal (mis. data cuaca atau *mobility*) untuk mendukung keputusan operasional dan strategis. Studi-studi terkini menegaskan percepatan adopsi BDA pada sektor kesehatan, industri/manufaktur, dan

layanan publik—walau banyak organisasi masih menghadapi hambatan pada kualitas data, tata kelola, dan kapabilitas SDM (Badshah *et al.*, 2024).

7.4.1 Kerangka Kerja Studi Kasus *Big Data Analytics*

Sebuah studi kasus BDA yang baik mengikuti langkah-langkah berikut:

- **Identifikasi *use-case* dengan nilai bisnis jelas** (ROI/*impact*). Mulai dari masalah terukur (mis. pengurangan biaya energi 10%, pengurangan false positive fraud 20%) (Rahul, Banyal and Arora, 2023).
- **Audit & akses data**: Peta sumber, frekuensi, kualitas, pastikan kepatuhan regulasi (Rahul, Banyal and Arora, 2023).
- **PoC (*Proof-of-Concept*)**: dengan subset data uji model sederhana dan proses ingestion (Sharma, 2023).
- **Skala & *engineering***: tentukan arsitektur (*on-prem/cloud*), otomasi pipeline (CI/CD for data & model) (Sharma, 2023).
- **Integrasi ke proses bisnis**: hasil analitik harus actionable (API, dashboad, alerts) (Sharma, 2023).
- ***Governance & monitoring**:* data catalog, lineage, model drift monitoring, periodic audits (Ma, Jørgensen and Ma, 2024).

7.4.2 Tantangan Implementasi dan Risk (Resiko)

- **Kualitas & heterogenitas data**: *missing, noise*, format tak konsisten.
- **Interoperabilitas & integrase**: antar-sumber data lintas-lembaga.

- **Privasi & regulasi**: khususnya data kesehatan dan finansial perlu enkripsi, anonymization, dan kebijakan akses.
- **SDM & kapabilitas organisasi**: kekurangan data engineers, data scientists dengan pemahaman domain.
- **Biaya & skalabilitas**: infrastruktur dan pemeliharaan.
- **Etika & fairness**: potensi bias dalam dataset yang menimbulkan keputusan tidak adil.

7.4.3 Ringkasan Tabel Aplikasi dan Studi Kasus

Tabel 7. 1 Aplikasi dan Studi Kasus

Sektor	Aplikasi Utama	Studi/Jurnal Terkait
Kesehatan	Prediksi penyakit, personalized medicine, IoT	(Stojanov and Daniel, 2024)
Industri/Manufaktur	Maintenance prediktif, optimasi energi/produksi	(Sarker *et al.*, 2022)
Keuangan & FinTech	Fraud detection, credit scoring, decision support	(Fahd and Miah, 2023)

Sektor	Aplikasi Utama	Studi/Jurnal Terkait
Pendidikan	Learning analytics, retention prediction, personalisasi	(Stojanov and Daniel, 2024)
Smart City / Publik	Mobilitas, sensor kota, layanan darurat	(Akbulut and Kaya, 2018)

7.4.4 Contoh Studi Kasus di Bidang Kesehatan *Big Data Analytics*

Analisis Big Data untuk Prediksi Penyebaran Penyakit Tuberkulosis (TB) di Kota Palembang Menggunakan Metode Naive Bayes.

1. Tujuan Aplikasi

a. Memprediksi potensi peningkatan kasus TB di wilayah tertentu.

b. Mengidentifikasi factor risiko utama berdasarkan data besar multidimensi.

c. Mendukung kebijakan berbasis data (*data driven policy*).

2. Sumber Data

Big data diperoleh dari berbagai sumber:

Tabel 7. 2 Aplikasi dan Studi Kasus

Jenis Data	Sumber	Volume	Frekuensi
Rekam medis TB	RS & Puskesmas	100.000+ entri	Harian
Data lingkungan (kepadatan, sanitasi)	BPS & Satgas	30.000 entri	Bulanan
Citra hasil rontgen dada	Rumah sakit rujukan	10.000+ file	Mingguan
Data sosial ekonomi	Dinas Sosial	25.000 entri	Tahunan

3. Arsitektur Sistem Big Data

Aplikasi ini menggunakan arsitektur Lambda Architectur yang terdiri dari:

1. ***Batch Layer*** – Menyimpan data historis di *Hadoop Distributed File System (HDFS)*.
2. ***Speed Layer*** – Mengolah data real-time dari fasilitas kesehatan menggunakan *Apache Kafka* dan *Spark Streaming*.
3. ***Serving Layer*** – Menyajikan hasil analitik melalui dashboard interaktif (*Tableau/Power BI*).

4. Proses Analisis Big Data

Tabel 7. 3 Proses Analisis Big Data

Tahapan	Deskripsi	Teknologi yang digunakan
Data Collection	Mengumpulkan data dari sistem informasi rumah sakit, BPS, dan citra medis	Kafka, Flume
Data Storage	Menyimpan data terstruktur & tidak terstruktur	Hadoop, HDFS
Data Cleaning	Menghapus duplikasi, mengisi missing values	PySpark
Feature Engineering	Ekstraksi variabel penting (usia, lokasi, hasil rontgen, sanitasi)	Python, Pandas
Modeling	Klasifikasi potensi TB menggunakan **Naive Bayes**	Scikit-learn
Visualization	Peta sebaran TB dan tingkat risiko kecamatan	Tableau, Plotly, QGIS

5. Proses Analisis Big Data

Contoh model prediksi sederhana:

```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
```

```
from sklearn.metrics import accuracy_score, confusion_matrix

import pandas as pd

# Contoh data sederhana

data = pd.DataFrame({

    'usia': [25, 40, 33, 60, 18, 55, 29, 70],

     'jenis_kelamin': ['L', 'P', 'L', 'L', 'P', 'L', 'P', 'L'],

    'hasil_rontgen': ['Positif', 'Negatif', 'Positif', 'Positif', 'Negatif', 'Positif', 'Negatif', 'Positif'],

'status_tb': ['Ya', 'Tidak', 'Ya', 'Ya', 'Tidak', 'Ya', 'Tidak', 'Ya']

})

# Konversi ke numerik

data_encoded = pd.get_dummies(data, drop_first=True)

X = data_encoded.drop('status_tb_Ya', axis=1)

y = data_encoded['status_tb_Ya']

# Split data

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Model Naive Bayes

model = GaussianNB()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

```
# Evaluasi
print("Akurasi:",
accuracy_score(y_test, y_pred))
print("Confusion          Matrix:\n",
confusion_matrix(y_test, y_pred))
```

Output Contoh :

```
Akurasi: 0.85
Confusion Matrix:
[[2 0]
  [1 3]]
```

Artinya, model dapat memprediksi status TB dengan tingkat akurasi 85%.

6. Hasil dan Visualisasi

Peta Risiko TB (Heatmap)

1. Kecamatan Ilir Timur dan Kalidoni menunjukkan *density* kasus TB tertinggi.
2. Faktor dominan: kepadatan penduduk > 9000/km^2 dan sanitasi buruk.
3. Visualisasi dibuat dengan **QGIS atau Tableau**, menampilkan zona merah risiko tinggi.

7. Hasil dan Visualisasi

1. Tren bulanan Kasus TB
2. Sebaran kasus berdasarkan usia dan jenis kelamin
3. Prediksi 3 bulan ke depan untuk setiap kecamatan

8. **Hasil dan Visualisasi**

   Manfaat Aplikasi:

   1. **Prediksi dini** memungkinkan intervensi cepat (misalnya fogging, kampanye kesehatan).
   2. **Efisiensi sumber daya**: alokasi tenaga medis ke daerah rawan.
   3. **Transparansi data** bagi pembuat kebijakan dan masyarakat.

9. **Tantangan Implementasi**

   - Integrasi data antar instansi.
   - Privasi dan keamanan data pasien.
   - Kualitas data input yang tidak seragam.
   - Kebutuhan SDM dengan literasi data tinggi.

7. 5 Tantangan dan Peluang Implementasi

7.5.1 Tantangan Utama

- Kualitas dan heterogenitas data

  Organisasi sering menghadapi data dalam format berbeda (terstruktur, semi-terstruktur, tidak terstruktur), dengan tingkat keandalan dan kebersihan yang bervariasi. Hal ini memperlambat proses persiapan data dan meningkatkan risiko hasil analitik yang kurang akurat. Sebuah tinjauan literatur menyebut "*skill shortages, dataset management, privacy, scalability, and intellectual property issues*" sebagai tantangan besar dalam 15 tahun riset big data (Tosi, Kokaj and Roccetti, 2024).

- Infrastruktur teknis dan skalabilitas

  Pemrosesan dan penyimpanan data berskala besar membutuhkan arsitektur yang kuat (data *lake/lakehouse, streaming, batch*) serta performa tinggi. Banyak organisasi mengalami kesulitan dalam memilih teknologi, mengelola biaya, dan mengoperasikan pipeline produksi secara andal. Sebuah studi menyimpulkan bahwa meskipun potensi besar, banyak pekerjaan masih harus dilakukan untuk "*seamlessly integrate Big Data into data-driven advanced software solutions*" (Tosi, Kokaj and Roccetti, 2024).

- *Governance*, privasi, dan regulasi

  Data besar sering mencakup informasi kesehatan, finansial, lokasi-sehingga memerlukan kontrol akses, anonymization, audit, dan kepatuhan regulasi. Tanpa tata kelola yang memadai, risiko kebocoran, pelanggaran etika, dan kepercayaan stakeholder meningkat. Misalnya, dalam konteks sistem informasi kesehatan, disebutkan "data privacy and security issues, complex data integration, and limited existing resources" sebagai tantangan utama (Ristevski, Savoska and Blazheska-Tabakovska, 2020).

- Kapabilitas SDM dan budaya organisasi

  Implementasi BDA bukan hanya soal teknologi, tetapi juga soal organisasi: ketersediaan data engineers, scientist, domain experts; budaya pengambilan keputusan berbasis data; adaptasi proses bisnis; dan resistensi terhadap perubahan. Studi kasus pada UKM mengungkap hambatan seperti "high implementation costs and lack of

skilled personnel" dalam adopsi big data (Noonpakdee, Phothichai and Khunkornsiri, 2018).

- Inovasi produk, layanan, dan model bisnis baru

  Data besar membuka peluang untuk model bisnis berbasis data ("*data-as-a-service*"), personalisasi layanan, dan insight kompetitif. Review literatur menunjukkan bahwa big data telah menjadi pilar dalam sektor keuangan, manufaktur, pendidikan, dan *smart cities.*

- Etika, bias, dan interpretabilitas model

  Model prediktif berbasis data besar bisa memperkuat bias historis atau menghasilkan keputusan yang tidak transparan. Organisasi di sektor regulasi tinggi (kesehatan, keuangan) perlu menjaga explainability (penjelasan model), fairness, dan auditabilitas. Sebuah review menyebut bias algoritmik dan privasi sebagai aspek yang harus dihadapi dalam penerapan big data (Badshah *et al.*, 2024).

7.5.2 Peluang yang Dapat Dimanfaatkan

- Efisiensi operasional dan penghematan biaya

  Dengan analisis big data yang tepat, organisasi dapat mengurangi downtime, memprediksi perawatan mesin, mengoptimalkan penggunaan energi, serta mempercepat respon operasional. Misalnya, dalam tinjauan aplikasi big data disebut bahwa "*big data applications transforming industries and shaping the future of how we live and work*" (Sharma, 2023).

- Peningkatan kualitas layanan dan pengambilan keputusan berbasis data

Di sektor kesehatan, misalnya, big data memungkinkan pemantauan pasien secara real-time, model prediktif penyakit, dan distribusi sumber daya yang lebih baik. Sebuah studi menemukan bahwa analitik big data memiliki potensi besar untuk mendukung keputusan dalam sistem informasi kesehatan (Jahani, Jain and Ivanov, 2023)

- Inovasi produk, layanan, dan model bisnis baru

  Data besar membuka peluang untuk model bisnis berbasis data ("*data-as-a-service*"), personalisasi layanan, dan insight kompetitif. Review literatur menunjukkan bahwa big data telah menjadi pilar dalam sektor keuangan, manufaktur, pendidikan, dan smart cities (Tosi, Kokaj and Roccetti, 2024).

- Kolaborasi lintas domain dan transformasi digital

  Penerapan big data mendorong kolaborasi antara data scientist, domain expert (dokter, insinyur, analis), dan manajemen bisnis. Ini memperkuat literasi data di organisasi dan mengubah proses menjadi lebih responsif dan adaptif terhadap perubahan pasar atau kondisi eksternal. Project transformasi publik menunjukkan bahwa integrasi AI dan big data dapat meningkatkan efisiensi dan transparansi pemerintahan jika didukung kebijakan dan kapasitas yang tepat (Ristevski, Savoska and Blazheska-Tabakovska, 2020).

- Peluang untuk penelitian & pengembangan lanjutan

  Karena masih banyak gap dalam standar, metodologi, dan implementasi real-world, terdapat peluang besar bagi organisasi dan peneliti untuk

mengeksplorasi arsitektur baru, teknik privasi, model hybrid, dan integrasi multi-sumber data. Sebuah review menyebut bahwa meskipun riset sudah banyak, "*a substantial amount of work remains to be done*" untuk integrasi penuh big data (Tosi, Kokaj and Roccetti, 2024).

7. 6 Kesimpulan

Analisis Big Data telah menjadi fondasi penting dalam pengambilan keputusan strategis di berbagai sektor, mulai dari kesehatan, keuangan, pendidikan, hingga industry. Pemanfaatan data dalam volume besar dengan kecepatan dan variasi tinggi telah memungkinkan organisasi memperoleh wawasan yang lebih mendalam untuk meningkatkan efisiensi dan inovasi. Meskipun demikian, tantangan terkait kualitas data, keamanan privasi, serta keterbatasan sumber daya manusia di bidang analitik masih menjadi hambatan utama dalam implementasi Big Data di berbagai negara berkembang, termasuk Indonesia.

Dari berbagai studi kasus yang telah diuraikan, dapat disimpulkan bahwa keberhasilan implementasi Big Data tidak hanya bergantung pada teknologi, tetapi juga pada strategi organisasi, tata kelola data (*data governance*), dan kesiapan sumber daya manusia. Peningkatan literasi data menjadi kunci agar hasil analisis Big Data dapat diterjemahkan menjadi keputusan yang efektif dan berkelanjutan (Ristevski, Savoska and Blazheska-Tabakovska, 2020).

DAFTAR PUSTAKA

Akbulut, D.H. and Kaya, I. (2018) "Big data analytics in financial reporting and accounting," *Pressacademia*, 7, pp. 256–259.

Badshah, A. *et al.* (2024) "Big data applications: overview, challenges and future," *Artificial Intelligence Review*, 57(11), p. 290.

Fahd, K. and Miah, S.J. (2023) "Designing and evaluating a big data analytics approach for predicting students' success factors," *Journal of Big Data*, 10(1), p. 159.

Jahani, H., Jain, R. and Ivanov, D. (2023) "Data science and big data analytics: a systematic review of methodologies used in the supply chain and logistics research," *Annals of Operations Research*, pp. 1–58.

Ma, Z., Jørgensen, B.N. and Ma, Z.G. (2024) "A systematic data characteristic understanding framework towards physical-sensor big data challenges," *Journal of Big Data*, 11(1), p. 84.

Noonpakdee, W., Phothichai, A. and Khunkornsiri, T. (2018) "Big data implementation for small and medium enterprises," in *2018 27th Wireless and Optical Communication Conference (WOCC)*. IEEE, pp. 1–5.

Nuha, N. *et al.* (2025) "Beyond the outbreak: a review of big data analytics in proactive infectious disease prevention for risk mitigation for COVID-19," *Journal of Big Data*, 12(1), p. 185.

Orji, U. *et al.* (2023) "Using Data Analytics to Derive Business Intelligence: A Case Study," in *The International Conference on Cybersecurity, Situational Awareness and Social Media*. Springer, pp. 35–46.

Rahul, K., Banyal, R.K. and Arora, N. (2023) "A systematic review on big data applications and scope for industrial processing and healthcare sectors," *Journal of Big Data*, 10(1), p. 133.

Ristevski, B., Savoska, S. and Blazheska-Tabakovska, N. (2020) "Opportunities for big data analytics in healthcare information systems development for decision support."

Sarker, S. *et al.* (2022) "A comprehensive review on big data for industries: challenges and opportunities," *IEEE Access*, 11, pp. 744–769.

Sharma, S. (2023) "Big data in finance: A systematic literature review," in *AIP Conference Proceedings*. AIP Publishing LLC, p. 30010.

Stojanov, A. and Daniel, B.K. (2024) "A decade of research into the application of big data and analytics in higher education: A systematic review of the literature," *Education and information technologies*, 29(5), pp. 5807–5831.

Tosi, D., Kokaj, R. and Roccetti, M. (2024) "15 years of Big Data: a systematic literature review," *Journal of Big Data*, 11(1), p. 73.

BAB 8

VISUALISASI, INTERPRETASI, DAN KOLABORASI PROYEK BIG DATA

8.1 Pendahuluan

Dalam era digital modern yang ditandai oleh pertumbuhan data yang masif, fenomena yang dikenal sebagai Big Data telah menjadi pusat perhatian dalam hampir setiap sektor kehidupan manusia. Big Data bukan sekadar istilah teknis, melainkan representasi dari revolusi informasi yang mengubah cara manusia memahami, berinteraksi, dan mengambil keputusan berdasarkan data. Setiap detik, jutaan perangkat digital, sensor, dan sistem informasi di seluruh dunia menghasilkan aliran data yang luar biasa besar, beragam, dan cepat. Data ini dapat berasal dari berbagai sumber seperti media sosial, transaksi keuangan, sistem kesehatan, jaringan transportasi, maupun sensor *Internet of Things* (IoT). Kecepatan dan volume data yang sangat besar tersebut melampaui kemampuan sistem konvensional dalam menyimpan, memproses, dan menganalisisnya, sehingga diperlukan pendekatan baru yang lebih terintegrasi dan cerdas untuk mengekstraksi makna dari tumpukan data tersebut. Di sinilah konsep Big Data memainkan peran fundamental: ia tidak hanya menekankan pada kuantitas data, tetapi pada kemampuan untuk mengekstraksi nilai, menemukan pola, dan mengubah data menjadi pengetahuan yang bermanfaat (Stephenson, 2018) .

Visualisasi data merupakan jembatan antara dunia digital yang penuh angka dan realitas manusia yang lebih mudah memahami informasi dalam bentuk visual. Otak manusia jauh lebih efisien dalam mengenali pola visual dibandingkan dengan membaca angka atau teks mentah. Melalui grafik, diagram, peta, atau dashboard interaktif, data yang kompleks dapat diubah menjadi informasi yang intuitif dan mudah dimengerti. Dalam konteks Big Data, visualisasi tidak lagi sekadar sarana pelengkap, tetapi menjadi elemen esensial yang menentukan keberhasilan proses analisis. Tanpa visualisasi yang baik, hasil analitik dari algoritma canggih pun sulit diterjemahkan menjadi kebijakan nyata atau keputusan bisnis yang efektif. Visualisasi memungkinkan para pemangku kepentingan dari berbagai latar belakang baik teknis maupun nonteknis untuk berkolaborasi dalam memahami fenomena yang diungkapkan oleh data.

Pentingnya visualisasi semakin meningkat ketika data yang dihadapi memiliki volume besar, beragam format, dan kecepatan tinggi. Misalnya, dalam analisis media sosial, data yang dihasilkan mencakup teks, gambar, video, serta metadata waktu dan lokasi. Untuk memahami sentimen publik terhadap suatu isu, analis perlu memvisualisasikan data tersebut dalam bentuk grafik tren, peta persebaran geografis, atau bahkan jaringan keterhubungan antar-pengguna. Teknologi seperti Tableau, Power BI dan Python dengan pustaka seperti Matplotlib, Seaborn, atau Plotly memberikan kemampuan visualisasi dinamis dan interaktif. Dashboard yang terhubung langsung ke database Big Data memungkinkan pengguna untuk melihat perubahan data secara *real-time,* menggali detail tertentu, dan membandingkan indikator antar periode. Dalam konteks pemerintahan, misalnya, dashboard Big Data dapat

digunakan untuk memantau distribusi vaksinasi, penyebaran penyakit, atau bahkan pergerakan ekonomi daerah secara langsung, membantu pengambil kebijakan membuat keputusan berbasis bukti yang cepat dan akurat (Härdle & Unwin, 2018).

Namun visualisasi hanyalah langkah awal dalam perjalanan memahami data besar. Setelah data divisualisasikan, langkah berikutnya adalah interpretasi, yaitu proses memberi makna terhadap pola yang muncul dalam visualisasi tersebut. Interpretasi membutuhkan keahlian multidisiplin yang menggabungkan pengetahuan statistik, logika analitik, dan pemahaman konteks domain. Seorang data scientist mungkin mampu mendeteksi pola matematis dari data penjualan, tetapi tanpa pemahaman bisnis, pola itu tidak akan berarti banyak. Sebaliknya, seorang manajer bisnis mungkin memahami konteks pasar dengan baik, namun tanpa interpretasi berbasis data, keputusannya bisa bersifat subjektif. Oleh karena itu, dalam proyek Big Data, kolaborasi antara berbagai pihak analis data, insinyur sistem, pakar domain, dan pemangku kepentingan menjadi kunci keberhasilan interpretasi yang valid dan bermanfaat.

Kolaborasi dalam proyek Big Data menuntut koordinasi lintas disiplin dan lintas fungsi. Seorang data engineer bertanggung jawab terhadap penyimpanan dan arsitektur data, seorang data scientist fokus pada pengembangan model dan algoritma analisis, sedangkan seorang business analyst berperan dalam menerjemahkan hasil analisis menjadi strategi atau rekomendasi praktis. Integrasi antarperan ini memerlukan komunikasi yang efektif dan alat kolaboratif yang mendukung. Platform seperti GitHub, JupyterHub, atau Google Colab memfasilitasi kerja sama tim dalam penulisan kode, analisis data, serta dokumentasi proses. Selain itu,

penggunaan sistem manajemen proyek seperti Jira atau Trello membantu tim menjaga konsistensi dan alur kerja yang transparan (Dougherty, 2021) .

Salah satu tantangan terbesar dalam proyek Big Data adalah menjaga konsistensi dan keakuratan data selama proses kolaborasi. Ketika data berasal dari berbagai sumber, kemungkinan terjadi duplikasi, inkonsistensi, atau kesalahan pencatatan sangat tinggi. Oleh karena itu, proses *data cleaning* dan *data validation* menjadi tahap krusial. Di sinilah visualisasi kembali berperan, bukan hanya untuk menampilkan hasil akhir, tetapi juga untuk mendeteksi anomali sejak tahap awal pemrosesan data. Misalnya, visualisasi distribusi nilai dapat membantu mendeteksi *outlier* atau data yang hilang sebelum digunakan dalam model analisis. Dengan demikian, visualisasi dan interpretasi tidak hanya saling melengkapi, tetapi juga membentuk siklus berulang yang memperbaiki kualitas analisis secara berkelanjutan.

Lebih jauh lagi, interpretasi data tidak selalu bersifat deterministik. Dalam banyak kasus, hasil analisis Big Data mengandung ketidakpastian yang harus dipahami dengan hati-hati. Sebuah korelasi yang ditemukan antara dua variabel tidak selalu menunjukkan hubungan sebab-akibat. Misinterpretasi terhadap pola data dapat menimbulkan kesimpulan yang menyesatkan, terutama ketika digunakan untuk kebijakan publik atau keputusan bisnis berskala besar. Karena itu, prinsip-prinsip ilmiah dan etika analisis data harus selalu dijaga. Interpretasi data yang bertanggung jawab berarti memahami keterbatasan model, menjelaskan tingkat kepercayaan hasil, dan mengkomunikasikan informasi secara transparan. Visualisasi yang jujur misalnya dengan menampilkan rentang ketidak pastian atau *margin of error*

membantu pengguna memahami konteks dan batasan temuan yang disajikan (Fawcett, 2013) .

Peran etika dalam proyek Big Data juga sangat penting dalam konteks kolaborasi dan visualisasi. Ketika data berisi informasi sensitif seperti data kesehatan, pendidikan, atau perilaku konsumen, privasi individu harus dilindungi. Visualisasi publik harus mempertimbangkan anonimisasi data dan menghindari pengungkapan informasi pribadi. Teknologi seperti *differential privacy* dan data masking digunakan untuk memastikan bahwa visualisasi tetap informatif tanpa melanggar kerahasiaan. Selain itu, kolaborasi yang melibatkan banyak pihak juga memerlukan kesepakatan terkait hak akses, kepemilikan data, dan tanggung jawab terhadap hasil analisis. Etika kolaborasi berarti membangun kepercayaan antaranggota tim, menghargai kontribusi setiap individu, serta menghindari manipulasi atau penyalahgunaan data untuk kepentingan tertentu.

Dalam konteks akademik dan penelitian, visualisasi dan interpretasi Big Data juga berperan penting dalam diseminasi ilmu pengetahuan. Data penelitian yang kompleks dapat dikomunikasikan dengan lebih efektif melalui visualisasi interaktif yang memungkinkan pembaca mengeksplorasi temuan secara mandiri. Misalnya, hasil pemodelan prediksi penyebaran penyakit dapat divisualisasikan dalam bentuk peta interaktif yang menunjukkan potensi area risiko tinggi berdasarkan waktu dan kondisi lingkungan. Kolaborasi antarpeneliti dari berbagai institusi juga semakin dimudahkan oleh platform berbasis cloud yang memungkinkan pertukaran data dan analisis secara langsung tanpa batas geografis. Pendekatan kolaboratif ini

mempercepat inovasi ilmiah, meningkatkan replikasi penelitian, serta memperluas dampak sosial dari hasil riset.

Big Data tidak hanya mengubah cara manusia memandang data, tetapi juga cara berpikir dan bekerja dalam organisasi modern. Paradigma lama yang berfokus pada intuisi kini bergeser ke arah pengambilan keputusan berbasis bukti. Dalam organisasi yang matang secara digital, setiap keputusan penting biasanya didukung oleh analisis data visual yang mudah diinterpretasikan oleh pimpinan. Misalnya, dalam sektor keuangan, analisis risiko kredit dilakukan dengan memvisualisasikan tren pembayaran, riwayat transaksi, dan perilaku pelanggan secara real-time. Dalam sektor kesehatan, rumah sakit menggunakan dashboard berbasis Big Data untuk memantau ketersediaan tempat tidur, jadwal operasi, dan efektivitas pengobatan. Semua itu tidak mungkin tercapai tanpa sinergi antara visualisasi, interpretasi, dan kolaborasi lintas fungsi.

Secara keseluruhan, pendahuluan ini menegaskan bahwa proyek Big Data bukan hanya soal teknologi penyimpanan atau pemrosesan data dalam skala besar, tetapi tentang bagaimana manusia mampu memahami data tersebut, memaknainya, dan bekerja sama untuk menghasilkan keputusan yang lebih baik. Visualisasi berperan sebagai media penerjemah data ke dalam bentuk yang dapat dipahami, interpretasi berperan dalam memberi makna dan konteks terhadap hasil visualisasi, sementara kolaborasi memastikan bahwa setiap proses berjalan secara terintegrasi dan berorientasi pada tujuan yang sama. Ketiganya saling berkelindan dan membentuk fondasi utama dalam ekosistem *Big Data modern.*

8.2 Peran Visualisasi dalam Big Data

Dalam ekosistem Big Data yang sarat akan volume data yang masif, kecepatan aliran informasi yang luar biasa, serta keberagaman format yang dihasilkan dari berbagai sumber digital seperti sensor IoT, media sosial, sistem transaksi daring, dan platform cloud, peran visualisasi data menjadi semakin krusial karena tanpa mekanisme penyajian visual yang tepat, hasil pengolahan data hanya akan menjadi deretan angka dan teks yang sulit dipahami oleh manusia, sehingga visualisasi berfungsi sebagai jembatan antara kompleksitas teknis dengan pemahaman intuitif pengguna, di mana proses ini mengubah kumpulan data mentah yang tersebar dan tak beraturan menjadi bentuk representasi visual yang informatif, menarik, dan mudah dicerna, seperti grafik, diagram, peta interaktif, serta dashboard dinamis yang mampu menampilkan pola tersembunyi, tren temporal, dan relasi antar variabel dalam konteks yang lebih bermakna, sehingga pengambil keputusan dapat memperoleh wawasan yang cepat, akurat, dan relevan dengan kebutuhan strategis organisasi. Dalam konteks Big Data, visualisasi tidak lagi dianggap sebagai tahap opsional di akhir proses analisis, melainkan menjadi komponen inti dalam keseluruhan siklus pengolahan data, karena visualisasi memungkinkan manusia untuk berinteraksi langsung dengan hasil analitik yang dihasilkan oleh sistem cerdas atau sistem penyimpanan terdistribusi, di mana data dalam jumlah besar yang telah diproses dapat divisualisasikan melalui alat-alat modern seperti Tableau, Power BI atau bahkan menggunakan bahasa pemrograman seperti Python dengan pustaka Matplotlib, Seaborn, dan Plotly, yang memberikan fleksibilitas tinggi dalam menghasilkan tampilan visual yang adaptif terhadap dinamika data (Healy, 2019).

Visualisasi berperan penting dalam menyederhanakan kompleksitas, sebab manusia memiliki kemampuan kognitif yang terbatas dalam memahami angka besar dan hubungan

matematis yang kompleks, namun sangat unggul dalam mengenali pola visual dan perbedaan warna, bentuk, atau posisi, sehingga dengan mengubah data kuantitatif ke dalam bentuk grafis, seseorang dapat dengan cepat mengidentifikasi anomali, pergerakan tren, atau korelasi antarvariabel tanpa perlu membaca tabel panjang yang membingungkan, dan hal ini menjadikan visualisasi sebagai bentuk komunikasi data yang efisien antara sistem komputasi dan pengguna.

Lebih jauh lagi, visualisasi dalam Big Data berperan sebagai alat demokratisasi informasi karena ia membuka akses bagi pengguna nonteknis untuk memahami hasil analisis yang sebelumnya hanya dapat dijangkau oleh pakar data, dengan menghadirkan antarmuka visual yang intuitif, interaktif, dan dapat dikustomisasi sesuai kebutuhan analisis masing-masing pengguna, misalnya pimpinan organisasi dapat memantau indikator kinerja utama melalui dashboard real-time yang menampilkan metrik produksi, penjualan, atau distribusi dalam satu layar, sedangkan analis dapat menggali lebih dalam setiap variabel dengan melakukan eksplorasi data interaktif yang memungkinkan pembaruan tampilan secara langsung berdasarkan filter tertentu seperti waktu, lokasi, atau kategori, dan kemampuan ini tidak hanya mempercepat pengambilan keputusan, tetapi juga meningkatkan transparansi serta akuntabilitas organisasi dalam memanfaatkan data secara efisien. Dalam proyek Big Data berskala besar, visualisasi juga berperan dalam mempercepat proses validasi dan komunikasi antaranggota tim multidisiplin, karena hasil analisis yang divisualisasikan dengan baik dapat memudahkan kolaborasi antara data engineer, data scientist, business analyst, dan pemangku kepentingan nonteknis, di mana semua pihak dapat memiliki persepsi yang sama terhadap fenomena yang sedang dianalisis untuk menampilkan sebaran kasus penyakit berdasarkan wilayah dan waktu, maka dokter, epidemiolog, dan pembuat kebijakan dapat berdiskusi berdasarkan

tampilan data yang sama tanpa harus memahami struktur tabel atau algoritma yang mendasarinya, sehingga visualisasi berfungsi sebagai bahasa universal yang menyatukan berbagai disiplin ilmu di bawah satu pemahaman data yang konsisten.

Selain sebagai sarana komunikasi, visualisasi Big Data juga menjadi alat eksplorasi yang memungkinkan pengguna menemukan pengetahuan baru yang sebelumnya tidak terlihat, karena dalam data yang sangat besar dan kompleks sering kali tersembunyi pola yang tidak dapat diungkap hanya dengan perhitungan statistik konvensional, tetapi menjadi jelas ketika divisualisasikan dalam bentuk multidimensi, misalnya visualisasi 3D atau *network graph* yang menampilkan keterkaitan antarentitas dalam analisis jejaring sosial dapat mengungkap hubungan pengaruh antarindividu atau kelompok dalam sebuah komunitas digital, sementara penggunaan *time series visualization* dengan *streaming data* dapat menunjukkan perubahan perilaku pelanggan secara real-time, memberikan peluang bagi perusahaan untuk menyesuaikan strategi pemasaran secara cepat. Teknologi visualisasi modern kini semakin terintegrasi dengan pendekatan berbasis kecerdasan buatan (AI) dan pembelajaran mesin (Machine Learning), di mana sistem dapat secara otomatis memilih bentuk visualisasi terbaik berdasarkan jenis data dan pola yang terdeteksi, bahkan menghasilkan *smart dashboard* yang menyesuaikan tampilan secara adaptif terhadap preferensi pengguna. Inovasi ini menjadikan visualisasi bukan hanya alat pasif untuk menampilkan hasil, tetapi juga sebagai asisten analitik yang proaktif membantu pengguna menafsirkan data dengan lebih cepat dan cerdas. Namun demikian, meskipun visualisasi memberikan manfaat besar, tantangan dalam penerapannya tetap signifikan, terutama terkait dengan skalabilitas, keakuratan, dan interpretabilitas.

8.1.1 Tujuan Visualisasi Data

Visualisasi data bertujuan untuk menyederhanakan kompleksitas informasi yang muncul dari kumpulan data besar agar mudah dipahami secara cepat. Dalam proyek big data, data yang dihasilkan sering kali terlalu banyak dan tidak terstruktur, sehingga visualisasi menjadi cara paling efektif untuk menampilkan informasi penting dalam bentuk yang lebih ringkas, jelas, dan intuitif. Melalui grafik, peta, atau dashboard, pengguna dapat langsung melihat gambaran umum tanpa harus membaca tabel panjang atau rumus yang rumit.

Selain menyederhanakan data, visualisasi membantu mengungkap pola, tren, dan anomali yang tersembunyi. Misalnya, dengan *heatmap* atau *line chart*, analis bisa mengenali kenaikan permintaan, perubahan perilaku pengguna, atau mendeteksi kejadian tidak normal yang bisa menjadi sinyal penting bagi strategi berikutnya. Hal ini menjadikan visualisasi sebagai alat eksplorasi awal sebelum analisis mendalam menggunakan metode statistik atau *machine learning*.

Visualisasi juga berperan penting dalam mendukung pengambilan keputusan berbasis data. Melalui alat seperti Tableau, Power BI, atau Google Data Studio, manajer dapat melihat indikator kinerja utama secara real-time dan segera mengambil tindakan berdasarkan bukti yang terlihat secara visual. Dengan begitu, keputusan yang diambil menjadi lebih cepat, akurat, dan terukur.

Terakhir, visualisasi meningkatkan komunikasi dan kolaborasi antaranggota tim lintas bidang. Karena tidak semua pihak memiliki latar belakang teknis yang sama, tampilan visual membantu menjembatani pemahaman antara analis data, peneliti, dan pengambil kebijakan. Data yang kompleks dapat diterjemahkan menjadi cerita

visual yang mudah dipahami semua pihak, sehingga memudahkan koordinasi dan kesamaan persepsi dalam menentukan langkah strategis. Secara keseluruhan, visualisasi data tidak hanya berfungsi untuk memperindah tampilan, tetapi menjadi alat penting untuk menyederhanakan kompleksitas, menemukan pola tersembunyi, mendukung keputusan berbasis bukti, dan memperkuat komunikasi di seluruh proses analisis big data.

8.1.2 Jenis Visualisasi Data

Dalam konteks Big Data, berbagai jenis visualisasi digunakan untuk menampilkan informasi sesuai dengan tujuan analisis dan karakteristik data yang diolah. Beberapa bentuk visualisasi yang umum digunakan antara lain sebagai berikut:

1. **Diagram Batang (*Bar Chart*)**

   Digunakan untuk menampilkan perbandingan antar-kategori atau kelompok data. Setiap batang mewakili nilai tertentu, sehingga perbedaan antar-kategori dapat dilihat dengan jelas. Visualisasi ini efektif untuk analisis data kategorikal seperti penjualan per produk, jumlah pengguna per wilayah, atau frekuensi kejadian tertentu.

2. **Diagram Garis (*Line Chart*)**

   Cocok digunakan untuk menggambarkan perubahan nilai terhadap waktu. Diagram garis membantu pengguna mengidentifikasi tren, pola musiman, atau fluktuasi dalam data deret waktu (*time series*), seperti pertumbuhan trafik website, harga saham, atau volume transaksi harian.

3. Diagram Lingkaran (*Pie Chart*)

Digunakan untuk menunjukkan proporsi atau persentase kontribusi setiap bagian terhadap keseluruhan data. Visualisasi ini umum dipakai untuk menampilkan komposisi seperti pangsa pasar, distribusi anggaran, atau proporsi jenis data tertentu. Meskipun sederhana, penggunaannya perlu hati-hati agar tidak membingungkan ketika kategori terlalu banyak.

4. Peta Panas (*Heatmap*)

Menampilkan distribusi intensitas data melalui gradasi warna. Semakin tinggi nilai suatu variabel, semakin kuat warna yang ditampilkan. *Heatmap* sering digunakan untuk analisis korelasi antarvariabel, distribusi aktivitas pengguna pada situs web, atau pola interaksi dalam sistem kompleks.

5. Graf Jaringan (*Network Graph*)

Digunakan untuk menggambarkan hubungan atau keterkaitan antar-entitas dalam suatu sistem. Setiap titik (*node*) mewakili entitas, sedangkan garis penghubung (*edge*) menunjukkan hubungan di antaranya. Jenis visualisasi ini sering digunakan dalam analisis jejaring sosial, koneksi antar-server, atau hubungan antar-topik dalam analisis teks.

6. Dashboard Interaktif

Merupakan kumpulan berbagai visualisasi dalam satu tampilan terpadu yang dapat diperbarui secara real-time. Dashboard memungkinkan pengguna memantau metrik penting dan berinteraksi dengan data secara langsung. Alat populer untuk membuat dashboard interaktif antara lain Tableau, Microsoft Power BI, dan Apache

Superset. Dashboard digunakan dalam berbagai bidang seperti bisnis, kesehatan, pendidikan, maupun pemerintahan untuk mendukung pengambilan keputusan berbasis data secara cepat dan visual.

8.1.3 Prinsip Desain Visualisasi

Agar visualisasi data dapat menyampaikan informasi secara efektif dan mudah dipahami, diperlukan penerapan prinsip desain yang tepat. Prinsip-prinsip ini membantu memastikan bahwa pesan yang disampaikan melalui visualisasi tetap jelas, akurat, dan relevan dengan tujuan analisis. Beberapa prinsip penting yang perlu diperhatikan antara lain:

1. **Kejelasan (*Clarity*)**

   Visualisasi harus mampu menyampaikan informasi secara langsung tanpa menimbulkan kebingungan. Hindari penggunaan elemen visual yang berlebihan seperti efek tiga dimensi, warna mencolok, atau dekorasi yang tidak memiliki fungsi informatif. Fokus utama harus diarahkan pada pesan yang ingin disampaikan agar pengguna dapat memahami data dengan cepat dan akurat.

2. **Konsistensi (*Consistency*)**

   Gunakan warna, simbol, skala, dan gaya tampilan yang seragam di seluruh visualisasi untuk menjaga keselarasan interpretasi. Konsistensi membantu pengguna mengenali pola dan membandingkan data antarbagian dengan lebih mudah, serta memperkuat identitas visual dari laporan atau dashboard yang ditampilkan.

3. Relevansi (*Relevance*)

Setiap elemen visual harus memiliki tujuan yang jelas dan mendukung analisis yang dilakukan. Hindari menampilkan data atau variabel yang tidak relevan karena dapat mengaburkan pesan utama. Visualisasi yang efektif adalah yang fokus pada informasi inti dan membantu pengguna menjawab pertanyaan analitik dengan cepat.

4. Interaktivitas (*Interactivity*)

Dalam konteks Big Data, interaktivitas menjadi aspek penting yang memungkinkan pengguna untuk mengeksplorasi data secara mandiri. Dengan fitur seperti *filter*, *drill-down*, atau *hover detail*, pengguna dapat menggali lapisan informasi yang lebih dalam sesuai kebutuhan analisis. Hal ini meningkatkan keterlibatan pengguna sekaligus memperkaya pemahaman terhadap data yang kompleks.

8.1.4 Teknik dan Alat Visualisasi dalam Big Data

Kemajuan teknologi informasi telah menghadirkan berbagai alat (*tools*) dan teknik yang memungkinkan visualisasi data dalam skala besar dilakukan secara lebih efisien, interaktif, dan real-time. Dalam konteks Big Data, visualisasi tidak hanya bertujuan untuk menampilkan grafik sederhana, tetapi juga untuk membantu memahami pola tersembunyi, tren waktu nyata, serta relasi kompleks antar-variabel. Beberapa kategori alat visualisasi utama yang banyak digunakan baik di lingkungan akademik maupun industri dapat dijelaskan sebagai berikut.

A. Tools Visualisasi

1. Tableau dan Power BI

Kedua platform ini merupakan alat visualisasi yang paling populer di dunia bisnis dan organisasi karena kemampuannya dalam membuat *dashboard* interaktif yang menarik dan mudah digunakan. Tableau memiliki kekuatan dalam *data blending*, *real-time analytics*, serta konektivitas dengan berbagai sumber data seperti SQL Server, Hadoop, dan Google BigQuery. Sementara itu, Power BI yang dikembangkan oleh Microsoft lebih terintegrasi dengan ekosistem Office 365 dan Azure, menjadikannya solusi ideal untuk organisasi yang sudah menggunakan produk Microsoft. Keduanya mendukung pembuatan laporan otomatis, berbagi hasil analisis secara kolaboratif, dan pemantauan indikator kinerja utama (*Key Performance Indicators/KPI*) secara real-time.

2. Python Libraries (Matplotlib, Seaborn, Plotly, Bokeh)

Bahasa pemrograman Python menjadi salah satu pilihan utama dalam dunia analisis data dan sains data karena ekosistem pustaka visualisasi yang sangat luas.

- **Matplotlib** merupakan pustaka dasar yang memungkinkan pembuatan grafik dua dimensi dengan kontrol penuh terhadap setiap elemen visual.
- **Seaborn**, dibangun di atas Matplotlib, menyederhanakan pembuatan grafik

statistik dengan desain yang lebih estetis dan intuitif.

- **Plotly** mendukung visualisasi interaktif berbasis web dengan kemampuan ekspor ke format HTML, cocok untuk dashboard analitik.
- **Bokeh** berfokus pada visualisasi interaktif berskala besar dan dapat diintegrasikan langsung dengan server aplikasi berbasis web. Kombinasi pustaka ini memungkinkan analis data menghasilkan visualisasi yang tidak hanya informatif tetapi juga dapat diotomatisasi dalam pipeline analitik.

3. R (ggplot2 dan Shiny)

R merupakan bahasa yang kuat dalam statistik dan analisis data, dengan ekosistem visualisasi yang kaya.

- **ggplot2** adalah paket paling terkenal untuk membuat grafik yang elegan dan kompleks menggunakan konsep *grammar of graphics*, di mana setiap elemen visual dapat dikombinasikan secara sistematis.
- **Shiny** memungkinkan pengguna membangun aplikasi web interaktif untuk menampilkan hasil analisis dan visualisasi data secara dinamis tanpa perlu menguasai pemrograman web secara mendalam.

Keduanya menjadikan R sangat efektif digunakan oleh peneliti, akademisi, maupun data scientist yang memerlukan pendekatan analitik berbasis statistik inferensial.

4. **Apache Superset dan Kibana** Dalam lingkungan Big Data, alat seperti **Apache Superset** dan **Kibana** sangat penting karena mampu menangani volume data yang sangat besar dan terdistribusi.

   - **Apache Superset**, dikembangkan oleh Apache Foundation, merupakan platform *open source* yang mendukung integrasi dengan berbagai sistem data seperti Hive, Presto, dan Druid. Superset memungkinkan pembuatan dashboard yang efisien dan visualisasi berskala besar tanpa memerlukan pengolahan manual yang berat.
   - **Kibana**, di sisi lain, merupakan bagian dari *Elastic Stack* (ELK: Elasticsearch, Logstash, Kibana) dan digunakan secara luas untuk menganalisis data log, keamanan siber, dan *real-time monitoring*. Kedua alat ini banyak digunakan dalam sistem berbasis *distributed computing* karena kemampuannya menampilkan data hasil query dalam waktu hampir seketika.

5. **D3.js (*Data-Driven Documents*)**

   D3.js adalah pustaka JavaScript yang sangat fleksibel untuk membuat visualisasi berbasis web yang dinamis dan interaktif. Dengan pendekatan berbasis DOM (*Document Object Model*), D3.js memungkinkan manipulasi elemen HTML dan SVG secara langsung berdasarkan data, sehingga hasil visualisasi

dapat disesuaikan sepenuhnya dengan kebutuhan pengguna. Visualisasi yang dihasilkan bersifat *responsive*, dapat dikustomisasi dengan efek animasi, serta diintegrasikan ke dalam situs web atau aplikasi analitik. Walaupun membutuhkan kemampuan pemrograman yang lebih tinggi dibandingkan alat visualisasi *drag-and-drop*, D3.js menjadi pilihan utama untuk proyek-proyek visualisasi yang memerlukan kebebasan desain dan fleksibilitas tingkat lanjut.

8.1.6 Tantangan Visualisasi Big Data

Meskipun visualisasi berperan penting dalam memahami dan mengkomunikasikan hasil analisis Big Data, penerapannya dihadapkan pada sejumlah tantangan teknis dan konseptual. Tantangan-tantangan ini muncul akibat karakteristik utama Big Data yang dikenal dengan istilah *5V* (Volume, Velocity, Variety, Veracity, dan Value). Oleh karena itu, visualisasi dalam konteks Big Data tidak hanya menuntut kemampuan menampilkan data dalam bentuk grafis, tetapi juga memastikan bahwa representasi tersebut relevan, cepat, dan mudah diinterpretasikan. Beberapa tantangan utama yang dihadapi dalam proses visualisasi Big Data antara lain:

1. Skalabilitas (*Scalability*)

Salah satu kendala terbesar dalam visualisasi Big Data adalah keterbatasan kemampuan sistem untuk menampilkan jutaan hingga miliaran titik data secara efisien dalam satu grafik. Ketika volume data sangat besar, tampilan visual dapat menjadi padat, tidak informatif, dan sulit dibaca. Pendekatan tradisional seperti *scatter plot* atau *bar chart* sering kali tidak mampu menangani kompleksitas data

dalam skala besar tersebut. Oleh karena itu, dibutuhkan teknik visualisasi yang lebih adaptif seperti *sampling*, *aggregation*, atau *progressive rendering* untuk menjaga keseimbangan antara detail dan keterbacaan. Selain itu, penggunaan infrastruktur berbasis *cloud computing* dan *distributed visualization frameworks* menjadi penting untuk mendukung skalabilitas tampilan data besar.

2. Kecepatan Pemrosesan (*Processing Speed*)

Dalam era analitik real-time, visualisasi tidak hanya menampilkan data statis tetapi juga harus mampu merefleksikan perubahan data secara langsung. Tantangan muncul ketika sistem harus memproses dan memperbarui visualisasi dalam waktu sangat singkat, terutama pada data yang bersumber dari *streaming platforms* seperti Apache Kafka atau Flink. Arsitektur yang tidak dioptimalkan dapat menyebabkan latensi tinggi, pembaruan grafik yang lambat, atau bahkan kegagalan dalam menampilkan data terbaru. Untuk mengatasinya, digunakan pendekatan *real-time data pipelines* dan *in-memory computation* agar visualisasi dapat menyesuaikan secara dinamis dengan aliran data yang terus berubah.

3. Interpretabilitas (Interpretability)

Tantangan lain yang sering diabaikan adalah bagaimana memastikan bahwa visualisasi yang dihasilkan benar-benar bermakna bagi pengguna. Sebuah grafik yang menarik secara estetika belum tentu menyampaikan informasi yang akurat atau mudah dipahami. Tanpa konteks yang jelas, pengguna dapat salah menafsirkan pola atau tren yang ditampilkan. Oleh karena itu, desain visualisasi

harus selalu mempertimbangkan audiens, tujuan analisis, serta narasi data yang ingin disampaikan. Penggunaan anotasi, legenda, dan penjelasan tambahan dapat membantu meningkatkan pemahaman terhadap hasil visualisasi yang kompleks.

Secara keseluruhan, keberhasilan visualisasi dalam Big Data sangat bergantung pada keseimbangan antara aspek teknis, desain, dan interpretasi. Visualisasi yang baik bukan hanya mampu menampilkan data dalam jumlah besar, tetapi juga menyederhanakan kompleksitas tersebut menjadi informasi yang dapat ditindaklanjuti secara cepat dan tepat.

8.2 Interpretasi Data: dari Visual ke *Insight*

Visualisasi data hanyalah langkah awal dalam memahami Big Data. Nilai sebenarnya muncul ketika data yang telah divisualisasikan diinterpretasikan untuk menghasilkan *insight* atau pemahaman yang bermakna. Interpretasi data adalah proses memberi makna terhadap pola, tren, atau hubungan yang muncul dari hasil visualisasi agar dapat digunakan dalam pengambilan keputusan.

Melalui interpretasi, grafik dan diagram tidak hanya menjadi tampilan visual, tetapi juga menjadi dasar untuk menjawab pertanyaan penting seperti *mengapa sesuatu terjadi* dan *apa yang harus dilakukan selanjutnya*. Misalnya, grafik peningkatan penjualan dapat diinterpretasikan sebagai hasil dari strategi pemasaran yang efektif atau adanya tren baru di pasar. Dengan interpretasi yang tepat, organisasi dapat mengambil keputusan yang lebih cepat, tepat, dan berbasis data nyata.

Namun, interpretasi yang salah bisa menimbulkan kesimpulan keliru. Tidak semua korelasi berarti sebab-akibat, sehingga konteks dan pemahaman terhadap domain analisis sangat diperlukan. Di sinilah kemampuan berpikir kritis dan pengetahuan lapangan menjadi faktor penting dalam membaca hasil visualisasi.

Selain itu, interpretasi data juga perlu dikomunikasikan dengan cara yang mudah dipahami oleh semua pihak. Melalui pendekatan *data storytelling*, hasil analisis dapat dijelaskan dalam bentuk narasi sederhana yang membantu pembuat kebijakan, peneliti, maupun masyarakat umum memahami makna di balik angka.

Dalam Big Data modern, proses interpretasi sering dibantu oleh teknologi seperti *machine learning* dan *predictive analytics*. Meski demikian, hasil dari sistem otomatis tetap memerlukan penilaian manusia agar sesuai dengan konteks dan tidak menimbulkan bias.

Secara keseluruhan, interpretasi adalah tahap penting yang mengubah data dari sekadar informasi visual menjadi pemahaman yang bernilai. Tanpa interpretasi, visualisasi hanyalah gambar; tetapi dengan interpretasi yang baik, data dapat menjadi dasar tindakan nyata dan keputusan strategis.

8.2.1 Proses Interpretasi

Proses interpretasi data bertujuan mengubah hasil visualisasi menjadi pemahaman yang bermakna dan dapat ditindaklanjuti. Tahapan utamanya meliputi:

1. Mengenali Pola dan Tren

Langkah pertama adalah mengidentifikasi pola, hubungan, atau tren dari hasil visualisasi. Misalnya, adanya peningkatan penjualan pada periode tertentu dapat menunjukkan pola musiman atau pengaruh kampanye promosi.

2. Menguji Hipotesis

Setelah pola ditemukan, tahap berikutnya adalah menguji dugaan atau hipotesis penyebabnya. Analisis ini membantu menentukan apakah perubahan yang terlihat disebabkan oleh kebijakan internal, kondisi pasar, atau faktor eksternal lainnya.

3. Menarik Kesimpulan Berbasis Bukti

Tahap terakhir adalah menyusun kesimpulan yang didukung oleh data dan konteks. Hasil interpretasi kemudian diterjemahkan menjadi rekomendasi yang dapat digunakan untuk pengambilan keputusan atau perencanaan strategi berikutnya.

8.2.2 Bias dan Kesalahan Interpretasi

Interpretasi data yang tidak tepat dapat menimbulkan kesimpulan keliru dan berpotensi menyesatkan proses pengambilan keputusan. Dalam analisis Big Data, bias dan kesalahan interpretasi sering terjadi karena pengabaian konteks, pemilihan data yang tidak seimbang, atau pemahaman yang terbatas terhadap hasil visualisasi. Beberapa bentuk bias umum yang perlu dihindari antara lain:

1. Confirmation Bias

Terjadi ketika analis hanya mencari dan menafsirkan data yang mendukung keyakinan atau hipotesis awal, sambil mengabaikan data lain yang mungkin bertentangan. Bias ini dapat menyebabkan hasil analisis menjadi tidak objektif dan menyesatkan.

2. Sampling Bias

Muncul ketika data yang digunakan tidak mewakili keseluruhan populasi. Akibatnya, interpretasi yang dihasilkan tidak akurat dan tidak dapat digeneralisasikan. Misalnya, menganalisis perilaku konsumen hanya dari satu wilayah tanpa mempertimbangkan daerah lain dengan karakteristik berbeda.

3. Overfitting Insight

Terjadi ketika peneliti terlalu fokus pada pola tertentu hingga melihat hubungan palsu antar-variabel yang sebenarnya tidak signifikan. Hal ini sering muncul saat data sangat besar dan kompleks, di mana kebetulan statistik dapat disalahartikan sebagai hubungan kausal.

Oleh karena itu, setiap proses interpretasi harus disertai validasi statistik, uji konsistensi data, serta tinjauan ulang terhadap konteks analisis. Langkah-langkah ini memastikan bahwa *insight* yang dihasilkan benar-benar mencerminkan kondisi nyata dan dapat digunakan sebagai dasar keputusan yang andal.

8.2.3 *Visual Storytelling*

Interpretasi data yang efektif tidak hanya bergantung pada analisis, tetapi juga pada cara hasil tersebut dikomunikasikan. Salah satu pendekatan yang semakin populer adalah visual storytelling, yaitu metode penyampaian pesan yang menggabungkan data, narasi, dan desain visual untuk membentuk alur cerita yang menarik dan mudah dipahami.

Melalui visual storytelling, data tidak hanya ditampilkan dalam bentuk grafik atau angka, tetapi

diubah menjadi cerita yang memiliki konteks dan makna. Pendekatan ini membantu audiens memahami pesan utama secara lebih emosional dan rasional, sehingga mempermudah proses pengambilan keputusan. Misalnya, grafik tren kesehatan publik yang disertai narasi tentang dampak kebijakan vaksinasi dapat memberikan gambaran yang lebih jelas mengenai pentingnya tindakan preventif dan mendorong partisipasi masyarakat.

Dengan menggabungkan elemen visual yang kuat dan narasi yang relevan, visual storytelling menjembatani kesenjangan antara data teknis dan pemahaman manusia. Pendekatan ini menjadikan hasil interpretasi tidak hanya informatif, tetapi juga inspiratif dan berdampak nyata.

8.3 Kolaborasi dalam Proyek Big Data

Proyek Big Data tidak dapat diselesaikan oleh satu individu atau satu bidang keahlian saja. Kompleksitas data yang sangat besar, beragam, dan terus berubah menuntut kerja sama lintas disiplin agar seluruh tahapan — mulai dari pengumpulan, penyimpanan, analisis, hingga interpretasi — dapat berjalan efektif. Oleh karena itu, kolaborasi menjadi elemen kunci dalam keberhasilan proyek Big Data.

Dalam praktiknya, kolaborasi melibatkan berbagai peran utama dengan tanggung jawab yang saling melengkapi:

1. ***Data Engineer***

   Bertugas membangun dan memelihara infrastruktur data seperti pipeline, sistem penyimpanan, dan integrasi antar-platform. Mereka memastikan data dapat diakses dengan cepat, bersih, dan siap untuk dianalisis.

2. ***Data Scientist***

Berperan dalam menganalisis data menggunakan metode statistik, algoritma machine learning, dan model prediktif. Mereka menerjemahkan data mentah menjadi insight yang bernilai dan mendukung pengambilan keputusan strategis.

3. ***Business Analyst***

Menjembatani antara tim teknis dan pihak manajemen. Business analyst memastikan hasil analisis data selaras dengan kebutuhan bisnis, serta mampu mengubah insight menjadi rekomendasi yang aplikatif.

4. ***Domain Expert***

Memberikan konteks dan pemahaman mendalam tentang bidang spesifik yang sedang dianalisis, seperti kesehatan, keuangan, pendidikan, atau pemasaran. Tanpa perspektif mereka, interpretasi data bisa kehilangan relevansi praktis.

Kolaborasi yang baik memerlukan komunikasi yang terbuka, penggunaan alat kerja kolaboratif (seperti Git, JupyterHub, atau platform cloud analytics), serta budaya berbagi pengetahuan antaranggota tim. Dengan sinergi antarprofesi ini, proyek Big Data dapat menghasilkan solusi yang tidak hanya akurat secara teknis, tetapi juga relevan, berdampak, dan berkelanjutan bagi pengambilan keputusan di berbagai sektor.

8.3.1 Platform Kolaborasi

Dalam proyek Big Data, kolaborasi yang efektif memerlukan dukungan platform digital yang memfasilitasi kerja tim lintas peran. Beberapa alat yang umum digunakan antara lain:

1. Google Colab / JupyterHub – memungkinkan analisis kode dan eksperimen data dilakukan secara kolaboratif dalam lingkungan berbasis notebook.
2. GitHub dan GitLab – menyediakan sistem manajemen versi untuk menyimpan, mengontrol, dan berbagi kode proyek secara terstruktur.
3. Slack / Microsoft Teams – digunakan untuk komunikasi, koordinasi, dan berbagi pembaruan antaranggota tim secara real-time.
4. DataOps dan MLOps Pipeline – mendukung otomatisasi alur kerja analisis dan model machine learning, sehingga integrasi antar-tahapan berjalan efisien dan konsisten.

8.3.2 Tantangan Kolaborasi dalam Big Data

Meskipun kolaborasi memberikan banyak manfaat, pelaksanaannya dalam proyek Big Data sering menghadapi sejumlah tantangan. Beberapa di antaranya meliputi:

1. **Perbedaan latar belakang keahlian** – anggota tim berasal dari bidang yang berbeda, seperti teknik data, bisnis, dan analisis statistik, sehingga sering muncul kesenjangan pemahaman teknis maupun konseptual.
2. **Manajemen data besar dan kompleks** – koordinasi dalam mengakses, menyimpan, serta memproses data berukuran terabyte atau petabyte dapat menimbulkan masalah sinkronisasi dan keamanan.
3. **Keterbatasan komunikasi lintas tim** – miskomunikasi dalam interpretasi hasil analisis dapat menyebabkan keputusan yang tidak tepat.
4. **Masalah privasi dan kepatuhan regulasi** – berbagi data antaranggota atau antarorganisasi

sering kali dibatasi oleh kebijakan keamanan dan peraturan perlindungan data.

5. **Kurangnya standar dokumentasi dan pelacakan versi** – tanpa sistem kontrol versi yang baik, perubahan pada kode, model, atau dataset dapat menyebabkan inkonsistensi hasil analisis.

Untuk mengatasi tantangan tersebut, tim perlu membangun budaya kolaboratif yang terbuka, menetapkan standar komunikasi yang jelas, serta menerapkan teknologi manajemen proyek dan keamanan data yang terintegrasi.

8.3.3 Studi Kasus: Kolaborasi dan Visualisasi dalam Analisis Data Kesehatan

Sebagai contoh penerapan nyata integrasi visualisasi, interpretasi, dan kolaborasi dalam proyek Big Data, dapat dilihat dari studi kasus analisis **data kesehatan ibu dan anak** yang dilakukan oleh sebuah tim riset universitas bekerja sama dengan pemerintah daerah. Tujuan utama proyek ini adalah untuk memahami faktor-faktor yang memengaruhi **angka stunting** dan merumuskan kebijakan intervensi yang lebih efektif.

Data dikumpulkan dari berbagai sumber, antara lain **rekam medis fasilitas kesehatan, sistem informasi kesehatan daerah, serta hasil survei lapangan**. Volume data yang besar dan bersifat heterogen ini menuntut proses pembersihan serta integrasi data sebelum dianalisis.

Setelah data siap, tim menggunakan **Tableau** dan **Power BI** untuk melakukan **visualisasi interaktif**, seperti peta distribusi stunting per wilayah, tren tahunan, serta grafik hubungan antara status pendidikan ibu, pendapatan keluarga, dan tingkat gizi anak. Melalui

visualisasi ini, pola penurunan angka stunting dapat diamati secara lebih jelas dan cepat dipahami oleh seluruh anggota tim, termasuk pihak non-teknis seperti pejabat kesehatan daerah.

Tim riset bersifat **multidisiplin**, terdiri atas **ahli gizi, dokter, epidemiolog, data scientist, dan analis kebijakan publik**. Kolaborasi lintas bidang ini sangat penting karena masing-masing memiliki perspektif berbeda dalam menafsirkan hasil analisis. Ahli gizi menilai hubungan antara pola makan dan status gizi, sementara data scientist memastikan model analitik dan visualisasi yang digunakan akurat secara statistik.

Hasil interpretasi menunjukkan adanya **korelasi signifikan antara tingkat pendidikan ibu dan status gizi anak**, terutama pada daerah dengan akses informasi kesehatan terbatas. Berdasarkan temuan tersebut, pemerintah daerah kemudian menyusun **program intervensi gizi terpadu**, yang melibatkan peningkatan edukasi ibu hamil serta kampanye pola makan sehat di tingkat rumah tangga.

Melalui **kolaborasi lintas bidang dan visualisasi yang efektif**, penelitian ini tidak hanya menghasilkan insight ilmiah, tetapi juga berdampak langsung terhadap perumusan kebijakan publik. Studi kasus ini menunjukkan bahwa integrasi antara data, teknologi, dan kerja sama manusia dapat mengubah data kompleks menjadi dasar pengambilan keputusan yang lebih **tepat sasaran dan berorientasi pada hasil nyata**.

8.4 Tren dan Masa Depan Visualisasi dan Kolaborasi dalam Big Data

Perkembangan teknologi data terus menghadirkan inovasi dalam cara manusia memvisualisasikan, menafsirkan, dan berkolaborasi dengan data. Di masa depan, visualisasi Big Data tidak hanya berfokus pada tampilan grafis, tetapi juga pada kemampuan adaptif, interaktif, dan prediktif untuk mendukung pengambilan keputusan secara real-time dan kolaboratif lintas platform serta disiplin ilmu.

Integrasi antara kecerdasan buatan (AI), pembelajaran mesin (*machine learning*), dan analisis visual akan semakin memperkuat peran manusia dalam memahami makna di balik data. Dengan dukungan kolaborasi digital yang semakin kuat, masa depan Big Data akan mengarah pada sistem yang lebih cerdas, transparan, dan berorientasi pada insight guna menghasilkan keputusan yang cepat, akurat, dan bertanggung jawab.

Selain itu, perkembangan visualisasi berbasis *augmented reality* (AR) dan *virtual reality* (VR) mulai membuka dimensi baru dalam eksplorasi data. Pengguna dapat berinteraksi langsung dengan data dalam ruang tiga dimensi, memungkinkan pemahaman yang lebih intuitif dan mendalam terhadap pola serta hubungan antarvariabel. Pendekatan ini sangat potensial dalam bidang kesehatan, pendidikan, hingga perencanaan kota berbasis data (*smart city*).

Di sisi lain, tren masa depan juga menekankan pentingnya kolaborasi manusia–mesin dalam proses interpretasi data. Mesin mampu mengolah dan menampilkan data dalam skala besar, sementara manusia memberikan konteks, intuisi, dan nilai etika dalam pengambilan keputusan. Sinergi antara keduanya akan menjadi fondasi bagi terciptanya ekosistem Big Data yang berkelanjutan, inklusif, dan berorientasi pada kemajuan sosial.

8.5 Tren Utama Visualisasi Big Data

Beberapa tren yang sedang berkembang dalam bidang visualisasi dan kolaborasi data antara lain:

1. **Visualisasi Real-Time dan Streaming Data**

   Dengan meningkatnya data yang dihasilkan dari sensor IoT, media sosial, dan aplikasi daring, kebutuhan akan visualisasi yang mampu menampilkan informasi secara *real-time* semakin penting. Teknologi seperti Apache Kafka dan Spark Streaming memungkinkan pembaruan data secara langsung sehingga pengguna dapat memantau perubahan secara dinamis.

2. **Integrasi Kecerdasan Buatan (AI) dan Pembelajaran Mesin (*Machine Learning*)**

   AI kini digunakan untuk membuat visualisasi yang lebih cerdas, misalnya dengan otomatis memilih jenis grafik paling sesuai atau mendeteksi anomali tanpa campur tangan manusia. Pendekatan ini membantu pengguna awam memahami data kompleks dengan lebih mudah.

3. **Visualisasi Berbasis *Augmented Reality* (AR) dan *Virtual Reality* (VR)**

   AR dan VR mulai digunakan untuk menghadirkan visualisasi data tiga dimensi yang imersif. Teknologi ini memungkinkan pengguna menjelajahi data dalam ruang virtual seperti melihat hubungan antarvariabel dalam bentuk ruang 3D yang memberikan pengalaman analisis lebih mendalam.

4. **Dashboard Adaptif dan Interaktif**

   Platform modern seperti Power BI, Looker, dan Tableau kini menawarkan *dashboard* yang menyesuaikan tampilan berdasarkan peran pengguna. Misalnya, eksekutif akan melihat metrik strategis,

sementara analis data melihat detail teknis. Pendekatan ini meningkatkan efisiensi dan relevansi informasi.

5. Kolaborasi Terintegrasi Berbasis Cloud

Kolaborasi kini berpindah ke lingkungan berbasis cloud, memungkinkan tim bekerja dari lokasi berbeda namun tetap terhubung pada dataset yang sama. Layanan seperti Google BigQuery, Databricks, dan Snowflake menjadi penghubung utama bagi kerja kolaboratif lintas organisasi.

8.6 Masa Depan Interpretasi dan Kolaborasi Data

Masa depan Big Data akan sangat ditentukan oleh bagaimana manusia berinteraksi dengan data. Tren menunjukkan bahwa analisis data akan semakin bersifat kolaboratif, multimodal, dan berorientasi konteks.

1. Analisis Kolaboratif Berbantuan AI

Sistem analitik di masa depan akan mampu memberikan rekomendasi otomatis, menyoroti insight penting, dan menyarankan langkah strategis. AI akan berperan sebagai "asisten analisis" yang mempercepat interpretasi data.

2. Demokratisasi Data (*Data Democratization*)

Akses terhadap data tidak lagi terbatas pada ahli statistik atau data scientist. Platform yang intuitif memungkinkan siapa pun di organisasi untuk melakukan eksplorasi data dan menghasilkan insight sendiri, dengan tetap menjaga tata kelola dan keamanan.

3. Kolaborasi Multidisiplin Berbasis Data

Kolaborasi akan melibatkan berbagai bidang keahlian teknologi, sosial, ekonomi, dan kebijakan public untuk menghasilkan solusi komprehensif terhadap

permasalahan kompleks seperti kesehatan, pendidikan, dan lingkungan.

4. **Integrasi Analitik dan Keputusan Otomatis (*Automated Decision Systems*)**

   Sistem pengambilan keputusan otomatis yang didukung analitik prediktif akan semakin banyak digunakan. Namun, manusia tetap memegang peran penting dalam menilai konteks dan etika dari keputusan yang dihasilkan sistem.

8.7 Kesimpulan

Tren dan masa depan visualisasi, interpretasi, dan kolaborasi dalam Big Data menunjukkan arah yang semakin cerdas, kolaboratif, dan beretika. Transformasi ini menandai pergeseran paradigma dari sekadar pengolahan data menjadi proses pemahaman yang menyeluruh, di mana manusia dan teknologi bekerja berdampingan untuk mengubah data mentah menjadi pengetahuan yang bermakna.

Visualisasi berperan penting sebagai bahasa universal dalam komunikasi data. Melalui tampilan grafis yang intuitif dan interaktif, visualisasi membantu menyederhanakan kompleksitas Big Data sehingga pola, hubungan, dan anomali dapat dipahami secara cepat. Di sisi lain, interpretasi menjadi proses yang memberi makna terhadap visualisasi tersebut, menghubungkannya dengan konteks nyata, serta menuntun pengambilan keputusan yang lebih akurat dan relevan.

Ketika visualisasi yang kuat dipadukan dengan interpretasi yang tepat dan kolaborasi lintas disiplin, terciptalah ekosistem analitik yang holistik. Kemampuan analitik mesin mempercepat proses pengolahan dan deteksi pola, sedangkan intuisi manusia memastikan hasil interpretasi selaras dengan nilai, etika, dan kebutuhan sosial.

Sinergi ini membentuk paradigma baru dalam pengambilan keputusan berbasis data lebih cepat, adaptif, dan manusiawi.

Keberhasilan masa depan Big Data tidak hanya bergantung pada teknologi, tetapi juga pada kemampuan manusia untuk berkolaborasi, menafsirkan, dan memvisualisasikan data dengan tanggung jawab serta integritas. Visualisasi bukan sekadar alat estetis, interpretasi bukan sekadar proses logis, dan kolaborasi bukan hanya kerja Bersama ketiganya adalah fondasi bagi budaya ilmiah yang mengedepankan transparansi, objektivitas, dan inovasi.

Dengan demikian, visualisasi, interpretasi, dan kolaborasi merupakan elemen kunci dalam membangun masa depan yang berbasis pengetahuan, bukti empiris, dan pemahaman mendalam terhadap data. Organisasi yang mampu mengintegrasikan ketiganya akan lebih siap menghadapi tantangan era digital, sekaligus mampu menciptakan solusi yang bermakna bagi masyarakat dan dunia yang semakin digerakkan oleh data.

DAFTAR PUSTAKA

Dougherty, J. (2021). *Hands-On Data Visualization Interactive Storytelling from Spreadsheets to Code*.

Fawcett, T. (2013). *Data Science for Business*. https://www.researchgate.net/publication/256438799

Härdle, W., & Unwin, A. (2018). *Chun-houh Chen Handbook of Data Visualization*. https://doi.org/10.1007/978-3-540-33037-0

Healy, K. (2019). *Kieran-Healy-Data-Visualization_-A-Practical-Introduction-Princeton-University-Press*.

Stephenson, D. (2018). *Big Data Demystified: How to Use Big Data, Data Science and AI to Make Better Business Decisions and Gain Competitive Advantage (David Stephenson, 2018) — Fokus pada Big Data dari perspektif bisnis, pengambilan keputusan, competitive advantage*.

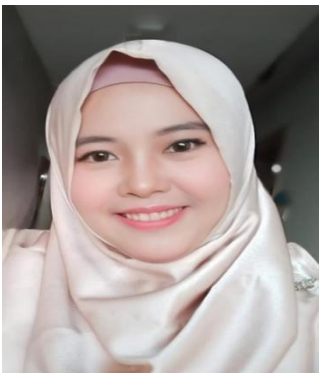
BIODATA PENULIS

Rudiansyah, S.Kom., M.Kom

Dosen Program Studi D4 Manajemen Informasi Kesehatan
Fakultas Kesehatan dan Teknologi Universitas Aisyiyah
Palembang

Penulis lahir di Palembang tanggal 20 November 1979. Penulis adalah dosen tetap pada Program Studi D4 Manajemen Informasi Kesehatan Universitas Aisyiyah Palembang. Menyelesaikan pendidikan S1 pada Jurusan Sistem Informasi Universitas Bina Darma dan melanjutkan S2 pada Jurusan Teknik Informatika Universitas Bina Darma. Penulis menekuni bidang Menulis.

BIODATA PENULIS



Ayu Meida, S.SI., M.Kom.

Dosen Program Studi Teknologi Informasi
Universitas 'Aisyiyah Palembang

Penulis lahir di Palembang pada tanggal 25 Mei 1989. Penulis adalah dosen tetap pada Program Studi Teknologi Informasi, Universitas 'Aisyiyah Palembang. Penulis menyelesaikan pendidikan D3 pada Jurusan Manajemen Informatika tahun 2006-2009 di Politeknik Negeri Sriwijaya, melanjutkan Pendidikan S1 pada Jurusan Sistem Informasi tahun 2009-2011 di Universitas Sriwijaya, melanjutkan S2 pada Program Studi Magister Ilmu Komputer tahun 2017-2019 di Universitas Sriwijaya. Saat ini penulis sedang melanjutkan Pendidikan S3 pada Program Studi Doktor Ilmu Komputer di Universitas Sriwijaya.

BIODATA PENULIS

Nyimas Sriwihajriyah, S.SI., M.Kom.
Dosen Program Studi Teknologi Informasi
Universitas 'Aisyiyah Palembang

Penulis lahir di Bengkulu pada tanggal 25 Agustus 1987. Penulis adalah dosen tetap pada Program Studi Teknologi Informasi, Universitas 'Aisyiyah Palembang. Penulis menyelesaikan pendidikan D3 pada Jurusan Teknik Komputer tahun 2005-2008 di Politeknik Negeri Sriwijaya, melanjutkan Pendidikan S1 pada Jurusan Sistem Informasi tahun 2009-2011 di Universitas Sriwijaya, melanjutkan S2 pada Program Studi Magister Teknik Informatika tahun 2012-2014 di Universitas Bina Dharma.

BIODATA PENULIS

Pita Rosemari, S.SI., M.Kom.
Dosen Program Studi Teknologi Informasi
Universitas 'Aisyiyah Palembang

Penulis lahir di Muara Penimbung pada tanggal 31 Agustus 1987. Penulis adalah dosen tetap pada Program Studi Teknologi Informasi, Universitas 'Aisyiyah Palembang. Penulis menyelesaikan pendidikan S1 pada Jurusan Sistem Informasi tahun 2006-2011 di Fakultas Ilmu Komputer Universitas Sriwijaya, melanjutkan Pendidikan S2 pada Program Studi Magister Ilmu Komputer tahun 2022-2024 di Universitas Sriwijaya.

BIODATA PENULIS

Theo Vhaldino, S.AP., S.Kom., M.Kom

Dosen Program Studi Teknologi Informasi
Fakultas Kesehatan dan Teknologi Universitas Aisyiyah Palembang

Penulis lahir di Palembang tanggal 24 Desember 1996. Penulis adalah dosen tetap pada Program Studi Teknologi Informasi Fakultas Kesehatan dan Teknologi Universitas Aisyiyah Palembang. Menyelesaikan pendidikan S1 pada Jurusan Ilmu Administrasi Negara dan Teknik Informatika dan melanjutkan S2 pada Jurusan Magister Teknik Informatika. Penulis menekuni bidang Sistem Informasi menyangkut tentang IT Governance dan aktif dalam kegiatan penelitian terkait pengampu mata kuliah yang dibidangi.

BIODATA PENULIS

Arpa Pauziah, Am.Keb.,S.Kom.,M.Kom.

Dosen Program Studi Manajemen Informasi Kesehatan
Universitas 'Aisyiyah Palembang

Penulis lahir di Palembang tanggal 26 November 1993. Saat ini Penulis menjadi Dosen tetap pada Program Studi Manajemen Informasi Kesehatan, Universitas 'Aisyiyah Palembang. Menyelesaikan D3 pada Jurusan Kebidanan, kemudian pendidikan S1 pada Jurusan Sistem Informasi dan melanjutkan S2 pada Jurusan Teknik Informatika. Penulis menekuni bidang pendidikan, teknologi informasi, dan Kesehatan. Ketertarikan terhadap dunia informatika mendorong saya untuk terus menggali konsep-konsep logika, sistem digital, serta penerapan teknologi dalam kehidupan sehari-hari.

Sebagai pendidik sekaligus peneliti, Arpa Pauziah aktif dalam kegiatan akademik dan pengembangan materi pembelajaran yang berorientasi pada penerapan logika dan teknologi secara praktis.

BIODATA PENULIS

Fitri Yani, S.SI., M.Kom.

Dosen Program Studi Teknologi Informasi
Universitas 'Aisyiyah Palembang

Penulis lahir di Medan pada tanggal 29 Mei 1988. Penulis adalah dosen tetap pada Program Studi Teknologi Informasi, Universitas 'Aisyiyah Palembang. Penulis menyelesaikan pendidikan D3 pada Jurusan Manajemen Informatika tahun 2006-2009 di Politeknik Negeri Sriwijaya, Kota Palembang. Melanjutkan Pendidikan S1 pada Jurusan Sistem Informasi tahun 2010-2014 di Universitas Sriwijaya Kota Palembang. Kemudian melanjutkan S2 pada Program Studi Magister Sistem Informasi tahun 2018-2021 di Universitas Diponegoro, Kota Semarang. Sampai saat ini penulis masih menjadi Dosen Tetap pada Program Studi Teknologi Informasi di Universitas 'Aisyiyah Palembang.

BIODATA PENULIS

Arif Fadillah, S.Kom., M.Kom

Dosen Program Studi Sistem Informasi
Fakultas Humaniora dan Informasi Universitas
Muhammadiyah Ahmad Dahlan Palembang

Penulis lahir di Palembang, pada tanggal 05 Maret 1983. Saat ini, penulis merupakan dosen tetap pada Program Studi Sistem Informasi, Fakultas Humaniora dan Informasi, Universitas Muhammadiyah Ahmad Dahlan Palembang.

Penulis menempuh pendidikan Sarjana (S1) pada Jurusan Sistem Informasi, kemudian melanjutkan studi Magister (S2) pada Jurusan Teknik Informatika.

Selain mengajar, penulis aktif menekuni bidang penulisan ilmiah dan akademik, khususnya dalam topik teknologi informasi, data science, dan sistem berbasis kecerdasan buatan. Melalui karya tulis dan penelitian yang dihasilkan, penulis berkomitmen untuk berkontribusi dalam pengembangan ilmu pengetahuan dan peningkatan kualitas pendidikan di bidang teknologi informasi di Indonesia.