BUKU AJAR

ILMU DATA

Arbi Haza Nasution
Winda Monika

PRESS

BUKU AJAR

ILMU DATA

Arbi Haza Nasution
Winda Monika

PRESS

Ilmu Data

Penulis:
Arbi Haza Nasution
Winda Monika

ISBN: 978-623-8687-36-7

Editor:
Arbi Haza Nasution

Penyunting:
Arbi Haza Nasution

Desain Sampul dan Tata Letak:
Arbi Haza Nasution

15,5 x 23 cm
Jumlah halaman: xv, 297 halaman

Penerbit:
UIR Press
Gedung Rektorat Lantai 3 Universitas Islam Riau (UIR)
Jalan Kaharuddin Nasution No. 113 Perhentian Marpoyan, Pekanbaru 28285
E-Mail : uirpress@uir.ac.id
Website : https://uirpress.uir.ac.id

Anggota IKAPI Riau
015 / Anggota Luar Biasa / RAU / 2022

Prakata

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas selesainya penyusunan buku ajar ini yang berjudul "Ilmu Data". Buku ini disusun sebagai bagian dari pengembangan materi pengajaran pada Program Studi Teknik Informatika jenjang Sarjana (S1), dan diharapkan dapat menjadi referensi utama yang komprehensif dan aplikatif dalam memahami konsep serta praktik Ilmu Data.

Ilmu Data kini menjadi salah satu disiplin ilmu yang paling berkembang pesat di era digital, di mana data telah menjadi aset utama dalam pengambilan keputusan di berbagai sektor. Oleh karena itu, kebutuhan akan pemahaman menyeluruh tentang pengolahan data – mulai dari akuisisi, pembersihan, eksplorasi, pemodelan hingga penyajian hasil – menjadi sangat penting bagi mahasiswa maupun praktisi teknologi informasi.

Buku ini dirancang dengan struktur sistematis, dilengkapi penjelasan teoretis, studi kasus nyata, kode program menggunakan bahasa Python, serta latihan pada akhir setiap bab. Pendekatan ini bertujuan agar pembaca tidak hanya memahami teori, tetapi juga memiliki kompetensi praktis dalam mengolah dan menganalisis data secara end-to-end.

Penulis menyadari bahwa penyusunan buku ajar ini masih memiliki keterbatasan, baik dari sisi kedalaman bahasan maupun ragam contoh kasus. Oleh karena itu, kritik dan saran yang membangun dari para pembaca, pengajar, dan mahasiswa sangat diharapkan untuk penyempurnaan edisi

selanjutnya.

Akhir kata, penulis menyampaikan terima kasih kepada semua pihak yang telah memberikan dukungan moral, akademik, dan teknis dalam proses penyusunan buku ini. Semoga buku ini dapat memberikan manfaat nyata bagi pembaca, khususnya mahasiswa yang ingin mengembangkan diri dalam bidang Ilmu Data.

Pekanbaru, Mei 2025

Assoc. Prof. Dr. Arbi Haza Nasution, M.IT
Winda Monika, M.Sc

Daftar Isi

1 PENDAHULUAN ILMU DATA

Tujuan Pembelajaran: Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Menjelaskan definisi dan ruang lingkup Ilmu Data.
2. Menelusuri sejarah dan evolusi Ilmu Data sebagai disiplin modern.
3. Mengidentifikasi hubungan antara Ilmu Data dengan bidang ilmu lainnya.
4. Mengenali berbagai profesi dan peran dalam ekosistem Ilmu Data.
5. Memahami aplikasi nyata dari Ilmu Data di berbagai sektor industri.

1.1 Apa Itu Ilmu Data?

Ilmu Data (Data Science) adalah sebuah disiplin multidisipliner yang memadukan berbagai cabang ilmu pengetahuan, seperti ilmu komputer, statistika, matematika, dan pemahaman konteks domain tertentu (domain knowledge), dengan tujuan utama untuk mengekstraksi pengetahuan (knowledge) dan wawasan yang bermakna (insight) dari berbagai jenis data, baik yang terstruktur maupun tidak terstruktur.[1]

Seiring perkembangan zaman, jumlah data yang dihasilkan meningkat secara eksponensial. Menurut laporan IDC Data Age 2025, diperkirakan volume data global akan mencapai 175 zettabyte pada tahun 2025. Dalam kondisi ini, Ilmu Data menjadi bukan lagi sekadar alat, tetapi kebutuhan. Tanpa ilmu data, data hanyalah kumpulan angka dan simbol tak bermakna. Namun dengan ilmu data, kita bisa mengubah data menjadi strategi, inovasi, bahkan solusi atas berbagai persoalan global — mulai dari krisis iklim, pandemi, ketimpangan sosial, hingga peningkatan efisiensi bisnis.

Data yang dimaksud dalam konteks ini bisa berasal dari tabel dalam basis data, catatan transaksi, sensor dari perangkat Internet of Things (IoT), media sosial, hingga bentuk data tidak terstruktur seperti dokumen teks, gambar, rekaman suara, bahkan video. Ilmu Data tidak hanya berfokus pada pengumpulan data semata, tetapi mencakup keseluruhan siklus hidup data (data lifecycle) yang meliputi akuisisi data, pembersihan data (cleaning), eksplorasi data (exploratory data analysis), visualisasi, pemodelan (modeling), hingga proses pengambilan keputusan yang berbasis pada hasil analisis tersebut.

Menurut Provost dan Fawcett (2013) dalam buku mereka yang berjudul Data Science for Business, Ilmu Data adalah pendekatan sistematis untuk memahami dan mengekstrak informasi dari data guna menghasilkan pengetahuan yang bisa digunakan secara nyata dalam konteks bisnis maupun kebijakan [2]. Dalam buku ini dijelaskan bahwa Ilmu Data bukan hanya tentang memanipulasi data atau membangun

model statistik, tetapi lebih kepada bagaimana menggunakan data untuk menyelesaikan masalah nyata, membangun nilai, dan memperkuat strategi.

Lebih lanjut, National Institute of Standards and Technology (NIST) mendefinisikan Ilmu Data sebagai “the extraction of actionable knowledge directly from data through a process of discovery, hypothesis, and analytical modeling” [3]. Definisi ini menekankan bahwa Ilmu Data merupakan proses ilmiah yang mencakup penemuan (discovery), perumusan hipotesis, dan pemodelan analitik untuk menghasilkan pengetahuan yang dapat langsung digunakan (actionable knowledge). Dengan kata lain, ilmu data berfokus pada pemanfaatan data untuk menghasilkan pemahaman yang dapat digunakan sebagai dasar dalam pengambilan keputusan yang strategis dan berbasis bukti (evidence-based decision making).

Selain itu, Peter Bruce dan Andrew Bruce dalam buku Practical Statistics for Data Scientists (2017) menjelaskan bahwa Ilmu Data adalah ”kombinasi dari keterampilan pemrograman, keahlian statistik, dan pengetahuan domain untuk mengubah data mentah menjadi insight yang berguna” [4]. Penjelasan ini menunjukkan pentingnya kolaborasi antara kemampuan teknis dan pemahaman konteks dalam setiap analisis data.

Dalam praktiknya, Ilmu Data bukan hanya tentang menggunakan algoritma canggih atau mengolah dataset dalam jumlah besar. Ia adalah proses berpikir yang terstruktur dan mendalam untuk memahami dunia melalui lensa data. De-

ngan data, kita bisa meninjau ulang asumsi, menantang intuisi, dan membuat keputusan yang lebih adil, lebih akurat, dan lebih berdampak.

Ilmu Data juga tidak terlepas dari nilai-nilai kemanusiaan. Di balik angka dan grafik, terdapat manusia, peristiwa, dan keputusan yang memengaruhi kehidupan nyata. Oleh sebab itu, pendekatan ilmu data yang humanistik menjadi penting. Etika penggunaan data, keadilan dalam algoritma, dan transparansi dalam pemodelan merupakan aspek yang kini menjadi perhatian utama dalam penerapan ilmu data, terutama ketika data digunakan untuk mengambil keputusan dalam sektor-sektor kritis seperti kesehatan, pendidikan, hukum, dan pelayanan publik.

Ilmu Data juga bersifat inklusif. Disiplin ini memberi ruang kepada siapa pun dari berbagai latar belakang untuk berkontribusi: seorang insinyur dapat menggunakan ilmu data untuk mengoptimalkan performa mesin, seorang guru dapat memahami gaya belajar siswa dari data pembelajaran digital, bahkan seorang aktivis lingkungan dapat memanfaatkan data untuk mengadvokasi kebijakan yang berkelanjutan. Ini menunjukkan bahwa ilmu data tidak hanya milik mereka yang berlatar belakang teknologi, tetapi milik siapa saja yang ingin memahami dunia dengan cara yang lebih objektif dan berbasis bukti.

Sehingga, Ilmu Data adalah jembatan antara data dan pengambilan keputusan yang bermakna. Ia bukan hanya bidang teknis yang mengandalkan pemrograman dan algoritma, tetapi juga bidang yang mengedepankan pemikiran

kritis, pemahaman kontekstual, dan empati terhadap manusia yang menjadi subjek dari data itu sendiri. Sebagai disiplin yang terus berkembang dan berevolusi, ilmu data telah menjadi salah satu fondasi utama dalam membentuk masa depan peradaban digital yang lebih cerdas, adil, dan inklusif.

Ilmu Data mengajarkan kita untuk tidak sekadar percaya pada intuisi, tetapi juga menghargai proses analitik yang berbasis bukti. Ia mengajarkan kita bahwa setiap data memiliki cerita, dan tugas kita sebagai ilmuwan data adalah menemukan, memahami, dan menyampaikan cerita itu dengan cara yang jujur dan bermanfaat. Dalam konteks pendidikan tinggi, khususnya bagi mahasiswa sarjana, memahami dasar-dasar ilmu data bukan hanya membuka peluang karier, tetapi juga memberikan bekal berpikir logis, kritis, dan bertanggung jawab di tengah dunia yang semakin kompleks dan didorong oleh data.

1.2 Evolusi dan Sejarah Ilmu Data

Ilmu data tidak lahir sebagai suatu disiplin tunggal secara tiba-tiba. Sebaliknya, ia merupakan hasil evolusi dan akumulasi pengetahuan dari berbagai bidang keilmuan, terutama statistika, ilmu komputer, kecerdasan buatan, dan teknologi informasi. Setiap fase dalam lintasan sejarah tersebut telah berkontribusi dalam membentuk fondasi kokoh bagi kemunculan ilmu data seperti yang kita kenal saat ini.

Secara garis besar, evolusi ilmu data dapat ditelusuri me-

lalui beberapa tonggak sejarah berikut:

- **Statistika Tradisional (abad ke-18)**: Statistika mulai berkembang sebagai alat utama dalam menganalisis data populasi, ekonomi, dan eksperimen ilmiah. Pada masa ini, peran statistika sangat sentral dalam mendukung pengambilan keputusan berbasis bukti di bidang sains dan kebijakan publik.
- **Machine Learning (1950-an)**: Pendekatan baru melalui perkembangan machine learning sebuah cabang dari kecerdasan buatan (AI) memungkinkan komputer belajar dari data. Teknologi ini membuka jalan bagi sistem prediktif dan otomatisasi dalam berbagai bidang.
- **Data Mining (1990-an)**: Data mining berkembang pesat seiring meluasnya penggunaan komputer personal dan basis data relasional. Para ilmuwan dan praktisi mulai menggali informasi tersembunyi dari kumpulan data besar untuk menemukan pola dan wawasan berharga.
- **Big Data (2005-an)**: Pertumbuhan eksponensial dalam volume, kecepatan, dan keragaman data menuntut pendekatan baru. Teknologi seperti Hadoop dan Spark dikembangkan untuk menangani serta memproses data berskala masif secara efisien.
- **Deep Learning (2010-an)**: Pembelajaran mendalam muncul sebagai pendekatan revolusioner dalam pengolahan data kompleks seperti gambar, suara, dan bahasa.

Dengan jaringan saraf dalam (deep neural networks), model ini meniru cara kerja otak manusia dalam mengenali pola.

- **Large Language Models (2022–sekarang)**: Model bahasa besar (LLM) seperti ChatGPT mengubah cara manusia berinteraksi dengan teknologi melalui kemampuan pemrosesan bahasa alami yang sangat canggih. Didukung oleh arsitektur Transformer, LLM kini digunakan luas dalam asisten virtual, penulisan otomatis, hingga platform edukasi dengan tingkat adopsi yang sangat cepat secara global.

1.3 Ruang Lingkup dan Elemen Inti Ilmu Data

Secara umum, ruang lingkup ilmu data dapat dirangkum ke dalam lima elemen inti berikut:

1. **Data Collection (Pengumpulan Data)**: Tahap awal ini mencakup proses memperoleh data dari berbagai sumber. "The data science process involves collecting data, preparing it for analysis, building predictive models, and integrating those models into decision-making systems." [2]. Teknik yang digunakan dapat berupa web crawling, web scraping, pengambilan data dari API, sensor-based acquisition, atau ekstraksi dari basis data.
2. **Data Preparation (Pembersihan & Transformasi)**: Merupakan teknik yang digunakan dalam membersihkan

dan mentransformasikan data [5].Proses ini melibatkan penanganan data hilang (missing values), penghapusan atau penyesuaian outlier, serta konversi data ke dalam format yang sesuai untuk analisis.

3. **Exploratory Data Analysis (Data dieksplorasi)**: untuk memahami struktur, pola, hubungan antarvariabel, dan tren yang mungkin tersembunyi. Visualisasi data sering digunakan untuk membantu proses eksplorasi ini, misalnya melalui grafik sebar, histogram, heatmap, dan lain sebagainya.

4. **Modeling**: Data digunakan untuk membangun model matematis atau statistik yang mampu membuat prediksi atau mengelompokkan data. Teknik yang digunakan meliputi supervised learning (seperti regresi dan klasifikasi) dan unsupervised learning (seperti klasterisasi dan reduksi dimensi) [6]. Pemilihan model bergantung pada jenis data dan tujuan analisis.

5. **Deployment & Evaluation**: Model yang telah dibangun kemudian diterapkan dalam lingkungan nyata, seperti aplikasi bisnis, sistem rekomendasi, atau perangkat lunak berbasis AI. Evaluasi dilakukan secara berkala untuk mengukur performa model menggunakan metrik tertentu.

1.4 Hubungan Ilmu Data dengan Disiplin Ilmu Lain

Ilmu Data adalah disiplin lintas bidang. Berikut ini hubungannya dengan bidang-bidang utama:

- **Statistika**: Statistika memberikan dasar konseptual dan metodologis dalam analisis data. Konsep-konsep seperti distribusi probabilitas, inferensi statistik, pengujian hipotesis, dan teknik estimasi menjadi pilar utama dalam proses analitik dan pengambilan keputusan berbasis data.
- **Ilmu Komputer**: Ilmu Komputer menyediakan kerangka teknis dan operasional yang mencakup algoritma, struktur data, pemrograman, manajemen basis data, serta pengembangan sistem informasi. Kemampuan komputasional yang ditawarkan oleh disiplin ini sangat penting dalam pengolahan data skala besar, otomatisasi proses analitik, serta implementasi model prediktif dalam sistem nyata.
- **Matematika**: Matematika, khususnya aljabar linier, kalkulus diferensial dan integral, serta teori probabilitas, menjadi fondasi utama bagi pengembangan berbagai model matematis dalam Ilmu Data. Pemahaman terhadap prinsip-prinsip matematis memungkinkan perancangan model yang tidak hanya akurat, tetapi juga stabil dan dapat diinterpretasikan secara ilmiah.

- **Kecerdasan Buatan**: Ilmu Data sangat erat kaitannya dengan Kecerdasan Buatan, terutama dalam penerapan algoritma pembelajaran mesin (machine learning) dan pembelajaran mendalam (deep learning). Teknik-teknik tersebut memungkinkan sistem untuk melakukan prediksi, klasifikasi, serta pengambilan keputusan berbasis data historis dan pola yang tersembunyi dalam data.
- **Etika & Hukum**: Dalam praktik Ilmu Data, dimensi etika dan hukum memiliki peranan yang sangat penting. Pengelolaan data harus memperhatikan aspek privasi, keamanan, dan hak-hak individu sebagaimana diatur dalam berbagai regulasi, seperti General Data Protection Regulation (GDPR) di Eropa dan Undang-Undang Informasi dan Transaksi Elektronik (UU ITE) di Indonesia. Kepatuhan terhadap prinsip etika dan regulasi hukum menjadi syarat mutlak dalam penerapan Ilmu Data yang bertanggung jawab dan berkelanjutan.

1.5 Peran Profesi dalam Ekosistem Ilmu Data

Di dunia kerja, Ilmu Data melahirkan banyak profesi dengan tanggung jawab berbeda:

Profesi	Tanggung Jawab Utama
Data Scientist	Membangun model prediksi berbasis data dan analisis statistik
Data Analyst	Membuat laporan dan visualisasi untuk pengambilan keputusan
Data Engineer	Menyusun pipeline data, ETL, dan arsitektur penyimpanan
Machine Learning Eng.	Mengembangkan dan menguji algoritma pembelajaran mesin
Business Intelligence	Menyusun dashboard interaktif dan menganalisis KPI bisnis

Tabel 1: Profesi terlahir dari Ilmu Data

1.6 Studi Kasus Aplikasi Ilmu Data

Kasus 1: Google Search dan Rekomendasi Otomatis Google memanfaatkan ilmu data untuk mempersonalisasi hasil pencarian. Algoritma RankBrain menggunakan pembelajaran mesin untuk menyesuaikan hasil pencarian sesuai perilaku pengguna.

Kasus 2: Netflix dan Rekomendasi Film Netflix menggabungkan collaborative filtering dan content-based filtering untuk memberikan rekomendasi yang sangat personal. Sistem ini dibangun berdasarkan miliaran data penayangan dan preferensi pengguna.

Kasus 3: Smart City dan Data Sensor Kota pintar seperti Barcelona menggunakan sensor IoT untuk mengelola lalu lintas, air, dan limbah secara efisien. Data dikumpulkan secara real-time dan dianalisis untuk mendukung kebijakan pemerintah kota.

1.7 Tantangan dalam Ilmu Data

Beberapa tantangan utama dalam bidang ini meliputi:

- **Data Berkualitas Rendah**: Salah satu tantangan paling mendasar dalam Ilmu Data adalah rendahnya kualitas data yang tersedia. Permasalahan seperti nilai yang hilang (missing values), kesalahan input, data duplikat, dan noise dapat secara signifikan memengaruhi akurasi dan validitas hasil analisis. Oleh karena itu, tahap pembersihan dan validasi data menjadi sangat krusial sebelum dilakukan pemodelan.
- **Volume dan Kecepatan Data**: Dengan munculnya era Big Data, data yang dihasilkan oleh berbagai sumber (sensor, media sosial, transaksi digital, dan sebagainya) memiliki volume dan kecepatan yang sangat tinggi. Tantangan ini memerlukan solusi teknologi yang skalabel dan efisien, seperti penggunaan sistem pemrosesan terdistribusi (misalnya Hadoop dan Apache Spark) untuk memastikan bahwa analisis dapat dilakukan secara real-time atau near realtime.
- **Interpretabilitas Model**: Model-model canggih seperti deep learning sering kali memberikan hasil yang akurat

namun sulit untuk dijelaskan secara intuitif. Hal ini menimbulkan tantangan interpretabilitas, terutama dalam konteks pengambilan keputusan yang membutuhkan transparansi dan akuntabilitas, seperti dalam bidang kesehatan, hukum, dan keuangan.

- **Keamanan dan Privasi Data**: Dalam pengelolaan data, terutama yang berkaitan dengan informasi pribadi atau sensitif seperti data medis, finansial, dan sosial perlindungan terhadap privasi individu menjadi tantangan yang sangat penting.

1.8 Alur Kerja Proyek Ilmu Data (Workflow)

1. **Define Problem**
    - Menetapkan tujuan proyek, pertanyaan penelitian, dan kriteria kesuksesan.
2. **Collect Data**
    - Mengumpulkan data dari berbagai sumber yang relevan, baik internal maupun eksternal.
3. **Clean and Explore Data**
    - Membersihkan data dari missing values, outliers, dan kesalahan.
    - Melakukan eksplorasi awal untuk memahami pola dan distribusi data.

4. **Modeling**

    - Mengembangkan model statistik atau machine learning yang sesuai dengan masalah yang ditetapkan.

5. **Evaluate**

    - Mengevaluasi performa model menggunakan metrik yang tepat.
    - Melakukan iterasi dan perbaikan model untuk mencapai hasil optimal.

6. **Deploy and Monitor**

    - Menerapkan model ke lingkungan produksi.
    - Memantau performa model secara berkala dan melakukan penyesuaian jika diperlukan.

1.9 LATIHAN / TUGAS AKHIR

1. **[Uraian]** Jelaskan siklus hidup proyek Ilmu Data dan contoh penerapannya di sektor transportasi!

2. **[Uraian Visual]** Gambar dan jelaskan hubungan antara Ilmu Data, Statistik, Machine Learning, dan Big Data dalam satu diagram Venn.

3. **[Coding]**

   Buatlah skrip Python menggunakan library `pandas` untuk:

   - Membaca file CSV
   - Menampilkan 5 baris pertama
   - Menghitung statistik deskriptif kolom numerik

4. **[Studi Kasus]** Teliti dan analisis sistem rekomendasi yang digunakan oleh TikTok. Jelaskan bagaimana Ilmu Data berperan dalam memberikan konten yang sesuai dengan preferensi pengguna.

5. **[Studi Kasus]** Temukan berita atau jurnal yang menunjukkan kasus penyalahgunaan data pribadi oleh perusahaan teknologi. Buatlah ringkasan dan refleksi etis terhadap kasus tersebut.

Daftar Pustaka

[1] A. H. Nasution, Y. Murakami **and** T. Ishida, "A generalized constraint approach to bilingual dictionary induction for low-resource language families," *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, **jourvol** 17, **number** 2, **pages** 1–29, 2017.

[2] F. Provost **and** T. Fawcett, *Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking*. Sebastopol, CA: O'Reilly Media, 2013, ISBN: 978-1-4493-6132-7.

[3] National Institute of Standards and Technology (NIST), "NIST Big Data Interoperability Framework: Volume 1, Definitions," U.S. Department of Commerce, techreport NIST SP 1500-1, 2015, Special Publication. **url**: `https://doi.org/10.6028/NIST.SP.1500-1`.

[4] P. Bruce **and** A. Bruce, *Practical Statistics for Data Scientists: 50 Essential Concepts*. Sebastopol, CA: O'Reilly Media, 2017, ISBN: 978-1-4919-0649-3.

[5] C. Zhang **and** Y. Zheng, "A Survey on Data Preparation Techniques for Data Mining," *ACM Computing Surveys*, **jourvol** 52, **number** 1, **pages** 1–38, 2020. DOI: 10.1145/3297753.

[6] J. Han, M. Kamber **and** J. Pei, *Data Mining: Concepts and Techniques*, 3 **edition**. Morgan Kaufmann, 2011.

2 JENIS DATA DAN REPRESENTASINYA

2.1 Tujuan Pembelajaran

Setelah menyelesaikan bab ini, mahasiswa diharapkan mampu:

1. Mengidentifikasi berbagai jenis data berdasarkan format dan struktur.
2. Membedakan antara data terstruktur, semi-terstruktur, dan tidak terstruktur.
3. Menggunakan Python untuk merepresentasikan dan mengelola tipe-tipe data umum.
4. Mengilustrasikan hubungan antar data dalam bentuk tabel dan grafik.
5. Memahami peran representasi data dalam proses analitik.

2.2 Pengertian Data

Data merupakan sekumpulan fakta, angka, simbol, atau hasil observasi yang belum diolah dan tidak memiliki makna tertentu hingga dilakukan proses interpretasi atau analisis. Sebagaimana yang dikemukakan oleh Provost dan Fawcett

(2013) ”Data are facts and numbers that are collected, measured, and analyzed to produce information and knowledge”. Data menjadi bahan dasar untuk menghasilkan informasi dan pengetahuan yang berguna bagi pengambilan keputusan. Selain itu, Data merupakan fondasi utama dalam proses analisis ilmiah, pengembangan model, dan pengambilan kebijakan berbasis bukti.

2.3 Tipe-Tipe Data

Dalam konteks Ilmu Data, pemahaman mengenai tipe-tipe data merupakan aspek fundamental yang menentukan metode analisis statistik, pemilihan algoritma pembelajaran mesin, serta teknik visualisasi yang sesuai. Tipe data merujuk pada klasifikasi variabel berdasarkan sifat dan karakteristiknya, yang secara umum dapat dibagi menjadi beberapa kategori utama [1], [2] sebagai berikut:

- **Data Numerik (Numerical Data)** : Data numerik, atau data kuantitatif, adalah data yang dinyatakan dalam bentuk angka dan memungkinkan dilakukan operasi matematis. Data ini dibagi menjadi dua subkategori:
  - *Data Diskret (Discrete Data)*: Merupakan data yang memiliki nilai-nilai terbatas dan dapat dihitung, seperti jumlah anak dalam keluarga atau jumlah kendaraan yang melintas di suatu jalan.
  - *Data Kontinu (Continuous Data)*: Merupakan data yang dapat mengambil nilai dalam rentang tertentu dan dapat diukur dengan tingkat presisi ter-

tentu, seperti tinggi badan, berat badan, atau suhu udara.

- **Data Kategorikal (Categorical Data)**: Data kategorikal, atau data kualitatif, adalah data yang mengelompokkan individu atau objek ke dalam kategori atau kelompok yang berbeda. Data ini tidak memiliki nilai numerik yang bermakna secara matematis dan dibagi menjadi dua jenis:
  - *Data Nominal (Nominal Data)*: Kategori yang tidak memiliki urutan atau peringkat intrinsik. Contohnya termasuk jenis kelamin, warna mata, atau jenis kendaraan.
  - *Data Ordinal (Ordinal Data)*: Kategori yang memiliki urutan atau peringkat tertentu, namun jarak antar kategori tidak dapat diukur secara pasti. Contohnya termasuk tingkat pendidikan (SD, SMP, SMA, Sarjana) atau tingkat kepuasan pelanggan (tidak puas, cukup puas, sangat puas).
- **Data Deret Waktu (Time Series Data)**: Data deret waktu adalah data yang dikumpulkan atau dicatat pada titik waktu tertentu dan dianalisis untuk mengidentifikasi pola atau tren seiring waktu. Contoh data deret waktu meliputi harga saham harian, suhu udara per jam, atau volume penjualan bulanan.

2.4 Struktur Data

Dalam konteks ilmu data dan manajemen informasi, struktur data dapat dikategorikan ke dalam tiga jenis utama berdasarkan tingkat keterstrukturannya [3], [4]:

2.4.1 Structured Data

Structured Data adalah data yang disimpan dalam format yang terorganisir dan mudah diproses oleh sistem komputer. Sebagaimana yang dikemukakan oleh Kitchin (2014) "Structured data are organized in a clear data model, typically tabular, and are easily searchable by simple, straightforward algorithms." [5]. Umumnya, data jenis ini disimpan dalam bentuk tabel dengan baris dan kolom yang jelas, seperti pada basis data relasional.

Contoh Tabel:

ID	Nama	Usia	Kota
01	Fajar	22	Surabaya
02	Anita	20	Medan

Tabel 2: Contoh data terstruktur

2.4.2 Semi-Structured Data

Semi-Structured Data tidak tersimpan dalam bentuk tabel secara kaku, namun masih memiliki tag atau metadata yang memungkinkan proses pengorganisasian dan interpretasi data. Contoh umum dari data jenis ini adalah format XML dan

JSON.

Contoh: XML, JSON.

```
{
  "nama": "Fajar",
  "usia": 22,
  "kota": "Surabaya"
}
```

2.4.3 Unstructured Data

Unstructured Data adalah data yang tidak memiliki format atau struktur yang terdefinisi dengan baik. Data ini sulit diproses secara langsung oleh sistem karena tidak memiliki skema yang jelas.

Contoh: video, gambar, dan dokumen PDF dan email.

2.5 Representasi Data dalam Python

Python menyediakan tipe data dan struktur data yang fleksibel untuk berbagai keperluan analisis dan pemrosesan data. Bagian ini menjelaskan representasi data sederhana hingga terstruktur menggunakan pustaka populer seperti `pandas`.

2.5.1 Tipe Data Dasar

Python memiliki tipe data dasar seperti bilangan bulat, bilangan desimal, teks, dan logika boolean. Berikut contoh penggunaannya:

```
angka = 10                # tipe data integer
harga = 2500.50           # tipe data float
nama = "Arbi"             # tipe data string
aktif = True              # tipe data boolean
```

2.5.2 Struktur Data Python untuk Ilmu Data

Untuk menangani data dalam format tabel, Python menyediakan pustaka `pandas`. Struktur data utama dari pustaka ini adalah `DataFrame`, yang menyerupai tabel relasional. *Python Syntax:*

```
import pandas as pd

data = {
    "Nama": ["Arbi", "Dina", "Rizki"],
    "Usia": [21, 23, 22],
    "Kota": ["Medan", "Jakarta", "Bandung"]
}
df = pd.DataFrame(data)
print(df)
```

Output:

```
    Nama  Usia     Kota
0   Arbi    21    Medan
1   Dina    23  Jakarta
2  Rizki    22  Bandung
```

2.6 Visualisasi Representasi Data

Visualisasi data adalah proses menampilkan informasi dalam bentuk grafis untuk membantu pemahaman, identifikasi pola, dan komunikasi hasil analisis. Teknik ini merupakan bagian penting dalam proses analisis data [6]–[8].

Beberapa jenis visualisasi data deskriptif yang umum digunakan antara lain:

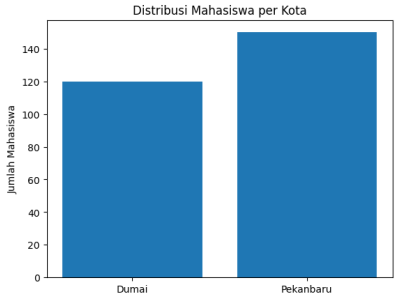
- **Bar Chart** — digunakan untuk membandingkan nilai antar kategori diskret.
- **Histogram** — menggambarkan distribusi frekuensi dari data numerik kontinu.
- **Pie Chart** — menunjukkan proporsi kategori terhadap total keseluruhan.

Contoh berikut menampilkan grafik batang (*bar chart*) dari jumlah mahasiswa berdasarkan kota asal:

Python Syntax:

```
import matplotlib.pyplot as plt

kota = ['Dumai', 'Pekanbaru']
jumlah = [120, 150]

plt.bar(kota, jumlah)
plt.title("Distribusi Mahasiswa per Kota")
plt.ylabel("Jumlah Mahasiswa")
plt.show()
```

Output:



Gambar 1: Distribusi jumlah mahasiswa berdasarkan kota asal

2.7 Studi Kasus: Analisis Data Demografis Mahasiswa

Kasus: Universitas ingin mengetahui distribusi asal mahasiswa berdasarkan provinsi.

Langkah Analisis:

1. Kumpulkan data dari bagian akademik.
2. Bersihkan duplikasi dan data kosong.
3. Representasikan data dalam bentuk tabel.
4. Visualisasikan data dengan bar chart atau pie chart.
5. Interpretasikan hasil untuk pengambilan keputusan promosi kampus.

2.8 LATIHAN / TUGAS AKHIR

1. **[Uraian]** Jelaskan perbedaan antara data ordinal dan data kategorikal disertai contoh masing-masing!

2. **[Uraian Visual]** Buatlah ilustrasi yang menunjukkan hirarki struktur data: structured → semi-structured → unstructured.

3. **[Coding]**

   Gunakan `pandas` untuk membaca file Excel berisi data mahasiswa dan tampilkan ringkasan statistik per kolom.

4. **[Studi Kasus]** Ambil 100 data publik dari open-data pemerintah (BPS, Kaggle, dsb) dan klasifikasikan strukturnya.

5. **[Studi Kasus]** Berikan contoh analisis sederhana untuk menentukan kota asal mahasiswa terbanyak di kelas Anda.

Daftar Pustaka

[1] P. Bruce **and** A. Bruce, *Practical Statistics for Data Scientists: 50 Essential Concepts*. Sebastopol, CA: O'Reilly Media, 2017, ISBN: 978-1-4919-0649-3.

[2] J. Han, J. Pei **and** M. Kamber, *Data Mining: Concepts and Techniques*, 3rd. Amsterdam: Elsevier, 2011, ISBN: 9780123814791

[3] S. Abiteboul, P. Buneman **and** D. Suciu, *Data on the Web: From Relations to Semistructured Data and XML*. Morgan Kaufmann, 2000, ISBN: 9781558606227.

[4] A. Gandomi **and** M. Haider, "Beyond the hype: Big data concepts, methods, and analytics," *International Journal of Information Management*, **journvol** 35, **number** 2, **pages** 137–144, 2015. DOI: 10.1016/j.ijinfomgt.2014.10.007.

[5] R. Kitchin, *The Data Revolution: Big Data, Open Data, Data Infrastructures and Their Consequences*. Sage Publications, 2014, ISBN: 9781446287484.

[6] J. S. Grosman, P. H. Furtado, A. M. Rodrigues, G. G. Schardong, S. D. Barbosa **and** H. C. Lopes, "Eras: Improving the quality control in the annotation process for natural language processing tasks," *Information Systems*, **journvol** 93, **page** 101 553, 2020.

[7] F. Gilardi, M. Alizadeh **and** M. Kubli, "Chatgpt outperforms crowd-workers for text-annotation tasks," *arXiv preprint arXiv:2303.15056*, 2023.

[8] Z. Liu, K. Yang, Q. Xie, T. Zhang **and** S. Ananiadou, "Emo-LLMs: A Series of Emotional Large Language Models and Annotation Tools for Comprehensive Affective Analysis," **in***Proceedings of ACM SIGKDD* 2024, **pages** 5487–5496. DOI: 10.1145/3637528.3671552.

3 PENGUMPULAN DAN AKUISISI DATA

3.1 Tujuan Pembelajaran

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Menjelaskan berbagai sumber data yang dapat digunakan dalam Ilmu Data.
2. Menggunakan metode scraping dan API untuk memperoleh data dari web.
3. Mengolah berbagai format penyimpanan data: CSV, JSON, dan XML.
4. Menerapkan teknik dasar pengumpulan data menggunakan Python.
5. Menyadari pentingnya etika dan legalitas dalam akuisisi data.

3.2 Pengantar Pengumpulan Data

Pengumpulan data (*data acquisition*) merupakan tahap awal yang fundamental dalam siklus Ilmu Data. Tahap ini sangat

menentukan kualitas, kelengkapan, dan relevansi hasil analisis yang diperoleh. Dalam era Big Data, pengumpulan data tidak hanya melibatkan jumlah data yang besar, tetapi juga kecepatan dan kompleksitas format data yang beragam [1]–[3]. Perubahan ini menjadikan pengumpulan data sebagai proses yang dinamis dan menantang, memerlukan infrastruktur teknologi serta pendekatan metodologis yang tepat.

3.3 Sumber-Sumber Data

Dalam proyek Ilmu Data, data dapat diperoleh dari berbagai sumber tergantung pada konteks dan tujuan analisis. Berikut adalah beberapa sumber data yang umum digunakan:

1. **Open Data**
   Merupakan data yang tersedia secara publik dan dapat diakses secara bebas.
   - *Contoh:* Kaggle, Data.go.id, Badan Pusat Statistik (BPS), World Bank.
   - Umumnya tersedia dalam format seperti CSV, XLSX, atau JSON.
2. **Web Data**
   Data yang berasal dari situs web umumnya diperoleh melalui teknik *web scraping*, yaitu proses ekstraksi informasi secara otomatis dari halaman web. Proses ini dapat dilakukan dengan membaca struktur halaman (HTML) atau memanfaatkan layanan akses data yang disediakan oleh situs terkait.

- Umum digunakan untuk mengumpulkan data dari portal berita, marketplace, atau media sosial.

3. **Sensor Data (Internet of Things)**
   Data yang dikumpulkan melalui perangkat sensor dalam ekosistem *Internet of Things (IoT)*. Data ini umumnya bersifat *real-time* dan disimpan dalam format deret waktu (*time-series*).

   - *Contoh:* perangkat wearable, sensor cuaca, dan smart meter.

4. **Enterprise Systems**
   Sumber data yang berasal dari sistem internal organisasi, seperti *Customer Relationship Management (CRM)*, *Enterprise Resource Planning (ERP)*, serta basis data transaksi operasional. Data ini umumnya bersifat terstruktur dan digunakan untuk mendukung pengambilan keputusan bisnis.

5. **Survei dan Kuesioner**
   Data yang diperoleh melalui instrumen survei, baik secara daring maupun luring. Sumber ini banyak digunakan untuk mengumpulkan data primer langsung dari responden.

   - *Contoh:* Google Form, SurveyMonkey, dan platform sejenis.
   - Dapat berupa data mentah (respon individu) maupun statistik ringkasan.

3.4 Web Scraping dengan Python

Web scraping adalah teknik dalam Ilmu Data yang digunakan untuk mengekstrak informasi secara otomatis dari halaman web [4]. Teknik ini berguna ketika data tidak tersedia dalam bentuk terstruktur seperti file CSV atau API, dan sering diterapkan untuk memperoleh teks, tabel, atau metadata dari berbagai situs web.

Dua pendekatan umum yang digunakan dalam *web scraping* dengan Python adalah menggunakan `BeautifulSoup` dan `Selenium`. `BeautifulSoup` cocok untuk halaman statis, sedangkan `Selenium` diperlukan untuk menangani konten dinamis yang dimuat oleh JavaScript [5], [6].

3.4.1 Menggunakan BeautifulSoup

`BeautifulSoup` memungkinkan pengguna untuk mem-parsing dokumen HTML atau XML. Bersama dengan pustaka `requests`, teknik ini cukup untuk mengambil data dari halaman web yang tidak membutuhkan interaksi pengguna atau pemrosesan JavaScript.

Contoh: Mengambil judul berita dari halaman web

```
1 from bs4 import BeautifulSoup
2 import requests
3
4
5 url = 'https://www.yahoo.com/news/mark-zuckerberg-suggested-
```

```
    wiping-everyone-110408822.html'
headers = {
    "User-Agent": "Mozilla/5.0"
}

response = requests.get(url, headers=headers)
soup = BeautifulSoup(response.text, 'html.parser')
judul = soup.find('h1').text
print("Judul Berita:", judul)
```

Kode di atas mengambil judul artikel dari elemen `<h1>` pada halaman berita. Permintaan HTTP dikirim melalui `requests`, dan konten HTML yang diterima diproses dengan `BeautifulSoup`.

Contoh: Mengambil data historis harga Bitcoin

```
from bs4 import BeautifulSoup
import pandas as pd
import requests

url = 'https://finance.yahoo.com/quote/BTC-USD/history/?p=BTC-
    USD&period1=1743465600&period2=1743811200'

headers = {
    "User-Agent": "Mozilla/5.0"
}

response = requests.get(url, headers=headers)
soup = BeautifulSoup(response.text, 'html.parser')

table = soup.find('table')
rows = table.find_all('tr')

data = []
for row in rows[1:]:
    cols = row.find_all('td')
```

```
    if len(cols) == 7:
        data.append([col.text.strip() for col in cols])

df = pd.DataFrame(data, columns=['Date', 'Open', 'High', 'Low'
    , 'Close', 'Adj Close', 'Volume'])
df.head()
```

Contoh ini menunjukkan bagaimana data historis harga Bitcoin diambil dari halaman Yahoo Finance. Data yang diperoleh dari elemen `<table>` diubah menjadi `DataFrame` untuk keperluan analisis lebih lanjut.

3.4.2 Menggunakan Selenium

Untuk halaman web yang memuat konten secara dinamis, seperti YouTube atau media sosial, `Selenium` digunakan karena mampu mengotomatisasi peramban web secara penuh. Hal ini memungkinkan interaksi seperti menggulir halaman dan menunggu elemen dimuat secara dinamis.

Instalasi di Google Colab

Berikut adalah perintah instalasi `Selenium`, Chrome, dan ChromeDriver di lingkungan Google Colab:

```
!sudo apt update -y
!sudo apt install wget curl gdebi-core unzip -y

!wget https://dl.google.com/linux/direct/google-chrome-
    stable_current_amd64.deb
!gdebi google-chrome-stable_current_amd64.deb -y

!wget -N https://storage.googleapis.com/chrome-for-testing-
    public/130.0.6723.58/linux64/chromedriver-linux64.zip -P /
    tmp/
!unzip -o /tmp/chromedriver-linux64.zip -d /tmp/

!chmod +x /tmp/chromedriver-linux64/chromedriver
!mv /tmp/chromedriver-linux64/chromedriver /usr/local/bin/
    chromedriver

!pip install selenium webdriver_manager
```

Contoh: Mengambil komentar dari video YouTube

```
from selenium.webdriver.support import expected_conditions as
    EC
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from IPython.display import HTML, display
from selenium import webdriver
from bs4 import BeautifulSoup
from tqdm import tqdm
import pandas as pd
import base64
import time
import os

```

```
def button(file_path, button_text):
    with open(file_path, 'rb') as file:
        data = file.read()
    b64 = base64.b64encode(data).decode()

    html = '''

    '''.format(file_name=os.path.basename(file_path), b64_data
    =b64, button_text=button_text)

    return HTML(html)

def main():

    chrome_options = Options()
    chrome_options.add_argument('--disable-dev-shm-usage')
    chrome_options.add_argument('--disable-extensions')
    chrome_options.add_argument('--start-maximized')
    chrome_options.add_argument('--disable-gpu')
    chrome_options.add_argument('--no-sandbox')
    chrome_options.add_argument('--headless')

    driver = webdriver.Chrome(options=chrome_options)

    try:
        url = 'https://www.youtube.com/watch?v=KH9txRUApUM'
        print("Membuka YouTube...")
        driver.get(url)

        wait = WebDriverWait(driver, 15)
        print("Menunggu komentar muncul...")
        wait.until(EC.presence_of_element_located((By.
    CSS_SELECTOR, 'ytd-comments#comments')))

        print("Jeda 10 detik sebelum mulai scroll...\n")
        time.sleep(10)
```

```
50
51      data = []
52      print("Mulai scroll dan mengambil komentar...")
53
54      for item in tqdm(range(10), desc='Scrolling'):
55          wait.until(EC.visibility_of_element_located((By.
    TAG_NAME, "body"))).send_keys(Keys.END)
56          time.sleep(15)
57
58      time.sleep(5)
59
60      page_source = driver.page_source
61      soup = BeautifulSoup(page_source, 'html.parser')
62
63      comment_elements = soup.select('ytd-comment-thread-
    renderer')
64
65      for comment in comment_elements:
66          comment_text_tag = comment.select_one('yt-
    attributed-string#content-text')
67          username_tag = comment.select_one('a#author-text
    span')
68
69          if comment_text_tag and username_tag:
70              comment_text = comment_text_tag.get_text(strip
    =True)
71              username = username_tag.get_text(strip=True)
72              data.append({'Username': username, 'Comment':
    comment_text})
73
74      print(f"Total komentar yang diambil: {len(data)}\n")
75
76      df = pd.DataFrame(data)
77      print(df.head())
78
79      file_name = 'dataset_youtube_comment.csv'
```

```
        df.to_csv(file_name, index=False)
        print(f'\nData berhasil disimpan:\n')

        display(button(file_name, 'Download File'))

    except Exception as e:
        print(f"Terjadi kesalahan: {e}")

    finally:
        driver.quit()

if __name__ == "__main__":
    main()
```

Contoh di atas menunjukkan bagaimana `Selenium` digunakan untuk menggulir halaman video YouTube secara otomatis hingga komentar dimuat, kemudian mengambil komentar dan nama pengguna menggunakan `BeautifulSoup`. Hasil akhir disimpan dalam format CSV untuk keperluan analisis lanjutan.

Catatan Etika: *Web scraping* harus dilakukan dengan memperhatikan ketentuan hukum dan kebijakan situs web. Disarankan untuk memeriksa `robots.txt` dan *Terms of Service* sebelum melakukan *scraping*. Pengambilan data tanpa izin dapat melanggar privasi atau peraturan yang berlaku.

3.5 Penggunaan API untuk Akses Data

API (*Application Programming Interface*) adalah antarmuka yang memungkinkan perangkat lunak untuk saling berko-

munikasi dan bertukar data. Dalam konteks ilmu data, API digunakan untuk memperoleh data dari berbagai layanan daring secara otomatis dan terstruktur.

Beberapa contoh API publik yang sering digunakan dalam analisis data antara lain:

- **Twitter API** — menyediakan akses ke data media sosial seperti tweet, tren, dan informasi pengguna.
- **OpenWeatherMap API** — menyediakan data terkait kondisi cuaca terkini dan prakiraan cuaca.
- **Google Maps API** — memberikan layanan geolokasi, pemetaan, dan rute.

Contoh: *Penggunaan API dari OpenWeatherMap untuk mengambil data cuaca terkini.*

```
import requests

url = "http://api.openweathermap.org/data/2.5/weather?q=
    Jakarta&appid=YOUR_API_KEY"
res = requests.get(url)
data = res.json()
print("Cuaca Jakarta:", data['weather'][0]['description'])
```

3.6 Format Penyimpanan Data

Dalam disiplin ilmu data, format penyimpanan data memegang peranan yang krusial karena berpengaruh terhadap kemudahan dalam membaca, menulis, dan memproses data. Pemilihan format yang sesuai juga berdampak pada interoperabilitas antarsistem, serta efisiensi dalam penyimpan-

an dan pemrosesan data. Beberapa format penyimpanan data yang umum digunakan meliputi CSV, XLSX, JSON, dan XML.

3.6.1 CSV (Comma-Separated Values)

CSV merupakan salah satu format paling umum untuk menyimpan data dalam bentuk tabular. Setiap baris pada file CSV merepresentasikan satu entri (*record*), sedangkan setiap nilai dipisahkan oleh tanda koma, titik koma, atau karakter pemisah lainnya. CSV bersifat ringan, mudah dibaca oleh manusia, dan didukung oleh berbagai perangkat lunak pengolah data.

- Sesuai untuk menyimpan data tabular satu lembar (*sheet*).
- Didukung secara luas oleh perangkat lunak seperti Microsoft Excel, Google Sheets, serta berbagai bahasa pemrograman, termasuk Python.
- Tidak mendukung format sel, rumus, maupun metadata lainnya.

Membaca file CSV menggunakan pustaka `pandas`:

```
import pandas as pd
df = pd.read_csv("data_csv.csv")
df.head()
```

Menyimpan data ke dalam file CSV:

```
import pandas as pd
data = {
```

```
    "nama": ["Andi", "Budi"],
    "usia": [21, 22],
    "jurusan": ["Informatika", "Sistem Informasi"]
}
df = pd.DataFrame(data)
df.to_csv("data_csv.csv", index=False)
```

3.6.2 XLSX (Excel Spreadsheet)

XLSX adalah format file spreadsheet yang digunakan oleh Microsoft Excel. Format ini mendukung berbagai fitur seperti lembar kerja ganda (*multiple sheets*), pemformatan sel, rumus, dan metadata, yang tidak tersedia pada format CSV.

- Mendukung banyak lembar kerja, pemformatan, dan penyimpanan metadata.
- Sesuai untuk pelaporan yang memerlukan struktur dan presentasi formal.
- Ukuran file cenderung lebih besar dibandingkan CSV.

Membaca file XLSX menggunakan `pandas`:

```
import pandas as pd
df = pd.read_excel("data_excel.xlsx")
df.head()
```

Menyimpan data ke dalam file XLSX:

```
import pandas as pd
data = {
    "nama": ["Andi", "Budi"],
    "usia": [21, 22],
    "jurusan": ["Informatika", "Sistem Informasi"]
}
```

```
df = pd.DataFrame(data)
df.to_excel("data_excel.xlsx", index=False)
```

3.6.3 JSON (JavaScript Object Notation)

JSON adalah format penyimpanan data berbasis teks yang dirancang untuk merepresentasikan struktur data kompleks, seperti objek dan array. Format ini banyak digunakan dalam pertukaran data melalui antarmuka pemrograman aplikasi (API), terutama dalam konteks pengembangan aplikasi web.

- Mendukung struktur data bersarang (*nested*) dan dinamis.
- Cocok untuk data hierarkis atau berorientasi objek.
- Umum digunakan dalam komunikasi antarsistem melalui API.

Membaca file JSON dengan kombinasi pustaka `json` dan `pandas`:

Pendekatan ini memanfaatkan pustaka `json` untuk memuat data mentah dari file, kemudian mengonversinya menjadi `DataFrame` menggunakan `pandas`. Metode ini berguna ketika data perlu diproses atau divalidasi terlebih dahulu sebelum dimasukkan ke dalam struktur tabular.

```
import pandas as pd
import json

with open("data_json.json", "r") as f:
```

```
    data = json.load(f)

df = pd.DataFrame(data)
df.head()
```

Membaca file JSON langsung menggunakan `pandas`:

Jika file JSON berbentuk larik objek (*list of records*), maka `pandas` dapat langsung memuatnya menjadi objek `DataFrame` tanpa perlu parsing manual.

```
import pandas as pd
df = pd.read_json("data_json.json")
df.head()
```

Menyimpan data ke dalam file JSON:

Terdapat dua pendekatan umum dalam menyimpan data ke format JSON, tergantung pada struktur data yang digunakan dan kebutuhan pemrosesan:

- **Menggunakan `pandas`**: Cocok ketika data sudah berada dalam format `DataFrame`, terutama jika data berasal dari proses analisis atau transformasi sebelumnya. Fungsi `to_json()` memungkinkan pengaturan format output, seperti orientasi data dan indentasi.
- **Menggunakan pustaka `json`**: Lebih fleksibel untuk menyimpan struktur data Python asli, seperti `list` atau `dict`, tanpa harus mengonversinya terlebih dahulu ke `DataFrame`. Cocok untuk menyimpan data JSON yang bersarang atau tidak berbentuk tabular.

```python
data = [
    {"nama": "Andi", "usia": 21, "jurusan": "Informatika"},
    {"nama": "Budi", "usia": 22, "jurusan": "Sistem Informasi"
    }
]

# Dengan pandas
import pandas as pd
df = pd.DataFrame(data)
df.to_json("data_json.json", orient="records", indent=4)

# Dengan json
import json
with open("data_json.json", "w") as f:
    json.dump(data, f, indent=4)
```

3.6.4 XML (Extensible Markup Language)

XML merupakan format markup yang digunakan secara luas sebelum populernya JSON. Format ini masih banyak dijumpai dalam sistem warisan (*legacy systems*) serta dokumen resmi yang membutuhkan validasi struktur melalui skema seperti XSD (XML Schema Definition).

- Merupakan format standar dalam pertukaran data di lingkungan enterprise dan pemerintahan.
- Mendukung struktur data kompleks serta validasi melalui skema.
- Umum digunakan dalam aplikasi lama atau dokumen institusional.

Membaca file XML menggunakan `pandas`:

Apabila struktur XML bersifat homogen dan konsisten (misalnya terdiri dari elemen-elemen yang berulang), maka dapat dibaca langsung menggunakan `pandas`.

```
import pandas as pd
df = pd.read_xml("data_xml.xml")
df.head()
```

Menyimpan data ke dalam file XML:

```
import xml.etree.ElementTree as ET

data = [
    {"nama": "Andi", "usia": 21},
    {"nama": "Budi", "usia": 22}
]

root = ET.Element("mahasiswa")
for item in data:
    mhs = ET.SubElement(root, "data")
    ET.SubElement(mhs, "nama").text = item["nama"]
    ET.SubElement(mhs, "usia").text = str(item["usia"])

tree = ET.ElementTree(root)
tree.write("data_xml.xml", encoding="utf-8", xml_declaration=
    True)
```

3.7 Studi Kasus: Scraping Berita Ekonomi

Masalah: Seorang analis ingin memperoleh berita ekonomi terbaru dari situs berita nasional.

Langkah:

1. Tentukan situs target dan identifikasi struktur HTML-nya.
2. Gunakan `requests` untuk mengambil halaman.
3. Gunakan `BeautifulSoup` untuk mengekstrak judul, tanggal, dan isi berita.
4. Simpan data ke dalam format CSV.

Ilustrasi naratif: "Data hasil *scraping* kemudian dimasukkan ke dalam DataFrame menggunakan pandas, lalu dianalisis frekuensi kemunculan kata-kata ekonomi seperti 'inflasi', 'bunga', dan 'BI rate'."

3.8 Tantangan dan Etika Pengumpulan Data

Pengumpulan data, khususnya dari sumber daring (online), memerlukan perhatian terhadap berbagai aspek teknis dan etis. Berikut ini adalah beberapa tantangan umum serta pertimbangan etis yang perlu diperhatikan dalam proses pengumpulan data, baik melalui *scraping* maupun antarmuka API.

1. **Legalitas dan Kepatuhan Hukum**
    - Beberapa situs web secara eksplisit melarang pengambilan data secara otomatis (*web scraping*) dalam ketentuan layanannya (*terms of service*).

- Disarankan untuk menggunakan antarmuka API resmi apabila tersedia, karena umumnya API disediakan dengan batasan dan dokumentasi yang lebih jelas serta disesuaikan untuk konsumsi data eksternal.

2. **Pembatasan Akses (Rate Limiting)**

   - Situs web dapat menerapkan pembatasan akses berdasarkan frekuensi permintaan, dan dalam beberapa kasus dapat memblokir alamat IP yang dianggap melakukan permintaan berlebihan.
   - Oleh karena itu, penting untuk menerapkan jeda (*delay*) antar permintaan, misalnya menggunakan fungsi `time.sleep()` dalam Python, guna menghindari deteksi sebagai aktivitas mencurigakan.

3. **Privasi dan Perlindungan Data Pribadi**

   - Pengumpulan data yang mengandung informasi pribadi harus memperhatikan prinsip-prinsip privasi dan mematuhi regulasi yang berlaku, seperti GDPR di Uni Eropa atau UU Perlindungan Data Pribadi di Indonesia.
   - Informasi yang bersifat sensitif sebaiknya dianonimkan atau dihindari pengumpulannya kecuali dengan persetujuan yang sah.

4. **Keaslian dan Validitas Data**

   - Data yang diperoleh dari proses *scraping* maupun API dapat mengandung ketidakakuratan, duplikasi, atau format yang tidak konsisten.
   - Oleh karena itu, proses validasi dan pembersihan data sangat penting dilakukan sebelum analisis lebih lanjut.

5. **Perlindungan Situs dan Teknologi Anti-Bot**

   - Beberapa situs menerapkan teknologi pelindung seperti `Cloudflare`, yang berfungsi untuk mendeteksi dan memblokir lalu lintas otomatis dari bot atau *skrip scraping*.
   - Teknologi ini dapat menghalangi akses langsung melalui pustaka *scraping* konvensional seperti `requests` atau `BeautifulSoup`, sehingga memerlukan pendekatan yang lebih kompleks (misalnya `Selenium` dengan dukungan browser, atau layanan proxy).
   - Dalam konteks etika, upaya untuk menghindari sistem perlindungan ini perlu dipertimbangkan secara hati-hati agar tidak melanggar hak penyedia data.

3.9 LATIHAN / TUGAS AKHIR

1. **[Uraian]** Bandingkan kelebihan dan kekurangan teknik *web scraping* menggunakan `BeautifulSoup` dan `Selenium`.

2. **[Uraian Visual]** Buatlah diagram alir proses pengumpulan data dari situs web hingga ke format DataFrame dalam Python.

3. **[Coding]**

   Gunakan `requests` dan `BeautifulSoup` untuk mengambil 5 judul berita dari situs berita lokal dan simpan ke file CSV.

4. **[Studi Kasus]** Analisis data cuaca dari API OpenWeatherMap selama 7 hari dan simpulkan pola cuaca di kota pilihan Anda.

5. **[Studi Kasus]** Ambil data dari BPS atau Data.go.id dan lakukan parsing menggunakan Python. Tampilkan 3 analisis awal dari data tersebut.

Daftar Pustaka

[1] D. Laney, "3D Data Management: Controlling Data Volume, Velocity and Variety," *META Group*, 2001, Technical Report.

[2] R. Kitchin, *The Data Revolution: Big Data, Open Data, Data Infrastructures and Their Consequences*. Sage Publications, 2014, ISBN: 9781446287484.

[3] A. Gandomi **and** M. Haider, "Beyond the hype: Big data concepts, methods, and analytics," *International Journal of Information Management*, **jourvol** 35, **number** 2, **pages** 137–144, 2015. DOI: 10.1016/j.ijinfomgt.2014.10.007.

[4] R. Mitchell, *Web Scraping with Python: Collecting More Data from the Modern Web*, 2nd. O'Reilly Media, 2018, ISBN: 9781491985571.

[5] R. Lawson, *Web Scraping with Python and BeautifulSoup*. Packt Publishing, 2015, ISBN: 9781783553299.

[6] V. Krotov **and** L. Silva, "Legality and Ethics of Web Scraping: A Discussion," *Journal of the Association for Information Systems*, **jourvol** 21, **number** 5, **pages** 1357–1376, 2020.

4 PEMBERSIHAN DATA (DATA CLEANING)

4.1 Tujuan Pembelajaran

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Menjelaskan pentingnya proses pembersihan data dalam proyek Ilmu Data.
2. Mengidentifikasi berbagai jenis masalah data seperti nilai hilang, duplikasi, dan outlier.
3. Menggunakan Python untuk melakukan proses cleaning menggunakan Pandas dan NumPy.
4. Memilih strategi *cleaning* yang sesuai dengan konteks dan jenis data.
5. Menilai dampak kualitas data terhadap hasil analisis dan model prediktif.

4.2 Mengapa Data Harus Dibersihkan?

Data yang diperoleh dari berbagai sumber sering kali tidak dalam kondisi ideal. Masalah-masalah seperti nilai kosong

(*missing values*), nilai ekstrem (*outlier*), format tidak konsisten, dan data duplikat dapat memengaruhi kualitas hasil analisis. Pembersihan data (*data cleaning*) adalah tahap kritis yang menentukan validitas dari *insight* yang dihasilkan dalam proses Ilmu Data [1]–[3].

Contoh Ilustratif: Seorang analis data ingin menghitung rata-rata usia pelanggan dari dataset e-commerce. Jika terdapat nilai usia 999 atau NULL, maka perhitungan rata-rata bisa menyesatkan tanpa pembersihan terlebih dahulu.

4.3 Jenis Masalah Data Umum

4.3.1 Missing Values (Nilai Hilang)

Dalam proses analisis data, sering kali ditemukan kasus di mana sebagian nilai pada dataset tidak tersedia atau hilang. Nilai hilang (*missing values*) dapat mengganggu proses analisis maupun pelatihan model prediktif jika tidak ditangani dengan tepat. Oleh karena itu, deteksi dan penanganan nilai hilang merupakan langkah penting dalam tahap pra-pemrosesan data.

Penyebab Umum Beberapa penyebab umum terjadinya nilai hilang dalam dataset antara lain:

- Data tidak terisi pada saat pengisian formulir (non-respons).
- Terjadi kesalahan saat input data atau pada proses pengambilan data dari sumber.

- Kesalahan teknis seperti konversi format file atau proses migrasi data.

Deteksi Nilai Hilang dengan `pandas` Dalam Python, pustaka `pandas` menyediakan fungsi untuk mendeteksi nilai hilang. Contoh kode berikut menunjukkan cara membaca data dari file CSV dan menghitung jumlah nilai hilang pada setiap kolom:

```
import pandas as pd
df = pd.read_csv('data.csv')
print(df.isnull().sum())
```

Fungsi `isnull()` akan menghasilkan DataFrame boolean yang menandai nilai hilang dengan `True`, dan `sum()` akan menghitung jumlah nilai hilang per kolom.

Strategi Penanganan Terdapat beberapa pendekatan umum dalam menangani nilai hilang, yang dapat disesuaikan dengan konteks dan tujuan analisis:

- Menghapus baris atau kolom yang mengandung nilai hilang menggunakan fungsi `dropna`. Pendekatan ini sesuai jika proporsi nilai hilang relatif kecil.
- Mengisi nilai hilang dengan nilai statistik tertentu seperti rata-rata (*mean*) atau median dari kolom tersebut menggunakan fungsi `fillna`.
- Melakukan imputasi lanjutan dengan teknik interpolasi atau prediksi berbasis model untuk memperkirakan nilai yang hilang.

Pemilihan strategi yang tepat memerlukan pemahaman terhadap konteks data serta pengaruh potensial terhadap hasil analisis atau performa model.

4.3.2 Duplicates (Duplikasi Data)

Duplikasi data merupakan salah satu isu yang umum dijumpai dalam proses pra-pemrosesan data. Hal ini biasanya terjadi ketika data berasal dari beberapa sumber yang berbeda atau akibat pengunduhan data yang dilakukan secara berulang. Keberadaan data duplikat dapat menyebabkan distorsi dalam analisis dan model prediktif, karena pengamatan yang identik dihitung lebih dari sekali.

Untuk mengidentifikasi baris yang terduplikasi, dapat digunakan fungsi `duplicated()` pada objek `DataFrame`. Fungsi ini akan mengembalikan nilai `True` untuk setiap baris yang dianggap duplikat (yaitu baris yang sudah pernah muncul sebelumnya dengan nilai yang sama pada semua kolom). Untuk menghitung total jumlah baris duplikat, cukup digunakan fungsi `sum()` terhadap hasil tersebut. Sementara itu, untuk menghapus seluruh baris yang teridentifikasi sebagai duplikat, digunakan fungsi `drop_duplicates()`.

Contoh implementasi dalam Python menggunakan pustaka `pandas` adalah sebagai berikut:

```
df.duplicated().sum()        # Cek jumlah duplikasi
df = df.drop_duplicates()    # Menghapus duplikasi
```

4.3.3 Inconsistent Formatting (Format Tidak Konsisten)

Format data yang tidak konsisten antar entri dapat menyebabkan kesalahan dalam proses analisis maupun pemodelan data. Ketidakkonsistenan ini sering terjadi dalam berbagai bentuk, antara lain:

- **Format tanggal yang beragam**, seperti `"2022/10/01"` dan `"01-10-2022"`, yang menyulitkan proses parsing menjadi objek `datetime`.
- **Kapitalisasi yang tidak seragam** pada data kategorikal, contohnya `"Jakarta"` dan `"jakarta"`. Walaupun secara semantik identik, komputer akan mengenalinya sebagai nilai yang berbeda.

Untuk menangani isu kapitalisasi, fungsi string pada pustaka `pandas` dapat digunakan untuk menyeragamkan format huruf menjadi huruf kecil. Contoh implementasi dalam `Python`:

```
df['kota'] = df['kota'].str.lower()
```

Langkah ini memastikan bahwa seluruh entri dalam kolom `kota` ditangani secara konsisten dan menghindari redundansi akibat perbedaan kapitalisasi.

4.3.4 Outliers (Nilai Ekstrem)

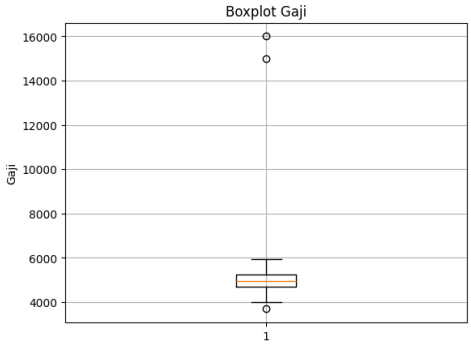
Outlier merujuk pada observasi dengan nilai yang jauh berbeda dari distribusi umum data. Nilai-nilai ekstrem ini dapat mengganggu analisis statistik, memengaruhi nilai rata-rata,

simpangan baku, dan bahkan mengganggu kinerja model prediktif. Oleh karena itu, identifikasi dan penanganan *outlier* menjadi langkah penting dalam tahap pra-pemrosesan data.

Salah satu metode visual yang umum digunakan untuk mendeteksi *outlier* adalah *boxplot*. Contoh penggunaan dalam Python:

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

# Contoh data
np.random.seed(42)
gaji_normal = np.random.normal(5000, 500, 100)
gaji_outliers = [15000, 16000]
gaji = np.append(gaji_normal, gaji_outliers)
df = pd.DataFrame({'gaji': gaji})

# Plot boxplot
plt.boxplot(df['gaji'])
plt.title('Boxplot Gaji')
plt.ylabel('Gaji')
plt.grid(True)
plt.show()
```

Hasil visualisasi menunjukkan bahwa titik-titik di luar batas atas (*upper whisker*) merupakan nilai yang secara statistik dianggap ekstrem.



Gambar 2: Boxplot variabel `gaji` untuk mendeteksi outlier.

Beberapa pendekatan yang dapat dilakukan untuk menangani *outlier*:

- **Menghapus outlier**, bila jumlahnya sedikit atau dianggap sebagai kesalahan pencatatan.
- **Transformasi data**, misalnya logaritma, akar kuadrat, atau *z-score*, untuk mengurangi pengaruh nilai ekstrem.
- **Winsorisasi**, yaitu membatasi nilai ekstrem pada batas tertentu dan menggantinya dengan nilai ambang.

4.4 Studi Kasus: Dataset Pasien Rumah Sakit

Masalah: Dataset rumah sakit memiliki data umur, jenis kelamin, hasil lab, dan diagnosa.

Masalah yang ditemukan:

- Beberapa kolom umur kosong

- Ada pasien dengan umur “-5” dan “250”
- Nama kota pasien ditulis dengan 5 variasi berbeda untuk “Surabaya”
- Diagnosa yang seharusnya “Diabetes” ditulis sebagai “diabtes”, “dbetes”, dll

Solusi Cleaning:

- Ganti umur negatif dengan nilai `NaN`, lalu isi median
- Koreksi kota menggunakan `.replace()` atau mapping
- Gunakan spell-checking/standardisasi pada nama diagnosa

4.5 Python Tools untuk Data Cleaning

Data cleaning merupakan tahap krusial dalam pemrosesan data, dan Python menyediakan berbagai pustaka yang efisien untuk menangani data yang tidak bersih atau tidak konsisten.

4.5.1 Pandas

`Pandas` merupakan pustaka paling umum digunakan untuk manipulasi dan pembersihan data tabular. Beberapa fungsi penting yang sering digunakan dalam proses *data cleaning* antara lain:

- `isnull()`, `dropna()`, `fillna()` untuk deteksi dan penanganan nilai hilang.
- `replace()`, `apply()`, `duplicated()` untuk transformasi data, fungsi kustom, dan deteksi duplikasi.

4.5.2 NumPy

`NumPy` mendukung operasi numerik yang efisien dan digunakan sebagai fondasi banyak operasi dalam `pandas`. Dalam konteks data cleaning, `NumPy` berguna untuk:

- Penanganan nilai hilang (misalnya `NaN`) dan nilai ekstrem.
- Operasi numerik yang membutuhkan efisiensi tinggi.

```
import numpy as np
df['umur'] = df['umur'].replace(-5, np.nan)
```

4.5.3 Regex untuk Cleaning Teks

Ekspresi reguler (*regular expressions*) sering digunakan untuk pembersihan teks, seperti menghapus simbol atau karakter tidak diinginkan dari kolom bertipe string. Contoh berikut menunjukkan pembersihan simbol dari kolom alamat:

```
import re
df['alamat'] = df['alamat'].str.replace(r'[^\w\s]', '', regex=
    True)
```

4.6 Strategi Cleaning Berdasarkan Konteks

Strategi pembersihan data tidak selalu bersifat universal. Tidak semua nilai hilang perlu dihapus secara langsung. Pemilihan metode cleaning bergantung pada beberapa faktor, seperti:

- Ukuran dataset
- Kepentingan kolom terhadap analisis
- Distribusi dan proporsi nilai hilang
- Pengetahuan domain (*domain knowledge*)

Contoh: Dalam analisis data medis, kolom tekanan darah memiliki prioritas lebih tinggi untuk dipertahankan dibandingkan kolom seperti nomor telepon pasien.

4.7 Evaluasi Kualitas Data

Setelah proses *cleaning*, langkah selanjutnya adalah melakukan evaluasi terhadap hasil pembersihan data. Evaluasi ini bertujuan memastikan bahwa data telah bebas dari nilai yang mencurigakan dan siap untuk dianalisis lebih lanjut. Beberapa langkah evaluasi antara lain:

- Memeriksa distribusi nilai setelah cleaning.
- Memastikan tidak ada *outlier* ekstrem yang tersisa.

- Melakukan visualisasi pasca-*cleaning* untuk konfirmasi.

```
df['gaji'].hist()
```

Data perlu diperiksa kembali untuk memastikan bahwa nilainya masuk akal secara statistik dan logika domain.

4.8 Tips dan Best Practice Cleaning

Berikut beberapa praktik terbaik yang disarankan dalam proses pembersihan data:

1. Hindari menghapus data secara langsung kecuali benar-benar diperlukan.
2. Selalu simpan salinan cadangan dari data mentah.
3. Lakukan eksplorasi menyeluruh sebelum tahap cleaning.
4. Catat semua perubahan yang dilakukan dalam log transformasi.
5. Gunakan pipeline atau fungsi seperti `transform()` untuk proses *cleaning* yang dapat direplikasi.

4.9 LATIHAN / TUGAS AKHIR

1. **[Uraian]** Jelaskan 3 jenis masalah data umum yang sering ditemukan dalam proyek Ilmu Data dan bagaimana cara menanganinya.

2. **[Uraian Visual]** Buatlah diagram alir yang menggambarkan proses cleaning dataset dari tahap deteksi hingga evaluasi.

3. **[Coding]**

   Bersihkan dataset CSV yang memiliki kolom `umur`, `kota`, dan `diagnosa`. Gunakan Pandas untuk:

   - Mengisi missing value
   - Menstandarkan penulisan kota
   - Menghapus nilai umur di luar rentang 0–120

4. **[Studi Kasus]** Ambil dataset COVID-19 terbuka dan identifikasi minimal 3 masalah data, lalu bersihkan menggunakan Python.

5. **[Studi Kasus]** Analisis dampak jika proses *cleaning* tidak dilakukan dengan benar dalam proyek analisis data bencana alam.

Daftar Pustaka

[1] E. Rahm **and** H. H. Do, “Data cleaning: Problems and current approaches,” *IEEE Data Engineering Bulletin*, **jourvol** 23, **number** 4, **pages** 3–13, 2000.

[2] W. M. P. van der Aalst, A. Adriansyah **and** B. F. van Dongen, *Principles of Data Cleaning in Process Mining*. Springer, 2020, ISBN: 9783030496626.

[3] T. Dasu **and** T. Johnson, *Exploratory Data Mining and Data Cleaning*. John Wiley & Sons, 2003, ISBN: 9780471268519.

5 EKSPLORASI DATA *(EXPLORATORY DATA ANALYSIS)*

5.1 Tujuan Pembelajaran

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Memahami tujuan dan manfaat eksplorasi data dalam proses Ilmu Data.
2. Menggunakan teknik statistik deskriptif untuk memahami karakteristik dataset.
3. Menerapkan teknik visualisasi untuk mendeteksi pola dan hubungan antar variabel.
4. Menggunakan Python (Pandas, Matplotlib, dan Seaborn) untuk analisis eksploratif.
5. Menyusun laporan eksplorasi data yang sistematis sebagai dasar pengambilan keputusan.

5.2 Apa Itu Eksplorasi Data?

Eksplorasi Data atau **Exploratory Data Analysis *exploratory data analysis*(EDA)** adalah tahap dalam analisis data

yang bertujuan untuk memahami struktur, pola, dan relasi dalam data [1]–[3]. EDA dilakukan sebelum membangun model agar analisis lebih terarah dan akurat.

Tujuan EDA:

- Mengenal distribusi data
- Menemukan nilai ekstrem (outlier)
- Mengidentifikasi korelasi antar variabel
- Menyusun hipotesis awal

5.3 Statistik Deskriptif dalam EDA

Statistik deskriptif adalah bagian penting dari *Exploratory Data Analysis* (EDA) yang bertujuan untuk memberikan ringkasan numerik mengenai karakteristik utama dari suatu kumpulan data. Tahap ini membantu peneliti memahami pola dasar, mendeteksi anomali, dan mengevaluasi asumsi sebelum melakukan analisis lanjut seperti pemodelan statistik atau pembelajaran mesin [4], [5].

Secara umum, statistik deskriptif terbagi ke dalam dua kelompok utama: ukuran pemusatan dan ukuran penyebaran [5].

5.3.1 Ukuran Pemusatan

Ukuran pemusatan memberikan informasi tentang titik tengah atau nilai representatif dari distribusi data. Tiga ukuran

yang paling umum digunakan meliputi:

- **Mean** — Rata-rata aritmatika dari sekumpulan data.
- **Median** — Nilai tengah ketika data diurutkan.
- **Modus** — Nilai yang paling sering muncul.

5.3.2 Ukuran Penyebaran

Ukuran penyebaran menginformasikan tingkat variabilitas atau sebaran data dari nilai pusatnya:

- **Range** — Selisih antara nilai maksimum dan minimum.
- **Standar Deviasi** — Ukuran seberapa jauh data menyebar dari rata-rata.
- **Varians** — Kuadrat dari standar deviasi.
- **IQR (Interquartile Range)** — Rentang antara kuartil ketiga dan kuartil pertama.

5.4 Visualisasi Data

Visualisasi data berfungsi sebagai media komunikasi untuk menyampaikan informasi yang kompleks secara visual dan intuitif. Visualisasi mendukung interpretasi pola, anomali, dan hubungan antar variabel. Dengan bantuan grafik, diagram, atau peta interaktif, visualisasi memungkinkan pengguna untuk memahami tren, pola tersembunyi, outlier, serta hubungan antar variabel dengan lebih cepat dibandingkan penyajian dalam bentuk tabel atau teks mentah [6]–[8].

Visualisasi membantu mempercepat proses analisis, mendukung pengambilan keputusan berbasis data, serta menjembatani pemahaman antara analis data dengan audiens non-teknis.

Untuk menggali wawasan dari data numerik dan kategorikal, beberapa jenis visualisasi dasar yang sering digunakan dalam tahap EDA antara lain:

5.4.1 Histogram

Histogram merupakan representasi grafis yang digunakan untuk menggambarkan distribusi frekuensi dari suatu variabel numerik. Histogram membantu dalam memahami persebaran nilai, identifikasi sebaran normal, serta deteksi potensi skewness pada data.

```
import matplotlib.pyplot as plt
df['usia'].hist(bins=10)
plt.title("Distribusi Usia Mahasiswa")
plt.xlabel("Usia")
plt.ylabel("Jumlah")
plt.show()
```

5.4.2 Boxplot

Boxplot (diagram kotak) digunakan untuk menggambarkan penyebaran data serta mendeteksi keberadaan nilai pencilan (*outlier*). Visualisasi ini menyajikan nilai minimum, kuartil pertama (Q1), median, kuartil ketiga (Q3), dan nilai maksimum.

```
import seaborn as sns
sns.boxplot(x=df['gaji'])
```

5.4.3 Scatter Plot

Scatter plot atau diagram pencar digunakan untuk mengevaluasi hubungan antara dua variabel numerik. Visualisasi ini memungkinkan identifikasi pola hubungan linier maupun non-linier serta keberadaan *outlier*.

```
sns.scatterplot(x='usia', y='nilai', data=df)
```

5.4.4 Heatmap Korelasi

Heatmap korelasi menyajikan hubungan antar variabel numerik dalam bentuk matriks korelasi. Warna dalam heatmap menunjukkan kekuatan dan arah hubungan (positif atau negatif), sehingga memudahkan identifikasi variabel yang memiliki asosiasi kuat.

```
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
```

5.5 EDA terhadap Data Kategorikal

Untuk variabel kategorikal, visualisasi dapat dilakukan dengan *bar chart* atau *count plot*:

```
sns.countplot(x='jenis_kelamin', data=df)
```

Untuk mengevaluasi hubungan antara dua variabel kategorikal, gunakan `crosstab`:

```
pd.crosstab(df['jurusan'], df['status_kelulusan'])
```

5.6 Studi Kasus: Analisis Data Keuangan Mahasiswa

Dataset: Berisi informasi mengenai pemasukan bulanan, pengeluaran, dan pinjaman mahasiswa.

Langkah Analisis:

1. Gunakan `.describe()` untuk memperoleh ringkasan statistik.
2. Visualisasikan distribusi pemasukan dengan histogram.
3. Gunakan *scatter plot* untuk mengevaluasi hubungan antara pemasukan dan pengeluaran.
4. Hitung koefisien korelasi antara pengeluaran dan pinjaman.

Insight: Mahasiswa dengan pemasukan kurang dari satu juta rupiah cenderung memiliki pinjaman lebih tinggi dan pengeluaran yang lebih tidak stabil.

5.7 Insight dan Narasi Eksploratif

Hasil eksplorasi data perlu disampaikan melalui narasi yang jelas dan berbasis data. Narasi eksploratif merupakan komponen penting dalam pelaporan analisis data.

- "Mayoritas mahasiswa berasal dari tiga kota besar..."
- "Pengeluaran mahasiswa meningkat seiring naiknya semester..."
- "Hubungan negatif ditemukan antara frekuensi pinjaman dan indeks prestasi kumulatif (IPK)."

5.8 Tools Tambahan untuk EDA

Beberapa alat bantu modern dapat digunakan untuk mempercepat dan memperluas eksplorasi data secara otomatis dan interaktif:

1. **Pandas Profiling** — Membuat laporan otomatis untuk dataset.

```
import pandas_profiling
df.profile_report().to_file("laporan_eda.html")
```

2. **Sweetviz** — Visualisasi interaktif untuk membandingkan dua subset data.
3. **D-Tale** — Antarmuka eksplorasi data interaktif berbasis browser.

5.9 Kesalahan Umum dalam EDA

Beberapa kesalahan yang sering ditemukan saat melakukan EDA meliputi:

- Mengandalkan rata-rata tanpa mempertimbangkan outlier.
- Tidak memeriksa tipe data (numerik vs. kategorikal).
- Melakukan analisis tanpa membersihkan data terlebih dahulu.
- Menghasilkan terlalu banyak grafik tanpa mengaitkannya dengan insight analitik.

5.10 LATIHAN / TUGAS AKHIR

1. **[Uraian]** Jelaskan peran statistik deskriptif dalam tahap eksplorasi data dan berikan contoh penerapannya!
2. **[Uraian Visual]** Gambar dan jelaskan contoh output histogram dan boxplot dari data pengeluaran bulanan mahasiswa.
3. **[Coding]**

   Lakukan eksplorasi pada dataset mahasiswa. Tampilkan:

   - Statistik deskriptif
   - Korelasi antar variabel numerik
   - Visualisasi: histogram, scatter plot, dan heatmap
4. **[Studi Kasus]** Analisis dataset pinjaman mahasiswa (student loan). Buat narasi insight dari EDA yang dilakukan.
5. **[Studi Kasus]** Unduh dataset keuangan rumah tangga dari BPS/Data.go.id. Lakukan EDA menyeluruh dan simpulkan hubungan pengeluaran dengan jenis pekerjaan kepala rumah tangga.

Daftar Pustaka

[1] J. W. Tukey, *Exploratory Data Analysis*. Addison-Wesley, 1977, ISBN: 9780201076165.

[2] P. Bruce **and** A. Bruce, *Practical Statistics for Data Scientists: 50 Essential Concepts*. Sebastopol, CA: O'Reilly Media, 2017, ISBN: 978-1-4919-0649-3.

[3] X. Yan **and** Z. Ma, *Data Exploration Using Example-Based Methods*. Springer, 2009, ISBN: 9780387715022.

[4] L.-P. Chen, *Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python: by Peter Bruce, Andrew Bruce, and Peter Gedeck, O'Reilly Media Inc., Boston, United States 2020. ISBN 9781492072942, pp. xiii+ 368 (Paperback),* 56.00, 2021.

[5] M. C. Data, M. Komorowski, D. C. Marshall, J. D. Salciccioli **and** Y. Crutain, "Exploratory data analysis," *Secondary analysis of electronic health records*, **pages** 185–203, 2016.

[6] S. Few, *Now you see it: simple visualization techniques for quantitative analysis*. Analytics Press, 2009.

[7] P. Buono **and** R. Lanzilotti, "Big Data Visualization and Visual Analytics," **in***Human-Computer Interaction in Various Application Domains* CRC Press, 2024, **pages** 398–421.

[8] M. O. Ward, G. Grinstein **and** D. Keim, *Interactive data visualization: foundations, techniques, and applications*. AK Peters/CRC Press, 2010.

6 TRANSFORMASI DAN FEATURE ENGINEERING

6.1 Tujuan Pembelajaran

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Memahami konsep transformasi data dan rekayasa fitur (*feature engineering*).
2. Mengimplementasikan berbagai teknik encoding dan normalisasi.
3. Mengubah data mentah menjadi fitur yang relevan dan dapat digunakan oleh model.
4. Membangun pipeline preprocessing dengan Scikit-learn.
5. Menerapkan feature selection dan *extraction* secara efisien.

6.2 Apa Itu Transformasi Data?

Transformasi data adalah tahapan penting dalam proses praproses data (*data preprocessing*) yang bertujuan untuk mengubah data mentah menjadi format yang lebih siap digunakan untuk analisis statistik, visualisasi, atau pemodelan machine learning. Tahapan ini dilakukan setelah proses pem-

bersihan data dan sebelum proses modeling, sebagai jembatan antara data asli dan kebutuhan algoritmik [1]–[3].

Transformasi data mencakup berbagai teknik yang tidak hanya mengubah bentuk atau representasi data, tetapi juga meningkatkan kualitas informasi yang dapat diekstrak darinya. Proses ini sangat krusial karena sebagian besar algoritma *machine learning* mensyaratkan data dalam bentuk numerik, terstandarisasi, dan terdistribusi secara proporsional agar mampu memberikan hasil prediksi yang akurat dan stabil.

6.3 Apa Itu Feature Engineering?

Feature Engineering adalah seni dan ilmu dalam menciptakan fitur (variabel input) baru dari data yang telah tersedia. Proses ini sangat krusial karena kualitas fitur secara langsung memengaruhi performa model prediktif.

Sebagai contoh sederhana: apabila dataset hanya memiliki variabel `tanggal_lahir`, kita dapat mengubahnya menjadi `usia`, yaitu fitur yang lebih relevan untuk keperluan analisis atau prediksi, misalnya dalam konteks risiko kesehatan atau kelayakan pinjaman.

6.4 Teknik-Teknik Transformasi

Teknik transformasi digunakan untuk menyelaraskan skala antar fitur, memperbaiki distribusi data, dan mengurangi pengaruh outlier. Pemilihan teknik transformasi yang tepat dapat berdampak besar terhadap kinerja dan stabilitas model. Berikut adalah dua teknik transformasi yang paling umum digunakan dalam analisis data dan *machine learning* :

6.4.1 Scaling (Normalisasi / Standardisasi)

- **StandardScaler**: Mengubah distribusi menjadi rata-rata = 0 dan standar deviasi = 1.
- **MinMaxScaler**: Menskalakan nilai ke dalam rentang 0 hingga 1.

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df[['gaji']] = scaler.fit_transform(df[['gaji']])
```

6.4.2 Log Transform

Digunakan untuk mengatasi distribusi miring (skewed), misalnya pada data pendapatan atau pengeluaran.

```
import numpy as np

df['log_pengeluaran'] = np.log1p(df['pengeluaran'])
```

6.5 Encoding Data Kategorikal

Dalam ilmu data dan pembelajaran mesin, algoritma umumnya memerlukan input dalam bentuk numerik. Oleh karena itu, data kategorikal (seperti jenis kelamin, status pernikahan, tipe produk) perlu dikonversi menjadi representasi numerik agar dapat diproses oleh model. Proses ini dikenal dengan nama *encoding* [2].

Pemilihan teknik *encoding* yang tepat sangat penting, terutama karena representasi numerik yang salah dapat mengakibatkan penyimpangan makna atau menimbulkan bias algoritmik.

Dua teknik *encoding* yang paling umum adalah:

6.5.1 One-Hot Encoding

Mengubah variabel kategori menjadi vektor biner, di mana setiap kategori direpresentasikan oleh satu kolom biner.

```
df = pd.get_dummies(df, columns=['jenis_kelamin'])
```

6.5.2 Label Encoding

Memberi nilai numerik ke kategori. Hati-hati dalam menerapkan pada data non-ordinal, karena label encoding dapat menyiratkan urutan yang tidak ada.

```python
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
df['provinsi'] = le.fit_transform(df['provinsi'])
```

6.6 Feature Selection (Seleksi Fitur)

Proses memilih fitur yang paling relevan terhadap target dan menghapus fitur yang tidak informatif atau redundan.

Teknik-teknik yang umum digunakan:

- **Korelasi**: Menghindari fitur yang sangat berkorelasi karena dapat menyebabkan multikolinearitas.
- **Uji Chi-square**: Cocok untuk data kategorikal.
- **Recursive Feature Elimination (RFE)**: Mengeliminasi fitur secara iteratif berdasarkan kepentingan.

```python
from sklearn.feature_selection import SelectKBest, chi2

X_new = SelectKBest(score_func=chi2, k=5).fit_transform(X, y)
```

6.7 Feature Extraction (Ekstraksi Fitur)

Mengubah atau menggabungkan fitur eksisting menjadi representasi baru yang lebih bermakna atau berdimensi lebih rendah.

- **Principal Component Analysis (PCA)**: Reduksi dimensi berdasarkan variansi data.
- **TF-IDF**: Representasi numerik dokumen dalam analisis teks.
- **Word Embeddings**: Representasi vektor untuk kata dalam NLP.

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
```

6.8 Studi Kasus: Dataset E-Commerce

Masalah: Dataset pembelian online berisi fitur-fitur berikut:

- Tanggal transaksi
- Kota
- Kategori produk
- Harga satuan
- Jumlah barang

Transformasi dan Engineering:

1. Ekstrak fitur `bulan` dan `hari` dari tanggal.
2. Hitung fitur baru: total belanja = `harga * jumlah`.

3. Lakukan *one-hot encoding* pada `kategori produk`.

4. Terapkan *log transform* pada total belanja karena distribusinya skewed.

Insight: Fitur-fitur baru tersebut terbukti meningkatkan performa model prediksi terhadap variabel target seperti `jumlah pengemb`

6.9 Pipelines dalam Scikit-learn

Pipeline menyederhanakan dan mengautomasi proses *feature engineering* dan pelatihan model. Semua langkah preprocessing dan modeling dapat digabungkan ke dalam satu alur kerja yang konsisten.

```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression

pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('model', LogisticRegression())
])

pipe.fit(X_train, y_train)
```

Keuntungan: Konsistensi, efisiensi, serta kemudahan dalam integrasi dengan `GridSearchCV` untuk *hyperparameter tuning*.

6.10 Feature Engineering untuk Data Waktu dan Teks

Feature engineering adalah proses menciptakan fitur baru atau merekayasa ulang fitur yang ada untuk meningkatkan kinerja model dalam machine learning. Dalam konteks data waktu (datetime) dan data teks (text), proses feature engineering memiliki karakteristik yang khas, karena jenis datanya tidak langsung cocok untuk digunakan oleh algoritma prediktif yang bekerja dengan data numerik [4].

Transformasi ini bertujuan untuk mengubah data waktu dan teks menjadi representasi numerik yang informatif, sehingga dapat digunakan secara efektif dalam model prediktif

6.10.1 Datetime

Feature engineering pada data waktu sering melibatkan ekstraksi informasi kalender.

```
df['tanggal'] = pd.to_datetime(df['tanggal'])
df['bulan'] = df['tanggal'].dt.month
df['hari'] = df['tanggal'].dt.dayofweek
```

6.10.2 Teks (TF-IDF)

Mengubah teks menjadi representasi numerik berbasis frekuensi kata.

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer()
X_tfidf = vectorizer.fit_transform(df['ulasan'])
```

6.11 Tantangan dalam Feature Engineering

Beberapa tantangan umum yang perlu diperhatikan dalam proses *feature engineering*:

- **Overfitting**: Terlalu banyak fitur spesifik bisa membuat model hanya cocok pada data latih.
- **Multikolinearitas**: Fitur yang berkorelasi tinggi dapat membingungkan model.
- **Ukuran dataset besar**: Dapat menyebabkan masalah performa dan konsumsi memori.
- **Kesalahan logika domain**: Misalnya membuat fitur "total tagihan" untuk memprediksi tagihan.

6.12 LATIHAN / TUGAS AKHIR BAB 6

1. **[Uraian]** Jelaskan perbedaan antara feature extraction dan feature selection, serta kapan masing-masing lebih tepat digunakan.
2. **[Uraian Visual]** Buatlah diagram transformasi data dari format mentah menjadi bentuk yang siap digunakan untuk model regresi.
3. **[Coding]**

   Dari dataset pelanggan e-commerce:

   - Ekstrak fitur waktu dari kolom `tanggal_pembelian`
   - Buat fitur `total_belanja`
   - Lakukan scaling pada fitur numerik
   - Terapkan One-Hot Encoding pada kolom `kategori_produk`
4. **[Studi Kasus]** Analisis fitur penting untuk prediksi churn pelanggan berdasarkan data transaksi dan aktivitas.
5. **[Studi Kasus]** Bandingkan performa model dengan dan tanpa feature engineering pada dataset pelanggan. Lakukan evaluasi dengan akurasi dan F1-score.

Daftar Pustaka

[1] J. D. Kelleher, B. Mac Namee **and** A. D'arcy, *Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies*. MIT press, 2020.

[2] L.-P. Chen, *Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python: by Peter Bruce, Andrew Bruce, and Peter Gedeck, O'Reilly Media Inc., Boston, United States 2020. ISBN 9781492072942, pp. xiii+ 368 (Paperback),* 56.00, 2021.

[3] W. I. D. Mining, "Data mining: Concepts and techniques," *Morgan Kaufinann*, **jourvol** 10, **number** 559-569, **page** 4, 2006.

[4] D. Jurafsky, *Speech & language processing*. Pearson Education India, 2000.

7 PENGANTAR STATISTIK UNTUK ILMU DATA

7.1 Tujuan Pembelajaran

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Memahami konsep dasar dalam statistik deskriptif dan inferensial.
2. Menggunakan ukuran pemusatan dan penyebaran data.
3. Menjelaskan berbagai jenis distribusi probabilitas umum.
4. Menghitung probabilitas sederhana dan memahami prinsip kombinatorial.
5. Menggunakan Python untuk perhitungan statistik dan visualisasi distribusi data.

7.2 Peran Statistik dalam Ilmu Data

Statistik merupakan fondasi utama dalam ilmu data (data science). Sebagai disiplin ilmu yang berfokus pada pengumpulan, pengolahan, analisis, dan interpretasi data, statistik menyediakan kerangka konseptual dan metodologis yang

sangat penting bagi seluruh siklus analisis data—mulai dari eksplorasi awal hingga pembuatan model prediktif dan pengambilan keputusan [1], [2].

Di tengah pesatnya perkembangan algoritma *machine learning* dan *artificial intelligence*, prinsip-prinsip statistik tetap menjadi tulang punggung untuk memvalidasi model, mengukur ketidakpastian, dan menghindari kesalahan interpretasi data [3]. Statistik berperan dalam:

- Memahami karakteristik data
- Membuat kesimpulan tentang populasi dari sampel
- Mengukur ketidakpastian dan resiko keputusan berbasis data

Tanpa pemahaman statistik yang kuat, proses data science hanya akan menjadi sekumpulan eksperimen tanpa dasar ilmiah. Oleh karena itu, penguasaan konsep statistik adalah syarat mutlak bagi setiap praktisi ilmu data yang ingin bekerja secara metodologis, akurat, dan terpercaya.

7.3 Statistik Deskriptif

7.3.1 Ukuran Pemusatan (Central Tendency)

- **Mean (Rata-rata)**: Jumlah semua nilai dibagi banyaknya data.
- **Median**: Nilai tengah dari data yang telah diurutkan.

- **Modus**: Nilai yang paling sering muncul.

```
import numpy as np

data = [1, 3, 4, 4, 5, 7]

print("Mean:", np.mean(data))
print("Median:", np.median(data))
```

7.3.2 Ukuran Penyebaran (Dispersion)

- **Range**: Selisih antara nilai maksimum dan minimum dalam suatu dataset.
- **Standar Deviasi (SD)**: Mengukur seberapa tersebar data terhadap nilai rata-rata.
- **Varians**: Kuadrat dari standar deviasi, menyatakan ragam penyebaran data.
- **IQR (Interquartile Range)**: Selisih antara kuartil ketiga (Q3) dan kuartil pertama (Q1), mencerminkan penyebaran data tengah.

```
print("Standar Deviasi:", np.std(data))
```

7.4 Konsep Distribusi Data

Distribusi data menunjukkan bagaimana nilai-nilai dalam dataset tersebar atau terdistribusi.

7.4.1 Distribusi Normal (Bell Curve)

- Berbentuk simetris di sekitar rata-rata.
- Banyak fenomena alam dan sosial mengikuti distribusi ini.

```
import matplotlib.pyplot as plt
import seaborn as sns

sns.histplot(data, kde=True)
```

7.4.2 Distribusi Binomial

- Termasuk distribusi diskret, untuk eksperimen dengan dua kemungkinan hasil (misal: sukses/gagal).
- Contoh: jumlah sukses dalam 10 kali lempar koin.

```
from scipy.stats import binom

prob = binom.pmf(k=3, n=10, p=0.5)
print("P(X=3):", prob)
```

7.4.3 Distribusi Poisson

- Cocok untuk kejadian langka yang terjadi dalam interval waktu atau ruang tertentu.
- Contoh: jumlah kecelakaan lalu lintas dalam satu hari.

7.5 Probabilitas Dasar

Probabilitas mengukur peluang suatu kejadian terjadi dalam ruang sampel tertentu.

7.5.1 Ruang Sampel dan Kejadian

- **Ruang Sampel**: himpunan semua kemungkinan hasil dari suatu percobaan.
- **Kejadian**: subset dari ruang sampel yang menjadi fokus perhatian.

7.5.2 Aturan Penjumlahan dan Perkalian

- **Aturan Penjumlahan**:
  $P(A \cup B) = P(A) + P(B) - P(A \cap B)$
- **Aturan Perkalian**: $P(A \cap B) = P(A) \times P(B|A)$

7.5.3 Permutasi dan Kombinasi

- **Permutasi**: Memperhitungkan urutan.
  $P(n, r) = \frac{n!}{(n-r)!}$
- **Kombinasi**: Tidak memperhitungkan urutan.
  $C(n, r) = \frac{n!}{r!(n-r)!}$

7.5.4 Inferensi Statistik

Inferensi statistik adalah proses membuat generalisasi atau kesimpulan tentang populasi berdasarkan data dari sampel.

7.5.5 Konsep Populasi dan Sampel

- **Populasi**: Seluruh elemen yang menjadi objek pengamatan atau kajian.
- **Sampel**: Subset dari populasi yang diambil untuk dianalisis.

7.5.6 Estimasi

- **Titik Estimasi**: Nilai tunggal yang digunakan untuk mengestimasi parameter populasi (misalnya rata-rata sampel).
- **Interval Estimasi**: Rentang nilai yang diyakini mengandung parameter populasi dengan tingkat kepercayaan tertentu (confidence interval).

7.5.7 Uji Hipotesis

- **Hipotesis Nol (H_0)**: Tidak terdapat perbedaan atau pengaruh.
- **Hipotesis Alternatif (H_1)**: Terdapat perbedaan atau pengaruh.

- Digunakan uji statistik seperti *t-test*, *chi-square*, dan *ANOVA* untuk menguji hipotesis.

7.6 Visualisasi Distribusi dan Probabilitas

Visualisasi sangat membantu dalam memahami bentuk distribusi dan karakteristik data.

```
sns.histplot(df['penghasilan'], bins=30, kde=True)
plt.title("Distribusi Penghasilan Mahasiswa")
```

Gunakan modul `scipy.stats` untuk menghitung probabilitas distribusi yang sesuai dengan konteks data.

7.7 Studi Kasus: Analisis Nilai Mahasiswa

Masalah: Seorang dosen ingin mengetahui apakah terdapat perbedaan signifikan rata-rata nilai akhir mahasiswa antara dua jurusan.

Langkah-Langkah Analisis:

1. Ambil sampel data nilai dari kedua jurusan.
2. Lakukan uji normalitas data pada masing-masing sampel.
3. Terapkan *independent two-sample t-test*.
4. Buat kesimpulan berdasarkan nilai *p-value*.

```python
from scipy.stats import ttest_ind

t_stat, p_value = ttest_ind(nilai_jurusan_A, nilai_jurusan_B)
print("p-value:", p_value)
```

Insight: Jika *p-value* < 0.05, maka hipotesis nol (H_0) ditolak. Ini berarti terdapat perbedaan rata-rata nilai yang signifikan secara statistik antara dua jurusan.

7.8 Pentingnya Statistik dalam Modeling

Sebelum membangun model pembelajaran mesin (machine learning):

- Statistik digunakan dalam proses seleksi fitur (*feature selection*).
- Membantu dalam menilai kualitas dan distribusi data.
- Berguna untuk mendeteksi outlier yang dapat memengaruhi performa model.
- Memastikan asumsi-asumsi statistik tertentu terpenuhi (misal: residu regresi linear harus berdistribusi normal).

7.9 LATIHAN / TUGAS AKHIR BAB 7

1. **[Uraian]** Jelaskan perbedaan antara mean, median, dan modus serta berikan contoh situasi di mana masing-masing lebih tepat digunakan.

2. **[Uraian Visual]** Gambarkan dan jelaskan bentuk distribusi normal, binomial, dan poisson.

3. **[Coding]**

   Buat program Python yang menghitung:

   - Mean, median, dan standar deviasi dataset
   - Distribusi normal dengan `sns.histplot()`
   - Probabilitas binomial dan poisson dari skenario yang Anda pilih

4. **[Studi Kasus]** Lakukan analisis uji t terhadap nilai IPK mahasiswa yang mengambil mata kuliah online dan tatap muka. Ambil kesimpulan statistiknya.

5. **[Studi Kasus]** Lakukan perhitungan kombinasi dan permutasi untuk kasus seleksi tim lomba data science dari 10 kandidat, di mana hanya 4 orang dipilih.

Daftar Pustaka

[1] L.-P. Chen, *Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python: by Peter Bruce, Andrew Bruce, and Peter Gedeck, O'Reilly Media Inc., Boston, United States 2020. ISBN 9781492072942, pp. xiii+ 368 (Paperback),* 56.00, 2021.

[2] Z. John Lu, *The elements of statistical learning: data mining, inference, and prediction*, 2010.

[3] J. D. Kelleher, B. Mac Namee **and** A. D'arcy, *Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies*. MIT press, 2020.

8 MACHINE LEARNING: KONSEP DASAR

8.1 Tujuan Pembelajaran

Setelah menyelesaikan bab ini, mahasiswa diharapkan mampu:

1. Memahami konsep dasar dan perbedaan utama dalam pendekatan machine learning.
2. Mengidentifikasi jenis-jenis machine learning: supervised, unsupervised, dan reinforcement learning.
3. Menjelaskan proses pelatihan model (training), evaluasi, dan prediksi.
4. Mengimplementasikan workflow sederhana machine learning menggunakan Scikit-learn.
5. Menilai performa model menggunakan metrik evaluasi dasar.

8.2 Apa Itu Machine Learning?

Machine Learning (ML) adalah cabang dari kecerdasan buatan (Artificial Intelligence/AI) yang berfokus pada pembuatan algoritma yang memungkinkan komputer belajar dari data tanpa diprogram secara eksplisit [1].

Definisi Praktis:

Machine learning memungkinkan sistem untuk belajar dari data, mengidentifikasi pola, dan membuat keputusan secara otomatis berdasarkan pengalaman [2].

8.3 Mengapa Machine Learning Penting?

- Mampu menangani data dalam jumlah besar.
- Digunakan dalam banyak aplikasi modern: rekomendasi produk, deteksi penipuan, pengenalan wajah, diagnosis medis [3].
- Meningkatkan efisiensi dan akurasi pengambilan keputusan berbasis data.

8.4 Klasifikasi Machine Learning

Machine Learning (ML) atau pembelajaran mesin merupakan pendekatan komputasional yang memungkinkan sistem belajar dari data dan membuat prediksi atau keputusan berdasarkan pola yang dikenali. Berdasarkan jenis data pelatihan yang digunakan, ML terbagi menjadi tiga kategori utama: *supervised learning*, *unsupervised learning*, dan *reinforcement learning* [4].

8.4.1 Supervised Learning

Supervised learning adalah metode pembelajaran di mana model dilatih menggunakan data yang telah dilabeli, yaitu memiliki pasangan input-output yang diketahui. Tujuan dari proses ini adalah membentuk model yang mampu memetakan input baru terhadap output yang sesuai berdasarkan pengalaman belajar dari data historis.

Contoh ilustratif:

- Input: Jumlah jam belajar seorang mahasiswa
- Output: Nilai ujian akhir yang diperoleh

Pendekatan ini banyak diterapkan dalam berbagai bidang seperti prediksi harga, deteksi anomali, klasifikasi email, hingga diagnosis medis.

Beberapa algoritma populer dalam *supervised learning*:

- **Linear Regression**: Digunakan untuk memodelkan hubungan linier antara variabel input dan output kontinu.
- **Decision Tree**: Membangun model berbentuk pohon keputusan yang mudah diinterpretasikan.
- **Support Vector Machine (SVM)**: Mencari hyperplane optimal yang memisahkan kelas secara maksimal.
- **K-Nearest Neighbors (KNN)**: Menentukan kelas suatu data berdasarkan mayoritas tetangga terdekat dalam ruang fitur.

- **Naive Bayes**: Menggunakan prinsip probabilistik dengan asumsi independensi antar fitur, sangat efektif dalam klasifikasi teks dan masalah berskala besar [5].

Contoh implementasi menggunakan `scikit-learn`:

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
```

Setelah proses pelatihan, model yang diperoleh dapat digunakan untuk melakukan prediksi terhadap data baru. Hasil prediksi kemudian dianalisis menggunakan metrik evaluasi seperti akurasi, presisi, recall, F1-score, atau mean squared error, bergantung pada jenis tugas (klasifikasi atau regresi).

8.4.2 Unsupervised Learning

Berbeda dengan *supervised learning*, metode *unsupervised learning* bekerja tanpa menggunakan label atau target output. Model dilatih hanya berdasarkan data input, dengan tujuan menemukan struktur, pola tersembunyi, atau keterkaitan alami dalam kumpulan data.

Unsupervised learning sangat berguna dalam tahap eksplorasi data awal, di mana belum tersedia anotasi atau klasifikasi sebelumnya, serta ketika ingin mengidentifikasi pengelompokan alami yang mungkin tidak langsung terlihat.

Contoh penerapan:

- Segmentasi pelanggan berdasarkan perilaku pembelian

- Pengelompokan dokumen ke dalam topik-topik utama
- Deteksi outlier dalam data keuangan

Beberapa algoritma populer dalam *unsupervised learning*:

- **K-Means Clustering**: Mengelompokkan data ke dalam k klaster berdasarkan kedekatan (misalnya jarak Euclidean). Algoritma ini bersifat iteratif dan mencari posisi centroid yang optimal.
- **Principal Component Analysis (PCA)**: Merupakan teknik reduksi dimensi yang memproyeksikan data ke dalam ruang berdimensi lebih rendah tanpa kehilangan informasi penting. Sangat berguna untuk visualisasi data berdimensi tinggi dan praproses sebelum pelatihan model.

Contoh implementasi K-Means Clustering:

```
from sklearn.cluster import KMeans

model = KMeans(n_clusters=3)
model.fit(X)
```

Setelah pelatihan, setiap data akan memiliki label klaster hasil prediksi model. Hasil ini kemudian dapat dianalisis lebih lanjut, misalnya dengan visualisasi dua dimensi menggunakan PCA, atau digunakan sebagai dasar dalam strategi bisnis, seperti personalisasi promosi atau pengembangan produk baru [6].

Catatan: Karena tidak ada label yang tersedia, evaluasi model *unsupervised* tidak sesederhana evaluasi klasifikasi

biasa. Metode seperti *silhouette score*, *inertia*, atau visualisasi klaster digunakan untuk menilai kualitas hasil pengelompokan.

8.4.3 Reinforcement Learning (RL)

Reinforcement Learning (RL) merupakan paradigma pembelajaran mesin di mana agen (agent) belajar berinteraksi dengan lingkungan (environment) melalui proses trial-and-error. Tujuan utama dari agen adalah memaksimalkan total *reward* yang diperoleh dari aksi-aksi yang diambil selama waktu tertentu.

Berbeda dari *supervised learning* yang menggunakan data berlabel, dan *unsupervised learning* yang tidak memiliki label sama sekali, RL mempelajari perilaku optimal berdasarkan umpan balik langsung dari lingkungan dalam bentuk *reward* (penghargaan) atau *penalty* (hukuman).

Komponen utama dalam Reinforcement Learning:

- **Agen (Agent)**: Entitas yang mengambil keputusan atau aksi.
- **Lingkungan (Environment)**: Dunia tempat agen berinteraksi.
- **Aksi (Action)**: Tindakan yang dapat diambil agen pada suatu keadaan.
- **Status (State)**: Representasi kondisi lingkungan saat ini.

- **Reward**: Nilai umpan balik yang diterima agen setelah mengambil suatu aksi.
- **Kebijakan (Policy)**: Strategi yang menentukan aksi mana yang akan diambil berdasarkan status.

Contoh penerapan RL dalam dunia nyata:

- **Robotika**: Agen belajar cara bergerak, menavigasi ruangan, atau menghindari rintangan.
- **Permainan (Game)**: Pengembangan agen yang dapat bermain catur, Go, atau permainan video seperti *Atari* secara mandiri.
- **Keuangan**: Agen beradaptasi dengan pasar untuk melakukan pengambilan keputusan investasi otomatis.

Catatan: RL sering kali dimodelkan menggunakan kerangka kerja *Markov Decision Process (MDP)*, dan digunakan bersama algoritma seperti Q-learning, Deep Q-Network (DQN), atau Proximal Policy Optimization (PPO) dalam konteks deep reinforcement learning [7].

Reinforcement Learning memiliki kekuatan dalam pengambilan keputusan berurutan, namun juga tantangan besar seperti eksplorasi vs eksploitasi, kestabilan pelatihan, serta kebutuhan akan simulasi lingkungan yang realistis dan terkontrol.

8.5 Proses Machine Learning

Pengembangan model pembelajaran mesin tidak hanya bergantung pada pemilihan algoritma, tetapi juga melibatkan

serangkaian tahapan sistematis [8] yang mencakup pemrosesan data, pelatihan, evaluasi, hingga implementasi model ke lingkungan nyata. Tahapan-tahapan ini membentuk sebuah alur kerja yang dikenal sebagai *machine learning pipeline*.

Berikut ini adalah tahapan utama dalam proses pengembangan model machine learning:

1. **Pengumpulan Data**: Tahap awal melibatkan pengumpulan data mentah dari berbagai sumber, seperti basis data internal, API, sensor IoT, atau hasil survei. Data yang diperoleh harus cukup merepresentasikan domain permasalahan yang ingin diselesaikan agar model yang dibangun relevan dan dapat diandalkan.

2. **Pembersihan dan Preprocessing**: Data mentah sering kali mengandung kekurangan seperti nilai kosong (missing values), duplikasi, atau outlier. Oleh karena itu, pembersihan diperlukan sebelum model dilatih. Selain itu, tahap ini juga mencakup transformasi fitur, seperti normalisasi, standardisasi, encoding variabel kategorikal, serta penanganan skala antar fitur.

3. **Split Data**: Dataset yang telah dibersihkan dibagi menjadi dua bagian utama: data pelatihan dan data pengujian. Rasio pemisahan yang umum digunakan meliputi 90:10, 80:20, dan 70:30. Rasio 90:10 digunakan ketika jumlah data sangat besar, sehingga data pengujian sebesar 10% sudah cukup representatif. Rasio 80:20 dan 70:30 lebih sering diterapkan dalam studi kasus umum,

guna memastikan bahwa model memiliki cukup data untuk belajar sambil tetap dapat dievaluasi dengan andal terhadap data yang belum pernah dilihat sebelumnya.

4. **Pemilihan Model**: Pada tahap ini, algoritma yang paling sesuai dipilih berdasarkan karakteristik data dan jenis masalah (klasifikasi, regresi, atau clustering). Sering kali beberapa model diuji secara paralel untuk dibandingkan performanya sebelum dipilih model final.

5. **Pelatihan Model**: Model dilatih menggunakan data pelatihan yang telah disiapkan. Selama pelatihan, parameter model disesuaikan agar dapat menangkap pola-pola yang tersembunyi dalam data. Kualitas pelatihan sangat bergantung pada representasi fitur dan kualitas data.

6. **Evaluasi Model**: Setelah model dilatih, performanya diuji menggunakan data pengujian. Beberapa metrik evaluasi yang umum digunakan antara lain *accuracy*, *precision*, *recall*, dan *F1-score* untuk klasifikasi, serta *mean squared error (MSE)* dan *R-squared* untuk regresi. Evaluasi ini bertujuan untuk mengukur seberapa baik model dapat menggeneralisasi terhadap data baru.

7. **Prediksi dan Deployment**: Model yang telah dievaluasi dan dinyatakan layak, kemudian digunakan untuk membuat prediksi terhadap data aktual (real-world data). Pada tahap ini, model dapat diintegrasikan ke dalam sistem produksi dan diakses melalui API, antarmu-

ka pengguna, atau otomatisasi proses. Deployment juga mencakup pemantauan model secara berkala untuk menangani penurunan performa akibat perubahan data (data drift).

Contoh implementasi pemisahan data dengan `scikit-learn`:

```
from sklearn.model_selection import train_test_split

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

Kode di atas memisahkan data menjadi 70% untuk pelatihan dan 30% untuk pengujian, dengan parameter `random_state` ditetapkan agar hasil pembagian data bersifat reprodusibel.

8.6 Overfitting vs Underfitting

Salah satu tantangan utama dalam pengembangan model pembelajaran mesin adalah mencapai keseimbangan antara akurasi model pada data pelatihan dan kemampuannya melakukan generalisasi terhadap data baru. Dua masalah ekstrem yang sering muncul adalah **overfitting** dan **underfitting**.

- **Overfitting**: Terjadi ketika model terlalu kompleks atau terlalu menyesuaikan diri terhadap data pelatihan, hingga menangkap noise atau fluktuasi acak sebagai pola yang signifikan. Akibatnya, performa model sangat baik pada data pelatihan, tetapi menurun drastis pada data pengujian karena generalisasi yang buruk.

- **Underfitting**: Terjadi ketika model terlalu sederhana atau tidak memiliki kapasitas cukup untuk menangkap hubungan atau pola yang terdapat dalam data. Kondisi ini menyebabkan performa rendah baik pada data pelatihan maupun pengujian.

Fenomena ini dapat diilustrasikan melalui kurva pembelajaran [9]: dalam kasus overfitting, akurasi pelatihan tinggi tetapi akurasi validasi menurun; sebaliknya, underfitting ditandai oleh akurasi rendah pada kedua jenis data.

Beberapa pendekatan yang dapat digunakan untuk mengatasi overfitting:

- **Regularisasi (L1/L2)**: Menambahkan penalti terhadap kompleksitas model dalam fungsi kerugian. Regularisasi L1 (Lasso) cenderung menghasilkan model yang lebih sederhana dengan mengeliminasi beberapa parameter, sedangkan L2 (Ridge) menekan nilai parameter menjadi kecil.
- **Cross-validation**: Metode validasi silang digunakan untuk mengevaluasi performa model pada berbagai subset data, sehingga dapat mendeteksi jika model mulai terlalu cocok terhadap subset tertentu.
- **Dropout**: Digunakan dalam arsitektur jaringan saraf tiruan (neural network), dengan secara acak menonaktifkan sejumlah neuron selama pelatihan untuk mencegah ketergantungan berlebih antar node.

Ilustrasi Deskriptif:

Sebuah grafik menunjukkan dua kurva: satu untuk akurasi pelatihan dan satu lagi untuk akurasi validasi. Pada kasus overfitting, akurasi pelatihan mendekati 100% sedangkan akurasi validasi menurun setelah titik tertentu. Hal ini menandakan bahwa model tidak mampu melakukan generalisasi terhadap data yang tidak dilihat sebelumnya.

8.7 Studi Kasus: Prediksi Kelulusan Mahasiswa

Masalah: Diberikan data seperti jumlah SKS, IPK, tingkat kehadiran, apakah mahasiswa lulus atau tidak?

Langkah:

1. Load data dan lakukan preprocessing.
2. Split menjadi X dan y.
3. Gunakan Decision Tree sebagai model.
4. Latih model dan evaluasi menggunakan confusion matrix.

```
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier()
model.fit(X_train, y_train)
```

Insight: Model bisa digunakan oleh universitas untuk mendeteksi mahasiswa yang perlu intervensi akademik [10].

8.8 Perbandingan Beberapa Algoritma Dasar

Setiap algoritma machine learning memiliki karakteristik, kelebihan, dan keterbatasan masing-masing. Pemilihan algoritma yang tepat sangat bergantung pada jenis data, tujuan analisis, serta konteks masalah yang dihadapi. Tabel berikut menyajikan perbandingan beberapa algoritma dasar yang umum digunakan dalam berbagai tugas pembelajaran mesin:

Algoritma	Tipe	Kelebihan	Kekurangan
Linear Regression	Supervised	Sederhana, interpretatif	Tidak cocok untuk pola kompleks
Decision Tree	Supervised	Mudah dipahami, cepat	Mudah overfitting
KNN	Supervised	Non-parametrik, mudah diimplementasi	Lambat untuk dataset besar
Naive Bayes	Supervised	Cepat, efektif untuk data teks	Asumsi independensi fitur sering tidak terpenuhi
K-Means	Unsupervised	Efisien untuk clustering	Perlu tentukan jumlah cluster

Tabel 3: Perbandingan Beberapa Algoritma Machine Learning Dasar

8.9 Tantangan dalam Penerapan Machine Learning

Meskipun machine learning menawarkan berbagai solusi cerdas dalam pengolahan data dan automasi pengambilan keputusan, implementasinya di dunia nyata tidak terlepas dari sejumlah tantangan teknis maupun non-teknis. Beberapa isu utama yang sering dihadapi antara lain:

- **Imbalanced Data** [11]: Terjadi ketika distribusi kelas target tidak merata, misalnya dalam kasus deteksi penipuan atau diagnosis penyakit langka. Model cenderung bias terhadap kelas mayoritas dan mengabaikan kelas minoritas yang justru sering menjadi fokus utama. Strategi seperti *oversampling*, *undersampling*, dan penggunaan metrik evaluasi khusus (seperti F1-score) dibutuhkan dalam situasi ini.
- **Missing Value dan Noise**: Data yang tidak lengkap atau mengandung kesalahan dapat menurunkan kualitas pelatihan model. Penanganan yang hati-hati diperlukan melalui teknik imputasi, pembersihan data, atau penggunaan model yang robust terhadap outlier.
- **Scalability**: Ketika ukuran data sangat besar (big data), banyak algoritma tradisional tidak dapat dijalankan secara efisien. Diperlukan pendekatan yang mendukung komputasi paralel atau distribusi, seperti penggunaan

framework *Spark MLlib* atau *dask* untuk mempercepat proses pelatihan dan prediksi.

- **Explainability**: Model yang sangat kompleks, seperti deep learning atau ensemble methods, sering kali dianggap sebagai *black box*, sehingga sulit dijelaskan kepada pihak non-teknis atau regulator. Tantangan ini mendorong berkembangnya bidang *Explainable AI (XAI)*, dengan teknik seperti SHAP (SHapley Additive exPlanations) dan LIME (Local Interpretable Model-agnostic Explanations) [12] untuk meningkatkan transparansi dan kepercayaan terhadap model.

8.10 LATIHAN / TUGAS AKHIR BAB 8

1. **[Uraian]** Jelaskan perbedaan antara supervised, unsupervised, dan reinforcement learning serta berikan contoh penerapannya di dunia nyata.
2. **[Uraian Visual]** Gambar workflow machine learning dari data mentah hingga deployment model.
3. **[Coding]**

   Gunakan dataset `iris.csv`:

   - Lakukan train-test split
   - Bangun model klasifikasi (Decision Tree)
   - Evaluasi dengan confusion matrix dan F1-score

4. **[Studi Kasus]** Buat sistem prediksi kelulusan mahasiswa menggunakan minimal 4 fitur input. Jelaskan proses modeling hingga evaluasi.

5. **[Studi Kasus]** Identifikasi risiko overfitting dalam proyek machine learning berbasis text classification. Bagaimana Anda menghindarinya?

Daftar Pustaka

[1] T. M. Mitchell, *Machine learning*. McGraw-Hill Education, 1997.

[2] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[3] M. I. Jordan **and** T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, **jourvol** 349, **number** 6245, **pages** 255–260, 2015.

[4] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[5] T. Hastie, R. Tibshirani **and** J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2 **edition**. Springer, 2009.

[6] G. James, D. Witten, T. Hastie **and** R. Tibshirani, *An introduction to statistical learning*. Springer, 2013.

[7] R. S. Sutton **and** A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[8] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2 **edition**. O'Reilly Media, 2019.

[9] F. Chollet, *Deep learning with Python*. Manning Publications Co., 2018.

[10] S. Raschka **and** V. Mirjalili, *Python machine learning*. Packt Publishing Ltd, 2019.

[11] A. Fernández, S. Garcia, M. Galar, R. C. Prati, B. Krawczyk **and** F. Herrera, "SMOTE for learning from imbalanced data: progress and challenges," *Progress in Artificial Intelligence*, **jourvol** 7, **number** 1, **pages** 1–36, 2018.

[12] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020.

9 KLASIFIKASI DAN REGRESI

9.1 Tujuan Pembelajaran

Setelah menyelesaikan bab ini, mahasiswa diharapkan mampu:

1. Membedakan antara tugas klasifikasi dan regresi dalam supervised learning.
2. Memahami berbagai algoritma populer untuk klasifikasi dan regresi.
3. Mengimplementasikan model klasifikasi dan regresi menggunakan Scikit-learn.
4. Mengevaluasi performa model menggunakan metrik yang sesuai.
5. Menganalisis hasil dan memahami interpretasi model secara praktis.

9.2 Perbedaan Klasifikasi dan Regresi

Dalam dunia *ilmu data* dan *pembelajaran mesin* (*machine learning*), terdapat dua jenis utama dari tugas pemodelan

prediktif, yaitu **klasifikasi** dan **regresi** [1], [2]. Meskipun keduanya bertujuan untuk memprediksi suatu nilai berdasarkan data input, terdapat perbedaan mendasar dalam hal jenis output yang dihasilkan, tujuan, serta algoritma dan metrik evaluasi yang digunakan.

Klasifikasi adalah proses untuk memetakan data input ke dalam *kategori* atau *kelas* tertentu. Masalah klasifikasi muncul ketika output yang ingin diprediksi bersifat *diskrit* atau *kategorikal*. Contoh klasik dari tugas klasifikasi adalah memprediksi apakah seorang siswa akan *lulus* atau *tidak lulus* ujian berdasarkan data nilai tugas, kehadiran, dan partisipasi. Hasil prediksi dari model klasifikasi biasanya berupa label seperti "ya" atau "tidak", atau label numerik seperti 0 dan 1.

Sebaliknya, **regresi** digunakan ketika variabel target (label) bersifat *kontinu*. Artinya, nilai yang ingin diprediksi berada pada rentang numerik yang luas. Contoh umum dari tugas regresi adalah memprediksi *harga rumah* berdasarkan ukuran, lokasi, dan jumlah kamar. Hasil dari model regresi bukanlah kategori, melainkan angka yang bersifat kuantitatif dan terus menerus.

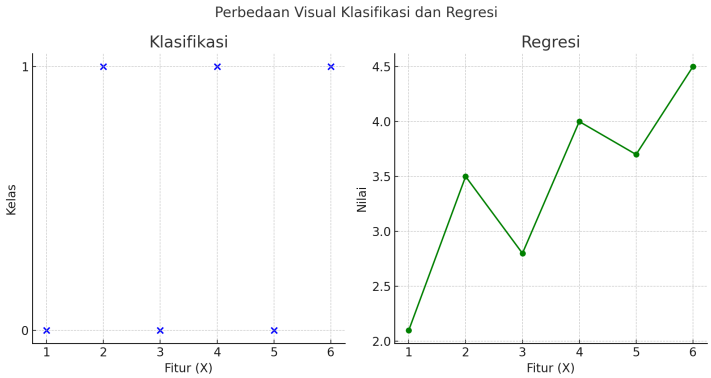
Meskipun klasifikasi dan regresi sering kali melibatkan proses pra-pemrosesan data yang serupa, seperti pembagian data, pembersihan, dan normalisasi, **pendekatan algoritma dan cara evaluasi hasilnya berbeda**. Klasifikasi menggunakan metrik evaluasi seperti *akurasi*, *precision*, *recall*, dan *F1-score*. Sementara itu, regresi menggunakan metrik seperti *MAE (Mean Absolute Error)*, *RMSE (Root Mean Squared*

Error), dan R^2 (koefisien determinasi).

Untuk memberikan perbandingan yang lebih jelas, berikut adalah tabel yang merangkum perbedaan utama antara klasifikasi dan regresi:

Aspek	**Klasifikasi**	**Regresi**
Tujuan	Memprediksi kategori/kelas	Memprediksi nilai kontinu
Output	Diskrit (cth: "lulus"/"tidak lulus")	Kontinu (cth: "harga rumah")
Contoh Algoritma	Decision Tree, Logistic Regression, SVM	Linear Regression, Random Forest Regressor
Metrik Evaluasi	Akurasi, Precision, Recall, F1	MAE, RMSE, R^2

Tabel 4: Perbandingan antara Klasifikasi dan Regresi



Gambar 3: Visualisasi perbedaan antara klasifikasi dan regresi.

9.3 Algoritma Klasifikasi

Dalam pembelajaran mesin, terdapat berbagai algoritma yang dapat digunakan untuk menyelesaikan masalah klasifikasi. Berikut ini adalah tiga algoritma populer beserta penjelasan dan contoh implementasi sederhananya.

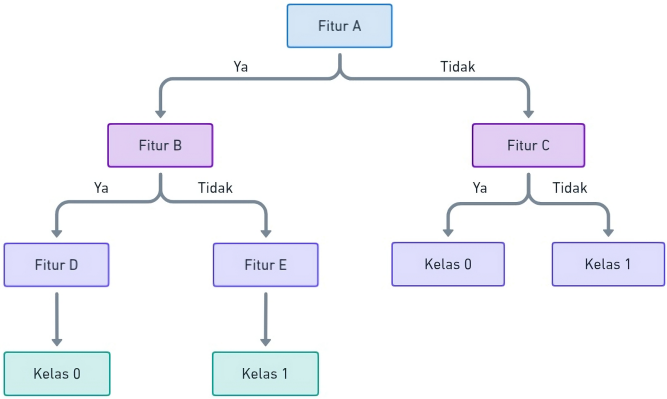
9.3.1 Decision Tree Classifier

Decision Tree merupakan algoritma yang menggunakan struktur pohon keputusan untuk mengklasifikasikan data. Algoritma ini bekerja dengan membagi dataset berdasarkan fitur yang memberikan *informasi maksimum*, biasanya diukur dengan *entropy* atau *Gini index* [2]. Setiap node dalam pohon mewakili sebuah fitur, dan setiap cabang mewakili aturan keputusan, sedangkan daun (*leaf*) mewakili hasil klasifikasi.

- Kelebihan: Mudah dipahami, visualisasinya intuitif, tidak memerlukan normalisasi fitur.
- Kekurangan: Rentan terhadap overfitting jika pohon terlalu dalam.

Contoh kode Python:

```
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier()
model.fit(X_train, y_train)
```



Gambar 4: Ilustrasi algoritma Decision Tree Classifier. Setiap node mewakili fitur, cabang menunjukkan hasil keputusan (ya/tidak), dan daun mewakili hasil akhir klasifikasi seperti Kelas 0 atau Kelas 1.

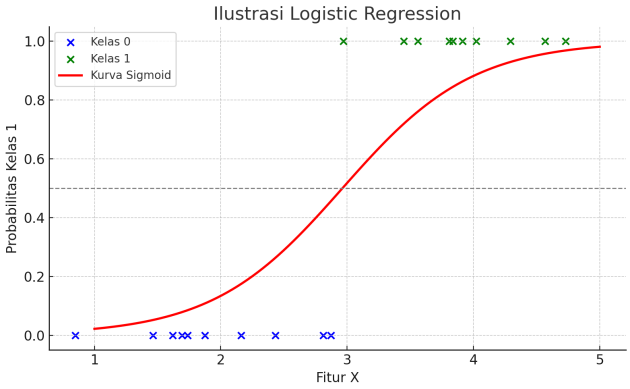
9.3.2 Logistic Regression

Logistic Regression adalah model statistik yang digunakan untuk memprediksi probabilitas dari suatu kelas. Meskipun namanya mengandung kata ”regression”, model ini digunakan untuk klasifikasi biner atau multikelas [3]. Model ini menggunakan fungsi *sigmoid* untuk mengubah output linear menjadi probabilitas antara 0 dan 1.

- Kelebihan: Cepat, efisien untuk klasifikasi linier, hasilnya mudah diinterpretasi.
- Kekurangan: Tidak cocok untuk relasi non-linier tanpa transformasi fitur.

Contoh kode Python:

```python
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
```



Gambar 5: Ilustrasi algoritma Logistic Regression. Titik-titik biru dan hijau menunjukkan dua kelas yang berbeda. Kurva merah menunjukkan fungsi sigmoid yang memetakan fitur input ke dalam probabilitas kelas. Garis putus-putus di $y = 0.5$ menjadi batas keputusan klasifikasi.

9.3.3 K-Nearest Neighbors (KNN)

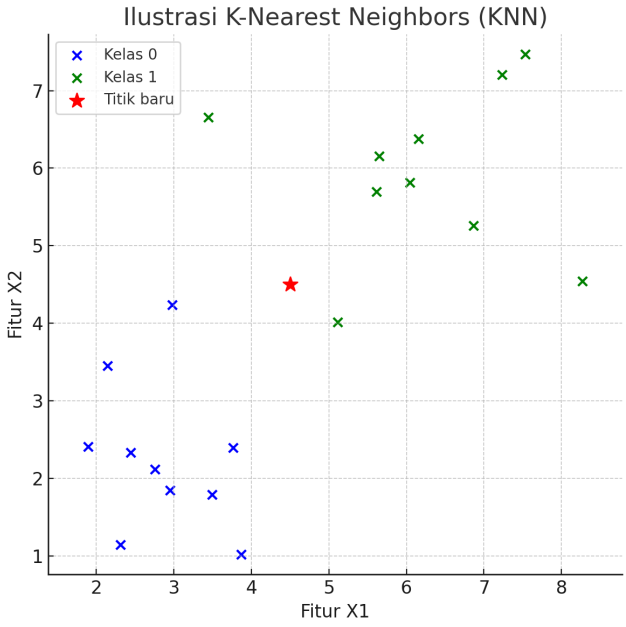
K-Nearest Neighbors (KNN) adalah algoritma berbasis instance-based learning yang mengklasifikasikan data baru berdasarkan mayoritas label dari *k* tetangga terdekatnya dalam ruang fitur. Jarak biasanya dihitung menggunakan Euclidean distance [1].

- Kelebihan: Sederhana, tidak memerlukan pelatihan model eksplisit.

- Kekurangan: Lambat untuk dataset besar, sensitif terhadap skala fitur.

Contoh kode Python:

```python
from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier(n_neighbors=5)
```



Gambar 6: Ilustrasi algoritma K-Nearest Neighbors (KNN). Titik-titik biru dan hijau mewakili dua kelas yang berbeda. Titik merah berbentuk bintang merupakan data baru yang ingin diklasifikasikan, dengan garis putus-putus menunjukkan jarak ke tetangga terdekat. Kelas mayoritas dari tetangga tersebut akan menjadi prediksi label untuk titik ini.

9.4 Algoritma Regresi

Regresi adalah pendekatan dalam pembelajaran mesin yang digunakan untuk memodelkan hubungan antara variabel input (independen) dan variabel output (dependen) yang bersifat kontinu. Tujuan dari algoritma regresi adalah untuk memprediksi nilai numerik yang akurat berdasarkan pola dalam data.

Berikut beberapa algoritma regresi yang umum digunakan:

9.4.1 Linear Regression

Linear Regression adalah salah satu algoritma paling sederhana dan paling umum dalam regresi. Model ini berasumsi bahwa terdapat hubungan linier antara variabel input dan output [2]. Persamaan dasar dari model regresi linier adalah:

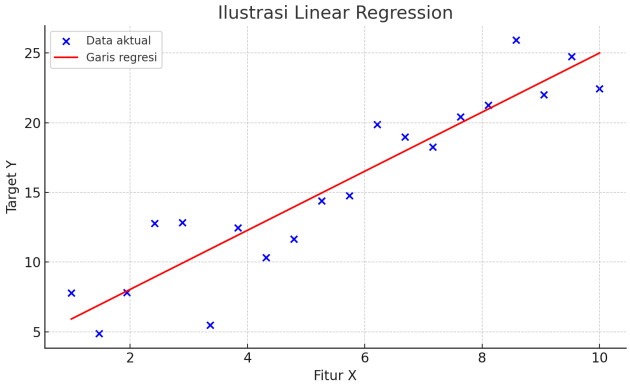
$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \varepsilon$$

di mana β_0 adalah intercept, β_i adalah koefisien regresi untuk masing-masing fitur x_i, dan ε adalah error.

- Kelebihan: Mudah dipahami dan diimplementasikan, hasilnya mudah diinterpretasi.
- Kekurangan: Tidak cocok untuk data dengan relasi non-linier, rentan terhadap outlier.

Contoh kode Python:

```python
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
```



Gambar 7: Ilustrasi algoritma Linear Regression. Titik-titik biru menunjukkan data aktual, sementara garis merah merupakan hasil prediksi model yang memodelkan hubungan linier antara fitur dan target.

9.4.2 Ridge dan Lasso Regression

Ridge Regression dan **Lasso Regression** merupakan variasi dari linear regression yang menggunakan teknik *regularisasi* untuk menghindari *overfitting*.

- Ridge Regression menambahkan penalti L_2 terhadap besar koefisien ($\sum \beta_i^2$).
- Lasso Regression menambahkan penalti L_1 yang dapat

menyusutkan beberapa koefisien menjadi nol, sehingga juga dapat digunakan untuk seleksi fitur.

Regularisasi membantu meningkatkan kemampuan generalisasi model pada data yang belum pernah dilihat sebelumnya.

Contoh kode Python (Ridge):

```
from sklearn.linear_model import Ridge

model = Ridge(alpha=1.0)
model.fit(X_train, y_train)
```

Contoh kode Python (Lasso):

```
from sklearn.linear_model import Lasso

model = Lasso(alpha=0.1)
model.fit(X_train, y_train)
```

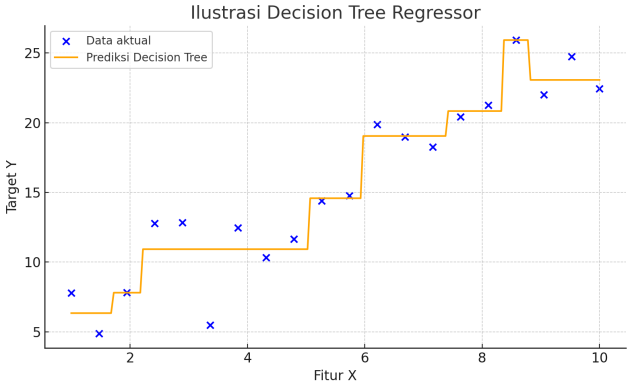
9.4.3 Decision Tree Regressor

Decision Tree Regressor adalah algoritma regresi non-linier yang menggunakan struktur pohon untuk memetakan input ke output kontinu. Algoritma ini bekerja dengan membagi data ke dalam kelompok-kelompok kecil berdasarkan fitur, kemudian memberikan prediksi rata-rata nilai target di setiap kelompok (leaf node).

- Kelebihan: Dapat memodelkan relasi non-linier dan interaksi antar fitur.
- Kekurangan: Rentan terhadap overfitting jika tidak dilakukan pemangkasan (pruning).

Contoh kode Python:

```python
from sklearn.tree import DecisionTreeRegressor

model = DecisionTreeRegressor()
model.fit(X_train, y_train)
```



Gambar 8: Ilustrasi algoritma Decision Tree Regressor. Titik-titik biru menunjukkan data aktual, sedangkan garis oranye menunjukkan hasil prediksi model pohon keputusan yang membagi ruang input ke dalam segmen-segmen dengan nilai konstan.

9.5 Evaluasi Model

Setelah membangun model klasifikasi atau regresi, langkah penting berikutnya adalah melakukan **evaluasi kinerja model**. Evaluasi ini bertujuan untuk mengukur seberapa baik model memprediksi data yang belum pernah dilihat sebelumnya (data uji). Pendekatan evaluasi berbeda antara tugas klasifikasi dan regresi.

9.5.1 Evaluasi Model Klasifikasi

Untuk tugas klasifikasi, kita menilai seberapa akurat model mengklasifikasikan data ke dalam kelas yang benar. Beberapa metrik yang umum digunakan adalah sebagai berikut:

- **Accuracy (Akurasi)** Mengukur proporsi prediksi yang benar terhadap total jumlah data. Dirumuskan sebagai:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

di mana **TP** (True Positive), **TN** (True Negative), **FP** (False Positive), dan **FN** (False Negative) diperoleh dari confusion matrix.

- **Precision dan Recall Precision** mengukur proporsi prediksi positif yang benar-benar positif:

$$Precision = \frac{TP}{TP + FP}$$

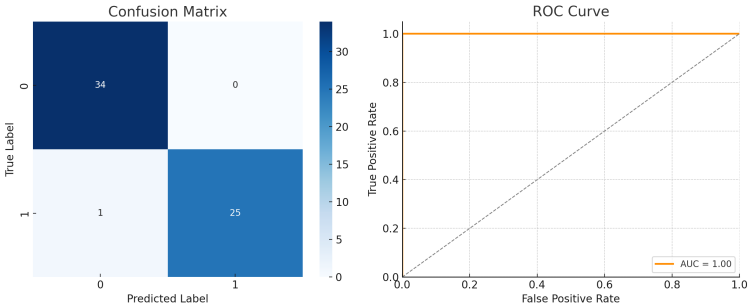
Recall mengukur proporsi data positif yang berhasil diprediksi oleh model:

$$Recall = \frac{TP}{TP + FN}$$

Keduanya sering digunakan bersama dalam metrik *F1-score* [1] untuk menyeimbangkan precision dan recall.

- **Confusion Matrix** Merupakan tabel yang menggambarkan kinerja model klasifikasi dengan menunjukkan jumlah TP, TN, FP, dan FN secara visual.

- **ROC-AUC (Receiver Operating Characteristic - Area Under Curve)** Digunakan untuk mengevaluasi model klasifikasi biner. ROC-AUC menunjukkan kemampuan model membedakan antara dua kelas [2]. Nilai AUC mendekati 1 menandakan performa yang baik.



Gambar 9: Ilustrasi evaluasi model klasifikasi. Kiri: Confusion Matrix menunjukkan jumlah prediksi benar dan salah. Kanan: ROC Curve menunjukkan kinerja model dalam membedakan dua kelas dengan nilai AUC sebagai indikator performa.

9.5.2 Evaluasi Model Regresi

Untuk tugas regresi, model dievaluasi berdasarkan kesalahan prediksi terhadap nilai sebenarnya. Beberapa metrik populer yang digunakan adalah:

- **Mean Absolute Error (MAE)** Mengukur rata-rata selisih absolut antara nilai prediksi dan nilai aktual.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- **Root Mean Squared Error (RMSE)** Mengukur akar dari rata-rata kuadrat kesalahan. Memberikan penalti lebih besar terhadap kesalahan besar dibanding MAE.

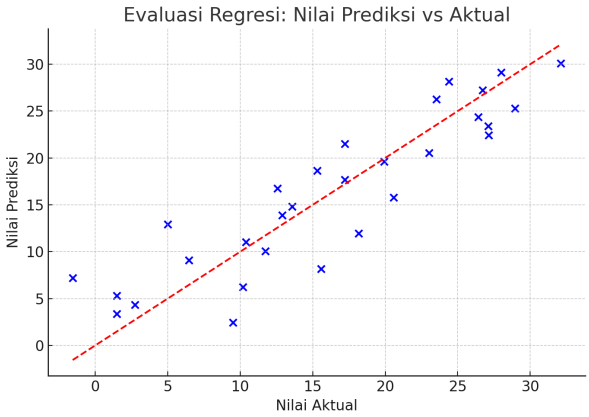
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- **R-squared (R^2)** Mengukur seberapa baik model menjelaskan variasi pada data. Nilai R^2 berkisar antara 0 hingga 1, dengan nilai lebih tinggi menunjukkan model yang lebih baik.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

Contoh kode Python:

```python
from sklearn.metrics import mean_squared_error, r2_score

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```



Gambar 10: Evaluasi model regresi. Titik-titik biru menunjukkan hubungan antara nilai aktual dan prediksi. Garis merah putus-putus menunjukkan prediksi sempurna. Semakin dekat titik-titik ke garis, semakin baik performa model.

9.6 Studi Kasus Klasifikasi: Prediksi Kelulusan Mahasiswa

1. **Fitur:** IPK, kehadiran, nilai tugas, partisipasi kelas
2. **Label:** Lulus / Tidak lulus

```
from sklearn.metrics import confusion_matrix,
    classification_report

y_pred = model.predict(X_test)

print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

Insight: Dosen dapat menggunakan model ini untuk memprediksi risiko mahasiswa drop out.

9.7 Studi Kasus Regresi: Prediksi Harga Rumah

1. **Fitur:** Luas bangunan, lokasi, jumlah kamar
2. **Target:** Harga dalam juta rupiah

```
model = LinearRegression()
model.fit(X_train, y_train)

print("R-squared:", model.score(X_test, y_test))
```

Insight: Model dapat membantu agen properti memberi harga lebih tepat sasaran.

9.8 Visualisasi Hasil Model

9.8.1 Untuk Regresi:

```
plt.scatter(y_test, y_pred)
plt.xlabel("Actual")
plt.ylabel("Predicted")
plt.title("Prediksi vs Realita Harga Rumah")
```

9.8.2 Untuk Klasifikasi:

```
from sklearn.metrics import plot_confusion_matrix

plot_confusion_matrix(model, X_test, y_test)
```

9.9 Tantangan dalam Klasifikasi & Regresi

Dalam penerapan algoritma klasifikasi dan regresi pada data dunia nyata, terdapat berbagai tantangan yang dapat mempengaruhi performa model. Pemahaman terhadap tantangan ini penting untuk memilih strategi pemodelan yang tepat dan meningkatkan kualitas prediksi.

9.9.1 Data Tidak Seimbang (Imbalanced Classes)

Dalam tugas klasifikasi, sering kali ditemukan bahwa jumlah data pada satu kelas jauh lebih besar dibanding kelas lainnya. Kondisi ini disebut **data tidak seimbang** (imbalanced classes), dan dapat menyebabkan model bias terhadap kelas mayoritas.

Solusi yang umum digunakan:

- **Oversampling:** Menambah jumlah data pada kelas minoritas, misalnya menggunakan teknik *SMOTE* (Synthe-

tic Minority Over-sampling Technique) [4] untuk membuat data sintetis.

- **Undersampling:** Mengurangi jumlah data pada kelas mayoritas agar seimbang dengan kelas minoritas.

9.9.2 Multikolinearitas Antar Fitur

Multikolinearitas terjadi ketika dua atau lebih fitur saling berkorelasi sangat tinggi. Hal ini dapat menyebabkan ketidakstabilan dalam estimasi koefisien regresi linier dan menurunkan interpretabilitas model.

Solusi:

- **Principal Component Analysis (PCA):** Mengurangi dimensi fitur dengan menggabungkan fitur yang berkorelasi menjadi komponen utama.
- **Lasso Regression:** Model regresi yang menggunakan regularisasi L_1 untuk menyusutkan koefisien fitur tidak penting menjadi nol, secara efektif melakukan seleksi fitur [2].

9.9.3 Outlier dalam Regresi

Outlier adalah data yang jauh berbeda dari sebagian besar data lain. Dalam regresi, outlier dapat sangat memengaruhi garis regresi, khususnya pada model linier.

Solusi:

- Gunakan model yang **robust terhadap outlier**, seperti **Random Forest Regressor** [5], karena model berbasis

pohon tidak terlalu terpengaruh oleh nilai ekstrem.

- Lakukan deteksi dan penanganan outlier melalui analisis distribusi atau metode statistik seperti Z-score dan IQR.

9.9.4 Kelebihan Fitur yang Tidak Relevan

Fitur yang tidak relevan atau redundan dapat menyebabkan overfitting, memperlambat pelatihan model, dan menurunkan performa prediksi.

Solusi:

- Lakukan **feature selection** untuk memilih subset fitur yang paling informatif, baik secara manual berdasarkan pemahaman domain, maupun menggunakan metode otomatis seperti *SelectKBest*, *Recursive Feature Elimination* (RFE), atau berdasarkan feature importance.

9.10 Perbandingan Model

Lakukan evaluasi pada beberapa model dan bandingkan hasilnya menggunakan metrik yang sama.

```
models = [LogisticRegression(), DecisionTreeClassifier(),
    KNeighborsClassifier()]

for model in models:
    model.fit(X_train, y_train)

```

```
6    print(model.__class__.__name__, model.score(X_test, y_test
     ))
```

9.11 LATIHAN / TUGAS AKHIR BAB 9

1. **[Uraian]** Jelaskan kapan harus menggunakan klasifikasi dan kapan regresi, lengkap dengan contoh nyata masing-masing.
2. **[Uraian Visual]** Gambar skema perbandingan Decision Tree untuk klasifikasi dan regresi.
3. **[Coding]**
   - Gunakan dataset rumah.csv
   - Bangun model regresi linear
   - Evaluasi dengan MAE dan R^2
   - Visualisasikan hasil prediksi vs aktual
4. **[Studi Kasus]** Klasifikasikan apakah mahasiswa akan mendapatkan beasiswa berdasarkan IPK, kehadiran, dan prestasi. Jelaskan model, evaluasi, dan insight.
5. **[Studi Kasus]** Gunakan dataset dari Kaggle untuk membangun model prediksi harga mobil. Bandingkan performa Linear Regression dan Decision Tree Regressor.

Daftar Pustaka

[1] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2 **edition**. O'Reilly Media, 2019.

[2] T. Hastie, R. Tibshirani **and** J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2 **edition**. Springer, 2009.

[3] M. Kuhn **and** K. Johnson, *Applied Predictive Modeling*. Springer, 2013.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall **and** W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, **jourvol** 16, **pages** 321–357, 2002.

[5] L. Breiman, "Random Forests," *Machine Learning*, **jourvol** 45, **number** 1, **pages** 5–32, 2001.

10 CLUSTERING DAN DIMENSIONALITY REDUCTION

10.1 Tujuan Pembelajaran

Setelah menyelesaikan bab ini, mahasiswa diharapkan mampu:

1. Memahami konsep dasar dari clustering dan dimensionality reduction.
2. Mengimplementasikan algoritma clustering seperti K-Means dan DBSCAN.
3. Menggunakan Principal Component Analysis (PCA) untuk mereduksi dimensi fitur.
4. Memvisualisasikan hasil clustering dan transformasi data.
5. Menerapkan teknik ini dalam kasus nyata seperti segmentasi pelanggan atau deteksi anomali.

10.2 Apa Itu Clustering?

Clustering adalah salah satu metode dalam *unsupervised learning* [1], [2] yang digunakan untuk mengelompokkan da-

ta ke dalam beberapa grup atau *cluster* berdasarkan kemiripan atau kedekatan antar data. Tidak seperti klasifikasi dan regresi yang bergantung pada label atau target output, clustering tidak memerlukan label apa pun. Tujuan dari clustering adalah untuk menemukan struktur atau pola tersembunyi dalam data secara otomatis.

Ciri khas clustering:

- Tidak menggunakan label atau target (unsupervised).
- Data dalam satu cluster memiliki kemiripan yang tinggi.
- Data antar cluster memiliki perbedaan yang signifikan.

Dalam konteks ini, model tidak ”dilatih” untuk memprediksi sesuatu yang spesifik, tetapi lebih kepada *mengorganisasi* data berdasarkan hubungan atau karakteristik alami dalam data tersebut. Proses ini sangat berguna dalam eksplorasi data, segmentasi pasar, atau sistem rekomendasi.

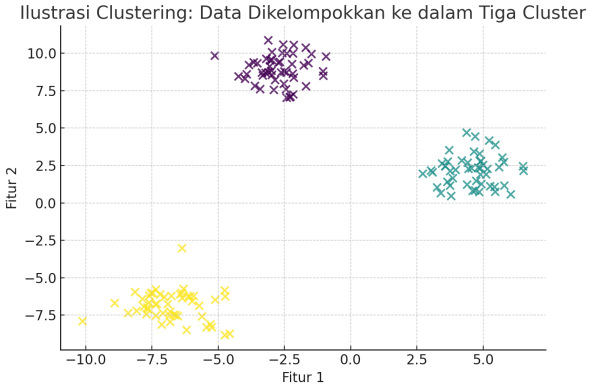
Contoh nyata:

Misalnya dalam bisnis e-commerce, kita dapat menggunakan clustering untuk *mengelompokkan pelanggan berdasarkan pola belanja mereka*. Pelanggan yang sering membeli produk elektronik akan tergabung dalam cluster yang berbeda dibanding pelanggan yang lebih sering membeli produk pakaian. Dengan demikian, perusahaan dapat menyesuaikan strategi pemasaran dan rekomendasi produk secara lebih personal.

Ilustrasi konsep:

- Pelanggan A dan B sering membeli gadget, maka mereka masuk dalam cluster yang sama.
- Pelanggan C dan D lebih suka produk rumah tangga, maka mereka dikelompokkan dalam cluster berbeda.

Clustering menjadi dasar penting dalam analisis segmentasi dan membantu pengambilan keputusan strategis berdasarkan pola perilaku data yang muncul secara alami.



Gambar 11: Ilustrasi clustering. Data yang memiliki kemiripan fitur dikelompokkan ke dalam cluster yang sama. Setiap warna menunjukkan cluster yang berbeda.

10.3 Algoritma Clustering

Beberapa algoritma populer digunakan dalam proses clustering. Masing-masing memiliki pendekatan dan keunggulan tersendiri dalam mengelompokkan data. Dua algoritma

yang paling sering digunakan adalah **K-Means Clustering** dan **DBSCAN**.

10.3.1 K-Means Clustering

K-Means adalah algoritma clustering berbasis centroid [3] yang bekerja dengan membagi data ke dalam k cluster berdasarkan jarak terdekat ke pusat cluster (*centroid*).

Langkah-langkah utama algoritma K-Means:

1. Tentukan jumlah cluster (k).
2. Inisialisasi centroid secara acak.
3. Hitung jarak antara tiap titik data ke centroid.
4. Tetapkan setiap titik data ke cluster dengan centroid terdekat.
5. Hitung ulang posisi centroid berdasarkan rata-rata titik-titik dalam cluster.
6. Ulangi langkah 3–5 hingga konvergen (centroid tidak berubah).

Tujuan utama: Meminimalkan jumlah total jarak kuadrat antara titik-titik data dan centroid-nya, atau disebut *within-cluster sum of squares (WCSS)* [4].

Contoh kode Python:

```
from sklearn.cluster import KMeans

model = KMeans(n_clusters=3)
model.fit(X) labels = model.labels_
```

Visualisasi hasil:

```
import matplotlib.pyplot as plt

plt.scatter(X[:, 0], X[:, 1], c=labels)
plt.title("Hasil Clustering K-Means")
```

10.3.2 DBSCAN (Density-Based Spatial Clustering)

DBSCAN adalah algoritma clustering berbasis kepadatan [5] yang membentuk cluster berdasarkan *density* atau jumlah titik yang berdekatan.

Karakteristik utama DBSCAN:

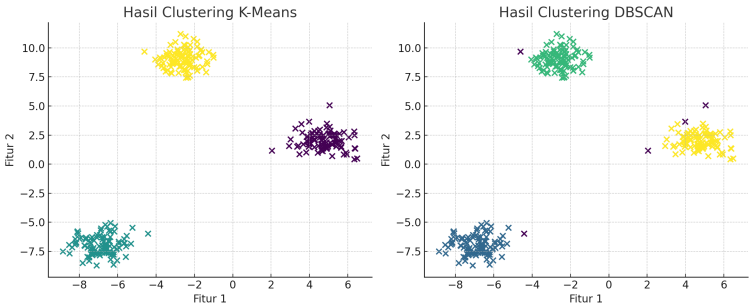
- Tidak perlu menentukan jumlah cluster di awal.
- Dapat mengidentifikasi outlier sebagai titik yang tidak termasuk ke dalam cluster mana pun.
- Cocok untuk data dengan bentuk cluster yang tidak beraturan atau tidak linier.

Parameter utama:

- `eps`: Jarak maksimum dua titik untuk dianggap dalam satu lingkungan (neighborhood).
- `min_samples`: Minimum jumlah titik dalam `eps` untuk membentuk cluster.

Contoh kode Python:

```
from sklearn.cluster import DBSCAN

model = DBSCAN(eps=0.5, min_samples=5) model.fit(X)
```



Gambar 12: Perbandingan hasil clustering menggunakan K-Means (kiri) dan DBSCAN (kanan). K-Means mengelompokkan berdasarkan centroid, sedangkan DBSCAN berdasarkan kepadatan titik.

10.4 Evaluasi Clustering

Berbeda dengan klasifikasi dan regresi yang memiliki label target sebagai dasar evaluasi, pada clustering kita tidak memiliki label yang diketahui sebelumnya. Oleh karena itu, evaluasi performa model clustering dilakukan dengan cara mengukur **kualitas internal** dari hasil pengelompokan.

Berikut ini adalah dua metrik evaluasi yang umum digunakan dalam clustering:

10.4.1 Silhouette Score

Silhouette Score adalah metrik evaluasi yang mengukur seberapa baik setiap titik data cocok dengan cluster-nya sendiri [6] dibandingkan dengan cluster lain. Nilai Silhouette berada dalam rentang -1 hingga 1:

- Nilai mendekati 1 menunjukkan bahwa titik berada sangat cocok dalam clusternya.
- Nilai mendekati 0 menunjukkan bahwa titik berada pada batas antara dua cluster.
- Nilai negatif menunjukkan bahwa titik mungkin ditempatkan dalam cluster yang salah.

Silhouette Score menggabungkan dua informasi:

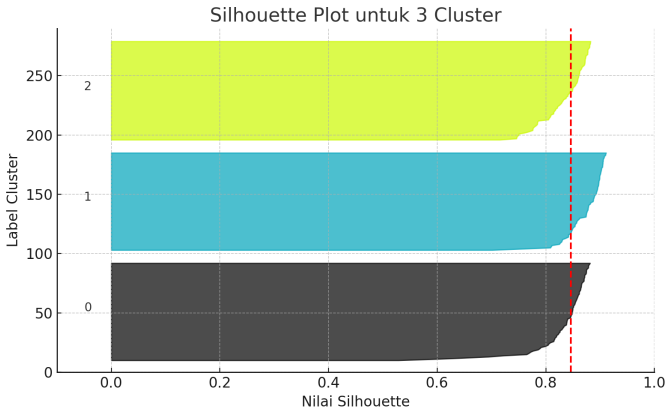
- *a(i)*: rata-rata jarak antara titik dan semua titik lain dalam cluster yang sama.
- *b(i)*: jarak minimum rata-rata antara titik dan semua titik dari cluster lain.

Formula:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Contoh kode Python:

```python
from sklearn.metrics import silhouette_score

score = silhouette_score(X, model.labels_)
print("Silhouette Score:", score)
```



Gambar 13: Silhouette Plot untuk hasil clustering dengan 3 cluster. Garis merah menunjukkan rata-rata nilai silhouette seluruh titik. Nilai yang tinggi menunjukkan cluster yang baik dan terpisah dengan jelas.

10.4.2 Inertia (Untuk K-Means)

Inertia adalah metrik khusus untuk algoritma K-Means yang mengukur jumlah total jarak kuadrat antara titik-titik data dengan centroid dari cluster-nya masing-masing. Semakin kecil nilai inertia, semakin baik pengelompokan data (karena berarti titik-titik data berada dekat dengan centroid-nya).

Formula:

$$Inertia = \sum_{i=1}^{n} \|x_i - \mu_{c_i}\|^2$$

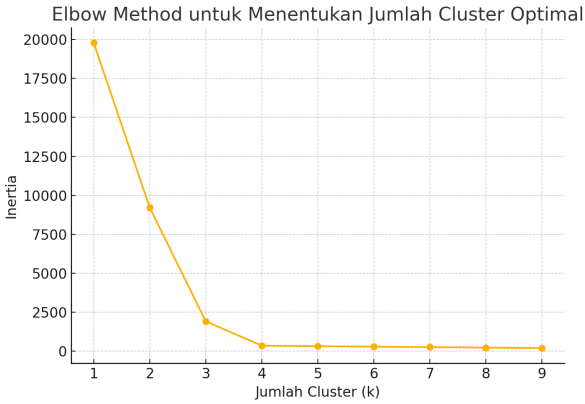
di mana:

- x_i adalah titik data ke-i,
- μ_{c_i} adalah centroid dari cluster c_i tempat x_i berada.

Namun, nilai inertia cenderung menurun seiring bertambahnya jumlah cluster. Oleh karena itu, evaluasi biasanya dikombinasikan dengan metode seperti **Elbow Method** untuk menentukan jumlah cluster optimal.

Contoh akses nilai inertia:

```
print("Inertia:", model.inertia_)
```



Gambar 14: Elbow Plot untuk menentukan jumlah cluster optimal berdasarkan nilai inertia. Titik tekukan (elbow) menunjukkan nilai k terbaik untuk digunakan.

10.5 Apa Itu Dimensionality Reduction?

Dimensionality reduction atau *reduksi dimensi* adalah proses untuk mengurangi jumlah fitur (dimensi) dalam dataset, dengan tujuan mempertahankan sebanyak mungkin informasi penting dari data asli. Teknik ini sangat berguna teruta-

ma dalam pengolahan data berdimensi tinggi (high-dimensional data), seperti data citra, teks, atau genomik.

10.5.1 Motivasi dan Tujuan Reduksi Dimensi

Reduksi dimensi bukan hanya sekadar menghapus kolom data, melainkan menerapkan transformasi matematis atau statistik untuk menyusun representasi baru dari data dengan dimensi yang lebih rendah. Tujuan utamanya meliputi:

- **Mempercepat komputasi** Model pembelajaran mesin bekerja lebih cepat ketika jumlah fitur lebih sedikit, karena beban komputasi menurun.
- **Menghindari Curse of Dimensionality** Dalam data berdimensi tinggi, performa model bisa menurun karena jarak antar titik menjadi kurang bermakna [7], dan risiko overfitting meningkat. Reduksi dimensi membantu menjaga kualitas generalisasi model.
- **Visualisasi Data** Data yang memiliki lebih dari dua dimensi sulit divisualisasikan. Teknik seperti *Principal Component Analysis (PCA)* dan *t-SNE* memungkinkan kita memproyeksikan data ke dalam dua atau tiga dimensi untuk analisis visual.

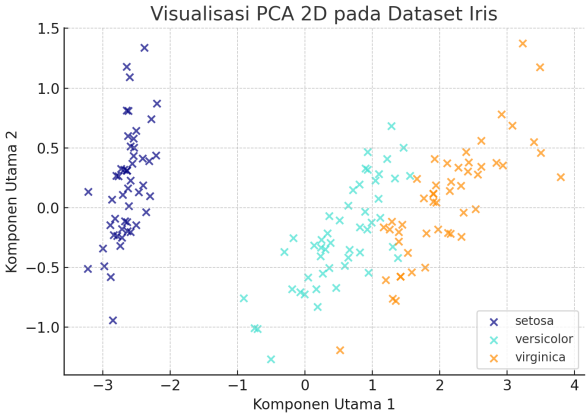
10.5.2 Contoh Aplikasi

- **PCA (Principal Component Analysis)** Teknik linear yang memproyeksikan data ke arah variansi maksimum. Co-

cok untuk mereduksi fitur sambil mempertahankan struktur utama data.

- **t-SNE (t-distributed Stochastic Neighbor Embedding)** Teknik non-linear untuk visualisasi data dalam 2D atau 3D, banyak digunakan untuk clustering hasil deep learning atau embedding teks [8].

Ilustrasi sederhana: Bayangkan dataset dengan 100 fitur. Tidak semua fitur memberikan informasi penting, beberapa mungkin redundant atau tidak relevan. Dengan reduksi dimensi, kita bisa menyaring informasi penting menjadi hanya 2 atau 3 fitur utama untuk keperluan analisis dan visualisasi lebih lanjut.



Gambar 15: Visualisasi PCA 2D pada dataset Iris. Data yang awalnya berdimensi empat direduksi menjadi dua dimensi utama untuk membantu eksplorasi dan visualisasi. Warna berbeda menunjukkan kelas yang berbeda.

10.6 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) adalah metode statistik [9] yang digunakan untuk melakukan *reduksi dimensi* dengan cara mentransformasikan kumpulan fitur asli ke dalam sekumpulan **komponen utama** (*principal components*). Komponen utama tersebut merupakan kombinasi linear dari fitur asli dan disusun berdasarkan jumlah variasi (varian) data yang dijelaskannya.

10.6.1 Prinsip Dasar PCA

PCA mencari arah baru dalam ruang fitur (disebut *komponen utama*) yang memaksimalkan variansi data. Komponen pertama (PC1) adalah arah dengan variansi tertinggi, komponen kedua (PC2) tegak lurus dengan PC1 dan menjelaskan variansi tertinggi berikutnya, dan seterusnya.

Tujuan utama:

- Mengurangi jumlah fitur tanpa kehilangan informasi penting.
- Mempermudah visualisasi data dalam dua atau tiga dimensi.
- Menghilangkan redundansi antar fitur dengan menggabungkannya ke dalam komponen yang tidak berkorelasi (orthogonal).

10.6.2 Langkah-Langkah PCA

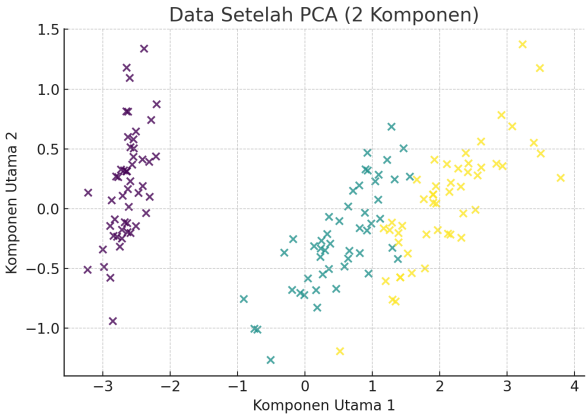
1. Standardisasi data (mean = 0, varian = 1).
2. Hitung matriks kovarians antar fitur.
3. Hitung eigenvalue dan eigenvector dari matriks kovarians.
4. Proyeksikan data ke ruang baru berdasarkan eigenvector dengan eigenvalue tertinggi.

10.6.3 Contoh Implementasi PCA dalam Python

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
```

Visualisasi hasil PCA:

```
import matplotlib.pyplot as plt

plt.scatter(X_pca[:, 0], X_pca[:, 1])
plt.title("Data Setelah PCA")
plt.xlabel("Komponen Utama 1")
plt.ylabel("Komponen Utama 2")
plt.grid(True)
plt.show()
```



Gambar 16: Visualisasi PCA dengan dua komponen utama pada dataset Iris. Warna menunjukkan kelas target. PCA digunakan untuk mereduksi dimensi dari 4 fitur menjadi 2 fitur utama.

10.6.4 Kelebihan dan Kekurangan PCA

Kelebihan:

- Efisien dalam mengurangi dimensi.
- Membantu mengatasi multikolinearitas antar fitur.
- Berguna untuk visualisasi data berdimensi tinggi.

Kekurangan:

- Komponen utama tidak memiliki makna interpretatif langsung terhadap fitur asli.
- PCA bersifat linear dan tidak cocok untuk pola non-linear (alternatif: t-SNE, UMAP).

10.7 Kombinasi PCA + Clustering

Dalam praktik nyata, teknik **reduksi dimensi** dan **clustering** sering digunakan secara bersamaan untuk menganalisis data berdimensi tinggi. Salah satu kombinasi paling umum adalah menggunakan **Principal Component Analysis (PCA)** untuk mereduksi dimensi data, kemudian menerapkan **clustering** seperti K-Means atau DBSCAN pada hasil transformasi PCA.

10.7.1 Langkah Umum

1. **Reduksi Dimensi:** Gunakan PCA untuk mengubah data berdimensi tinggi (misalnya dengan 10 atau lebih fitur) menjadi 2 atau 3 komponen utama. Hal ini bertujuan untuk mengekstrak struktur utama dari data sekaligus mengurangi noise dan redundansi fitur.

2. **Clustering:** Terapkan algoritma clustering (seperti K-Means, DBSCAN) pada data hasil transformasi PCA. Karena data telah disederhanakan, proses pengelompokan akan lebih efisien dan akurat.

3. **Visualisasi:** Visualisasikan hasil clustering pada ruang dua dimensi (jika menggunakan 2 komponen PCA). Visualisasi ini sangat membantu dalam interpretasi, eksplorasi data, dan pengambilan keputusan.

10.7.2 Manfaat Kombinasi PCA + Clustering

Kombinasi ini menggabungkan kekuatan PCA untuk menangani data berdimensi tinggi dan kemampuan clustering untuk menemukan struktur atau pola dalam data tanpa label.

Insight dan Aplikasi Nyata:

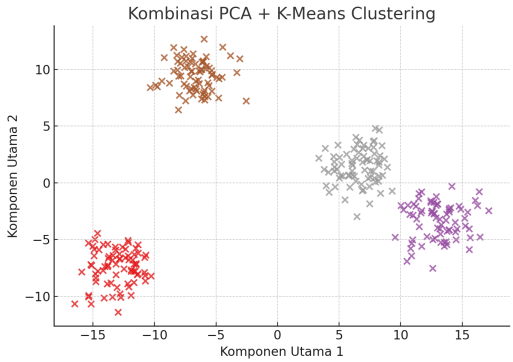
- **Segmentasi Pelanggan:** Mengelompokkan pelanggan berdasarkan perilaku belanja, preferensi produk, atau riwayat transaksi untuk meningkatkan personalisasi layanan.
- **Deteksi Fraud:** Mengidentifikasi pola transaksi abnormal dalam data finansial atau e-commerce.
- **Bioinformatika:** Mengelompokkan sampel genetik atau ekspresi gen berdasarkan profil molekuler yang kompleks.

Keuntungan utama:

- Meningkatkan interpretabilitas hasil clustering.
- Mengurangi noise yang dapat mengganggu algoritma clustering.
- Mempercepat proses pelatihan model karena data menjadi lebih ringkas.

10.7.3 Contoh Kode Python (Skema Umum)

```python
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans

# Langkah 1: Reduksi dimensi dengan PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
# Langkah 2: Clustering dengan K-Means
model = KMeans(n_clusters=3)
labels = model.fit_predict(X_pca)
# Langkah 3: Visualisasi hasil
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=labels)
plt.title("K-Means setelah PCA")
plt.show()
```



Gambar 17: Kombinasi PCA dan K-Means Clustering. Data berdimensi tinggi direduksi menjadi dua dimensi utama, lalu dikelompokkan menggunakan K-Means untuk mempermudah interpretasi visual.

10.8 Studi Kasus: Segmentasi Pelanggan E-Commerce

- **Fitur:** Frekuensi belanja, rata-rata pembelian, jenis produk, waktu kunjungan terakhir
- **Langkah:**

1. Normalisasi data
2. Terapkan PCA ke 2 dimensi
3. Gunakan K-Means untuk clustering
4. Visualisasikan hasil

Output: 3 klaster pelanggan: loyal, reguler, dan pasif

```
from sklearn.preprocessing import StandardScaler

X_scaled = StandardScaler().fit_transform(X)
X_pca = PCA(n_components=2).fit_transform(X_scaled)
labels = KMeans(n_clusters=3).fit_predict(X_pca)
```

10.9 Studi Kasus: Deteksi Anomali

Clustering juga bisa digunakan untuk menemukan anomali (keluar dari cluster).

Contoh: Transaksi yang berbeda jauh dari cluster lainnya → dicurigai sebagai penipuan.

10.10 Tantangan Clustering & Reduksi Dimensi

Meskipun clustering dan reduksi dimensi merupakan teknik yang sangat kuat dalam eksplorasi dan analisis data, keduanya juga memiliki berbagai tantangan yang harus diperhatikan dalam penerapannya. Berikut ini adalah beberapa tantangan utama yang sering ditemui:

10.10.1 Menentukan Jumlah Cluster Optimal

Pada algoritma seperti **K-Means**, pengguna harus menentukan jumlah cluster (k) di awal. Namun, dalam banyak kasus nyata, informasi tentang jumlah cluster yang ideal tidak tersedia.

Solusi yang umum digunakan:

- **Elbow Method:** Metode visual untuk menemukan titik "tekukan" (elbow) pada grafik inertia terhadap jumlah cluster. Titik ini menunjukkan jumlah cluster optimal sebelum penurunan inertia menjadi tidak signifikan.
- **Silhouette Score:** Memberikan nilai rata-rata yang mengukur kualitas pemisahan antar cluster. Semakin tinggi nilainya, semakin baik struktur cluster.

10.10.2 Interpretasi Hasil PCA Tidak Selalu Intuitif

Principal Component Analysis (PCA) membentuk fitur-fitur baru (komponen utama) sebagai kombinasi linear dari fitur-fitur asli. Komponen ini sering kali sulit untuk diinterpretasikan karena:

- Setiap komponen menggabungkan banyak fitur sekaligus.
- Arah komponen tidak selalu memiliki makna langsung terhadap domain asli data.

Meskipun PCA membantu dalam mengurangi dimensi dan meningkatkan efisiensi komputasi, pengguna perlu berhati-hati dalam menafsirkan hasil visualisasi atau kesimpulan berdasarkan komponen utama.

10.10.3 Hasil Clustering Sensitif terhadap Skala Data

Clustering, terutama algoritma berbasis jarak seperti K-Means atau DBSCAN, sangat **tergantung pada skala data**. Fitur yang memiliki skala lebih besar dapat mendominasi hasil jarak, sehingga memengaruhi proses pengelompokan.

Solusi:

- **Normalisasi atau Standardisasi:** Sebelum melakukan clustering atau PCA, sebaiknya semua fitur dinormali-

sasi agar berada pada skala yang sebanding, misalnya menggunakan:

- `StandardScaler` untuk menghasilkan mean 0 dan standar deviasi 1.
- `MinMaxScaler` untuk mereskalakan ke rentang [0, 1].

Contoh kode normalisasi:

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Dengan memahami dan mengantisipasi tantangan ini, kita dapat menghindari kesalahan interpretasi dan memastikan proses clustering dan reduksi dimensi berjalan secara optimal dan akurat.

10.11 Elbow Method untuk Menentukan K

Salah satu tantangan dalam algoritma **K-Means Clustering** adalah menentukan jumlah cluster optimal (K) sebelum melakukan pelatihan model. **Elbow Method** adalah pendekatan visual yang populer untuk memilih nilai K terbaik secara empiris.

10.11.1 Prinsip Dasar Elbow Method

K-Means bekerja dengan meminimalkan *inertia*, yaitu jumlah total jarak kuadrat antara setiap titik data dan centroid

dari cluster-nya. Semakin besar jumlah cluster, inertia akan menurun karena tiap cluster lebih kecil dan lebih dekat ke data yang dikandungnya.

Namun, setelah titik tertentu, penurunan inertia menjadi semakin kecil. Titik di mana penurunan ini mulai melambat disebut sebagai **elbow** (siku) dan dianggap sebagai jumlah cluster optimal.

Langkah-langkah:

1. Jalankan K-Means untuk berbagai nilai k (misalnya dari 1 hingga 10).
2. Simpan nilai inertia untuk masing-masing model.
3. Plot grafik k vs inertia.
4. Amati titik siku (elbow) pada grafik sebagai indikasi jumlah cluster terbaik.

10.11.2 Contoh Implementasi Python

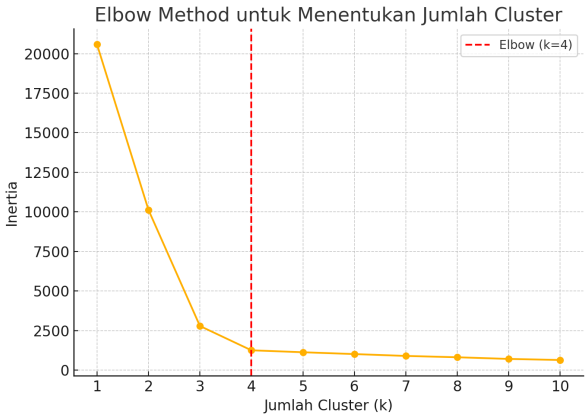
```
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

inertias = []
for k in range(1, 10):
    kmeans = KMeans(n_clusters=k)
    kmeans.fit(X)
    inertias.append(kmeans.inertia_)

plt.plot(range(1, 10), inertias, marker='o')
plt.title("Elbow Method")
plt.xlabel("Jumlah Cluster (k)")
```

```
plt.ylabel("Inertia")
plt.grid(True)
plt.show()
```

10.11.3 Interpretasi

- Grafik akan menunjukkan penurunan inertia secara signifikan saat jumlah cluster bertambah.
- Titik di mana grafik mulai *melandai* adalah titik siku atau **elbow**.
- Titik tersebut adalah nilai k yang sebaiknya digunakan karena menyeimbangkan antara kompleksitas model dan kualitas clustering.

Insight: Elbow Method sangat bermanfaat dalam eksplorasi data awal dan membantu menghindari pemilihan jumlah cluster yang terlalu sedikit (underfitting) atau terlalu banyak (overfitting).



Gambar 18: Visualisasi Elbow Method. Titik siku (elbow) pada $k = 4$ menunjukkan jumlah cluster optimal karena setelah titik tersebut penurunan inertia menjadi tidak signifikan.

10.12 LATIHAN / TUGAS AKHIR BAB 10

1. **[Uraian]** Jelaskan perbedaan utama antara K-Means dan DBSCAN, termasuk kelebihan dan kekurangan masing-masing.
2. **[Uraian Visual]** Buat skema visual yang menjelaskan proses PCA dari data mentah hingga dua komponen utama.
3. **[Coding]**
    - Gunakan dataset pelanggan.csv
    - Lakukan standardisasi

- Terapkan PCA ke 2 dimensi
- Lakukan K-Means dan tampilkan hasilnya dalam plot

4. **[Studi Kasus]** Terapkan DBSCAN untuk mengelompokkan titik lokasi GPS pelanggan dan analisis cluster potensial untuk membuka cabang baru.

5. **[Studi Kasus]** Gabungkan PCA dan clustering untuk analisis data pasar saham. Tentukan apakah terdapat pola atau anomali.

Daftar Pustaka

[1] R. Xu **and** D. Wunsch, "Survey of Clustering Algorithms," *IEEE Transactions on Neural Networks*, **jourvol** 16, **number** 3, **pages** 645–678, 2005.

[2] A. K. Jain, "Data Clustering: 50 years beyond K-means," *Pattern Recognition Letters*, **jourvol** 31, **number** 8, **pages** 651–666, 2010.

[3] J. B. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," **in***Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* **volume** 1, 1967, **pages** 281–297.

[4] T. M. Kodinariya **and** P. R. Makwana, "Review on determining number of Cluster in K-Means Clustering," *International Journal of Advance Research in Computer Science and Management Studies*, **jourvol** 1, **number** 6, **pages** 90–95, 2013.

[5] M. Ester, H.-P. Kriegel, J. Sander **and** X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," **in***Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD)* 1996, **pages** 226–231.

[6] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, **jourvol** 20, **pages** 53–65, 1987.

[7] R. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

[8] L. van der Maaten **and** G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, **jourvol** 9, **number** Nov, **pages** 2579–2605, 2008.

[9] I. T. Jolliffe, *Principal Component Analysis*, 2 **edition**. Springer, 2002.

11 TEXT MINING DAN ANALISIS DATA TEKS

11.1 Tujuan Pembelajaran

Setelah menyelesaikan bab ini, mahasiswa diharapkan mampu:

1. Memahami karakteristik dan tantangan data teks.
2. Melakukan proses cleaning, tokenisasi, dan representasi vektor teks.
3. Menggunakan metode Bag of Words dan TF-IDF untuk ekstraksi fitur dari teks.
4. Menerapkan teknik dasar analisis sentimen dan klasifikasi teks dengan Python.
5. Mengembangkan pemahaman etis tentang penggunaan data teks, termasuk privasi dan bias.

11.2 Apa Itu Text Mining?

Text Mining atau *text data mining* adalah proses otomatis atau semi-otomatis untuk mengekstraksi pengetahuan atau informasi bernilai dari data berbentuk teks [1]. Berbeda dari data terstruktur (seperti angka dalam tabel), data teks bersifat tidak terstruktur sehingga memerlukan pendekatan khusus untuk diproses dan dianalisis.

Dalam konteks *Ilmu Data*, text mining melibatkan beberapa tahapan mulai dari pra-pemrosesan teks, representasi fitur, hingga penerapan algoritma pembelajaran mesin untuk menemukan pola atau membuat prediksi [2].

11.2.1 Tujuan Utama Text Mining

Tujuan utama dari text mining adalah mengubah data teks mentah menjadi informasi yang berguna dan terstruktur, yang dapat dimanfaatkan untuk pengambilan keputusan, analisis tren, atau otomatisasi tugas berbasis teks.

11.2.2 Tahapan Umum Text Mining

1. **Ekstraksi Data Teks:** Mengambil data teks dari berbagai sumber seperti media sosial, dokumen PDF, email, atau situs berita.
2. **Pra-pemrosesan Teks:** Pembersihan teks seperti penghapusan stopword, tokenisasi, stemming, dan normalisasi.
3. **Representasi Teks:** Mengubah teks menjadi bentuk numerik seperti Bag of Words (BoW), TF-IDF, atau embedding.
4. **Analisis dan Model:** Menerapkan algoritma seperti klasifikasi, clustering, atau analisis sentimen untuk mendapatkan wawasan.

11.2.3 Contoh Penerapan Text Mining

Text mining memiliki banyak aplikasi dalam berbagai bidang industri dan riset. Berikut beberapa contohnya:

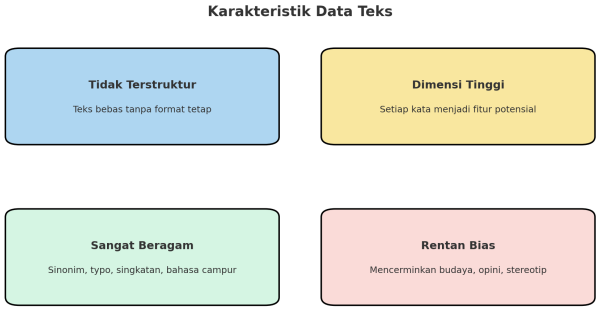
- **Analisis Sentimen Media Sosial:** Menentukan apakah opini atau komentar pengguna di media sosial bersifat positif, negatif, atau netral.
- **Klasifikasi Dokumen:** Mengkategorikan teks ke dalam topik tertentu secara otomatis, seperti berita politik, ekonomi, atau hiburan.
- **Sistem Rekomendasi Berbasis Ulasan:** Menggunakan ulasan pelanggan untuk memahami preferensi dan memberikan rekomendasi produk yang relevan.
- **Pendeteksian Hoaks dan Berita Palsu:** Menganalisis struktur bahasa dan konten artikel untuk mendeteksi informasi yang tidak valid atau menyesatkan.

11.2.4 Hubungan dengan Ilmu Data

Text mining menjadi semakin penting dalam ilmu data karena sebagian besar data digital saat ini berbentuk teks, seperti email, dokumen, obrolan daring, dan ulasan. Kemampuan untuk mengekstrak informasi dari teks menjadi keunggulan kompetitif bagi banyak organisasi.

11.3 Karakteristik Data Teks

Berbeda dengan data numerik atau kategorikal yang bersifat terstruktur dan mudah dikendalikan, **data teks** memiliki karakteristik unik yang membuat proses analisisnya menjadi lebih kompleks. Pemahaman terhadap karakteristik ini sangat penting sebelum menerapkan teknik *text mining* atau pembelajaran mesin berbasis teks.



Gambar 19: Empat karakteristik utama data teks: tidak terstruktur, berdimensi tinggi, sangat beragam, dan rentan terhadap bias. Pemahaman terhadap karakteristik ini penting dalam tahap awal text mining.

11.3.1 Tidak Terstruktur (Unstructured Data)

Data teks umumnya tidak memiliki struktur yang baku. Informasi tersimpan dalam kalimat bebas, paragraf, atau do-

kumen panjang yang tidak mengikuti format tabel atau skema tertentu. Hal ini menyulitkan pemrosesan otomatis jika tidak dilakukan pra-pemrosesan terlebih dahulu.

Contoh:

- Kalimat pengguna media sosial
- Artikel berita
- Transkrip wawancara atau percakapan

11.3.2 Dimensi Tinggi (High Dimensionality)

Setiap kata dalam korpus teks berpotensi menjadi satu fitur. Artinya, semakin banyak kata yang muncul, semakin tinggi jumlah dimensi yang harus dianalisis. Ini disebut sebagai *sparse high-dimensional space*, karena kebanyakan dokumen hanya mengandung sebagian kecil dari seluruh kosa kata yang tersedia.

Implikasi:

- Model menjadi sulit dilatih karena jumlah fitur sangat besar
- Diperlukan teknik representasi dan reduksi dimensi seperti TF-IDF dan PCA [3]

11.3.3 Sangat Beragam

Bahasa alami memiliki keragaman luar biasa. Kata-kata yang berbeda bisa memiliki makna yang sama (sinonim), dan ka-

ta yang sama bisa bermakna berbeda tergantung konteks. Selain itu, teks juga sering mengandung:

- *Typo* atau kesalahan ketik
- Singkatan informal (contoh: “gpp” untuk “nggak apa-apa”)
- Bahasa gaul dan campuran bahasa

Tantangan: Normalisasi teks menjadi langkah penting agar model tidak salah dalam memahami arti kata.

11.3.4 Rentan terhadap Bias

Data teks mencerminkan cara berpikir, budaya, dan pandangan penulisnya. Ini berarti teks sangat rentan terhadap:

- Bias budaya dan stereotip
- Subjektivitas dan opini
- Ketidakseimbangan representasi kelompok sosial tertentu

Jika tidak dikendalikan, bias ini dapat terbawa ke dalam model pembelajaran mesin dan memengaruhi hasil analisis secara tidak adil.

11.4 Pra-Pemrosesan Data Teks (Text Preprocessing)

Sebelum data teks dapat dianalisis secara statistik atau dimasukkan ke dalam model pembelajaran mesin, perlu di-

lakukan tahapan awal yang disebut **pra-pemrosesan teks** [4]. Tujuan dari proses ini adalah untuk membersihkan dan menyederhanakan teks sehingga dapat direpresentasikan secara numerik dan bermakna bagi algoritma analitik.

Berikut adalah beberapa langkah umum dalam pra-pemrosesan teks:

11.4.1 Lowercasing

Langkah pertama adalah mengubah seluruh huruf dalam teks menjadi huruf kecil (lowercase). Hal ini bertujuan untuk menyamakan bentuk kata seperti "Data" dan "data" yang semestinya dianggap sama.

Contoh kode Python:

```
text = text.lower()
```

11.4.2 Tokenization

Tokenisasi adalah proses memecah kalimat atau paragraf menjadi unit-unit terkecil, seperti kata atau frasa. Tokenisasi mempermudah analisis karena teks diubah menjadi list dari kata-kata.

Contoh kode Python:

```
from nltk.tokenize import word_tokenize
tokens = word_tokenize(text)
```

Hasil: Kalimat "Saya sedang belajar text mining." akan diubah menjadi `['Saya', 'sedang', 'belajar', 'text', 'mining', '.']`

11.4.3 Stopwords Removal

Stopwords adalah kata-kata umum seperti "yang", "dan", "adalah" yang sering muncul dalam teks tetapi tidak membawa informasi penting dalam konteks analisis. Stopwords dihapus untuk mengurangi dimensi data dan meningkatkan kualitas fitur.

Contoh kode Python:

```
from nltk.corpus import stopwords
stop_words = set(stopwords.words('indonesian'))
filtered_tokens = [w for w in tokens if w not in stop_words]
```

11.4.4 Stemming dan Lemmatization

Stemming adalah proses mengubah kata ke bentuk dasarnya dengan menghapus akhiran atau imbuhan. **Lemmatization** bertujuan serupa, tetapi mempertimbangkan konteks linguistik untuk mengembalikan kata ke bentuk lemanya (bentuk kamus).

Contoh kode Python (Stemming Bahasa Indonesia dengan Sastrawi):

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
stemmer = StemmerFactory().create_stemmer()
stemmed_text = stemmer.stem('berlari-lari')
```

Hasil: `"berlari-lari"` akan diubah menjadi `"lari"`

11.5 Representasi Teks sebagai Angka

Agar teks dapat digunakan dalam model *machine learning*, langkah penting yang harus dilakukan adalah mengubah data teks menjadi bentuk numerik. Proses ini disebut **representasi teks** atau *text vectorization*, yaitu mengubah dokumen teks menjadi *vektor fitur* yang dapat dihitung secara matematis.

Berikut adalah dua metode representasi teks yang paling umum:

11.5.1 Bag of Words (BoW)

Bag of Words adalah metode representasi teks yang mendasarkan diri pada **frekuensi kemunculan kata** dalam dokumen. Setiap dokumen diubah menjadi vektor berdimensi sebesar jumlah kosa kata unik dalam seluruh korpus.

Ciri utama:

- Tidak mempertimbangkan urutan kata
- Hanya memperhatikan kuantitas (frekuensi)

Contoh kode Python:

```
from sklearn.feature_extraction.text import CountVectorizer

corpus = [
    "Saya suka kopi",
    "Kopi membuat saya semangat",
    "Saya tidak suka teh"
```

```
]

vectorizer = CountVectorizer()
X = vectorizer.fit_transform(corpus)
```

Contoh representasi vektor:

Dokumen	saya	suka	kopi	membuat	semangat	tidak	teh
D1	1	1	1	0	0	0	0
D2	1	0	1	1	1	0	0
D3	1	1	0	0	0	1	1

Tabel 5: Contoh representasi Bag of Words (BoW)

11.5.2 TF-IDF (Term Frequency – Inverse Document Frequency)

TF-IDF adalah metode representasi teks yang memberikan bobot pada kata tidak hanya berdasarkan frekuensinya di dalam dokumen (TF), tetapi juga mempertimbangkan keunikan kata tersebut terhadap seluruh dokumen dalam korpus (IDF).

Rumus:

$$TF - IDF(t, d) = TF(t, d) \times \log\left(\frac{N}{DF(t)}\right)$$

di mana:

- $TF(t, d)$ = frekuensi kata t dalam dokumen d
- N = jumlah total dokumen
- $DF(t)$ = jumlah dokumen yang mengandung kata t

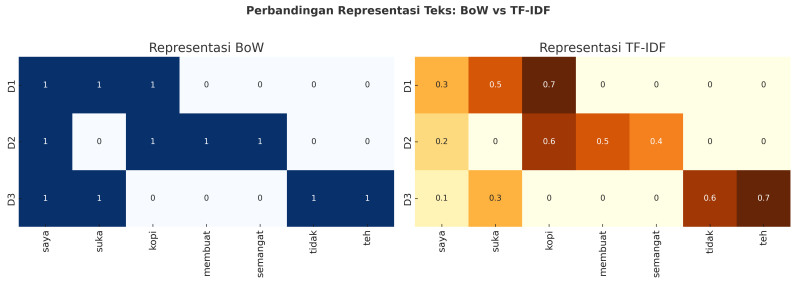
Contoh kode Python:

```python
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(corpus)
```

11.5.3 Perbandingan BoW vs TF-IDF

- **BoW:** Fokus pada kuantitas kata; kata yang sering muncul mendapat bobot tinggi meskipun mungkin tidak penting.
- **TF-IDF:** Memberi bobot lebih pada kata yang penting secara kontekstual; kata umum seperti ”saya” akan mendapatkan bobot rendah karena muncul di hampir semua dokumen.

Insight: TF-IDF lebih cocok digunakan saat kita ingin menekankan kata-kata unik yang mewakili isi dokumen, sedangkan BoW cocok untuk model sederhana atau baseline.



Gambar 20: Perbandingan visual antara representasi BoW dan TF-IDF pada tiga dokumen. TF-IDF memberikan bobot yang lebih tinggi pada kata-kata yang lebih spesifik.

11.6 Analisis Sentimen

Analisis Sentimen (*Sentiment Analysis*) adalah cabang dari *text mining* yang bertujuan untuk mengidentifikasi dan mengklasifikasikan opini atau emosi dalam teks, apakah bersifat **positif**, **negatif**, atau **netral**. Teknik ini banyak digunakan dalam analisis ulasan produk, media sosial, survei kepuasan pelanggan, dan bidang lain yang melibatkan opini publik.

11.6.1 Contoh Kasus

- `"Layanan pelanggan sangat mengecewakan."`
  → **Negatif**
- `"Saya sangat puas dengan pembeliannya."`
  → **Positif**
- `"Barang dikirim sesuai jadwal."`
  → **Netral**

11.6.2 Langkah-langkah Umum

1. **Pra-pemrosesan teks:** Bersihkan teks menggunakan tokenisasi, lowercasing, stopwords removal, dan stemming.
2. **Representasi fitur:** Ubah teks menjadi bentuk numerik, misalnya menggunakan TF-IDF.
3. **Pelabelan data:** Tentukan label untuk setiap teks (positif, negatif, netral).

4. **Pelatihan model klasifikasi:** Gunakan algoritma klasifikasi seperti Naive Bayes atau SVM.

5. **Evaluasi dan prediksi:** Uji model pada data baru untuk mengklasifikasikan sentimen.

11.6.3 Contoh Implementasi: Naive Bayes

Naive Bayes adalah salah satu algoritma yang sering digunakan untuk klasifikasi teks karena efisien dan cocok untuk data berdimensi tinggi seperti teks [5].

Contoh kode Python:

```
from sklearn.naive_bayes import MultinomialNB

model = MultinomialNB()
model.fit(X_train, y_train)
```

Keterangan:

- `X_train`: Data fitur yang telah direpresentasikan (misalnya hasil TF-IDF)
- `y_train`: Label sentimen dari data (positif, negatif, netral)

11.6.4 Kelebihan dan Kekurangan

- **Kelebihan:**
  - Cepat dan efisien untuk dataset besar
  - Mudah diimplementasikan

- **Kekurangan:**
  - Mengasumsikan independensi antar fitur
  - Rentan terhadap data yang tidak seimbang

11.6.5 Aplikasi Nyata

- Analisis ulasan e-commerce (Shopee, Tokopedia)
- Monitoring opini publik di media sosial (Twitter, Facebook)
- Evaluasi layanan pelanggan dari survei atau chat log

11.7 Studi Kasus: Analisis Ulasan Produk

Pada bagian ini, kita akan menerapkan teknik **analisis sentimen** dalam studi kasus nyata menggunakan dataset ulasan pelanggan dari platform e-commerce. Tujuan utama adalah untuk mengklasifikasikan apakah suatu ulasan bersifat **positif** atau **negatif**, sehingga perusahaan dapat memahami opini konsumen secara otomatis.

11.7.1 Dataset

Dataset yang digunakan berisi ulasan produk dalam bahasa Indonesia, lengkap dengan label sentimen:

- **Teks ulasan:** Teks asli dari pelanggan
- **Label sentimen:** 1 untuk positif, 0 untuk negatif

11.7.2 Langkah-langkah Analisis

1. **Pra-pemrosesan Teks** Bersihkan data teks dengan langkah-langkah seperti:
   - Mengubah ke huruf kecil (lowercasing)
   - Tokenisasi
   - Menghapus tanda baca dan stopwords
   - Stemming (menggunakan Sastrawi)
2. **Transformasi ke TF-IDF** Representasikan teks dalam bentuk numerik menggunakan *TF-IDF vectorizer* agar dapat diproses oleh model machine learning.
3. **Pemberian Label Sentimen** Tentukan label untuk setiap teks:
   - `1` untuk ulasan positif
   - `0` untuk ulasan negatif
4. **Pelatihan Model Klasifikasi** Gunakan model klasifikasi seperti:
   - **Logistic Regression:** model linier yang cocok untuk klasifikasi biner
   - **Naive Bayes:** sangat efisien untuk data teks
5. **Pengujian Model dan Prediksi** Uji model menggunakan data uji untuk melihat akurasi prediksi sentimen. Contoh hasil prediksi:
   - `"Barang bagus dan pengiriman cepat."` → **Positif**

- `"Kualitas produk mengecewakan."`
  → **Negatif**

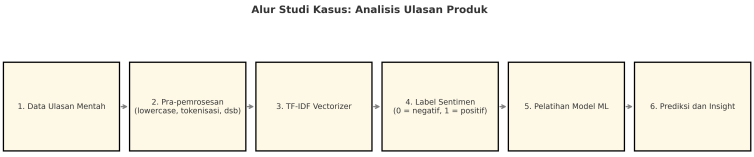
11.7.3 Insight

Analisis ulasan produk dengan teknik ini memberikan banyak manfaat, seperti:

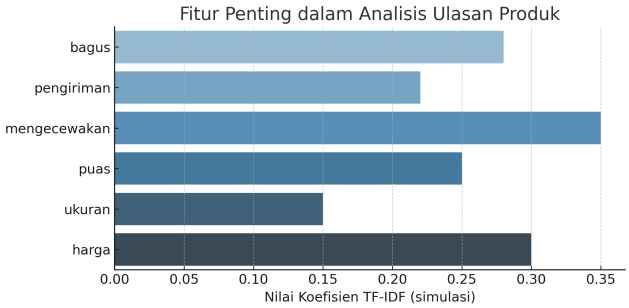
- Mengidentifikasi kekuatan dan kelebihan produk berdasarkan pujian pelanggan
- Menemukan kelemahan atau masalah umum yang sering dikeluhkan
- Mendukung pengambilan keputusan bisnis dan pengembangan produk

Contoh aplikasi:

- Analisis ribuan ulasan Tokopedia atau Shopee secara otomatis
- Sistem rekomendasi berbasis opini pengguna
- Dashboard pemantauan kepuasan pelanggan secara real-time



Gambar 21: Alur proses studi kasus analisis ulasan produk, mulai dari data mentah hingga insight akhir.



Gambar 22: Kata-kata yang paling berpengaruh dalam klasifikasi sentimen berdasarkan bobot TF-IDF (simulasi).

11.8 Word Cloud dan Visualisasi Kata

Salah satu cara populer dan intuitif untuk memahami isi teks secara umum adalah dengan menggunakan **Word Cloud**. Word Cloud menyajikan kata-kata yang paling sering muncul dalam korpus teks dalam bentuk visual, di mana ukuran huruf menunjukkan frekuensi kemunculan kata tersebut.

11.8.1 Apa itu Word Cloud?

Word Cloud (awan kata) adalah representasi visual dari kata-kata dalam dokumen. Kata yang sering muncul ditampilkan dengan ukuran huruf yang lebih besar, sementara kata yang jarang muncul ditampilkan lebih kecil atau tidak muncul sama sekali.

Manfaat Word Cloud:

- Mengidentifikasi kata-kata penting dalam dokumen secara cepat
- Memberikan gambaran topik utama secara visual
- Berguna untuk eksplorasi awal sebelum analisis lebih lanjut

11.8.2 Contoh Kode Python

Berikut adalah contoh penggunaan Word Cloud dalam Python:

```python
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Anggap 'corpus' adalah list of strings
text = ' '.join(corpus)

# Buat dan tampilkan Word Cloud
wc = WordCloud(width=800, height=400).generate(text)
plt.imshow(wc, interpolation='bilinear')
plt.axis('off')
plt.show()
```

11.8.3 Catatan Penting

- Sebaiknya lakukan **pra-pemrosesan teks** terlebih dahulu (lowercasing, stopwords removal, dll) sebelum membuat Word Cloud.
- Word Cloud hanya menunjukkan frekuensi kata, tidak mempertimbangkan konteks atau makna kata tersebut.

- Tidak cocok digunakan sebagai satu-satunya alat analisis; sebaiknya digunakan bersama metode kuantitatif lainnya.

11.8.4 Contoh Aplikasi

- Menyimpulkan topik utama dari kumpulan ulasan pelanggan
- Visualisasi hasil survei opini terbuka
- Eksplorasi cepat isi artikel berita atau pidato

Gambar 23: Contoh Word Cloud dari kumpulan ulasan pelanggan. Semakin besar ukuran kata, semakin sering kata tersebut muncul.

11.9 Evaluasi Model Teks

Setelah membangun model klasifikasi teks, langkah penting berikutnya adalah melakukan evaluasi untuk menilai sebe-

rapa baik model bekerja dalam memprediksi data baru. Evaluasi ini umumnya dilakukan menggunakan metrik yang sama seperti pada klasifikasi data numerik.

11.9.1 Accuracy

Akurasi menunjukkan proporsi prediksi yang benar dari seluruh prediksi yang dilakukan model.

$$Accuracy = \frac{Jumlahprediksibenar}{Totalprediksi}$$

Akurasi cocok digunakan jika jumlah data positif dan negatif seimbang. Namun, pada dataset yang tidak seimbang, akurasi bisa menyesatkan.

11.9.2 Precision, Recall, dan F1-Score

- **Precision (Presisi)**: Proporsi prediksi positif yang benar dari seluruh prediksi positif.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall (Sensitivitas)**: Proporsi prediksi positif yang benar dari seluruh data yang benar-benar positif.

$$Recall = \frac{TP}{TP + FN}$$

- **F1-Score**: Rata-rata harmonik dari precision dan recall.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Catatan:

- TP = True Positive
- FP = False Positive
- FN = False Negative

Metrik ini sangat penting ketika data tidak seimbang, seperti ketika hanya sebagian kecil ulasan yang negatif.

11.9.3 Confusion Matrix

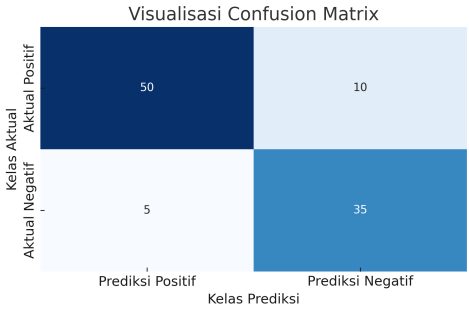
Confusion matrix adalah tabel yang menunjukkan jumlah prediksi benar dan salah untuk setiap kelas. Contoh matriks untuk klasifikasi dua kelas (positif dan negatif):

Aktual \Prediksi	Positif	Negatif
Positif	50 (TP)	10 (FN)
Negatif	5 (FP)	35 (TN)

Tabel 6: Contoh Confusion Matrix untuk Klasifikasi Teks

Interpretasi:

- TP (True Positive): Model memprediksi positif dan benar
- TN (True Negative): Model memprediksi negatif dan benar
- FP (False Positive): Model memprediksi positif tapi salah
- FN (False Negative): Model memprediksi negatif tapi salah



Gambar 24: Visualisasi confusion matrix untuk evaluasi model klasifikasi teks.

11.10 Tantangan dalam Text Mining

Meskipun teknik *text mining* telah berkembang pesat, penerapannya masih menghadapi berbagai tantangan, terutama dalam memahami kompleksitas bahasa alami. Tantangan ini muncul baik dari sisi bahasa itu sendiri, maupun dari karakteristik data yang digunakan.

11.10.1 Ambiguitas Bahasa

Bahasa alami bersifat ambigu, artinya satu kata bisa memiliki banyak makna tergantung pada konteks penggunaannya. Ini disebut **polisemi**.

- **Contoh:** Kata *"bisa"* dapat berarti kemampuan (*"saya bisa berenang"*) atau racun (*"ular berbisa"*).

Tantangan: Model teks harus memahami konteks agar tidak salah menafsirkan makna.

11.10.2 Ironi dan Sarkasme

Ironi dan sarkasme sering kali bertentangan dengan makna literal kalimat, sehingga sulit dideteksi secara otomatis.

- **Contoh:** *"Wah, pelayanannya luar biasa... bikin kapok!"*

Tantangan: Model klasifikasi sentimen bisa salah mengklasifikasikan kalimat sarkastik sebagai positif padahal bermakna negatif.

11.10.3 Bahasa Campuran

Dalam praktiknya, teks dalam bahasa Indonesia sering kali menggunakan kata-kata dari bahasa Inggris, terutama di media sosial, e-commerce, dan ulasan teknologi.

- **Contoh:** *"Produk ini sangat worth it dan fast delivery."*

Tantangan: Diperlukan strategi khusus seperti *normalisasi*, *token multilingual*, atau *word embedding* lintas bahasa.

11.10.4 Bias Data

Model pembelajaran mesin akan belajar dari data yang tersedia. Jika data tersebut mengandung bias, model juga dapat memperkuat bias tersebut [2].

- **Contoh:** Jika mayoritas ulasan positif berasal dari kelompok tertentu, model bisa mempelajari pola yang tidak adil terhadap kelompok lain.

Tantangan: Diperlukan proses *auditing* dan penyeimbangan dataset agar hasil model lebih adil dan representatif.

11.11 Etika dalam Analisis Data Teks

Dalam era digital saat ini, data teks banyak bersumber dari interaksi manusia di internet, seperti media sosial, ulasan produk, forum, dan dokumen digital. Oleh karena itu, penting untuk mempertimbangkan **aspek etika** dalam setiap proses analisis data teks.

Etika dalam analisis teks tidak hanya berhubungan dengan regulasi hukum, tetapi juga menyangkut tanggung jawab moral terhadap pengguna dan masyarakat luas.

11.11.1 Menghormati Privasi Pengguna

Data teks sering kali mengandung informasi pribadi atau sensitif, terutama ketika berasal dari media sosial atau percakapan daring.

- **Contoh:** Nama, lokasi, nomor telepon, atau keluhan pribadi dalam komentar publik.
- **Etika:** Jangan menyimpan, mempublikasikan, atau menganalisis data sensitif tanpa penyamaran atau pengha-

pusan identitas (*anonymization*).

11.11.2 Mendapatkan Izin atau Menggunakan Data Publik

Tidak semua data yang tersedia di internet dapat dianalisis secara bebas. Penting untuk memahami sumber data dan hak pengguna.

- **Gunakan data terbuka (open data)** yang secara eksplisit tersedia untuk penelitian.
- Jika menggunakan data dari platform tertentu, baca *terms of service* untuk memastikan legalitas penggunaannya.
- Hindari *web scraping* dari situs yang tidak memberikan izin eksplisit.

11.11.3 Menghindari Pelabelan Otomatis yang Diskriminatif

Model klasifikasi berbasis data dapat secara tidak sadar memperkuat stereotip atau membuat keputusan bias terhadap kelompok tertentu.

- **Contoh:** Model yang mengasosiasikan kata ”agresif” lebih sering pada ulasan dari kelompok gender atau etnis tertentu.
- **Etika:** Perlu dilakukan evaluasi berkala terhadap bias model dan memastikan dataset representatif.

11.11.4 Prinsip Umum Etika Analisis Teks

- **Transparansi:** Jelaskan bagaimana data dikumpulkan dan dianalisis.
- **Akunbilitas:** Bertanggung jawab atas dampak dari model yang dibangun.
- **Non-discrimination:** Hindari hasil yang berpotensi merugikan kelompok tertentu.
- **Privasi:** Lindungi identitas dan data pribadi pengguna.

11.12 LATIHAN / TUGAS AKHIR BAB 11

1. **[Uraian]** Jelaskan proses preprocessing teks mulai dari teks mentah hingga siap untuk machine learning.
2. **[Uraian Visual]** Buatlah bagan alir alur kerja analisis sentimen pada ulasan produk.
3. **[Coding]**
    - Ambil data ulasan (bisa dummy)
    - Bersihkan teks
    - Gunakan TF-IDF
    - Bangun model klasifikasi sentimen
    - Evaluasi akurasinya
4. **[Studi Kasus]** Ambil 100 tweet dengan tagar #Pendidikan dan lakukan klasifikasi opini secara otomatis. Sajikan hasil analisis dalam bentuk grafik.
5. **[Studi Kasus]** Kembangkan sistem pelacakan reputasi merek dari ulasan produk online dan berita media.

Daftar Pustaka

[1] C. D. Manning, P. Raghavan **and** H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[2] D. Jurafsky **and** J. H. Martin, *Speech and Language Processing (3rd ed. draft)*. 2023, Available online: `https://web.stanford.edu/~jurafsky/slp3/`.

[3] H. Keisha **and** K. Muslim, "Analysis of TF-IDF and TF-RF Feature Extraction on Product Review Sentiment," *Sinkron: Jurnal dan Penelitian Teknik Informatika*, **jourvol** 8, **number** 2, 2024, Available online: `https://www.researchgate.net/publication/379458320`.

[4] E. Haddi, X. Liu **and** Y. Shi, "The Role of Text Pre-processing in Sentiment Analysis," *Procedia Computer Science*, **jourvol** 17, **pages** 26–32, 2013.

[5] J. Camacho-Collados **and** M. T. Pilehvar, "On the Role of Text Preprocessing in Neural Network Architectures: An Evaluation Study on Text Categorization and Sentiment Analysis," *arXiv preprint arXiv:1707.01780*, 2017.

12 BIG DATA DAN HADOOP ECOSYSTEM

12.1 Tujuan Pembelajaran

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Menjelaskan konsep dan karakteristik Big Data.
2. Mengenal teknologi dan arsitektur Hadoop Ecosystem.
3. Memahami cara kerja HDFS, MapReduce, dan YARN.
4. Mengetahui peran alat bantu seperti Hive, Pig, Spark, dan HBase dalam analisis data besar.
5. Memahami tantangan dan penerapan Big Data dalam industri.

12.2 Apa Itu Big Data?

Big Data adalah istilah yang merujuk pada kumpulan data dalam jumlah sangat besar dan kompleks yang tidak dapat dikelola, disimpan, atau dianalisis menggunakan metode dan sistem pengolahan data tradisional [1]–[3].

Big Data muncul sebagai konsekuensi dari meningkatnya aktivitas digital, seperti penggunaan media sosial, perangkat IoT, sistem transaksi daring, dan sensor yang meng-

hasilkan data secara terus-menerus dalam jumlah besar dan beragam format.

Contoh:

- Data transaksi harian jutaan pelanggan di e-commerce
- Ratusan juta postingan media sosial setiap hari
- Streaming data dari sensor kendaraan atau perangkat medis

12.2.1 Ciri Khas Big Data: 5V

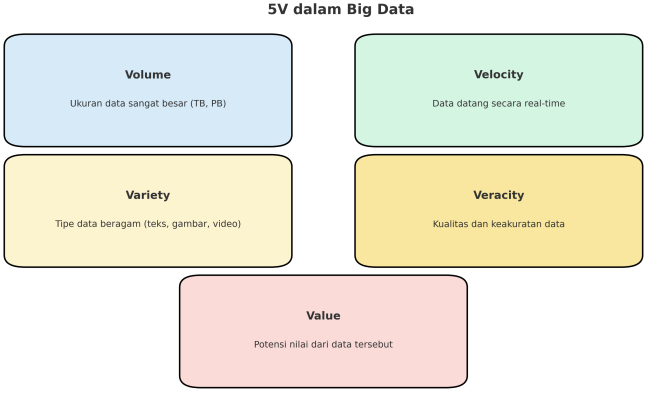
Big Data biasanya dikenali melalui lima karakteristik utama yang dikenal sebagai **5V** [1], [2]:

V	Penjelasan
Volume	Ukuran data yang sangat besar, bisa mencapai Terabyte (TB) hingga Petabyte (PB) atau lebih
Velocity	Kecepatan tinggi dalam menghasilkan dan memproses data secara real-time atau mendekati real-time
Variety	Keragaman tipe data, seperti teks, gambar, audio, video, log sistem, dan data sensor
Veracity	Tingkat keakuratan dan keandalan data; data besar sering kali mengandung *noise* atau ketidakpastian
Value	Potensi nilai yang dapat diperoleh dari analisis data, seperti insight bisnis, prediksi perilaku, atau efisiensi operasional

Tabel 7: Lima Karakteristik Big Data (5V)

12.2.2 Mengapa Big Data Penting?

- Big Data membuka peluang untuk memahami perilaku pengguna dan tren pasar secara lebih akurat.
- Organisasi dapat membuat keputusan berbasis data (*data-driven decision making*) dalam skala besar.
- Penggunaan Big Data mendorong inovasi di berbagai bidang seperti kesehatan, pendidikan, transportasi, dan pemerintahan [4].



Gambar 25: Lima karakteristik utama Big Data yang dikenal dengan istilah 5V: Volume, Velocity, Variety, Veracity, dan Value.

12.3 Sumber Data Big Data

Big Data berasal dari berbagai sumber yang tersebar luas dan terus berkembang seiring kemajuan teknologi digital.

Sumber-sumber ini menghasilkan data dalam volume besar, kecepatan tinggi, dan dalam berbagai format.

Berikut adalah beberapa kategori utama sumber data Big Data:

12.3.1 Media Sosial

Platform seperti **Twitter**, **Instagram**, dan **TikTok** menghasilkan jutaan konten setiap hari. Data ini meliputi:

- Teks (status, komentar)
- Gambar dan video
- Reaksi dan interaksi pengguna (like, share, repost)

Karakteristik: Tidak terstruktur, real-time, sangat beragam dan opini-driven.

12.3.2 Transaksi Online

Data dari **e-commerce**, **mobile banking**, dan sistem pembayaran digital mencakup:

- Riwayat pembelian
- Data pembayaran
- Preferensi dan perilaku pelanggan

Karakteristik: Terstruktur, sensitif, dan memiliki nilai tinggi untuk analisis perilaku konsumen.

12.3.3 Sensor dan IoT (Internet of Things)

Perangkat seperti kamera, sensor suhu, GPS, dan kendaraan otonom menghasilkan data dalam jumlah besar secara kontinu.

Contoh:

- Sistem *smart city* yang mengatur lalu lintas dan lingkungan
- Mobil otonom yang menggunakan sensor untuk navigasi

Karakteristik: Streaming data, berbasis waktu, dan membutuhkan respons cepat [4].

12.3.4 Log Sistem dan Infrastruktur TI

Setiap aktivitas di server, aplikasi, atau jaringan dicatat dalam bentuk log.

- Log aktivitas pengguna
- Log kesalahan sistem
- Log keamanan jaringan

Karakteristik: Semi-terstruktur, penting untuk pemantauan dan keamanan [5].

12.3.5 Multimedia

Data dalam bentuk **gambar**, **video**, dan **audio** dari berbagai sumber seperti CCTV, panggilan suara, dan rekaman konferensi.

Contoh:

- Video pengawasan kota
- Audio dari call center layanan pelanggan
- Foto produk dan review pengguna

Karakteristik: Tidak terstruktur, membutuhkan kapasitas penyimpanan dan pemrosesan besar, serta teknik analisis khusus seperti computer vision dan audio processing.

12.4 Tantangan dalam Pengelolaan Big Data

Meskipun Big Data menawarkan potensi besar untuk menghasilkan wawasan dan nilai strategis, pengelolaannya memunculkan berbagai tantangan teknis, operasional, dan etis. Tantangan-tantangan ini perlu dipahami agar solusi Big Data dapat diterapkan secara efektif dan berkelanjutan.

12.4.1 Skalabilitas Sistem Penyimpanan dan Pemrosesan

Big Data memerlukan sistem yang dapat menangani pertumbuhan data secara eksponensial. Infrastruktur penyim-

panan dan pemrosesan harus mampu **diskalakan** baik secara vertikal (meningkatkan kapasitas server) maupun horizontal (menambah jumlah server).

Solusi: Gunakan arsitektur *distributed computing* seperti Hadoop dan Spark yang memungkinkan pemrosesan paralel dan penyimpanan terdistribusi.

12.4.2 Integrasi Data dari Banyak Sumber

Data dalam lingkungan Big Data berasal dari berbagai sumber dan dalam format yang berbeda-beda (terstruktur, semi-terstruktur, tidak terstruktur).

- **Contoh:** Menggabungkan data transaksi (terstruktur) dengan komentar media sosial (tidak terstruktur)

Tantangan: Menyamakan skema, format, dan kualitas data agar dapat dianalisis secara terpadu.

12.4.3 Privasi dan Keamanan Data

Semakin besar dan beragam data yang dikumpulkan, semakin tinggi pula risiko kebocoran data pribadi dan penyalahgunaan informasi.

- **Isu utama:** Akses tanpa izin, pelanggaran regulasi, identifikasi pengguna tanpa sepengetahuan mereka.

Solusi: Implementasi enkripsi, autentikasi ganda, serta kepatuhan terhadap regulasi seperti GDPR dan UU Perlindungan Data Pribadi.

12.4.4 Biaya Penyimpanan dan Infrastruktur

Big Data memerlukan penyimpanan dengan kapasitas tinggi, server yang kuat, dan koneksi jaringan yang cepat, yang semuanya membutuhkan biaya besar.

Alternatif: Menggunakan layanan *cloud computing* seperti AWS, GCP, atau Azure yang menawarkan fleksibilitas biaya dan kapasitas sesuai kebutuhan.

12.4.5 Pengolahan Real-Time vs Batch

Tidak semua data perlu dianalisis secara langsung (real-time), tetapi ada kasus-kasus kritis seperti deteksi fraud atau pemantauan sistem yang menuntut pemrosesan instan.

- **Batch Processing:** Data dikumpulkan terlebih dahulu, lalu diproses secara berkala (contoh: laporan mingguan)
- **Stream Processing:** Data diproses segera saat diterima (contoh: monitoring media sosial)

Tantangan: Menyeimbangkan kebutuhan akan kecepatan (real-time) dan efisiensi sumber daya (batch).

12.5 Pengenalan Hadoop Ecosystem

Apache Hadoop adalah kerangka kerja (*framework*) open-source yang dirancang untuk menyimpan dan memproses **Big Data** secara efisien dengan cara *terdistribusi* melalui jaringan komputer (klaster) [6].

Hadoop memungkinkan organisasi untuk menangani data dalam skala petabyte dengan biaya relatif rendah menggunakan perangkat keras biasa (komoditas).

12.5.1 Karakteristik Utama Hadoop

- Open-source dan skalabel
- Cocok untuk pemrosesan data dalam jumlah besar secara paralel
- Dirancang untuk fault-tolerant (tahan terhadap kegagalan node)

12.5.2 Komponen Utama dalam Ekosistem Hadoop

Hadoop bukan hanya satu perangkat lunak, tetapi terdiri dari berbagai komponen yang saling terintegrasi. Tabel berikut merangkum komponen-komponen utama dalam Hadoop Ecosystem:

Komponen	Fungsi
HDFS	Sistem penyimpanan file terdistribusi
MapReduce	Model pemrosesan data paralel
YARN	Pengelolaan sumber daya dan penjadwalan pekerjaan
Hive	SQL-like interface untuk querying data di Hadoop
Pig	Bahasa scripting untuk analisis data kompleks
HBase	Database kolom untuk data besar yang membutuhkan akses cepat
Spark	Pemrosesan in-memory untuk kecepatan tinggi

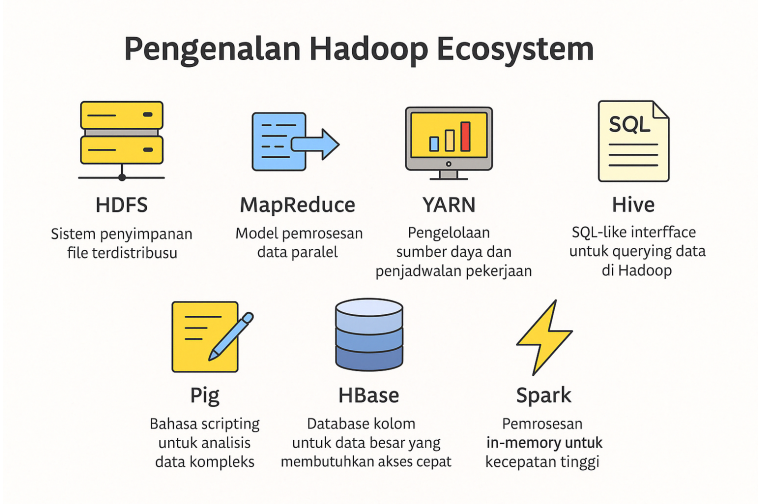
Tabel 8: Komponen dan Fungsi pada Ekosistem Hadoop

12.5.3 Mengapa Hadoop Penting dalam Big Data?

- Mendukung pengolahan data skala besar secara paralel dan terdistribusi
- Mampu menangani berbagai format data (teks, log, gambar)
- Menjadi fondasi dari banyak sistem analitik modern

12.5.4 Contoh Penggunaan Hadoop

- Analisis log pengguna dari jutaan kunjungan ke situs web
- Pemrosesan data sensor dari kendaraan atau pabrik
- Analisis transaksi besar di e-commerce



Gambar 26: Komponen utama dalam Hadoop Ecosystem: HDFS, MapReduce, YARN, Hive, Pig, HBase, dan Spark, masing-masing berperan dalam penyimpanan, pemrosesan, query, dan manajemen data berskala besar.

12.6 HDFS – Hadoop Distributed File System

Hadoop Distributed File System (HDFS) adalah komponen inti dalam ekosistem Hadoop yang bertugas menyimpan data dalam jumlah besar secara terdistribusi di banyak node dalam sebuah klaster.

Dirancang untuk lingkungan yang membutuhkan throughput tinggi dan fault-tolerance, HDFS memungkinkan pengelolaan data dalam skala besar dengan biaya relatif rendah.

12.6.1 Konsep Dasar HDFS

HDFS bekerja dengan membagi file menjadi blok-blok besar, yang kemudian disimpan di beberapa **DataNode** dalam klaster.

- Ukuran blok default adalah **128MB**, tetapi dapat dikonfigurasi.
- Setiap blok disalin (replicated) ke beberapa node untuk menjamin ketersediaan.
- Umumnya, HDFS menyimpan **3 salinan** dari setiap blok secara otomatis (replication factor = 3).

12.6.2 Struktur Arsitektur HDFS

1. NameNode

- Bertugas mengelola metadata sistem file, seperti struktur direktori, lokasi blok, dan izin akses file.
- Tidak menyimpan data secara langsung.
- Komponen kritis—jika NameNode gagal, seluruh sistem terganggu kecuali ada mekanisme cadangan.

2. DataNode

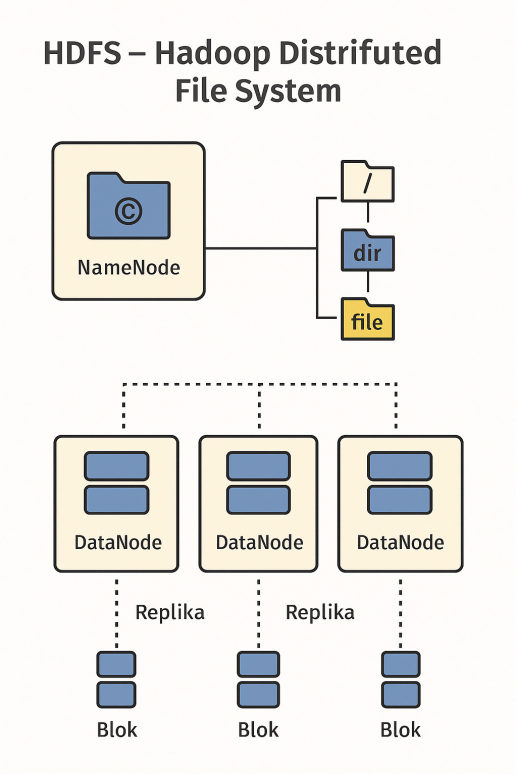
- Menyimpan blok-blok data secara fisik.
- Melayani permintaan baca/tulis data dari klien atau perintah dari NameNode.
- Dapat ditambahkan secara dinamis untuk meningkatkan kapasitas sistem.

12.6.3 Keunggulan HDFS

- **Redundansi dan Replikasi:** Data disalin ke beberapa node, memastikan ketersediaan bahkan jika beberapa node gagal.
- **Fault-Tolerance:** Sistem tetap berjalan meskipun ada DataNode yang rusak.
- **Skalabilitas Horizontal:** Mudah menambah kapasitas hanya dengan menambahkan node baru ke klaster.

12.6.4 Ilustrasi Singkat:

- File sebesar 256MB akan dipecah menjadi dua blok (128MB + 128MB)
- Setiap blok akan disimpan di tiga DataNode berbeda
- NameNode mencatat di mana setiap blok disimpan



Gambar 27: Diagram arsitektur HDFS: NameNode mengelola metadata, sedangkan DataNode menyimpan blok-blok data dan replikasinya secara terdistribusi.

12.7 MapReduce – Model Pemrosesan Terdistribusi

MapReduce adalah paradigma pemrograman dalam Hadoop yang memungkinkan pemrosesan data dalam jumlah besar secara paralel dan terdistribusi. Model ini sangat cocok digunakan dalam lingkungan Big Data karena mampu menangani pembagian beban kerja dan penggabungan hasil secara efisien.

12.7.1 Konsep Dasar MapReduce

MapReduce bekerja dalam dua tahap utama:

1. **Map:** Tahap ini memecah pekerjaan besar menjadi unit-unit kecil dalam bentuk *key-value pair*. Setiap *mapper* bekerja secara independen pada sebagian data.
2. **Reduce:** Tahap ini menggabungkan dan menganalisis hasil dari proses map berdasarkan *key* yang sama, menghasilkan output akhir yang telah diringkas atau dihitung.

12.7.2 Contoh Sederhana: Menghitung Jumlah Kemunculan Kata

Misalkan kita ingin menghitung berapa kali setiap kata muncul dalam dokumen besar.

- **Input:** `"data science is data"`
- **Tahap Map:**

```
(data, 1), (science, 1), (is, 1), (data, 1)
```

- **Tahap Reduce:**

```
(data, [1, 1]) → (data, 2)
(science, [1]) → (science, 1)
(is, [1]) → (is, 1)
```

12.7.3 Keunggulan MapReduce

- Dapat digunakan untuk pemrosesan data dalam skala petabyte
- Proses berjalan paralel di banyak node, meningkatkan efisiensi
- Dirancang untuk fault-tolerant: jika salah satu node gagal, proses bisa dialihkan ke node lain

12.7.4 Keterbatasan MapReduce

- Kurang cocok untuk pemrosesan real-time karena berbasis batch
- Tidak fleksibel untuk analitik kompleks (misalnya, machine learning)

- Relatif lambat dibanding model baru seperti Apache Spark

12.7.5 Kapan Menggunakan MapReduce?

MapReduce cocok digunakan untuk proses yang bersifat **batch processing** dalam volume data besar, seperti:

- Analisis log
- Penghitungan agregat data
- Sortir dan pencarian teks masif

12.8 YARN – Yet Another Resource Negotiator

YARN (Yet Another Resource Negotiator) adalah komponen utama dalam ekosistem Hadoop yang berfungsi sebagai **manajer sumber daya dan penjadwal eksekusi tugas** pada klaster Hadoop.

YARN memisahkan fungsi pengelolaan sumber daya dari fungsi pemrosesan data, sehingga lebih fleksibel dan efisien dalam mendukung berbagai jenis engine pemrosesan.

12.8.1 Fungsi Utama YARN

YARN memiliki tiga peran inti dalam sistem Hadoop:

- **Alokasi Sumber Daya** YARN mengatur distribusi sumber daya seperti RAM dan CPU kepada aplikasi atau job yang berjalan dalam klaster.
- **Penjadwalan dan Eksekusi Tugas** YARN memutuskan kapan dan di mana suatu tugas akan dijalankan, serta mengatur lifecycle-nya, termasuk pemantauan status dan penanganan kegagalan.
- **Dukungan Multi-Engine** Selain MapReduce, YARN juga mendukung berbagai engine pemrosesan lain seperti Apache Spark, Apache Flink, dan Tez.

12.8.2 Komponen Utama YARN

- **ResourceManager** Komponen pusat yang bertugas mengelola semua sumber daya dalam klaster, termasuk pengambilan keputusan global untuk penjadwalan job.
- **NodeManager** Komponen yang berjalan di setiap node. Bertanggung jawab memantau penggunaan sumber daya di node tersebut dan melaporkan ke ResourceManager.
- **ApplicationMaster** Unit yang dijalankan per aplikasi untuk mengelola eksekusi job-nya sendiri. ApplicationMaster berinteraksi langsung dengan ResourceManager untuk meminta resource dan mengatur eksekusi task-nya.
- **Container** Unit logis dari resource (RAM dan CPU) yang dialokasikan untuk menjalankan task tertentu. Semua aplikasi dijalankan dalam container.

12.8.3 Keunggulan YARN

- Meningkatkan pemanfaatan sumber daya karena lebih fleksibel dan dinamis
- Memungkinkan berbagai engine berjalan dalam satu ekosistem Hadoop
- Mendukung skalabilitas tinggi dan pemisahan tanggung jawab

12.8.4 Analogi Sederhana

Bayangkan YARN seperti *manajer gedung coworking*:

- **ResourceManager** = resepsionis yang menentukan siapa dapat ruangan mana
- **NodeManager** = penjaga setiap ruangan yang memastikan sumber daya cukup
- **ApplicationMaster** = koordinator tim proyek masing-masing
- **Container** = ruang kerja untuk setiap tugas dalam proyek

12.9 Hive dan Pig

Dalam ekosistem Hadoop, dua alat yang sangat berguna untuk analisis data di atas HDFS adalah **Hive** dan **Pig**. Keduanya memudahkan pengguna dalam menulis logika pem-

rosesan data tanpa harus menulis kode MapReduce secara manual.

12.9.1 Hive: SQL untuk Big Data

Apache Hive adalah alat query berbasis **SQL-like** yang dirancang untuk bekerja di atas HDFS. Hive mengubah query SQL menjadi job MapReduce, sehingga pengguna yang familiar dengan SQL dapat mengolah data besar dengan mudah.

Fitur Hive

- Mendukung sintaks SQL standar (disebut HiveQL)
- Cocok untuk data yang tersimpan dalam bentuk tabel
- Berfungsi sebagai *data warehouse* di Hadoop

Contoh Query Hive

```
SELECT gender, COUNT(*)
FROM users
GROUP BY gender;
```

Penjelasan: Query ini menghitung jumlah pengguna berdasarkan jenis kelamin dari tabel `users`.

Kelebihan Hive:

- Mudah digunakan oleh pengguna SQL
- Cocok untuk laporan, agregasi, dan query analitis

Kekurangan Hive:

- Tidak cocok untuk pemrosesan real-time
- Kurang fleksibel untuk logika transformasi kompleks

12.9.2 Pig: Scripting untuk Transformasi Data

Apache Pig adalah platform pemrosesan data yang menggunakan bahasa bernama **Pig Latin**, sebuah bahasa scripting yang dirancang untuk menulis pipeline transformasi data secara deklaratif.

12.9.3 Fitur Pig

- Cocok untuk proses ETL (Extract, Transform, Load)
- Lebih fleksibel daripada Hive untuk transformasi kompleks
- Dapat dijalankan di atas MapReduce

12.9.4 Contoh Alur Pig Latin

```
users = LOAD 'users.csv' USING PigStorage(',') AS (id:int,
    name:chararray, gender:chararray);
grouped = GROUP users BY gender;
counts = FOREACH grouped GENERATE group, COUNT(users);
DUMP counts;
```

Penjelasan: Script ini memuat data pengguna dari file CSV, mengelompokkan berdasarkan jenis kelamin, lalu menghitung jumlah pengguna per kelompok.

Kelebihan Pig:

- Lebih ringkas dibanding MapReduce mentah
- Mendukung alur pemrosesan yang kompleks

Kekurangan Pig:

- Kurang familiar bagi pengguna non-programmer
- Kurang optimal untuk query tabular sederhana

Perbandingan Hive vs Pig

Aspek	Hive	Pig
Bahasa	SQL-like (HiveQL)	Pig Latin (scripting)
Pengguna utama	Analis data, pengguna SQL	Data engineer, programmer
Gaya pemrograman	Deklaratif (query tabular)	Prosedural (alur transformasi data)
Kelebihan utama	Mudah dipahami oleh pengguna SQL	Fleksibel dan ringkas untuk ETL
Kekurangan utama	Kurang fleksibel untuk transformasi kompleks	Kurang familiar bagi non-programmer
Cocok untuk	Query laporan, agregasi data besar	Transformasi data kompleks dan pipeline

Gambar 28: Perbandingan antara Hive dan Pig dalam ekosistem Hadoop berdasarkan bahasa, pengguna, kelebihan, kekurangan, dan kasus penggunaan.

12.10 Apache Spark: Pemrosesan Lebih Cepat

Apache Spark adalah platform pemrosesan Big Data generasi baru yang dirancang untuk menggantikan keterbatasan Hadoop MapReduce. Spark dapat memproses data secara **in-memory** (langsung di memori), sehingga mempercepat analisis data yang kompleks hingga puluhan kali lipat dibanding MapReduce tradisional [7], [8].

12.11 Mengapa Spark Lebih Cepat?

- Spark menyimpan data dalam **RAM** selama pemrosesan (in-memory processing), bukan membaca dan menulis dari disk seperti MapReduce.
- Mendukung **lazy evaluation** dan **DAG (Directed Acyclic Graph)** untuk mengoptimalkan alur eksekusi.
- Dapat digunakan untuk berbagai beban kerja: *batch processing*, *stream processing*, *machine learning*, dan *graph analytics*.

12.11.1 Fitur Utama Apache Spark

- **In-Memory Computing:** Proses data langsung di RAM, tidak harus selalu disimpan ke disk.
- **Multibahasa:** Mendukung Python (PySpark), Scala, Java, R.
- **Spark SQL:** Mendukung query SQL di atas data besar.
- **Spark Streaming:** Untuk pemrosesan data secara real-time.
- **MLlib:** Library machine learning bawaan Spark.

12.11.2 Contoh Penggunaan dengan PySpark

Berikut adalah contoh sederhana membaca file CSV menggunakan PySpark:

```
from pyspark.sql import SparkSession

# Membuat session Spark
spark = SparkSession.builder.appName("Contoh").getOrCreate()

# Membaca data dari file CSV
df = spark.read.csv("data.csv", header=True, inferSchema=True)

# Menampilkan isi DataFrame
df.show()
```

12.11.3 Kelebihan Apache Spark

- Jauh lebih cepat dibanding MapReduce (terutama untuk iterative jobs)
- Mendukung real-time analytics
- API yang intuitif dan mudah digunakan
- Terintegrasi dengan Hadoop dan HDFS

12.11.4 Keterbatasan Apache Spark

- Konsumsi memori lebih tinggi
- Perlu pengelolaan sumber daya yang cermat di klaster besar

- Tidak optimal untuk semua jenis beban kerja, terutama jika hanya bersifat sekali jalan dan sangat besar

12.11.5 Contoh Penggunaan Spark di Dunia Nyata

- Analisis real-time media sosial
- Deteksi penipuan pada transaksi keuangan
- Analisis perilaku pengguna dalam aplikasi mobile
- Sistem rekomendasi e-commerce

12.12 HBase – Database Kolom

Apache HBase adalah sistem manajemen basis data non-relasional (NoSQL) berbasis kolom yang berjalan di atas Hadoop dan HDFS. HBase dirancang untuk menyimpan dan mengakses **data besar** secara cepat dan efisien, terutama dalam skenario akses acak dan skala besar.

HBase terinspirasi oleh sistem *Bigtable* milik Google dan menawarkan penyimpanan terstruktur untuk data yang tidak cocok dengan basis data relasional tradisional.

12.12.1 Karakteristik Utama HBase

- **Model data berbasis kolom:** Data disimpan dalam bentuk tabel, tetapi setiap kolom bisa sangat fleksibel dan bersifat semi-terstruktur.

- **Dibangun di atas HDFS:** Memanfaatkan penyimpanan terdistribusi Hadoop.
- **Skalabilitas horizontal:** Dapat ditambahkan node baru untuk menangani pertumbuhan data.
- **Akses acak:** Mendukung baca/tulis cepat pada baris tertentu, tidak seperti HDFS yang optimal untuk proses sekuensial.

12.12.2 Kapan Menggunakan HBase?

HBase sangat cocok digunakan untuk kasus-kasus berikut:

- **Data time-series:** Seperti data sensor, log perangkat, atau transaksi keuangan yang terus bertambah secara kronologis.
- **Akses acak pada data besar:** Jika aplikasi membutuhkan baca atau tulis cepat ke lokasi data tertentu (misalnya ID pengguna tertentu).
- **Data semi-terstruktur:** Misalnya catatan pengguna yang mungkin memiliki atribut yang bervariasi dan tidak cocok untuk skema relasional.

12.12.3 Struktur Data HBase

- **Table:** Koleksi data seperti tabel di database.
- **Row key:** Identitas unik untuk setiap baris.
- **Column Family:** Grup kolom yang disimpan bersama secara fisik.

- **Qualifier:** Nama kolom di dalam column family.
- **Timestamp:** Setiap sel menyimpan versi berdasarkan waktu.

12.12.4 Kelebihan HBase

- Waktu akses rendah untuk data besar
- Cocok untuk update data secara real-time
- Mendukung jutaan baris dan kolom

12.12.5 Keterbatasan HBase

- Tidak mendukung SQL standar (perlu integrasi dengan tools seperti Apache Phoenix)
- Tidak optimal untuk query agregasi atau laporan kompleks
- Membutuhkan perencanaan skema dan partisi dengan cermat

12.12.6 Contoh Aplikasi Dunia Nyata

- Penyimpanan data pengguna untuk layanan media sosial
- Log clickstream dalam sistem periklanan
- Sistem deteksi fraud berbasis histori transaksi

12.13 Studi Kasus: Analisis Log Server Besar

Masalah: Sebuah perusahaan cloud ingin menganalisis 5TB log server untuk mendeteksi aktivitas tidak normal.

Solusi:

1. Simpan log dalam HDFS
2. Gunakan Spark untuk parsing dan filtering log
3. Simpan hasil dalam Hive dan query untuk laporan

Insight: Hadoop memungkinkan pemrosesan data besar dalam hitungan menit, bukan jam.

12.14 Tren Terkini Big Data

Seiring perkembangan teknologi, pemanfaatan Big Data tidak lagi terbatas pada penyimpanan dan pemrosesan skala besar, tetapi telah berkembang menjadi bagian integral dari berbagai inovasi digital, termasuk kecerdasan buatan (AI), Internet of Things (IoT), dan layanan cloud modern.

Berikut adalah beberapa tren utama dalam ekosistem Big Data saat ini:

12.14.1 Integrasi dengan Machine Learning dan AI

Big Data kini menjadi **pondasi utama** untuk pengembangan model machine learning dan kecerdasan buatan.

- Dataset besar digunakan untuk melatih model klasifikasi, prediksi, deteksi anomali, dan sistem rekomendasi.
- Framework seperti Spark MLlib, TensorFlow, dan Scikit-learn banyak digunakan bersama data besar.
- Contoh aplikasi: rekomendasi produk, deteksi penipuan, chatbot cerdas.

12.14.2 Penggunaan Streaming Data

Big Data kini tidak hanya dianalisis secara batch, tetapi juga **secara real-time** melalui teknologi *data streaming*.

- Tools populer: **Apache Kafka**, **Apache Flink**, **Spark Streaming**
- Cocok untuk monitoring media sosial, transaksi finansial, IoT
- Memungkinkan pengambilan keputusan langsung saat data masuk

12.14.3 Kombinasi dengan Cloud Computing

Organisasi semakin banyak memindahkan beban kerja Big Data ke layanan **cloud** seperti:

- **Amazon Web Services (AWS):** S3, EMR, Redshift
- **Google Cloud Platform (GCP):** BigQuery, Dataflow
- **Microsoft Azure:** Azure Synapse, HDInsight

Keuntungan:

- Fleksibel dan elastis dalam skala penyimpanan dan komputasi
- Bayar sesuai pemakaian (pay-as-you-go)
- Tidak perlu infrastruktur fisik sendiri

12.14.4 Perpindahan ke Serverless dan Edge Computing

Dua paradigma baru yang kini banyak diadopsi:

- **Serverless Big Data:** Eksekusi tugas tanpa mengelola server secara langsung. Contoh: AWS Lambda, Google Cloud Functions.
- **Edge Computing:** Pemrosesan data dilakukan di dekat sumber data (misalnya di perangkat IoT) untuk mengurangi latensi dan beban jaringan.

Contoh Aplikasi:

- Smart camera yang menganalisis video langsung di perangkat
- Sensor pabrik yang memproses data di edge sebelum mengirim ke cloud

12.15 LATIHAN / TUGAS AKHIR BAB 12

1. **[Uraian]** Jelaskan 5V dari Big Data dan berikan contoh nyata masing-masing.
2. **[Uraian Visual]** Buat diagram arsitektur Hadoop: HDFS, YARN, dan MapReduce.
3. **[Coding]**
    - Buat SparkSession sederhana
    - Baca file CSV dan hitung jumlah baris
    - Filter baris berdasarkan kolom tertentu dan tampilkan
4. **[Studi Kasus]** Buat desain sistem Hadoop untuk menyimpan dan menganalisis data sensor dari ribuan kendaraan secara real-time.
5. **[Studi Kasus]** Bandingkan arsitektur Big Data berbasis Hadoop vs Spark untuk kasus analisis data transaksi keuangan nasional.

Daftar Pustaka

[1] A. Gandomi **and** M. Haider, "Beyond the hype: Big data concepts, methods, and analytics," *International Journal of Information Management*, **jourvol** 35, **number** 2, **pages** 137–144, 2015. DOI: 10.1016/j.ijinfomgt.2014.10.007.

[2] M. Chen, S. Mao **and** Y. Liu, "Big data: A survey," *Mobile networks and applications*, **jourvol** 19, **pages** 171–209, 2014.

[3] V. Mayer-Schönberger **and** K. Cukier, "Big data for development: Challenges and opportunities," *Journal of International Development*, **jourvol** 26, **number** 5, **pages** 591–608, 2014.

[4] J. Tang, B. Liu, Y. Wang, H. Wang, Y. Hu **and** H. Wang, "A survey on big data analytics: Challenges, open research issues and tools," *The Journal of Supercomputing*, **jourvol** 76, **number** 3, **pages** 2065–2083, 2020.

[5] Z. M. Khine **and** Z. Wang, "A survey on data storage and placement methodologies for Cloud-Big Data ecosystem," *The Journal of Supercomputing*, **jourvol** 73, **number** 6, **pages** 2410–2435, 2017.

[6] T. White, *Hadoop: The definitive guide*. O'Reilly Media, Inc., 2015.

[7] H. Karau, A. Konwinski, P. Wendell **and** M. Zaharia, *Learning Spark: Lightning-fast big data analysis*. O'Reilly Media, Inc., 2015.

[8] M. Zaharia, M. Chowdhury, T. Das **andothers**, “Apache Spark: a unified engine for big data processing,” *Communications of the ACM*, **journvol** 59, **number** 11, **pages** 56–65, 2016.

13 VISUALISASI DATA LANJUT

13.1 Tujuan Pembelajaran

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Memahami prinsip-prinsip desain visualisasi data yang baik.
2. Menggunakan berbagai jenis grafik untuk menampilkan pola dan hubungan antar data.
3. Membangun dashboard interaktif menggunakan Plotly, Dash, atau Streamlit.
4. Mengaplikasikan visualisasi spasial (peta) dengan GeoPandas.
5. Menafsirkan dan menyampaikan insight melalui media visual dengan tepat.

13.2 Pentingnya Visualisasi Data

Visualisasi data adalah proses menyajikan data dalam bentuk grafik, diagram, atau representasi visual lainnya agar lebih mudah dipahami oleh manusia. Dalam konteks ilmu da-

ta, visualisasi bukan sekadar membuat grafik yang menarik secara estetika, melainkan merupakan **alat komunikasi yang efektif** untuk menyampaikan insight dari data yang kompleks [1], [2].

Visualisasi berperan penting dalam semua tahap analisis data, mulai dari eksplorasi awal, pembentukan hipotesis, hingga pelaporan hasil analisis kepada pemangku kepentingan.

13.2.1 Manfaat Visualisasi Data

- **Mengidentifikasi Tren dan Outlier** Visualisasi memudahkan deteksi pola atau kecenderungan dalam data (misalnya kenaikan penjualan), serta mengungkap nilai-nilai ekstrem atau anomali yang tidak mudah terlihat dari tabel angka.
- **Mendukung Komunikasi dengan Non-Teknis** Tidak semua pemangku kepentingan memiliki latar belakang teknis. Representasi visual seperti diagram batang, peta panas, atau grafik garis membantu menyampaikan hasil analisis secara ringkas dan intuitif.
- **Validasi Asumsi dan Hipotesis** Visualisasi digunakan untuk menguji dugaan awal atau model statistik. Misalnya, scatter plot dapat membantu melihat hubungan antara dua variabel secara visual sebelum melakukan regresi.
- **Meningkatkan Daya Tarik dan Pemahaman** Grafik yang dirancang dengan baik tidak hanya memperindah pre-

sentasi, tetapi juga memperkuat pesan utama yang ingin disampaikan oleh analis data [3].

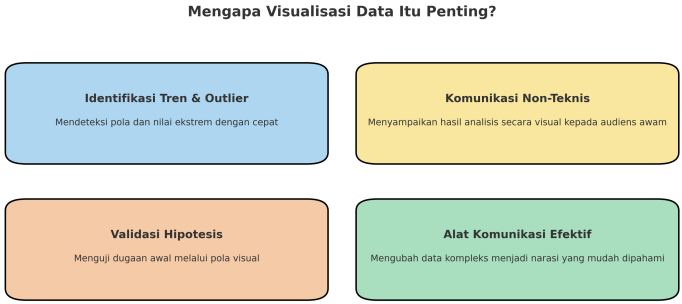
13.2.2 Contoh Aplikasi Visualisasi

- **Grafik Garis:** Memantau tren penjualan per bulan
- **Histogram:** Melihat distribusi usia pelanggan
- **Peta Panas (Heatmap):** Menilai korelasi antar fitur dalam dataset
- **Diagram Batang:** Membandingkan total penjualan per wilayah

13.2.3 Prinsip Penting

> *"Visualisasi bukan hanya tentang membuat grafik yang bagus, tapi tentang menyampaikan cerita di balik data."*

Oleh karena itu, seorang analis data harus memilih jenis visualisasi yang tepat, menggunakan warna dan skala dengan bijak, serta selalu mempertimbangkan audiens yang akan menerima informasi tersebut.



Gambar 29: Empat alasan utama pentingnya visualisasi data: mengidentifikasi pola, menjembatani komunikasi, memvalidasi hipotesis, dan menyampaikan pesan data secara efektif.

13.3 Dasar-Dasar Visualisasi yang Efektif

Visualisasi yang baik tidak hanya menarik secara estetika, tetapi juga menyampaikan informasi dengan jelas, ringkas, dan sesuai tujuan [2]. Dalam ilmu data, visualisasi digunakan untuk menjelaskan pola, menjawab pertanyaan, dan mendukung pengambilan keputusan.

Berikut adalah prinsip-prinsip dasar dalam membuat visualisasi data yang efektif:

13.3.1 Gunakan Jenis Grafik Sesuai Tujuan

Memilih jenis grafik yang tepat sangat penting untuk memastikan pesan yang ingin disampaikan dapat diterima dengan baik oleh audiens [1].

- **Tren Waktu** → **Line Chart** Cocok untuk menggambarkan perubahan suatu nilai sepanjang waktu (misal: penjualan bulanan).
- **Distribusi** → **Histogram atau Boxplot** Menunjukkan penyebaran data, outlier, dan sebaran nilai.
- **Perbandingan Kategori** → **Bar Chart atau Heatmap** Cocok untuk membandingkan nilai antar kelompok atau wilayah.
- **Relasi Antar Variabel** → **Scatter Plot atau Bubble Chart** Menunjukkan hubungan antara dua atau lebih variabel numerik.

13.3.2 Hindari "Chartjunk"

Chartjunk adalah elemen visual yang tidak relevan dengan data, seperti efek 3D, bayangan berlebihan, atau dekorasi yang justru mengaburkan informasi [3].

- Fokuslah pada kejelasan informasi, bukan sekadar hiasan.
- Hapus gridline, label, atau elemen yang tidak membantu pembacaan data.

13.3.3 Sertakan Elemen Konteks yang Jelas

Visualisasi yang efektif harus selalu disertai dengan:

- **Judul:** Menjelaskan isi visualisasi secara ringkas.
- **Label Sumbu:** Memberi arti pada sumbu X dan Y (misalnya "Waktu (bulan)" dan "Jumlah Penjualan").
- **Legenda:** Diperlukan jika ada kategori atau warna yang digunakan untuk membedakan data.

13.3.4 Gunakan Warna dengan Bijak

Warna harus digunakan untuk **membedakan dan mengarahkan perhatian**, bukan untuk menghias.

- Hindari penggunaan terlalu banyak warna
- Gunakan kontras yang cukup untuk membedakan kategori
- Gunakan palet warna konsisten dan ramah untuk buta warna (misalnya: viridis, colorbrewer) [1]

13.3.5 Uji Visualisasi Anda

Tanyakan:

- Apakah pesan utama tersampaikan dalam 5 detik?
- Apakah audiens yang tidak familiar dengan data bisa mengerti?

- Apakah visual ini membantu pengambilan keputusan?

13.4 Visualisasi dengan Matplotlib & Seaborn

Dalam Python, dua pustaka paling populer untuk membuat visualisasi data adalah **Matplotlib** dan **Seaborn**. Keduanya menyediakan alat yang fleksibel dan kaya fitur untuk menyajikan data dalam bentuk grafik.

13.4.1 Matplotlib

Matplotlib adalah pustaka visualisasi dasar di Python yang menyediakan fungsi untuk membuat grafik garis, batang, sebar, histogram, dan lainnya [4]. Modul yang paling sering digunakan adalah `matplotlib.pyplot`, yang berfungsi seperti antarmuka MATLAB.

13.4.2 Seaborn

Seaborn dibangun di atas Matplotlib dan menyediakan antarmuka tingkat tinggi untuk membuat visualisasi statistik yang menarik dan informatif [5].

Kelebihan Seaborn:

- Antarmuka deklaratif dan sederhana
- Integrasi langsung dengan `pandas DataFrame`
- Palet warna otomatis yang estetis

- Dukungan visualisasi statistik seperti distribusi, regresi, dan korelasi

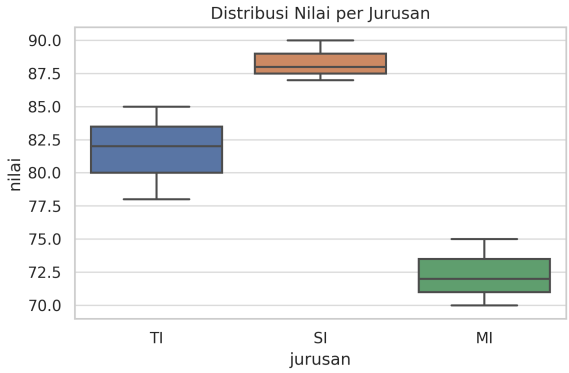
13.4.3 Contoh: Boxplot per Jurusan

Kode berikut digunakan untuk melihat distribusi nilai mahasiswa berdasarkan jurusan menggunakan **boxplot**:

```
import matplotlib.pyplot as plt
import seaborn as sns

sns.set(style="whitegrid")
sns.boxplot(x="jurusan", y="nilai", data=df)
plt.title("Distribusi Nilai per Jurusan")
plt.show()
```

Penjelasan:

- `boxplot()` membuat diagram box untuk setiap kategori kolom `jurusan`
- Visualisasi ini berguna untuk melihat median, kuartil, dan outlier dalam data numerik



Gambar 30: Distribusi nilai mahasiswa berdasarkan jurusan menggunakan boxplot.

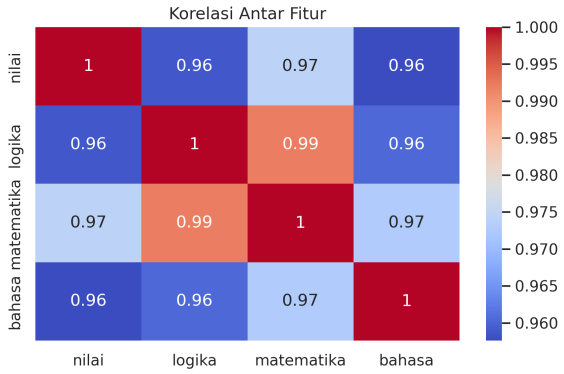
13.4.4 Contoh: Korelasi Antar Fitur

Berikut adalah contoh visualisasi korelasi antar fitur numerik dalam dataset:

```
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.title("Korelasi Antar Fitur")
```

Penjelasan:

- `df.corr()` menghitung matriks korelasi antar fitur numerik.
- `sns.heatmap()` menampilkan korelasi tersebut sebagai peta panas.
- `annot=True` menunjukkan nilai korelasi secara eksplisit di setiap kotak.
- `cmap="coolwarm"` menggunakan skema warna untuk memperjelas kekuatan dan arah korelasi.



Gambar 31: Peta panas (heatmap) korelasi antar fitur numerik dalam dataset.

13.4.5 Tips Praktis

- Gunakan `plt.tight_layout()` untuk mencegah label grafik saling tumpang tindih.
- Manfaatkan parameter `palette` atau `hue` di Seaborn untuk membedakan kategori melalui warna.
- Kombinasikan dengan `pandas` untuk eksplorasi cepat menggunakan `df.plot()`, `df.hist()`, dan fungsi visual lainnya.

13.5 Visualisasi Interaktif dengan Plotly

Plotly adalah pustaka visualisasi Python modern yang mendukung grafik interaktif berbasis web. Berbeda dengan Matplotlib dan Seaborn yang statis, visualisasi dengan Plotly me-

mungkinkan interaksi seperti **zoom, hover, klik, filter**, dan eksplorasi langsung oleh pengguna.

13.5.1 Fitur Utama Plotly

- Mendukung visualisasi interaktif berbasis JavaScript
- Cocok untuk eksplorasi data secara langsung (E-D-A)
- Dapat digunakan bersama framework dashboard seperti **Dash** dan **Streamlit**
- Mendukung ekspor sebagai HTML untuk dibagikan atau diintegrasikan ke situs web

13.5.2 Contoh Visualisasi: Scatter Plot Interaktif

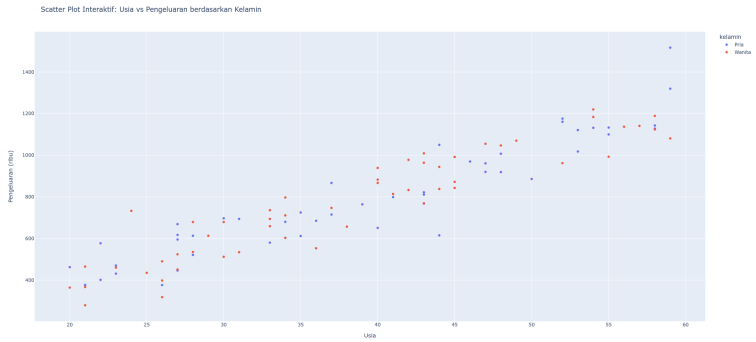
Kode berikut menghasilkan diagram sebar interaktif berdasarkan jenis kelamin:

```
import plotly.express as px

fig = px.scatter(df, x="usia", y="pengeluaran", color="kelamin")
fig.show()
```

Penjelasan:

- `px.scatter()` membuat scatter plot berdasarkan dua variabel numerik
- Parameter `color="kelamin"` memberikan warna berbeda untuk setiap kategori

- Grafik yang dihasilkan dapat di-zoom, dihover untuk detail nilai, dan disaring secara interaktif



Gambar 32: Scatter plot statis: hubungan antara usia dan pengeluaran berdasarkan jenis kelamin.

13.5.3 Keuntungan Menggunakan Plotly

- Sangat cocok untuk **eksplorasi data** karena memungkinkan interaksi langsung
- Ideal untuk presentasi dinamis atau integrasi ke **dashboard web**
- Mendukung berbagai jenis grafik: line, bar, box, histogram, 3D plot, dan peta geografis

13.5.4 Penggunaan dalam Dashboard

Plotly dapat digunakan sebagai komponen visualisasi utama dalam:

- **Dash:** Framework dari Plotly untuk membangun dashboard berbasis Python murni
- **Streamlit:** Framework sederhana dan cepat untuk membuat aplikasi web interaktif dari skrip Python

13.6 Membuat Dashboard dengan Streamlit

Streamlit adalah framework Python open-source yang memungkinkan pengguna membuat aplikasi web interaktif untuk analisis data hanya dengan beberapa baris kode. Berbeda dengan framework web tradisional seperti Flask atau Django, Streamlit dirancang khusus untuk kebutuhan *data science* dan *machine learning*.

13.7 Kelebihan Streamlit

- Sangat mudah digunakan: hanya perlu Python, tanpa HTML/JS
- Mendukung pemuatan data, visualisasi, dan kontrol UI secara langsung
- Dapat diintegrasikan dengan pustaka visualisasi populer seperti Matplotlib, Seaborn, dan Plotly
- Ideal untuk prototipe cepat dan berbagi analisis secara online

13.7.1 Instalasi Streamlit

Untuk mulai menggunakan Streamlit, jalankan perintah berikut di terminal:

```
pip install streamlit
```

13.7.2 Contoh Aplikasi Dashboard Sederhana

Contoh berikut menunjukkan bagaimana membuat dashboard sederhana untuk menampilkan data mahasiswa:

```
# app.py
import streamlit as st
import pandas as pd

st.title("Dashboard Analisis Mahasiswa")
df = pd.read_csv("mahasiswa.csv")
st.dataframe(df)
```

Penjelasan:

- `st.title()` digunakan untuk menampilkan judul halaman
- `st.dataframe()` menampilkan isi tabel dari file CSV secara interaktif

13.7.3 Menjalankan Aplikasi Streamlit

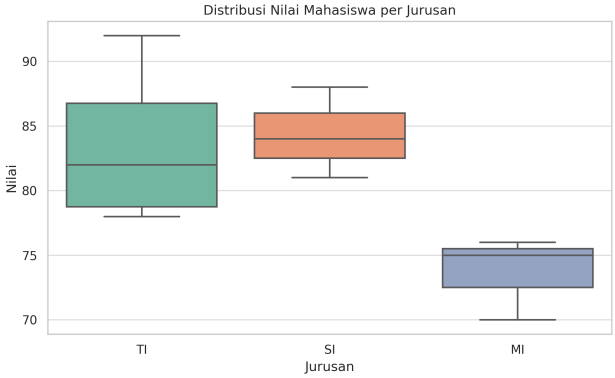
Setelah file `app.py` siap, jalankan dengan perintah:

```
streamlit run app.py
```

Browser akan terbuka secara otomatis dan menampilkan dashboard interaktif Anda.

13.7.4 Fitur Lanjutan Streamlit

- `st.file_uploader()`: untuk mengunggah file CSV langsung dari browser
- `st.sidebar`: membuat sidebar untuk kontrol filter, checkbox, slider
- `st.plotly_chart()`, `st.pyplot()`: menampilkan grafik interaktif dari Plotly, Matplotlib, dll.
- `st.map()`, `st.metric()`: untuk peta dan visual indikator metrik



Gambar 33: Visualisasi boxplot dalam dashboard Streamlit untuk menganalisis distribusi nilai mahasiswa berdasarkan jurusan.

13.8 Visualisasi Spasial (Pemetaan) dengan GeoPandas

GeoPandas adalah pustaka Python yang memperluas kemampuan pandas untuk bekerja dengan data spasial atau geospasial, seperti data peta, koordinat, dan wilayah geografis [6]. GeoPandas memungkinkan pemrosesan dan visualisasi data spasial secara langsung, seperti membuat peta berbasis warna (choropleth), plotting titik koordinat, dan overlay antar wilayah.

13.8.1 Fitur Utama GeoPandas

- Mendukung pemrosesan data spasial: poligon, titik, garis
- Dapat membaca file `shapefile`, GeoJSON, dan data geografis lainnya
- Integrasi erat dengan matplotlib dan shapely
- Memungkinkan pembuatan visualisasi seperti **choropleth map** dan **pemetaan titik**

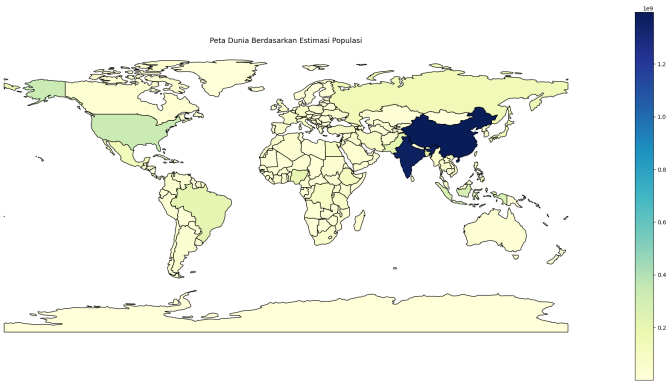
13.8.2 Contoh Kode Visualisasi Peta Global Berdasarkan Estimasi Populasi

```
#!/usr/bin/env python3
import geopandas as gpd
import matplotlib.pyplot as plt

def main():
    # Path to your Natural Earth shapefile
    shapefile_path = "data/ne_110m_admin_0_countries.shp"
    # Output image path
    output_path = "peta_populasi_dunia.png"

    # 1. Read the shapefile
    gdf = gpd.read_file(shapefile_path)

    # 2. Normalize column names to lowercase
    gdf.columns = gdf.columns.str.lower()

    # 3. Sanity check: make sure 'pop_est' is present
    if 'pop_est' not in gdf.columns:
        raise KeyError(
            f"'pop_est' column not found. Available columns: {
    gdf.columns.tolist()}"
        )

    # 4. Plot
    fig, ax = plt.subplots(figsize=(12, 6))
    gdf.plot(
        column='pop_est',       # now matches the lowercase
    column
        cmap='YlGnBu',
```

```
        legend=True,
        edgecolor='black',
        ax=ax
    )

    # 5. Styling
    ax.set_title('Peta Dunia Berdasarkan Estimasi Populasi',
    fontsize=13)
    ax.set_axis_off()
    plt.tight_layout()

    # 6. Save and show
    plt.savefig(output_path, dpi=300)
    plt.show()

if __name__ == "__main__":
    main()
```

Penjelasan:

- `read_file()` digunakan untuk membaca data shapefile atau GeoDataFrame
- Parameter `column='gdp_md_est'` menunjukkan variabel yang digunakan untuk mewarnai peta
- `cmap='Oranges'` menentukan skema warna
- `legend=True` menambahkan legenda otomatis



Gambar 34: Peta dunia berdasarkan estimasi populasi per negara. Warna yang lebih gelap menunjukkan jumlah penduduk yang lebih besar. Data divisualisasikan menggunakan GeoPandas.

13.8.3 Contoh Aplikasi Nyata

- **Penyebaran COVID-19 per provinsi:** Memvisualisasikan jumlah kasus di tiap wilayah.
- **Visualisasi Titik Kejahatan:** Menandai lokasi insiden kriminal untuk analisis keamanan wilayah.
- **Pemetaan Pelanggan Potensial:** Mengidentifikasi wilayah dengan konsentrasi pelanggan tinggi berdasarkan data koordinat.

13.8.4 Jenis Visualisasi yang Didukung GeoPandas

- Choropleth map (warna berdasarkan nilai variabel)

- Titik lokasi (Point)
- Overlay spasial antar wilayah
- Buffer area untuk radius analisis

13.9 Visualisasi Time Series

Visualisasi time series digunakan untuk menganalisis data yang berubah dari waktu ke waktu. Ini sangat penting dalam banyak bidang seperti keuangan, penjualan, cuaca, dan kesehatan [7], [8]. Visualisasi ini membantu dalam memahami pola, tren jangka panjang, dan variasi musiman dalam data.

13.9.1 Pentingnya Time Series

Data deret waktu (*time series*) mengandung informasi berurutan berdasarkan waktu. Dengan memvisualisasikannya, kita dapat:

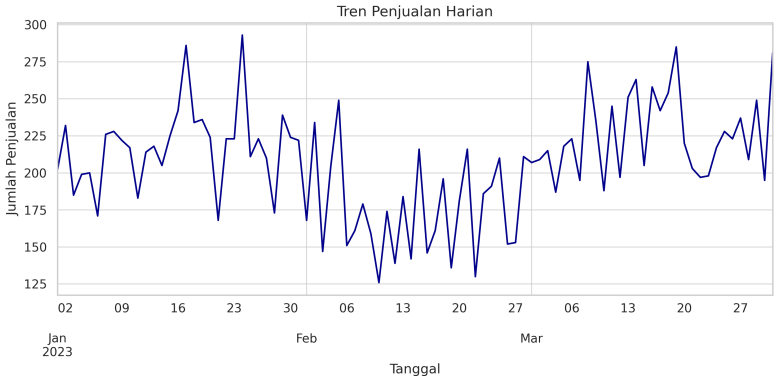
- Melihat tren jangka panjang (naik/turun)
- Mengidentifikasi pola musiman (mingguan, bulanan, tahunan)
- Mendeteksi anomali atau outlier
- Menilai dampak suatu kejadian (misalnya promosi, pandemi)

13.9.2 Contoh Visualisasi Time Series dengan Matplotlib

```
df['tanggal'] = pd.to_datetime(df['tanggal'])  # Konversi ke format waktu
df.set_index('tanggal')['penjualan'].plot()
plt.title("Tren Penjualan Harian")
```

Penjelasan:

- `pd.to_datetime()` mengubah kolom teks menjadi format datetime
- `set_index('tanggal')` menjadikan tanggal sebagai indeks
- `plot()` membuat grafik garis dari kolom penjualan



Gambar 35: Visualisasi time series penjualan harian dengan pola musiman. Grafik ini membantu mengidentifikasi tren dan variasi dari waktu ke waktu.

13.9.3 Jenis Pola dalam Time Series

- **Tren:** Perubahan jangka panjang, misalnya penjualan meningkat dari tahun ke tahun
- **Musiman:** Pola yang berulang secara reguler (misalnya lonjakan pembelian saat Lebaran atau akhir tahun)
- **Siklus:** Perubahan jangka panjang tetapi tidak beraturan (misalnya fluktuasi ekonomi)
- **Noise:** Variasi acak yang tidak dapat diprediksi

13.9.4 Insight dari Visualisasi Time Series

- Apakah penjualan meningkat secara konsisten?
- Apakah ada lonjakan di akhir bulan atau akhir tahun?
- Apakah ada efek dari kampanye promosi atau libur nasional?
- Apakah ada anomali atau penurunan drastis?

13.10 Studi Kasus: Visualisasi COVID-19 Global

Pada masa pandemi COVID-19, visualisasi data menjadi alat yang sangat penting untuk membantu pemerintah, lembaga kesehatan, dan masyarakat umum dalam memahami si-

tuasi secara cepat dan akurat. Salah satu bentuk visualisasi yang sangat berguna adalah peta interaktif berbasis data kasus harian dan kematian dari seluruh dunia.

13.10.1 Dataset

Data yang digunakan dalam studi kasus ini berasal dari **World Health Organization (WHO)**, berisi:

- Jumlah kasus dan kematian harian
- Nama negara dan wilayah
- Tanggal laporan

13.10.2 Langkah-Langkah Visualisasi

1. **Parsing dan Agregasi Data per Negara:** Menggabungkan data berdasarkan negara dan waktu, serta menghitung total kasus, kematian, dan jumlah kasus aktif.
2. **Visualisasi Tren Waktu Global:** Membuat grafik time series jumlah kasus global per hari untuk melihat puncak pandemi, tren naik/turun, serta dampak kebijakan.
3. **Pemetaan Kasus Aktif per Negara:** Menggunakan peta dunia untuk menampilkan jumlah kasus aktif per negara. Negara dengan jumlah kasus lebih tinggi ditampilkan dalam warna lebih gelap (choropleth).
4. **Dashboard Interaktif:** Menyediakan filter interaktif berdasarkan negara, tanggal, dan metrik (kasus, kematian,

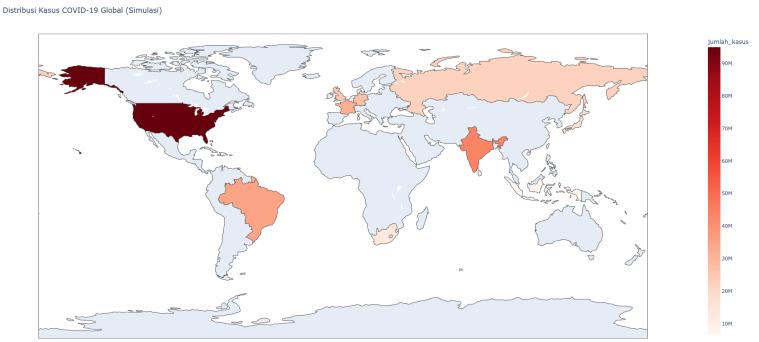
kesembuhan). Dapat dibangun menggunakan Plotly + Streamlit atau Dash.

13.10.3 Contoh Kode: Choropleth Map dengan Plotly

```
import plotly.express as px

fig = px.choropleth(df,
    locations="negara",
    locationmode="country names",
    color="jumlah_kasus",
    hover_name="negara"
)
fig.show()
```

Penjelasan:

- `locations="negara"` menunjuk ke kolom berisi nama negara
- `color="jumlah_kasus"` mengontrol intensitas warna berdasarkan nilai
- `hover_name="negara"` menampilkan nama negara saat kursor diarahkan



Gambar 36: Peta choropleth kasus COVID-19 global (simulasi) berdasarkan jumlah kasus per negara. Warna merah menunjukkan tingkat keparahan penyebaran.

13.10.4 Output dan Manfaat

- Menampilkan "zona merah" secara real-time di seluruh dunia
- Memudahkan pengambil kebijakan dalam mengambil keputusan berdasarkan kondisi terkini
- Dapat digunakan untuk alokasi sumber daya, pelaporan media, dan edukasi publik

13.11 Tips Penyampaian Insight Visual

Visualisasi data yang baik bukan hanya soal grafik yang indah, tetapi juga soal bagaimana grafik tersebut menyam-

paikan **pesan atau insight** yang kuat dan mudah dipahami. Di sinilah pentingnya **penyampaian insight visual** yang efektif, yang menggabungkan desain grafis, pemilihan data yang tepat, serta narasi yang kuat.

13.11.1 Fokus pada Storytelling

Visualisasi yang efektif harus membentuk narasi yang jelas dan logis, layaknya bercerita. Setiap grafik sebaiknya menjawab satu pertanyaan utama:

- Apa tren yang sedang terjadi?
- Apakah ada perubahan signifikan?
- Apa kesimpulan yang harus diambil oleh pembaca?

Contoh: Alih-alih hanya menunjukkan total penjualan, tampilkan grafik yang memperlihatkan penjualan naik pesat setelah kampanye iklan.

13.11.2 Tunjukkan Perbandingan, Bukan Hanya Nilai Tunggal

Angka absolut seringkali kurang bermakna jika tidak dibandingkan dengan konteksnya. Gunakan grafik untuk membandingkan antar waktu, wilayah, atau kelompok.

- Gunakan bar chart untuk membandingkan pendapatan antar divisi
- Gunakan line chart untuk membandingkan tren dua produk

Tujuan: Membantu pembaca memahami *"dibandingkan dengan apa"*.

13.11.3 Gunakan Anotasi untuk Highlight

Tambahkan teks atau tanda visual untuk menyoroti informasi penting dalam grafik:

- Tanda panah, garis bantu, atau lingkaran untuk menarik perhatian
- Label langsung pada puncak atau penurunan tajam
- Catatan singkat seperti "terjadi lonjakan saat promo"

Manfaat: Anotasi membantu memperjelas pesan utama grafik dan mencegah interpretasi yang salah.

13.11.4 Integrasikan Narasi dengan Grafik

Gabungkan grafik dengan narasi dalam satu alur yang saling melengkapi, baik di dalam laporan maupun dashboard interaktif.

- Tambahkan judul deskriptif (bukan hanya "Grafik 1" tapi misalnya "Penjualan Meningkat Setelah Promo")
- Berikan ringkasan insight di bawah grafik
- Gunakan filter interaktif agar pembaca bisa mengeksplorasi sendiri

13.12 Tantangan Visualisasi Data

Visualisasi data adalah alat penting dalam ilmu data untuk menyampaikan informasi secara intuitif. Namun, visualisasi juga memiliki tantangan tersendiri yang dapat menyebabkan misinterpretasi data jika tidak dirancang dengan baik. Berikut adalah beberapa tantangan umum dalam visualisasi data:

13.12.1 Overplotting pada Scatter Plot dengan Data Besar

Overplotting terjadi ketika terlalu banyak titik data diplot dalam satu grafik scatter, sehingga titik-titik saling menumpuk dan sulit dibedakan. Akibatnya, pola atau distribusi data menjadi tidak terlihat jelas.

Solusi:

- Gunakan transparansi (alpha) pada titik-titik.
- Terapkan teknik *sampling* atau *binning*.
- Gunakan visualisasi alternatif seperti hexbin plot atau density plot.

13.12.2 Pemilihan Skala Tidak Tepat (Linear vs Log)

Pemilihan skala yang tidak sesuai dapat menyebabkan distorsi dalam interpretasi data, terutama jika data memiliki distribusi yang tidak normal atau memiliki nilai ekstrem.

Contoh:

- Menggunakan skala linear untuk data pendapatan yang memiliki rentang dari ribuan hingga jutaan akan membuat nilai kecil tidak terlihat.
- Skala logaritmik lebih sesuai untuk data eksponensial, seperti pertumbuhan populasi atau data biologis.

Solusi: Pertimbangkan karakteristik distribusi data sebelum memilih skala sumbu.

13.12.3 Warna Tidak Ramah untuk Penderita Buta Warna

Penggunaan warna yang tidak mempertimbangkan aksesibilitas dapat menyulitkan audiens dengan gangguan penglihatan warna (color blindness) dalam memahami grafik.

Solusi:

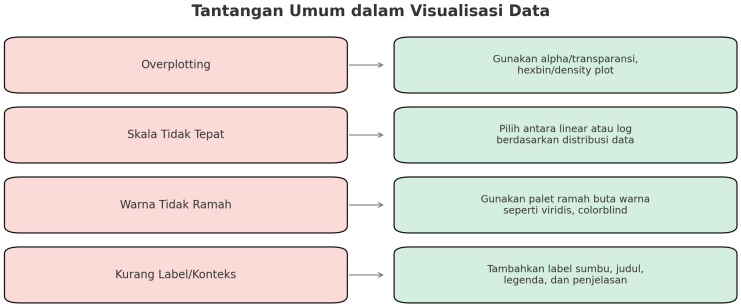
- Gunakan palet warna ramah buta warna (seperti palet `colorblind` dari Seaborn atau `viridis` dari Matplotlib).
- Hindari mengandalkan warna saja untuk membedakan kategori – tambahkan bentuk, garis, atau label.

13.12.4 Visualisasi Membingungkan Tanpa Label atau Konteks

Grafik yang tidak memiliki label sumbu, judul, atau legenda dapat menyebabkan kebingungan dan interpretasi yang salah. Audiens tidak akan memahami apa yang digambarkan jika konteks tidak jelas.

Solusi:

- Selalu berikan judul, label sumbu X dan Y, serta legenda jika menggunakan warna atau simbol yang berbeda.
- Tambahkan keterangan singkat atau insight yang dapat membantu pembaca memahami makna visualisasi.



Gambar 37: Tantangan dan solusi umum dalam visualisasi data. Desain visualisasi yang baik mempertimbangkan aspek teknis, interpretatif, dan aksesibilitas.

13.13 LATIHAN / TUGAS AKHIR BAB 13

1. **[Uraian]** Jelaskan prinsip desain visualisasi yang baik, dan bagaimana visualisasi buruk dapat menyesatkan.
2. **[Uraian Visual]** Gambar desain mockup dashboard sederhana untuk menampilkan data nilai dan kehadiran mahasiswa.
3. **[Coding]**
    - Buat histogram, boxplot, dan heatmap dari dataset kampus
    - Bangun aplikasi Streamlit sederhana untuk upload dan tampilkan data CSV
    - Buat peta kasus COVID-19 Indonesia dengan GeoPandas
4. **[Studi Kasus]** Bangun dashboard interaktif menggunakan Plotly atau Dash untuk data penjualan online selama 1 tahun.
5. **[Studi Kasus]** Identifikasi kesalahan visualisasi dari beberapa grafik media online (sumber data boleh simulasi), lalu perbaiki dan jelaskan alasannya.

Daftar Pustaka

[1] C. O. Wilke, *Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures*. O'Reilly Media, 2019.

[2] S. R. Midway, "Principles of Effective Data Visualization," *Patterns*, **jourvol** 1, **number** 6, **page** 100 141, 2020.

[3] S. D. H. Evergreen, *Effective Data Visualization: The Right Chart for the Right Data*. SAGE Publications, 2016.

[4] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in science & engineering*, **jourvol** 9, **number** 3, **pages** 90–95, 2007.

[5] M. L. Waskom, "Seaborn: statistical data visualization," *Journal of Open Source Software*, **jourvol** 6, **number** 60, **page** 3021, 2021.

[6] K. Jordahl, *GeoPandas: Python tools for geographic data*, `https://geopandas.org`, Accessed April 2025, 2014.

[7] R. J. Hyndman **and** G. Athanasopoulos, *Forecasting: Principles and Practice*. OTexts, 2018, Accessed April 2025. **url**: `https://otexts.com/fpp3/`.

[8] C. Chatfield, *The Analysis of Time Series: An Introduction*. Chapman **and** Hall/CRC, 2003.

14 ETIKA, PRIVASI, DAN HUKUM DALAM ILMU DATA

14.1 Tujuan Pembelajaran

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Menjelaskan prinsip-prinsip dasar etika dalam Ilmu Data.
2. Memahami pentingnya privasi data dan hak individu atas informasi mereka.
3. Mengenal peraturan hukum seperti GDPR dan UU ITE terkait pengelolaan data.
4. Menganalisis kasus nyata penyalahgunaan data dan implikasinya.
5. Menerapkan prinsip-prinsip etis dalam proyek data science.

14.2 Pentingnya Etika dalam Ilmu Data

Ilmu Data (*Data Science*) telah menjadi kekuatan utama dalam era digital modern [1], [2]. Dengan kemampuan untuk mengolah data dalam skala besar, menemukan pola tersembunyi, dan membuat prediksi yang sangat akurat, ilmu data dapat membantu dalam pengambilan keputusan strategis di berbagai sektor seperti kesehatan, keuangan, pemerintahan, dan pendidikan.

Namun, seperti halnya teknologi yang kuat lainnya, ilmu data juga membawa potensi risiko yang besar. Ketika digunakan tanpa pertimbangan etika, analisis data dapat menghasilkan konsekuensi serius [3] bagi individu maupun masyarakat.

14.2.1 Dampak Negatif Jika Ilmu Data Disalahgunakan

- **Diskriminasi Algoritmik** Model prediktif dapat memperkuat bias yang sudah ada dalam data historis [4]. Misalnya, sistem perekrutan otomatis yang melatih model pada data yang bias gender dapat menolak pelamar dari kelompok tertentu tanpa alasan yang adil.
- **Kebocoran Data Pribadi** Tanpa perlindungan privasi yang kuat, informasi sensitif individu dapat terekspos ke pihak yang tidak berwenang, menimbulkan risiko

keamanan, peretasan, atau penyalahgunaan identitas.

- **Manipulasi Opini Publik** Data dapat digunakan untuk menargetkan individu dengan informasi yang menyesatkan, seperti dalam kasus *microtargeting* politik atau penyebaran disinformasi melalui media sosial.
- **Pelanggaran Hak Individu** Aktivitas seperti pelacakan lokasi, pengenalan wajah massal, atau analisis data medis tanpa persetujuan dapat melanggar prinsip otonomi dan hak privasi seseorang.

14.2.2 Prinsip Etika dalam Ilmu Data

"With great data comes great responsibility."

Ilmuwan data memiliki tanggung jawab untuk tidak hanya memaksimalkan akurasi model [5], tetapi juga mempertimbangkan konsekuensi sosial dan etis dari sistem yang mereka bangun.

Beberapa prinsip utama yang harus dipegang:

- **Transparansi:** Penjelasan tentang bagaimana data dikumpulkan, dianalisis, dan digunakan harus terbuka.
- **Akunabilitas:** Tanggung jawab harus diambil terhadap kesalahan atau dampak yang dihasilkan dari sistem berbasis data.
- **Keadilan (Fairness):** Sistem harus dirancang untuk menghindari bias dan diskriminasi terhadap kelompok tertentu.

- **Privasi dan Perlindungan Data:** Informasi pribadi harus dijaga dan hanya digunakan sesuai dengan persetujuan yang diberikan.

14.3 Prinsip Etika dalam Ilmu Data

Etika dalam ilmu data tidak hanya tentang mengikuti hukum atau regulasi, tetapi juga mencakup tanggung jawab moral dan sosial dalam merancang, membangun, dan menerapkan sistem berbasis data. Untuk itu, ada sejumlah prinsip dasar yang wajib dijunjung tinggi oleh setiap ilmuwan data dalam setiap proyek yang mereka kerjakan.

14.3.1 Prinsip-Prinsip Etika Utama

Prinsip	Penjelasan
Akuntabilitas	Bertanggung jawab atas penggunaan data dan hasil prediksi
Transparansi	Menjelaskan bagaimana data digunakan dan model dibangun
Non-diskriminasi	Tidak membuat keputusan yang bias terhadap kelompok tertentu
Konsen (Consent)	Pengumpulan data harus dilakukan atas izin pemilik data
Keamanan	Melindungi data dari akses dan penggunaan tidak sah

Tabel 9: Prinsip Penggunaan dan Pengelolaan Data

14.3.2 Implementasi dalam Praktik

Mengikuti prinsip-prinsip etika ini berarti:

- Melibatkan tim multidisiplin (data scientist, ahli hukum, etika, dan perwakilan publik).
- Melakukan uji fairness dan audit bias secara rutin terhadap model.
- Mengembangkan dokumentasi (datasheet/model card) untuk tiap dataset dan model.
- Menyediakan kanal keluhan dan koreksi jika terjadi dampak negatif dari sistem.

14.4 Privasi dan Perlindungan Data Pribadi

Data pribadi adalah setiap informasi yang dapat digunakan secara langsung atau tidak langsung untuk mengidentifikasi individu tertentu [6]. Dalam konteks ilmu data, pelanggaran terhadap privasi dapat berdampak serius terhadap hak individu, reputasi lembaga, bahkan melanggar hukum perlindungan data.

14.4.1 Contoh Data Pribadi

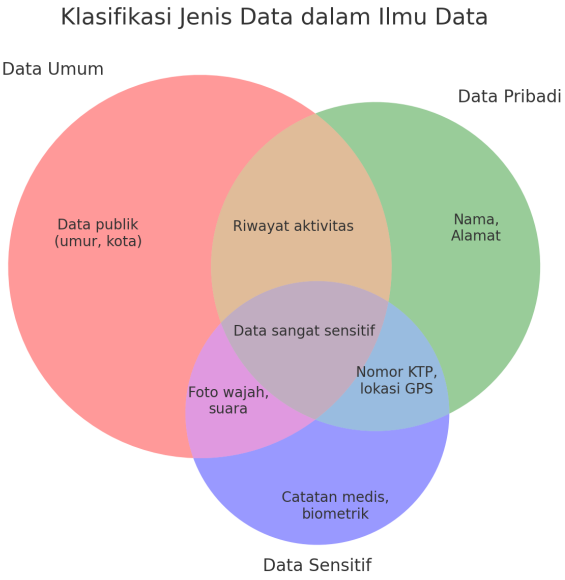
Beberapa jenis informasi yang tergolong sebagai data pribadi antara lain:

- **Informasi identitas**: nama lengkap, alamat rumah, nomor identitas (KTP, paspor).
- **Data lokasi**: koordinat GPS, histori lokasi dari aplikasi ponsel.
- **Riwayat aktivitas**: transaksi belanja online, histori pencarian, rekaman percakapan.
- **Data sensitif**: catatan medis, kondisi kesehatan, riwayat psikologis.
- **Biometrik**: foto wajah, sidik jari, suara, pola retina.

14.4.2 Prinsip Privasi yang Harus Diterapkan

Untuk melindungi privasi individu dalam proyek ilmu data, beberapa prinsip berikut harus dijunjung tinggi:

- **Minimasi Data** Kumpulkan hanya data yang benar-benar dibutuhkan untuk tujuan analisis. Hindari pengumpulan data berlebihan atau yang tidak relevan.
- **Anonimisasi dan Pseudonimisasi** Proses untuk menyamarkan identitas pengguna dari data yang dikumpulkan.
  - **Anonimisasi:** menghapus semua informasi identitas sehingga data tidak dapat dikaitkan dengan individu tertentu.
  - **Pseudonimisasi:** mengganti data identitas dengan kode unik, namun masih bisa dikembalikan dengan kunci tertentu.
- **Pembatasan Akses** Data sensitif harus hanya dapat diakses oleh pihak yang berwenang, dengan pengamanan seperti autentikasi ganda, enkripsi, dan kontrol hak akses.



Gambar 38: Diagram Venn klasifikasi data: umum, pribadi, dan sensitif. Pemahaman jenis data ini penting untuk menentukan strategi perlindungan yang sesuai.

14.5 Hukum dan Regulasi Terkait

Etika dalam ilmu data tidak dapat dilepaskan dari aspek hukum dan regulasi yang mengatur perlindungan data pribadi [7], [8]. Regulasi ini penting untuk memastikan bahwa penggunaan data dilakukan secara sah, adil, dan bertanggung jawab. Beberapa peraturan penting yang relevan bagi praktisi ilmu data antara lain:

14.5.1 GDPR (General Data Protection Regulation) – Uni Eropa

GDPR adalah regulasi utama Uni Eropa yang mulai berlaku sejak 25 Mei 2018, bertujuan untuk melindungi data pribadi individu dalam wilayah Eropa.

Fitur utama GDPR:

- Memberikan **hak akses** bagi individu terhadap data pribadi mereka.
- Memberikan **hak untuk koreksi dan penghapusan data** (termasuk *right to be forgotten*).
- Mewajibkan organisasi untuk melaporkan kebocoran data kepada otoritas dalam waktu maksimum **72 jam**.
- Memberikan sanksi yang sangat berat: hingga **€20 juta** atau **4% dari total pendapatan global tahunan** [9].

14.5.2 Undang-Undang ITE (Informasi dan Transaksi Elektronik) – Indonesia

UU No. 11 Tahun 2008 tentang Informasi dan Transaksi Elektronik (ITE) merupakan landasan hukum pertama di Indonesia terkait transaksi elektronik dan perlindungan data.

Poin penting:

- Melindungi keamanan informasi elektronik dan transaksi digital.

- **Pasal 26** mengatur bahwa penggunaan informasi pribadi seseorang harus dilakukan atas persetujuan subjek data **manik2020cyberlaw**.
- Memberikan sanksi pidana atas penyalahgunaan atau peretasan data pribadi dan sistem elektronik.

14.5.3 RUU PDP (Rancangan Undang-Undang Perlindungan Data Pribadi)

RUU PDP adalah inisiatif pemerintah Indonesia untuk menghadirkan regulasi yang setara dengan GDPR dan mengisi kekosongan hukum perlindungan data pribadi yang lebih komprehensif.

Poin kunci dalam RUU PDP:

- Menetapkan **peran pengendali data** dan **prosesor data** beserta tanggung jawabnya.
- Menentukan bahwa pelanggaran terhadap data pribadi harus dilaporkan kepada otoritas dalam waktu tertentu.
- Meningkatkan perlindungan terhadap hak subjek data, termasuk hak untuk mengetahui, memperbaiki, dan menghapus data.

14.6 Studi Kasus: Penyalahgunaan Data oleh Facebook – Cambridge Analytica

Kasus Facebook–Cambridge Analytica adalah salah satu skandal terbesar dalam sejarah pelanggaran data pribadi. Kasus ini menjadi contoh nyata bagaimana kegagalan dalam menerapkan prinsip etika dan regulasi dapat berujung pada konsekuensi sosial, hukum, dan reputasi yang sangat serius.

14.6.1 Kronologi Kasus

1. Sebuah aplikasi kuis kepribadian bernama *"thisisyourdigitallife"* dikembangkan oleh peneliti Aleksandr Kogan dan tersedia di Facebook.

2. Aplikasi ini mengklaim sebagai alat penelitian akademik, tetapi secara diam-diam mengumpulkan data pribadi tidak hanya dari pengguna yang menginstal aplikasi, tetapi juga dari teman-teman mereka — tanpa persetujuan eksplisit.

3. Data yang dikumpulkan mencakup profil pribadi, preferensi, dan aktivitas pengguna. Sebanyak **87 juta akun Facebook** terdampak.

4. Data ini kemudian dijual kepada **Cambridge Analyti-**

ca, perusahaan konsultan politik yang menggunakannya untuk membangun profil psikografis pemilih.

5. Profil tersebut digunakan untuk melakukan *microtargeting* dalam kampanye politik, termasuk dalam pemilihan Presiden AS 2016 dan kampanye *Brexit*.
6. Facebook dianggap gagal menjaga privasi data penggunanya dan tidak segera menginformasikan kebocoran data kepada publik.

14.6.2 Dampak dan Sanksi

- Facebook didenda **$5 miliar** oleh Federal Trade Commission (FTC) — denda terbesar dalam sejarah terkait privasi data.
- Reputasi Facebook mengalami penurunan drastis. Kepercayaan publik merosot, dan gerakan *#DeleteFacebook* sempat viral.
- CEO Facebook, Mark Zuckerberg, harus memberikan kesaksian di hadapan Kongres AS dan Parlemen Eropa.

14.6.3 Pelajaran Etis

- **Transparansi dan Konsen:** Pengumpulan dan penggunaan data harus dilakukan dengan persetujuan jelas dari pengguna.
- **Akuntabilitas:** Platform harus bertanggung jawab atas aplikasi pihak ketiga yang berjalan di atas sistem mereka.

- **Pengawasan Regulatif:** Kasus ini mendorong reformasi kebijakan privasi dan memperkuat perlindungan hukum atas data pribadi secara global.

  Pelajaran penting: Bahkan perusahaan teknologi terbesar di dunia bisa gagal menjaga privasi dan akuntabilitas — menunjukkan bahwa tata kelola data yang baik adalah kebutuhan universal.

14.7 Bias dan Diskriminasi dalam Model Data

Model pembelajaran mesin dibangun berdasarkan pola yang ditemukan dalam data historis. Namun, jika data yang digunakan mengandung bias—baik yang eksplisit maupun implisit—maka model yang dihasilkan pun berisiko mereproduksi atau bahkan memperkuat ketidakadilan sosial yang sudah ada.

14.7.1 Bagaimana Bias Terjadi?

Bias dapat masuk ke dalam sistem data science melalui berbagai jalur:

- **Bias dalam data pelatihan:** Jika data historis mencerminkan ketimpangan atau diskriminasi yang terjadi di masyarakat, model akan belajar dan meniru pola tersebut.
- **Bias dalam pengumpulan data:** Representasi yang tidak merata, misalnya lebih banyak data dari kelompok

mayoritas, akan mengurangi akurasi prediksi pada kelompok minoritas.

- **Bias dalam rancangan fitur dan algoritma:** Pemilihan variabel prediktor atau parameter algoritma yang tidak netral bisa memperkuat bias tersembunyi.

14.7.2 Contoh Kasus Nyata

- **Skor Kredit Otomatis:** Algoritma kredit di beberapa negara diketahui menolak lebih banyak aplikasi pinjaman dari etnis minoritas, karena dilatih pada data historis yang mencerminkan diskriminasi institusional.
- **Sistem Rekrutmen Otomatis:** Sebuah perusahaan teknologi besar mengembangkan sistem penyaringan CV otomatis, namun model cenderung memilih pelamar pria karena dilatih dari data historis yang menunjukkan dominasi pria di bidang teknologi.

14.7.3 Solusi untuk Mengurangi Bias

- **Audit Data dan Model Secara Berkala** Lakukan analisis bias dan ketidakadilan pada data dan hasil model. Gunakan metrik seperti disparate impact atau equality of opportunity.
- **Gunakan Teknik Fairness-aware Machine Learning** Gunakan algoritma yang dirancang untuk mempertimbangkan keadilan (fairness) dalam pelatihan, misalnya

reweighting, adversarial debiasing, atau post-processing untuk mengurangi diskriminasi.

- **Libatkan Tim Multidisiplin** Libatkan ahli sosial, hukum, dan etika dalam proses pengembangan model agar perspektif keadilan dan keberagaman turut dipertimbangkan.

14.8 Etika dalam Proyek Ilmu Data

Setiap proyek ilmu data harus didasarkan pada prinsip-prinsip etis yang tidak hanya menjaga kepatuhan terhadap hukum, tetapi juga menjamin keadilan, transparansi, dan perlindungan hak individu. Untuk itu, diperlukan **pemeriksaan etis (ethical checklist)** sebelum proyek dijalankan.

14.8.1 Checklist Etis Proyek Ilmu Data

Berikut adalah beberapa pertanyaan kunci yang harus dijawab oleh tim pengembang proyek data science sebelum memulai proses pengumpulan data, pelatihan model, dan implementasi sistem:

- ✓ Apakah data dikumpulkan dengan izin?
  Pastikan bahwa setiap data pribadi dikumpulkan melalui proses yang sah, dengan persetujuan eksplisit dari pemilik data (*informed consent*).

- ✓ Apakah pemilik data tahu bagaimana datanya digunakan?
  Transparansi sangat penting. Individu yang datanya digunakan harus diberi tahu tujuan, risiko, dan batas penggunaan data mereka.

- ✓ Apakah hasil model dapat dijelaskan ke publik awam?
  Model harus dapat dijelaskan (*interpretable*) agar hasil dan dampaknya bisa dimengerti oleh pengguna akhir dan pihak yang terkena dampaknya.

- ✓ Apakah keputusan model mempengaruhi kehidupan orang?
  Jika ya, maka model tersebut harus diaudit lebih ketat, terutama jika digunakan untuk keputusan penting seperti rekrutmen, kredit, atau diagnosis.

- ✓ Apakah ada sistem pelaporan untuk penyalahgunaan?
  Proyek harus menyediakan mekanisme yang memungkinkan publik atau pengguna untuk melaporkan kesalahan, bias, atau dampak negatif dari sistem.

14.8.2 Pentingnya Etika sebagai Bagian dari Proses

Etika bukanlah langkah tambahan yang dilakukan di akhir, melainkan komponen penting yang harus **terintegrasi** di setiap tahap:

- Saat merancang tujuan proyek

- Saat memilih data dan algoritma
- Saat mengevaluasi performa dan dampak
- Saat melakukan deployment dan pemantauan sistem

14.9 Peran Data Scientist sebagai Profesional

Profesi **data scientist** bukan hanya tentang keterampilan teknis seperti pemrograman, statistik, atau machine learning, tetapi juga mencakup tanggung jawab sosial dan etika. Dalam dunia yang semakin bergantung pada keputusan berbasis data, data scientist memegang peranan penting dalam menentukan bagaimana teknologi digunakan — dan untuk siapa.

14.9.1 Tanggung Jawab Profesional

Sebagai profesional, seorang data scientist harus:

- **Memahami Legal Landscape**
  Data scientist wajib memahami peraturan dan hukum yang berlaku di negara tempat mereka bekerja, termasuk regulasi perlindungan data pribadi seperti GDPR (Uni Eropa), UU ITE, atau RUU PDP (Indonesia). Ketidaktahuan hukum tidak membebaskan dari tanggung jawab hukum.
- **Berperan Aktif dalam Kebijakan Etis**
  Data scientist diharapkan menjadi agen perubahan di

dalam perusahaan. Mereka harus mendorong penggunaan data yang adil, transparan, dan akuntabel, serta memberi masukan terhadap kebijakan internal yang menyangkut privasi, keamanan, dan fairness.

- **Menolak Proyek yang Tidak Etis**
  Dalam situasi di mana proyek data science berpotensi melanggar hak asasi manusia, merugikan kelompok rentan, atau menyalahi hukum, profesional data harus memiliki keberanian untuk menolak berpartisipasi. Etika lebih tinggi dari target performa model.

- **Menjadi Gatekeeper Teknologi**
  Data scientist adalah penjaga (gatekeeper) yang memastikan bahwa teknologi yang dikembangkan — seperti sistem rekomendasi, prediksi risiko, atau algoritma penilaian — tidak membawa dampak negatif pada masyarakat luas.

14.9.2 Analogi Profesi

"Seperti dokter terikat oleh kode etik medis, praktisi data pun terikat oleh kode etik informasi."

Kode etik ini mencakup prinsip-prinsip seperti menjaga kerahasiaan, bertanggung jawab atas dampak sosial, serta menempatkan kesejahteraan manusia sebagai prioritas utama.

14.10 LATIHAN / TUGAS AKHIR BAB 14

1. **[Uraian]** Jelaskan prinsip-prinsip utama etika dalam Ilmu Data dan berikan contoh pelanggaran dari masing-masing prinsip.
2. **[Uraian Visual]** Buat diagram alir proses penanganan data pribadi dari tahap pengumpulan hingga penghapusan sesuai prinsip GDPR.
3. **[Coding]**
    - Simulasikan data pengguna (nama, email, lokasi)
    - Lakukan proses pseudonimisasi dan anonimisasi dengan Python
    - Simpan data dalam format terenkripsi sederhana
4. **[Studi Kasus]** Analisis dan tulis laporan kritis atas kasus Cambridge Analytica, fokus pada aspek teknis dan etis.
5. **[Studi Kasus]** Buat proposal proyek data science untuk instansi pemerintahan dan tambahkan bagian khusus tentang prinsip etika dan privasi.

Daftar Pustaka

[1] C. O'Neil, *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown Publishing Group, 2016.

[2] D. Boyd **and** K. Crawford, "Critical questions for big data: Provocations for a cultural, technological, and scholarly phenomenon," *Information, Communication & Society*, **jourvol** 15, **number** 5, **pages** 662–679, 2012.

[3] B. D. Mittelstadt, P. Allo, M. Taddeo, S. Wachter **and** L. Floridi, "The ethics of algorithms: Mapping the debate," *Big Data & Society*, **jourvol** 3, **number** 2, **page** 2 053 951 716 679 679, 2016.

[4] S. Barocas **and** A. D. Selbst, "Big data's disparate impact," *California Law Review*, **jourvol** 104, **number** 3, **pages** 671–732, 2016.

[5] L. Floridi, J. Cowls, M. Beltrametti **andothers**, "AI4People—An ethical framework for a good AI society: Opportunities, risks, principles, and recommendations," *Minds and Machines*, **jourvol** 28, **number** 4, **pages** 689–707, 2018.

[6] D. J. Solove, "A taxonomy of privacy," *University of Pennsylvania Law Review*, **jourvol** 154, **number** 3, **pages** 477–564, 2006.

[7] S. Wachter, B. Mittelstadt **and** L. Floridi, "Why a right to explanation of automated decision-making does not exist in the General Data Protection Regulation," *Inter-*

national Data Privacy Law, **jourvol** 7, **number** 2, **pages** 76–99, 2017. DOI: 10.1093/idpl/ipx005.

[8] P. M. Schwartz, "Information privacy in the cloud," *University of Pennsylvania Law Review*, **jourvol** 161, **number** 6, **pages** 1623–1662, 2011.

[9] P. Voigt **and** A. Von dem Bussche, *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer International Publishing, 2017. DOI: 10.1007/978-3-319-57959-7.

15 PROYEK AKHIR ILMU DATA

15.1 Tujuan Pembelajaran

Setelah menyelesaikan bab ini, mahasiswa diharapkan mampu:

1. Merancang proyek Ilmu Data lengkap dari perumusan masalah hingga presentasi hasil.
2. Mengintegrasikan seluruh tahapan analisis data: akuisisi, pembersihan, eksplorasi, pemodelan, dan visualisasi.
3. Membangun sistem prediktif sederhana menggunakan data nyata.
4. Menerapkan prinsip etika dan dokumentasi yang baik dalam proyek.
5. Menyusun laporan proyek dalam format profesional.

15.2 Tujuan dan Manfaat Proyek Akhir

Proyek akhir merupakan komponen penting dalam pembelajaran Ilmu Data yang memberikan kesempatan bagi mahasiswa untuk menerapkan secara langsung semua penge-

tahuan dan keterampilan yang telah diperoleh selama perkuliahan [1].

15.2.1 Tujuan Proyek Akhir

Tujuan utama dari proyek akhir ini adalah untuk:

- **Mengaplikasikan Konsep dan Keterampilan Ilmu Data**
  Mahasiswa diharapkan mampu menerapkan proses end-to-end dalam ilmu data: mulai dari eksplorasi data, pembersihan data, analisis statistik, pemodelan machine learning, hingga visualisasi dan interpretasi hasil.
- **Membangun Portofolio Profesional**
  Proyek akhir menjadi bagian penting dalam portofolio mahasiswa yang dapat ditunjukkan kepada calon pemberi kerja atau lembaga penelitian. Portofolio ini mencerminkan kemampuan teknis dan pemahaman konseptual mahasiswa.
- **Melatih Kemandirian dan Berpikir Sistematis**
  Melalui proyek ini, mahasiswa dilatih untuk merancang alur kerja data secara mandiri, menyusun dokumentasi, serta mengambil keputusan berbasis data secara logis dan terstruktur.

- **Mengembangkan Kemampuan Komunikasi Analitik**
  Mahasiswa akan belajar menyampaikan hasil analisis kepada publik, baik dalam bentuk laporan tertulis maupun presentasi visual yang jelas, ringkas, dan berdampak.

15.2.2 Catatan Penting

Proyek akhir ini dirancang agar *skalabel* dan dapat diselesaikan dalam waktu 2–4 minggu.

Dengan durasi tersebut, proyek difokuskan pada skenario nyata yang sederhana namun bermakna, seperti analisis data publik, klasifikasi teks, prediksi penjualan, atau eksplorasi sentimen media sosial. Mahasiswa tidak perlu membangun sistem produksi penuh, melainkan cukup membuktikan pemahaman konsep dan ketepatan teknis.

15.3 Struktur Proyek Ilmu Data

Setiap proyek ilmu data yang baik harus mengikuti alur kerja (*workflow*) yang terstruktur dan sistematis. Struktur ini membantu mahasiswa tidak hanya dalam mengorganisasi pekerjaan, tetapi juga dalam menyampaikan proses berpikir secara logis kepada dosen pembimbing atau audiens profesional.

15.3.1 Masalah atau Pertanyaan Bisnis

Langkah awal adalah merumuskan masalah atau pertanyaan yang ingin dijawab dengan data. Pertanyaan harus jelas, terukur, dan relevan.

- **Contoh:** Apakah kita bisa memprediksi mahasiswa yang berisiko drop out (DO) berdasarkan data akademik mereka?
- **Tujuan:** Menentukan output yang diharapkan: klasifikasi, regresi, rekomendasi, atau segmentasi.

15.3.2 Pengumpulan Data

Data dapat dikumpulkan dari berbagai sumber:

- Dataset open source (Kaggle, UCI, data pemerintah)
- API publik (Twitter, Spotify, Weather)
- Data dummy hasil simulasi atau data sintetis yang menyerupai skenario nyata

15.3.3 Pembersihan dan Preprocessing

Langkah penting untuk menyiapkan data sebelum dianalisis atau dimodelkan. [2] Termasuk:

- Menangani *missing value*
- Encoding variabel kategorikal

- Normalisasi atau standardisasi fitur numerik
- Deteksi dan penanganan outlier

15.3.4 Exploratory Data Analysis (EDA)

EDA bertujuan memahami struktur data, distribusi nilai, serta pola-pola penting sebelum membangun model. [3]

- Gunakan visualisasi: histogram, scatter plot, box plot
- Buat korelasi antar variabel
- Temukan insight awal dan hipotesis eksploratif

15.3.5 Pemodelan

Langkah inti dari proyek ilmu data. Mahasiswa dapat memilih pendekatan yang sesuai [4]:

- Klasifikasi (contoh: prediksi DO)
- Regresi (contoh: prediksi nilai IPK)
- Clustering (contoh: segmentasi tipe mahasiswa)
- Time Series (contoh: prediksi jumlah pendaftar tiap semester)

15.3.6 Evaluasi Model

Gunakan metrik evaluasi yang sesuai dengan jenis model [5]:

- **Klasifikasi:** akurasi, precision, recall, F1-score, AUC
- **Regresi:** MAE, MSE, RMSE, R^2
- **Clustering:** silhouette score, Davies–Bouldin index

15.3.7 Visualisasi dan Interpretasi

Presentasikan hasil model dan temuan penting secara visual [6]:

- Gunakan grafik: confusion matrix, feature importance, SHAP plot
- Sajikan insight dalam bahasa non-teknis agar mudah dipahami stakeholder

15.3.8 Laporan dan Presentasi

Dokumentasikan seluruh proses dan hasil dalam bentuk:

- Laporan proyek akhir seperti *mini research paper* (abstrak, metodologi, eksperimen, hasil)
- Presentasi final kepada dosen atau mitra industri

15.4 Panduan Topik Proyek

Pemilihan topik proyek merupakan langkah krusial dalam proses pengerjaan proyek akhir. Topik yang tepat akan membantu mahasiswa lebih fokus, termotivasi, dan menunjukkan kompetensi yang dimiliki. Dalam panduan ini, ditawarkan beberapa topik yang tidak hanya bervariasi secara doma-

in, tetapi juga mencerminkan keragaman jenis model dalam ilmu data.

15.4.1 Topik dan Jenis Model yang Direkomendasikan

No	Topik Proyek	Jenis Model
1	Prediksi drop-out mahasiswa berdasarkan data akademik	Klasifikasi
2	Segmentasi pelanggan e-commerce	Clustering
3	Analisis sentimen ulasan produk	Text Classification
4	Prediksi harga rumah berdasarkan fitur properti	Regresi
5	Dashboard penyebaran COVID-19 di Indonesia	Visualisasi Interaktif
6	Deteksi penipuan transaksi e-wallet	Anomali Detection

Tabel 10: Topik Proyek dan Jenis Model

15.4.2 Tips Memilih Topik

- Pilih topik yang sesuai dengan minat pribadi atau latar belakang keilmuan.
- Pastikan topik memiliki sumber data yang dapat diakses (open dataset atau dummy data).
- Mulailah dari ruang lingkup sederhana dan kembangkan jika waktu memungkinkan.

- Pertimbangkan topik yang dapat dikembangkan lebih lanjut sebagai skripsi atau portofolio karier.

15.5 Contoh Proyek: Prediksi Drop-Out Mahasiswa

Bagian ini menyajikan contoh proyek lengkap yang dapat dijadikan acuan oleh mahasiswa dalam menyusun dan mengerjakan proyek akhir. Studi kasus ini menggunakan pendekatan klasifikasi untuk mengidentifikasi mahasiswa yang berisiko tidak menyelesaikan studi tepat waktu (drop-out).

15.5.1 Rumusan Masalah

> **Bagaimana cara memprediksi mahasiswa yang berisiko tidak menyelesaikan studi tepat waktu berdasarkan data akademik dan aktivitas kampus?**

Pertanyaan ini ditujukan untuk mendeteksi risiko drop-out sedini mungkin agar pihak kampus dapat memberikan intervensi tepat waktu.

15.5.2 Dataset

Dataset bersumber dari data internal perguruan tinggi, dengan fitur-fitur utama sebagai berikut:

- **Nilai IPK semester 1–4** – menunjukkan performa akademik.

- **Kehadiran kuliah** – diukur dalam persentase atau jumlah absensi.
- **Jenis pembiayaan** – misalnya mandiri, beasiswa pemerintah, beasiswa swasta.
- **Aktivitas organisasi** – indikator partisipasi dalam kegiatan non-akademik.

15.5.3 Tahapan Proyek

1. **Akuisisi Data**
   Data dikumpulkan dari sistem informasi akademik (SIAKAD) kampus. Jika tidak tersedia, dapat digunakan data simulasi (dummy) yang meniru struktur data nyata.
2. **Data Cleaning**
   - Menangani *missing values* pada kolom kehadiran atau IPK.
   - Deteksi dan penanganan *outlier* pada nilai IPK yang ekstrem.
   - Encoding untuk fitur kategorikal seperti jenis pembiayaan.
3. **Eksplorasi Visual dan Statistik**
   - Scatter plot antara IPK dan kehadiran untuk melihat hubungan visual.
   - Korelasi antar fitur untuk mengetahui pengaruh relatif.

4. **Pemodelan Klasifikasi**
   Dua algoritma digunakan untuk membandingkan performa:

   - **Decision Tree**
   - **Logistic Regression**

   Target variabel: `status_studi` (lanjut / DO)

5. **Evaluasi Model**
   Gunakan metrik:

   - Akurasi: seberapa sering prediksi benar secara keseluruhan
   - Recall: seberapa sensitif model dalam mendeteksi kasus DO

6. **Interpretasi Fitur Penting**
   Dengan Decision Tree, fitur penting dapat ditelusuri melalui struktur pohon. Logistic Regression juga menunjukkan pengaruh setiap variabel.

7. **Simulasi Intervensi**
   Berdasarkan prediksi model, dilakukan simulasi kebijakan seperti:

   - Mengirim notifikasi akademik untuk mahasiswa berisiko
   - Menawarkan program mentoring bagi kelompok tertentu

15.6 Rubrik Penilaian Proyek

Untuk memastikan bahwa proyek akhir dievaluasi secara adil, menyeluruh, dan objektif, digunakan rubrik penilaian yang mencakup berbagai aspek penting dalam proses ilmu data [7]. Setiap aspek memiliki bobot tertentu yang mencerminkan kontribusinya terhadap kualitas keseluruhan proyek.

15.6.1 Tabel Rubrik Penilaian Proyek Ilmu Data

Aspek	Bobot (%)
Perumusan Masalah	10%
Kualitas Data dan Preprocessing	15%
Analisis dan EDA	15%
Pemilihan dan Evaluasi Model	25%
Visualisasi dan Interpretasi	15%
Dokumentasi dan Laporan	10%
Etika dan Privasi	10%

Tabel 11: Pembobotan Aspek Evaluasi Proyek

15.6.2 Catatan untuk Dosen Penguji

- Penilaian dapat dilakukan secara individu atau tim (jika proyek kelompok).
- Disarankan memberikan catatan kualitatif di samping nilai numerik.

- Penekanan pada etika dan privasi penting sebagai pembeda proyek teknis biasa dan proyek profesional.

15.7 Format Penulisan Laporan

Laporan proyek akhir adalah dokumen utama yang merekam seluruh proses, analisis, dan hasil dari proyek ilmu data yang telah dikerjakan [8]. Laporan ini berfungsi sebagai bentuk pertanggungjawaban akademik sekaligus portofolio profesional mahasiswa.

15.7.1 Struktur Laporan yang Direkomendasikan

Laporan disarankan disusun secara sistematis sebagai berikut [9]:

1. **Judul Proyek**
   Judul yang jelas, ringkas, dan mencerminkan fokus utama proyek.
2. **Penulis dan Identitas**
   Nama lengkap, NIM, program studi, dan institusi.
3. **Abstrak**
   Ringkasan proyek dalam satu paragraf (100–150 kata) yang mencakup latar belakang, tujuan, metode, dan hasil utama.
4. **Pendahuluan**
   Menyampaikan latar belakang masalah, urgensi, dan

relevansi topik yang diangkat.

5. **Tujuan dan Ruang Lingkup**
   Merinci apa yang ingin dicapai dan batasan-batasan proyek (misalnya jenis data, waktu pengerjaan, dll).

6. **Metodologi**
   Menjelaskan alur kerja proyek mulai dari pengumpulan data, preprocessing, analisis eksploratif, pemodelan, evaluasi, hingga interpretasi.

7. **Eksperimen dan Hasil**
   Menyajikan hasil pemodelan, tabel evaluasi, dan grafik performa. Dapat dibandingkan lebih dari satu model.

8. **Pembahasan**
   Interpretasi hasil, diskusi terhadap hipotesis awal, serta implikasi atau keterbatasan proyek.

9. **Kesimpulan dan Rekomendasi**
   Merangkum temuan utama dan menyarankan langkah lanjutan, baik untuk penelitian maupun implementasi riil.

10. **Referensi**
    Daftar pustaka dalam format standar (IEEE, APA, atau lainnya), mencakup sumber dataset, pustaka ilmiah, dan alat yang digunakan.

11. **Lampiran (Opsional)**
    Kode penting, grafik tambahan, tangkapan layar visualisasi interaktif, atau struktur folder proyek.

15.7.2 Format Penyerahan Laporan

- Format dokumen: **PDF** [10]
- Panjang maksimal: **15 halaman** (tidak termasuk lampiran)
- Ukuran font dan margin mengikuti panduan institusi, atau default akademik (misal: Times New Roman 12 pt, spasi 1.5, margin 3-3-2.5-2.5)

15.7.3 Catatan untuk Mahasiswa

Laporan ini akan menjadi salah satu portofolio yang bernilai tinggi jika disusun dengan baik [11]. Oleh karena itu, penting untuk:

- Menulis secara jelas dan lugas, hindari bahasa terlalu teknis tanpa penjelasan.
- Menyusun struktur dokumen dengan heading yang konsisten.
- Menyisipkan grafik atau tabel pendukung untuk memperjelas narasi.

15.8 Tools dan Template Pendukung

Untuk mendukung pengerjaan proyek akhir secara efisien, profesional, dan terdokumentasi dengan baik, mahasiswa di-

anjurkan menggunakan sejumlah perangkat lunak dan pustaka berikut:

15.8.1 Jupyter Notebook / Google Colab

- **Jupyter Notebook** adalah lingkungan pemrograman interaktif berbasis web yang sangat cocok untuk eksperimen dan dokumentasi proyek data science.
- **Google Colab** menyediakan fitur serupa namun berbasis cloud, mendukung eksekusi kode Python tanpa perlu instalasi lokal, dan dapat disimpan langsung ke Google Drive.
- Keduanya mendukung integrasi Markdown, visualisasi, dan eksekusi kode bertahap, ideal untuk dokumentasi proyek akhir.

15.8.2 Pustaka Python: Pandas, Matplotlib, Seaborn, Scikit-learn

- **Pandas** digunakan untuk manipulasi data tabular, pembersihan data, dan analisis statistik dasar.
- **Matplotlib** dan **Seaborn** digunakan untuk membuat berbagai jenis visualisasi seperti histogram, box plot, heatmap, dan scatter plot.

- **Scikit-learn** menyediakan pustaka lengkap untuk pemodelan machine learning, termasuk klasifikasi, regresi, clustering, evaluasi model, dan preprocessing.

15.8.3 Streamlit / Dash (Opsional untuk Visualisasi Interaktif)

- **Streamlit** dan **Dash** adalah framework Python untuk membangun dashboard interaktif berbasis web dengan cepat.
- Cocok digunakan untuk proyek yang ingin menyajikan hasil analisis secara interaktif kepada audiens non-teknis.
- Mahasiswa yang ingin menampilkan prediksi model, filter interaktif, atau peta visualisasi dapat memanfaatkan alat ini.

15.8.4 GitHub

- **GitHub** adalah platform untuk menyimpan dan berbagi kode sumber, serta mendokumentasikan proyek menggunakan `README.md`.
- Mahasiswa disarankan menggunakan GitHub untuk:
  - Menerapkan kontrol versi (versioning)
  - Menyimpan notebook, dataset, dan visualisasi
  - Membangun portofolio proyek yang dapat dibagikan ke calon pemberi kerja

- Repositori GitHub yang rapi dengan dokumentasi yang baik meningkatkan profesionalisme dan visibilitas online.

15.9 LATIHAN / TUGAS AKHIR BAB 15

1. **[Uraian]** Buat kerangka proyek Anda sendiri menggunakan 8 tahap standar proyek data science.
2. **[Uraian Visual]** Rancang alur kerja proyek Anda dalam bentuk diagram tahapan (flowchart).
3. **[Coding]**
    - Kumpulkan data dari open source (Kaggle, Data.go.id)
    - Lakukan EDA dan model sederhana (regresi atau klasifikasi)
    - Tampilkan hasil dan metrik performa
4. **[Studi Kasus]** Bangun proyek mini prediksi harga laptop berdasarkan spesifikasi teknis. Evaluasi model dan presentasikan dalam dashboard Streamlit.
5. **[Studi Kasus]** Buat sistem prediksi beasiswa mahasiswa berbasis IPK dan faktor ekonomi. Sertakan pertimbangan etis dalam seleksi otomatis ini.

15.10 Penutup

Proyek akhir bukan sekadar tugas akademik, melainkan latihan menyeluruh yang merefleksikan seluruh kompetensi inti seorang ***data scientist***. Dengan menyelesaikan proyek ini, mahasiswa telah melalui proses yang menyerupai siklus kerja profesional di dunia nyata.

15.10.1 Capaian Utama Mahasiswa

Melalui proyek ini, mahasiswa telah:

- **Berpikir kritis** dalam merumuskan masalah yang relevan dan mengubahnya menjadi pertanyaan berbasis data.
- **Bekerja dengan data nyata**, mulai dari akuisisi, pembersihan, hingga analisis.
- **Mengintegrasikan metode** statistik, machine learning, dan visualisasi dalam satu kerangka kerja terpadu.
- **Menerapkan etika**, termasuk privasi data, fairness, dan transparansi dalam proses pengambilan keputusan berbasis algoritma.
- **Mempresentasikan hasil** dengan percaya diri dalam format profesional, baik dalam laporan tertulis maupun secara visual dan lisan.

15.10.2 Refleksi Akhir

Proyek akhir ini adalah fondasi untuk memasuki dunia data yang lebih luas, baik dalam bentuk riset lanjutan, kontribusi pada masyarakat, maupun karier profesional.

“Kuasai data. Pahami realita. Ciptakan solusi.”

Dengan semangat ini, mahasiswa tidak hanya menjadi pengguna teknologi, tetapi juga pencipta solusi nyata yang berbasis data dan berdampak bagi dunia.

15.10.3 Selamat Berkarya!

Proyek ini adalah awal dari perjalanan profesionalmu di bidang ilmu data. Jadikan pengalaman ini sebagai dasar untuk terus belajar, bereksperimen, dan berkontribusi secara etis dalam era digital.

Daftar Pustaka

[1] A. Mourad, *Data Science Projects with Python*. Packt Publishing, 2019. DOI: 10.1007/978-1-4842-3966-1.

[2] G. James, D. Witten, T. Hastie **and** R. Tibshirani, *An introduction to statistical learning*. Springer, 2013.

[3] J. W. Tukey, *Exploratory Data Analysis*. Addison-Wesley, 1977, ISBN: 978-0201076167.

[4] T. Hastie, R. Tibshirani **and** J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2 **edition**. Springer, 2009.

[5] S. Mehtab **and** S. Sharma, "Model Evaluation Metrics for Machine Learning Algorithms: A Survey," *International Journal of Computer Science and Information Security*, **jourvol** 16, **number** 6, **pages** 1–8, 2018.

[6] W. McKinney, *Data Analysis with Python: A Guide to Using Pandas, NumPy, and Matplotlib*. O'Reilly Media, 2010, ISBN: 978-1449319793.

[7] S. Bansal, *Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking*. O'Reilly Media, 2016, ISBN: 978-1491903635.

[8] A. Morales, *Guidelines for Writing Data Science Reports*. Springer, 2021. DOI: 10.1007/978-3-030-50123-2.

[9] K. Healy, *Data Visualization: A Practical Introduction*. Princeton University Press, 2018, ISBN: 978-0691181617.

[10] D. S. Moore, *Research Methods in Data Science*. Wiley, 2020, ISBN: 978-1119624131.

[11] J. Grus, *Data Science from Scratch: First Principles with Python*. O'Reilly Media, 2019, ISBN: 978-1492041139.

Glosarium

Istilah	Definisi
Data Science	Ilmu yang mempelajari bagaimana mengekstraksi pengetahuan dari data dengan metode ilmiah.
Big Data	Kumpulan data dalam volume besar, bervariasi, dan bergerak cepat, sulit ditangani sistem tradisional.
Machine Learning	Cabang AI yang membuat sistem belajar dari data tanpa diprogram secara eksplisit.
Supervised Learning	Teknik pembelajaran mesin dengan data berlabel sebagai acuan pelatihan model.
Unsupervised Learning	Teknik pembelajaran mesin tanpa label, digunakan untuk menemukan pola atau kelompok.
Regression	Teknik untuk memprediksi nilai numerik kontinu berdasarkan variabel input.
Classification	Proses memetakan data ke dalam kategori/kelas tertentu.
Clustering	Pengelompokan data berdasarkan kemiripan tanpa label.
Feature Engineering	Proses membuat fitur baru dari data mentah untuk meningkatkan performa model.

Data Cleaning	Proses membersihkan data dari kesalahan, duplikasi, nilai kosong, dan inkonsistensi.
EDA (Exploratory Data Analysis)	Tahapan eksplorasi data awal untuk memahami struktur dan pola dalam data.
Overfitting	Kondisi saat model terlalu menyesuaikan diri dengan data pelatihan dan gagal pada data baru.
Underfitting	Kondisi saat model terlalu sederhana dan gagal menangkap pola dalam data.
Confusion Matrix	Tabel untuk menilai performa klasifikasi, berisi TP, TN, FP, dan FN.
TF-IDF	Skema pembobotan teks untuk menghitung pentingnya kata dalam dokumen dan seluruh korpus.
PCA (Principal Component Analysis)	Teknik reduksi dimensi dengan mengubah data ke dalam komponen utama.
K-Means	Algoritma clustering yang mengelompokkan data ke dalam k kelompok berdasarkan jarak centroid.
MapReduce	Model pemrosesan data besar dalam dua tahap: pemetaan (map) dan peringkasan (reduce).
HDFS	Hadoop Distributed File System, sistem penyimpanan data besar secara terdistribusi.

YARN	Yet Another Resource Negotiator, pengatur sumber daya dan eksekusi pada Hadoop.
Streamlit	Library Python untuk membuat web dashboard data interaktif dengan cepat.
API (Application Programming Interface)	Antarmuka yang memungkinkan aplikasi mengakses data dari sistem lain.
Anonymization	Proses menyembunyikan identitas individu dalam data untuk melindungi privasi.
Bias	Kecenderungan sistem yang menghasilkan prediksi tidak adil akibat data atau desain model.

Indeks

Ilmu Data

Di era digital yang penuh dengan data, kemampuan untuk mengolah, menganalisis, dan memahami data menjadi keterampilan esensial. Buku Ilmu Data ini hadir sebagai panduan komprehensif dan praktis untuk mahasiswa, dosen, maupun praktisi yang ingin memahami konsep dasar hingga lanjutan dalam ilmu data.

Dengan bahasa yang lugas dan sistematis, buku ini membahas berbagai aspek penting seperti jenis dan representasi data, proses pengumpulan dan pembersihan data, eksplorasi dan transformasi data, hingga pengenalan pada pembelajaran mesin (machine learning), text mining, dan big data. Selain itu, buku ini juga menyoroti aspek kritis seperti visualisasi data serta isu etika dan privasi.

Setiap bab disusun secara bertahap dan dilengkapi dengan studi kasus dan proyek akhir, menjadikan buku ini bukan hanya sumber belajar, tetapi juga alat praktis untuk memecahkan masalah nyata berbasis data.

Temukan bagaimana data dapat bercerita, dan jadikan ilmu data sebagai alat strategis dalam pengambilan keputusan di berbagai bidang.

Arbi Haza Nasution adalah *Associate Professor* di Prodi Teknik Informatika, Universitas Islam Riau (UIR), Indonesia. Ia meraih gelar Ph.D. di bidang Informatika dari Kyoto University pada tahun 2018, setelah sebelumnya menyelesaikan studi Sarjana di bidang Ilmu Komputer (2010) dan Magister di bidang Sistem Informasi Manajemen (2012) dari Universiti Kebangsaan Malaysia. Bidang penelitian yang diminatinya meliputi linguistik komputasional, pemrosesan bahasa alami, model bahasa besar, dan pembelajaran mesin. Di UIR, beliau mengampu beberapa mata kuliah utama, yaitu Ilmu Data, Pemrosesan Bahasa Alami, dan Basisdata Grafik, membekali mahasiswa dengan fondasi kuat untuk menghadapi tantangan dunia data dan teknologi modern.

Winda Monika adalah dosen di Program Studi Ilmu Perpustakaan, Fakultas Humaniora, Universitas Lancang Kuning (Unilak), Indonesia. Ia meraih gelar Sarjana dari Universitas Pendidikan Indonesia pada tahun 2013 dan gelar Magister dari University of Tsukuba, Jepang, pada tahun 2018. Bidang penelitian yang diminatinya mencakup metadata, pemrosesan bahasa alami, model bahasa besar, dan digital humaniora. Di Unilak, ia mengampu mata kuliah Manajemen Pengetahuan, di mana ia membekali mahasiswa dengan pengetahuan dan keterampilan untuk mengelola dan memanfaatkan pengetahuan dalam organisasi, serta memperkenalkan mereka pada konsep-konsep dasar dalam manajemen teknologi informasi dan data mining.

ISBN 978-623-8687-36-7

9 786238 687367

UIR PRESS

UIR Press merupakan penerbitan buku teks/ajar dan buku umum yang telah berkiprah dalam menerbitkan berbagai buku yang ditulis oleh para dosen di lingkungan internal UIR sendiri maupun masyarakat luas dari berbagai kalangan profesi. UIR Press melayani penerbitan buku-buku teks ilmiah dan buku umum karya para dosen dan cendekiawan berbagai bidang ilmu pengetahuan.

uirpress.ui.ac.id uirpress@uir.ac.id @poncorbituirpress 085374018353

Kantor: Gedung Serbaguna Universitas Islam Riau (UIR)
Jalan Kaharuddin Nasution No. 113, Pekanbaru 28285