Modul Pelatnas IOAI Indonesia

Modul 1: Materi Dasar Artificial Intelligence (AI)

Tim Pembina IOAI Indonesia
sc.ioai.id@gmail.com

Juli 2025

Modul Pelatnas IOAI Indonesia

Modul 1: Materi Dasar Artificial Intelligence (AI)

Penyusun

Tim Pembina IOAI Indonesia

sc.ioai.id@gmail.com

Daftar Isi

Pengantar

Puji syukur kehadirat Tuhan YME atas berkat limpahan rahmat dan karnuia-Nya, Buku Modul Pelatnas IOAI ini telah berhasil kami selesaikan. Buku Modul ini kami susun sebagai salah satu referensi rangkaian pembinaan/pelatihan nasional bagi siswa peserta didik yang mengikuti Pelatnas dalam rangka membentuk tim yang akan mewakili Indonesia pada ajang International Olympiad in Artificial Intelligence (IOAI).

Terima kasih yang sebesar-besarnya kami sampaikan kepada para Pembina, Asisten Pembina, dan para Alumni ajang OSN bidang Informatika dan IOI (TOKI), serta semua pihak yang telah berkontribusi sehingga Buku Modul ini dapat terwujud. Kami menyadari masih banyak kekurangan dalam penulisan Buku Modul ini. Untuk itu kami mohon maaf dan kami sangat mengharapkan masukan untuk perbaikan dan penyempurnaan selanjutnya, sehingga keberadaan Buku Modul ini dapat memberikan manfaat yang sebesar-besarnya bagi semua pihak.

Semoga Buku Modul Pelatnas IOAI ini dapat digunakan sebaik-baiknya untuk menunjang kegiatan Pelatnas IOAI dan mampu membantu menghasilkan calon-calon talenta Indonesia di bidang AI yang mampu memberikan prestasi yang membanggakan di tingkat internasional.

BAB 1

Pengenalan Machine Learning

1.1 Apa Itu Machine Learning?

Machine Learning (ML), atau pembelajaran mesin, adalah cabang dari kecerdasan buatan (AI) yang memungkinkan komputer untuk "belajar" dari data. Alih-alih diberi instruksi eksplisit untuk setiap tugas, mesin belajar dari contoh dan pengalaman, kemudian dapat membuat keputusan atau prediksi secara mandiri.

Contoh: Jika kita ingin komputer mengenali apakah sebuah foto berisi kucing atau anjing, kita tidak perlu menulis aturan yang menjelaskan semua ciri-ciri hewan. Kita hanya perlu memberikan banyak contoh gambar kucing dan anjing beserta labelnya. Komputer akan mempelajari pola dari data tersebut dan belajar mengenali hewan-hewan tersebut dalam gambar baru.

Analogi: Seperti halnya manusia belajar mengenali benda dari pengalaman, mesin pun belajar dari data. Semakin banyak dan berkualitas datanya, semakin baik kemampuannya dalam belajar dan mengambil keputusan.

1.2 Jenis-jenis Machine Learning

Machine Learning dibagi menjadi beberapa jenis utama berdasarkan cara mesin belajar dari data:

1.2.1 Supervised Learning

Pada supervised learning, data pelatihan memiliki pasangan input dan output yang diketahui. Model dilatih untuk memetakan input ke output dengan benar.

Contoh:

- Memprediksi apakah email adalah spam atau bukan.
- Memprediksi harga rumah berdasarkan ukuran dan lokasi.

1.2.2 Unsupervised Learning

Dalam unsupervised learning, data tidak memiliki label. Tujuan algoritma adalah menemukan pola tersembunyi atau struktur dalam data.

Contoh:

- Mengelompokkan pelanggan berdasarkan kebiasaan belanja.

- Menemukan pola dalam data genetik.

1.2.3 Reinforcement Learning (sekilas)

Dalam reinforcement learning, agen belajar dengan mencoba-coba di suatu lingkungan dan menerima umpan balik berupa "reward" atau "hukuman". Tujuannya adalah memaksimalkan total reward jangka panjang.

Contoh:

- Robot belajar berjalan di medan tertentu.
- Program bermain catur yang belajar dari setiap langkahnya.

1.3 Tugas-Tugas dalam Machine Learning

1.3.1 Klasifikasi

Menentukan kategori dari suatu input.

Contoh:

- Mendeteksi jenis bunga berdasarkan panjang kelopak dan mahkota.
- Memprediksi apakah pasien memiliki penyakit tertentu.

1.3.2 Regresi

Memprediksi nilai numerik kontinu.

Contoh:

- Memprediksi harga rumah.
- Memprediksi suhu udara.

1.3.3 Clustering

Mengelompokkan data ke dalam grup berdasarkan kemiripan.

Contoh:

- Segmentasi pelanggan dalam bisnis.
- Mengelompokkan dokumen berita.

1.3.4 Tugas Lainnya (sekilas)

Selain tugas utama di atas, ML juga digunakan untuk:

- Deteksi anomali (misalnya mendeteksi transaksi mencurigakan).
- Sistem rekomendasi (misalnya produk di e-commerce).
- Reduksi dimensi (memadatkan informasi).

1.4 Contoh Penerapan Machine Learning dalam Kehidupan

Machine learning telah digunakan dalam berbagai bidang untuk membantu pekerjaan manusia, meningkatkan efisiensi, dan membuat sistem yang cerdas. Berikut beberapa contoh penerapannya:

- **Kesehatan:** Diagnosis penyakit dari citra medis (misalnya X-ray).
- **Keuangan:** Deteksi penipuan kartu kredit.
- **Transportasi:** Sistem navigasi dan kendaraan otonom.
- **Edukasi:** Sistem rekomendasi materi belajar.
- **Media sosial:** Penyaringan konten dan personalisasi feed.

Machine learning menjadi fondasi banyak teknologi modern dan terus berkembang pesat, membuka berbagai kemungkinan baru dalam sains dan teknologi.

Apa selanjutnya?

Memahami jenis pembelajaran dan tugas-tugas dalam machine learning hanyalah langkah awal. Untuk dapat membangun sistem AI yang efektif, kita juga harus mempersiapkan data dengan baik, memahami representasi numerik, serta memahami bagaimana model belajar dari data. Bab-bab selanjutnya akan membimbing Anda secara bertahap melalui aspek-aspek teknis tersebut, dimulai dari dasar/fondasi matematika, statistika dan optimasi, dan juga dasar pemrograman yang diperlukan di *machine learning*.

Referensi dan Bahan Bacaan Lanjutan

- **Google: Machine Learning Crash Course** https://developers.google.com/machine-learning/crash-course
- **Teachable Machine by Google** (untuk eksperimen klasifikasi interaktif) https://teachablemachine.withgoogle.com/
- **Fast.ai: Introduction to Machine Learning for Coders** https://course.fast.ai/ml

BAB 2

Dasar-Dasar Matematika

2.1 Pendahuluan

Machine learning dibangun di atas fondasi matematis yang kuat, terutama dari dua cabang utama yaitu aljabar linear dan kalkulus (turunan). Untuk memahami cara kerja model-model pembelajaran mesin—baik dalam merancang algoritma, menghitung gradien, atau merepresentasikan data—pemahaman yang baik terhadap ketiga bidang ini sangat penting.

Bab ini akan memberikan landasan konseptual yang dibutuhkan untuk memahami dan menerapkan machine learning secara mendalam. Kita akan mulai dengan aljabar linear, dilanjutkan dengan konsep turunan dalam kalkulus dan dasar teori optimasi dengan kalkulus.

2.2 Aljabar Linear

Aljabar linear adalah bahasa universal untuk merepresentasikan dan memanipulasi data dalam machine learning. Data dalam dunia nyata—baik gambar, teks, maupun sinyal—dapat direpresentasikan sebagai vektor atau matriks. Operasi-operasi seperti rotasi, proyeksi, dan transformasi ruang data semuanya menggunakan prinsip-prinsip dari aljabar linear.

2.2.1 Skalar, Vektor, dan Matriks

Skalar adalah bilangan tunggal yang merepresentasikan satu nilai. Misalnya, suhu = 37°C adalah skalar.

Vektor adalah susunan dari beberapa bilangan (skalar) yang disusun dalam satu baris atau kolom. Vektor merepresentasikan data berdimensi satu, seperti kecepatan dalam ruang 3D: $\vec{v} = [3, -2, 5]$.

Matriks adalah kumpulan bilangan yang disusun dalam baris dan kolom. Matriks digunakan untuk menyimpan data berdimensi dua atau sebagai transformasi linear. Contoh matriks 2×3:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

Matriks juga dapat digunakan untuk mewakili kumpulan vektor atau sebagai representasi dari dataset (misalnya, setiap baris adalah satu sampel, dan setiap kolom adalah fitur).

2.2.2 Operasi Matriks

Beberapa operasi dasar yang sering digunakan dalam machine learning melibatkan manipulasi matriks. Operasi-operasi ini merupakan bagian penting dalam pemrosesan data dan perhitungan model.

1. Penjumlahan Matriks

Dua matriks dapat dijumlahkan jika memiliki dimensi yang sama, yaitu jumlah baris dan kolom yang sama. Operasi dilakukan dengan menjumlahkan elemen yang bersesuaian.

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 0 & 1 & 3 \\ 4 & 1 & 5 & 2 \\ 0 & 3 & 1 & 1 \end{bmatrix}$$

$$A + B = \begin{bmatrix} 3 & 2 & 4 & 7 \\ 9 & 7 & 12 & 10 \\ 9 & 13 & 12 & 13 \end{bmatrix}$$

2. Perkalian Matriks dengan Skalar

Dalam perkalian skalar, setiap elemen dari matriks dikalikan dengan satu bilangan tetap (skalar).

$$C = \begin{bmatrix} 1 & -2 \\ 0 & 5 \end{bmatrix}, \quad k = 3$$

$$k \cdot C = 3 \cdot \begin{bmatrix} 1 & -2 \\ 0 & 5 \end{bmatrix} = \begin{bmatrix} 3 & -6 \\ 0 & 15 \end{bmatrix}$$

3. Perkalian Matriks

Perkalian dua matriks $A \cdot B$ hanya valid jika jumlah kolom di A sama dengan jumlah baris di B. Jika A berukuran $m \times n$ dan B berukuran $n \times p$, maka hasilnya adalah matriks $m \times p$.

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$A \cdot B = \begin{bmatrix} 1 \cdot 5 + 2 \cdot 7 & 1 \cdot 6 + 2 \cdot 8 \\ 3 \cdot 5 + 4 \cdot 7 & 3 \cdot 6 + 4 \cdot 8 \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

4. Transpose Matriks

Transpose dari suatu matriks diperoleh dengan menukar baris menjadi kolom, dan sebaliknya.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \quad \Rightarrow \quad A^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

Transpose sering digunakan dalam pembentukan sistem persamaan dan perhitungan turunan dalam model machine learning.

2.2.3 Invers Matriks

Invers matriks A^{-1} adalah matriks yang ketika dikalikan dengan A menghasilkan matriks identitas I, yaitu:

$$A \cdot A^{-1} = A^{-1} \cdot A = I$$

Tidak semua matriks memiliki invers; hanya matriks persegi (jumlah baris = jumlah kolom) yang *non-singular* atau memiliki determinan tidak nol yang memiliki invers.

Invers berguna dalam menyelesaikan sistem persamaan linear dan juga dalam beberapa model statistik (misalnya regresi linear dalam bentuk formula tertutup).

2.2.4 Ketergantungan dan Kebebasan Linear

Definisi Formal: Sekumpulan vektor $v_1, v_2, \ldots, v_n$ dikatakan **linear dependent** (tergantung secara linear) jika terdapat skalar-skalar $a_1, a_2, \ldots, a_n$ yang tidak semuanya nol, sehingga:

$$a_1 v_1 + a_2 v_2 + \cdots + a_n v_n = 0$$

Jika satu-satunya solusi dari persamaan tersebut adalah $a_1 = a_2 = \cdots = a_n = 0$, maka vektor-vektor tersebut dikatakan **linear independent** (bebas secara linear).

Penjelasan intuitif: - Sekumpulan vektor adalah bebas linear jika tidak ada vektor yang bisa ditulis sebagai kombinasi linear dari vektor lainnya. - Jika salah satu vektor bisa dibentuk dari vektor-vektor lainnya, maka seluruh himpunan menjadi tergantung secara linear.

Contoh 1 (Ketergantungan Linear):

Misalkan tiga vektor di ruang $\mathbb{R}^3$:

$$v_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \quad v_2 = \begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix}, \quad v_3 = \begin{bmatrix} 3 \\ 5 \\ 7 \end{bmatrix}$$

Kita dapat melihat bahwa:

$$v_2 = 2 \cdot v_1$$

Artinya, v_2 bergantung secara linear pada v_1. Maka, himpunan $\{v_1, v_2, v_3\}$ adalah tergantung secara linear, karena setidaknya ada satu vektor (yakni v_2) yang dapat dinyatakan sebagai kombinasi dari vektor lainnya.

Contoh 2 (Kebebasan Linear):

Misalkan:

$$v_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad v_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad v_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Ketiga vektor tersebut merupakan vektor basis kanonik di $\mathbb{R}^3$, dan jelas tidak ada vektor yang merupakan kombinasi linear dari dua vektor lainnya. Maka, $\{v_1, v_2, v_3\}$ adalah **bebas secara linear**.

Catatan: Dalam konteks machine learning, kebebasan linear sering digunakan untuk menentukan apakah fitur-fitur dalam dataset membawa informasi yang unik, atau apakah beberapa fitur saling redundant. Redundansi dalam fitur dapat menyebabkan overfitting atau perhitungan yang tidak stabil, sehingga penting untuk mendeteksinya sejak awal.

2.2.5 Rank Matriks

Rank dari sebuah matriks menyatakan jumlah maksimum baris (atau kolom) yang bebas linear. Rank berguna untuk mengetahui:

- ◇ Dimensi ruang vektor yang dibentangkan oleh baris-baris atau kolom-kolom matriks.
- ◇ Apakah sistem persamaan linear memiliki solusi unik, tak hingga solusi, atau tidak ada solusi.

Dalam machine learning, rank penting dalam memahami struktur data, seperti pada reduksi dimensi dan dekomposisi matriks.

2.2.6 Eigenvalue dan Eigenvektor

Dalam aljabar linear, **eigenvalue** dan **eigenvektor** merupakan konsep penting yang berkaitan dengan transformasi linear pada ruang vektor.

Misalkan $\mathbf{A}$ adalah sebuah matriks persegi berukuran $n \times n$, dan $\mathbf{v}$ adalah vektor kolom berdimensi n. Jika hasil perkalian antara matriks $\mathbf{A}$ dan vektor $\mathbf{v}$ menghasilkan sebuah vektor baru yang searah (hanya berubah skala) dengan $\mathbf{v}$, maka:

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$$

dengan: - λ adalah **eigenvalue** (nilai eigen) - $\mathbf{v}$ adalah **eigenvektor** yang tidak nol.

Makna Geometris

Secara geometris, jika sebuah transformasi linear (diwakili oleh matriks $\mathbf{A}$) diterapkan pada vektor $\mathbf{v}$, maka hasilnya umumnya akan mengubah arah dan panjang vektor tersebut. Namun, untuk vektor-vektor tertentu (eigenvektor), transformasi hanya mengubah panjang (dengan skala λ), namun tidak mengubah arah.

Persamaan Karakteristik

Untuk mencari eigenvalue, kita gunakan:

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0$$

Persamaan ini disebut **persamaan karakteristik**, dan menghasilkan polinomial dengan akar-akar berupa eigenvalue λ. Untuk setiap λ, kita substitusi ke $(\mathbf{A} - \lambda\mathbf{I})\,\mathbf{v} = 0$ untuk menemukan vektor eigen $\mathbf{v}$.

Contoh Sederhana

Misalkan:

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

Kita cari nilai λ yang memenuhi:

$$\det\left(\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) = 0 \Rightarrow \det\left(\begin{bmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix}\right) = 0$$

$$(2-\lambda)^2 - 1 = \lambda^2 - 4\lambda + 3 = 0 \Rightarrow \lambda = 1 \text{ atau } 3$$

Setelah mendapat λ, kita bisa mencari eigenvektor dengan menyelesaikan $(\mathbf{A} - \lambda\mathbf{I})\,\mathbf{v} = 0$.

Aplikasi Eigenvalue dan Eigenvektor

- Digunakan dalam **Principal Component Analysis (PCA)** untuk mereduksi dimensi data.
- Dalam dinamika sistem (misalnya fisika dan ekonomi), eigenvalue menunjukkan **stabilitas** sistem.
- Dalam **graf** dan teori jaringan, eigenvalue digunakan untuk menganalisis konektivitas dan pentingnya node (misalnya PageRank).

2.2.7 Faktorisasi Matriks: LU dan SVD

Faktorisasi matriks adalah proses memecah sebuah matriks menjadi hasil perkalian beberapa matriks lain yang memiliki struktur tertentu. Proses ini sangat berguna untuk menyederhanakan berbagai operasi seperti penyelesaian sistem persamaan linear, invers matriks, dan kompresi data.

LU Decomposition (Faktorisasi LU)

Faktorisasi LU memecah matriks persegi $\mathbf{A}$ menjadi dua matriks:

$$\mathbf{A} = \mathbf{LU}$$

dengan:

- $\mathbf{L}$ adalah matriks segitiga bawah (Lower triangular)
- $\mathbf{U}$ adalah matriks segitiga atas (Upper triangular)

Kegunaan:

- Mempercepat penyelesaian sistem persamaan linear $\mathbf{Ax} = \mathbf{b}$ dengan menyelesaikan dua sistem:
$$\mathbf{Ly} = \mathbf{b}, \quad \mathbf{Ux} = \mathbf{y}$$
- Digunakan dalam metode numerik seperti metode Gauss.

SVD (Singular Value Decomposition)

Singular Value Decomposition (SVD) adalah generalisasi dari faktorisasi eigen untuk matriks bukan persegi. SVD menyatakan matriks $\mathbf{A}$ sebagai:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

dengan:

- $\mathbf{U}$ adalah matriks ortogonal berisi **left singular vectors**
- $\mathbf{V}$ adalah matriks ortogonal berisi **right singular vectors**
- $\mathbf{\Sigma}$ adalah matriks diagonal (atau hampir diagonal) berisi **singular values**

Makna Geometris: SVD menyatakan bahwa transformasi linear $\mathbf{A}$ dapat direpresentasikan sebagai rotasi (oleh $\mathbf{V}^T$), skala (oleh $\mathbf{\Sigma}$), dan rotasi lain (oleh $\mathbf{U}$).

Aplikasi SVD:

- **Reduksi dimensi:** Dengan menyimpan hanya singular values terbesar, kita dapat mengaproksimasi matriks asli dengan kompleksitas lebih rendah.
- **PCA:** PCA secara implisit menggunakan SVD untuk menghitung komponen utama.
- **Kompresi citra:** SVD digunakan untuk menyimpan citra dalam representasi yang lebih hemat memori.
- **Rekomendasi sistem:** Dalam sistem seperti Netflix atau Spotify, SVD digunakan untuk memperkirakan preferensi pengguna.

2.3 Kalkulus: Turunan dan Aplikasinya

Turunan adalah konsep inti dalam kalkulus yang berperan sangat penting dalam machine learning, terutama dalam proses optimasi, pembaruan parameter model, dan pelatihan jaringan saraf. Turunan menggambarkan laju perubahan suatu fungsi dan dapat digunakan untuk mencari nilai minimum (atau maksimum) dari suatu fungsi.

2.3.1 Dasar dan Arti dari Turunan

Turunan dari suatu fungsi menggambarkan seberapa cepat nilai fungsi berubah terhadap perubahannya pada variabel input. Secara geometris, turunan dari fungsi $f(x)$ di titik $x = a$ adalah kemiringan garis singgung terhadap kurva $f(x)$ di titik tersebut.

$$f'(a) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h}$$

Contoh: Misalkan $f(x) = x^2$, maka:

$$f'(x) = \frac{d}{dx}\left(x^2\right) = 2x$$

Sehingga $f'(3) = 6$ menunjukkan bahwa pada $x = 3$, nilai fungsi f sedang naik dengan kecepatan 6 unit per satuan perubahan x.

2.3.2 Turunan Pertama dan Kedua

Turunan pertama memberikan laju perubahan fungsi terhadap variabel input.

Turunan kedua memberikan laju perubahan dari turunan pertama—yakni bagaimana kecepatan perubahan itu sendiri berubah.

$$f(x) = x^3 \Rightarrow f'(x) = 3x^2, \quad f''(x) = 6x$$

Turunan kedua sering digunakan untuk menentukan sifat dari titik-titik ekstrem (minimum/maksimum) pada suatu fungsi.

2.3.3 Turunan untuk Fungsi Banyak Variabel

Dalam machine learning, kita sering berhadapan dengan fungsi dari banyak variabel, seperti fungsi kerugian $L(w_1, w_2, ..., w_n)$ yang tergantung pada banyak parameter.

Untuk fungsi $f(x, y)$, kita dapat mengambil **turunan parsial** terhadap masing-masing variabel:

$$\frac{\partial f}{\partial x}, \quad \frac{\partial f}{\partial y}$$

Contoh: Misalkan $f(x, y) = x^2y + y^3$, maka:

$$\frac{\partial f}{\partial x} = 2xy, \quad \frac{\partial f}{\partial y} = x^2 + 3y^2$$

Turunan parsial menunjukkan bagaimana fungsi berubah jika hanya satu variabel yang berubah, sementara variabel lainnya dianggap tetap.

2.3.4 Turunan Berarah dan Gradien

Turunan berarah adalah laju perubahan fungsi ke arah vektor tertentu. Dalam ruang berdimensi banyak, arah tercepat untuk menaikkan fungsi adalah arah **gradien**.

Gradien adalah vektor yang terdiri dari semua turunan parsial dari suatu fungsi:

$$\nabla f(x, y) = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

Gradien menunjukkan arah kenaikan tercepat dari fungsi. Sebaliknya, arah negatif gradien adalah arah penurunan tercepat.

2.3.5 Titik Kritis dan Uji Turunan Kedua

Titik kritis suatu fungsi adalah titik di mana turunan pertama sama dengan nol:

$$f'(x) = 0$$

Titik ini bisa menjadi minimum, maksimum, atau titik belok. Untuk menentukan jenisnya, kita gunakan **uji turunan kedua** (hanya untuk satu variabel):

- Jika $f''(x) > 0$, maka x adalah **minimum lokal** - Jika $f''(x) < 0$, maka x adalah **maksimum lokal** - Jika $f''(x) = 0$, maka diperlukan analisis lanjutan

Contoh:

$$f(x) = x^2 \Rightarrow f'(x) = 2x, f''(x) = 2 \Rightarrow f'(0) = 0 \text{ dan } f''(0) > 0 \Rightarrow \text{minimum di } x = 0$$

2.3.6 Gradient Descent

Gradient descent adalah algoritma optimasi yang digunakan untuk meminimalkan fungsi, khususnya fungsi kerugian (*loss*) dalam pembelajaran mesin (*machine learning*). Secara singkat, fungsi kerugian (*loss function*) dalam *machine learning* adalah fungsi yang mengukur seberapa besar kesalahan model *machine learning* tersebut. Semakin kecil nilai fungsi tersebut, semakin baik model yang didapatkan.

Prinsip dasar dari gradient descent adalah memulai dari sebuah titik awal (biasanya acak), lalu bergerak ke arah negatif gradien dari fungsi untuk menuruni nilai fungsi tersebut secara bertahap, sehingga akhirnya menemukan titik minimumnya.

$$x_{\text{baru}} = x_{\text{lama}} - \eta \cdot f'(x)$$

Di mana:

- η adalah **learning rate**, yaitu ukuran langkah (step size) yang menentukan seberapa besar perubahan setiap iterasi.
- $f'(x)$ adalah turunan (gradien) dari fungsi $f(x)$ pada titik x.

Contoh:

Kita ingin meminimalkan fungsi:

$$f(x) = x^2 \Rightarrow f'(x) = 2x$$

Misalkan kita mulai dari $x_0 = 4$ dan gunakan $\eta = 0.1$. Maka:

$$\begin{aligned}
x_1 &= x_0 - \eta \cdot f'(x_0) = 4 - 0.1 \cdot 2 \cdot 4 = 4 - 0.8 = 3.2 \\
x_2 &= x_1 - 0.1 \cdot 2 \cdot 3.2 = 3.2 - 0.64 = 2.56 \\
x_3 &= x_2 - 0.1 \cdot 2 \cdot 2.56 = 2.56 - 0.512 = 2.048 \\
x_4 &= x_3 - 0.1 \cdot 2 \cdot 2.048 = 2.048 - 0.4096 = 1.6384 \\
x_5 &= x_4 - 0.1 \cdot 2 \cdot 1.6384 = 1.6384 - 0.32768 = 1.31072
\end{aligned}$$

Dengan hanya 5 iterasi, nilai x sudah mendekati titik minimum global $x = 0$.

Gradient descent sederhana ini bekerja sangat baik pada fungsi-fungsi konveks dan mulus. Dalam machine learning modern, konsep ini diperluas ke fungsi multivariat (dengan banyak parameter) dan dimanfaatkan dalam pelatihan model seperti regresi, jaringan saraf turuan (*artificial neural network*), dan sebagainya.

Optimasi Fungsi Multivariabel dan Gradien Vektor

Dalam machine learning, fungsi yang kita optimasi biasanya bergantung pada banyak parameter (misalnya bobot dalam jaringan saraf). Oleh karena itu, kita perlu memperluas konsep turunan dari satu variabel ke banyak variabel.

Misalkan kita memiliki fungsi kerugian $L(w_1, w_2, ..., w_n)$ di mana w_i adalah parameter model. Maka gradien dari L terhadap parameter-parameter tersebut adalah vektor:

$$\nabla L = \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \frac{\partial L}{\partial w_2} \\ \vdots \\ \frac{\partial L}{\partial w_n} \end{bmatrix}$$

Vektor gradien ini menunjukkan arah naik tercepat dari fungsi. Maka untuk menurunkan nilai fungsi (misalnya meminimalkan fungsi loss), kita bergerak ke arah negatif gradien.

Gradient Descent Multivariat:

$$\vec{w}_{\text{baru}} = \vec{w}_{\text{lama}} - \eta \cdot \nabla L$$

di mana:

- $\vec{w}$ adalah vektor parameter (misalnya $[w_1, w_2, ..., w_n]$),
- η adalah learning rate (ukuran langkah),
- ∇L adalah gradien vektor.

Contoh:
Misalkan fungsi kerugian adalah:

$$L(w_1, w_2) = (w_1 - 2)^2 + (w_2 + 3)^2$$

Maka:

$$\frac{\partial L}{\partial w_1} = 2(w_1 - 2), \quad \frac{\partial L}{\partial w_2} = 2(w_2 + 3)$$

Gradiennya:

$$\nabla L = \begin{bmatrix} 2(w_1 - 2) \\ 2(w_2 + 3) \end{bmatrix}$$

Langkah update parameter:

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix}_{\text{baru}} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}_{\text{lama}} - \eta \cdot \begin{bmatrix} 2(w_1 - 2) \\ 2(w_2 + 3) \end{bmatrix}$$

Interpretasi Geometris: Permukaan dari fungsi $L(w_1, w_2)$ berbentuk mangkuk (kuadratik konveks). Gradien menunjukkan arah curam. Gradient descent akan menuruni permukaan ini menuju titik minimum ($w_1 = 2, w_2 = -3$).

Catatan: Pada fungsi yang kompleks (seperti loss neural network), bentuk permukaan bisa mengandung *saddle point* dan lokal minima. Oleh karena itu, teknik gradient

descent dimodifikasi menjadi varian lain seperti *Stochastic Gradient Descent (SGD)* dan *ADAM*, yang akan dibahas di modul berikutnya.

Referensi dan Bahan Bacaan Lanjutan

Untuk pendalaman konsep matematika dasar yang digunakan dalam machine learning, lihat:

- Deisenroth, M. P., Faisal, A. A., & Ong, C. S. (2020). *Mathematics for Machine Learning*. Cambridge University Press. [DFO20]
- Duke University. *Data Science Math Skills*. Kursus online di Coursera. https://www.coursera.org/learn/datasciencemathskills [Uni23]
- A Gentle Introduction To Gradient Descent Procedure https://machinelearningmastery.com/
- Khan Academy: Multivariabel Calculus & Gradient Descent https://www.khanacademy.org/math/multivariable-calculus/

BAB 3

Dasar-dasar Statistika

3.1 Pendahuluan: Data dan Variabel Acak

Statistika adalah ilmu yang berfokus pada pengumpulan, penyajian, analisis, dan interpretasi data. Dalam konteks machine learning, statistika sangat penting untuk memahami pola data, melakukan analisis awal (*Exploratory Data Analysis*/EDA), dan mengevaluasi performa model.

3.1.1 Apa itu Data dan Datum?

Datum adalah satuan terkecil dari informasi, yaitu satu pengamatan atau nilai tunggal. Sedangkan **data** adalah kumpulan dari banyak datum. Data dapat berbentuk angka, kategori, teks, atau simbol, tergantung pada konteks dan tujuan analisis.

Contoh:

- Datum: Suhu di Jakarta hari ini = 31°C
- Data: Suhu harian di Jakarta selama seminggu = [31, 30, 29, 32, 33, 30, 28]

3.1.2 Variabel dan Variabel Acak

Dalam statistika, **variabel** adalah karakteristik atau atribut yang diukur atau diamati.

- **Variabel deterministik**: nilainya tetap dan pasti. Contoh: jumlah sisi pada dadu = 6.
- **Variabel acak (random variable)**: nilainya tidak pasti, bergantung pada hasil dari suatu proses acak.

Contoh: Lempar sebuah dadu, hasilnya bisa 1 sampai 6. Maka variabel X = hasil lemparan dadu adalah variabel acak diskret.

3.1.3 Jenis Data dan Variabel

Data dapat dibedakan berdasarkan jenisnya:

- **Data numerik (kuantitatif)**:
  - *Diskret*: bilangan bulat yang dihitung (contoh: jumlah anak)

  - *Kontinu*: nilai dari pengukuran (contoh: tinggi badan, suhu)

- **Data kategorikal (kualitatif)**:

  - *Nominal*: tidak memiliki urutan (contoh: warna, jenis kelamin)
  - *Ordinal*: memiliki urutan, tetapi jarak antar nilai tidak bermakna (contoh: tingkat kepuasan: rendah, sedang, tinggi)

3.1.4 Contoh Ilustratif: Dataset Mahasiswa

Misalkan kita memiliki dataset mahasiswa dengan atribut berikut:

- Nama (string, kategorikal)
- Umur (angka, numerik kontinu)
- Jurusan (kategorikal nominal)
- IPK (angka, numerik kontinu)
- Tingkat kepuasan terhadap perkuliahan (ordinal: rendah, sedang, tinggi)

Dalam machine learning, penting untuk mengetahui jenis setiap variabel karena:

- Menentukan metode statistik yang tepat untuk analisis
- Menentukan jenis encoding yang digunakan saat pemrosesan data
- Menentukan distribusi peluang yang sesuai untuk pemodelan

3.1.5 Representasi Variabel Acak

Variabel acak biasanya dinotasikan dengan huruf kapital (X, Y, dll), dan nilainya berasal dari suatu **ruang sampel**.

Contoh: Jika X adalah jumlah kepala dalam dua kali lempar koin, maka nilai-nilai yang mungkin dari X adalah:

$$\text{Ruang sampel} = \{HH, HT, TH, TT\}, \quad X \in \{0, 1, 2\}$$

Nilai X tergantung pada hasil proses acak. Kita bisa menetapkan **distribusi peluang** ke setiap nilai yang mungkin dari X—yang akan dibahas lebih lanjut dalam bab ini.

3.2 Peluang dan Kombinatorika

3.2.1 Apa itu Peluang?

Peluang (probabilitas) adalah ukuran kemungkinan terjadinya suatu kejadian acak. Nilai peluang selalu berada antara 0 dan 1:

$$0 \leq P(A) \leq 1$$

dengan:

- $P(A) = 0$: kejadian A pasti tidak terjadi.
- $P(A) = 1$: kejadian A pasti terjadi.
- $P(A) = 0.5$: kejadian A dan komplementernya sama-sama mungkin terjadi.

3.2.2 Ruang Sampel dan Kejadian

Ruang sampel (S) adalah himpunan semua hasil yang mungkin dari suatu percobaan acak. **Kejadian** (A) adalah himpunan bagian dari ruang sampel.

Contoh: Melempar satu dadu.

$$S = \{1, 2, 3, 4, 5, 6\}, \quad A = \{\text{angka genap}\} = \{2, 4, 6\}$$

Maka peluang munculnya angka genap:

$$P(A) = \frac{|A|}{|S|} = \frac{3}{6} = 0.5$$

3.2.3 Aturan Penjumlahan Peluang

Jika dua kejadian A dan B saling lepas (tidak bisa terjadi bersamaan), maka:

$$P(A \cup B) = P(A) + P(B)$$

Contoh: Melempar dadu, berapa peluang keluar angka 2 atau 5?

$$P(\{2\}) = \frac{1}{6}, \quad P(\{5\}) = \frac{1}{6} \Rightarrow P(2 \text{ atau } 5) = \frac{1}{6} + \frac{1}{6} = \frac{2}{6} = \frac{1}{3}$$

Jika A dan B tidak saling lepas:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

3.2.4 Aturan Perkalian Peluang

Untuk dua kejadian A dan B yang independen:

$$P(A \cap B) = P(A) \cdot P(B)$$

Contoh: Peluang mendapat "kepala" dua kali berturut-turut saat melempar koin dua kali:

$$P(HH) = P(H) \cdot P(H) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$$

Untuk kejadian bersyarat (dependent), digunakan:

$$P(A \cap B) = P(A) \cdot P(B|A)$$

3.2.5 Kombinatorika Dasar

Kombinatorika membantu menghitung jumlah kemungkinan dalam situasi kompleks tanpa harus menuliskan semua kasus.

1. Aturan Perkalian (Multiplikasi)

Jika suatu proses terdiri dari dua langkah, dan langkah pertama memiliki m kemungkinan, dan langkah kedua memiliki n kemungkinan, maka total kombinasi $= m \cdot n$.

Contoh: Memilih 1 minuman dari 3 jenis dan 1 makanan dari 4 jenis $\rightarrow$ total $= 3 \cdot 4 = 12$ kombinasi.

2. Aturan Penjumlahan

Jika ada dua kejadian yang tidak dapat terjadi bersamaan, dan masing-masing memiliki m dan n cara, maka total cara $= m + n$.

3. Permutasi dan Kombinasi

- **Permutasi**: pengurutan penting

$$P(n, r) = \frac{n!}{(n-r)!}$$

Contoh: Susunan 3 dari 5 peserta lomba.

- **Kombinasi**: urutan tidak penting

$$C(n, r) = \binom{n}{r} = \frac{n!}{r!\,(n-r)!}$$

Contoh: Memilih 2 dari 4 siswa untuk maju tanpa memperhatikan urutan.

Contoh Kombinasi:

$$\binom{4}{2} = \frac{4!}{2!2!} = \frac{24}{4} = 6$$

Contoh Soal: Dari 3 koin dilempar sekaligus, berapa peluang mendapatkan tepat 2 kepala?

- Ruang sampel: $\{HHH, HHT, HTH, HTT, THH, THT, TTH, TTT\} \rightarrow 8$ total - Kejadian $= \{HHT, HTH, THH\} \rightarrow 3$ kasus

$$P(\text{2 kepala}) = \frac{3}{8}$$

Contoh 1: Memilih Tim dari Sekelompok Orang

Dari 10 siswa, akan dipilih 3 orang secara acak untuk membentuk tim. Berapa peluang bahwa siswa bernama A termasuk dalam tim?

Langkah 1: Hitung total cara memilih 3 orang dari 10:

$$\text{Total kombinasi} = \binom{10}{3} = 120$$

Langkah 2: Hitung jumlah cara memilih tim yang mencakup A:

- Kita sudah memilih A (1 orang), tinggal memilih 2 orang lagi dari 9 siswa lainnya:

$$\text{Kombinasi dengan A} = \binom{9}{2} = 36$$

Hasil:

$$P\,(\text{A terpilih}) = \frac{36}{120} = 0.3$$

Contoh 2: Menyusun Kata dari Huruf

Dari huruf-huruf dalam kata `STATISTIK`, akan dibentuk kata baru sepanjang 5 huruf. Berapa peluang bahwa huruf pertama adalah 'S'?

Langkah 1: Hitung total kemungkinan kata 5 huruf dari huruf-huruf tersebut.

Huruf-huruf dalam "STATISTIK" terdiri dari: - S: 2 buah - T: 3 buah - A, I, K: masing-masing 1 buah

Karena huruf ada yang berulang, kita harus gunakan teknik permutasi dengan pengulangan. Tapi karena kombinatorika eksak dari multiset ini cukup rumit, kita sederhanakan dengan asumsi: - Kita ambil 5 huruf secara acak dari huruf-huruf tersebut (tanpa memperhatikan urutan duplikat) - Kita hitung total kemungkinan permutasi dari 5 huruf (tanpa duplikat)

Langkah 2: Hitung jumlah cara memilih dan menyusun kata dengan huruf pertama adalah S:

- Pilih 4 huruf lainnya dari huruf selain S - Hitung permutasinya (dengan mempertimbangkan pengulangan jika ada)

Untuk mempermudah, kita bisa jadikan ini contoh konseptual saja dan bahas idenya:

Ide penyelesaian:

- Total kemungkinan = jumlah semua permutasi 5 huruf dari huruf-huruf di kata "STATISTIK"
- Hasil yang diinginkan = jumlah permutasi 5 huruf yang dimulai dengan 'S'
- Maka:

$$P\,(\text{huruf pertama S}) = \frac{\text{jumlah permutasi yang dimulai dengan S}}{\text{jumlah semua permutasi 5 huruf}}$$

Catatan: Untuk kasus seperti ini, aturan kombinatorika sering digunakan sebagai teknik penghitungan ruang sampel, terutama jika enumerasi eksplisit terlalu besar atau tidak praktis.

3.3 Statistika Deskriptif

Statistika deskriptif digunakan untuk meringkas dan menyajikan data agar lebih mudah dipahami. Teknik ini membantu kita mengenali pola umum, penyebaran, dan karakteristik utama dari kumpulan data sebelum melakukan analisis lebih lanjut.

Statistika deskriptif biasanya dibagi menjadi tiga kelompok utama:

1. Ukuran pemusatan (central tendency)
2. Ukuran penyebaran (dispersion)
3. Ukuran posisi (position)

3.3.1 Ukuran Pemusatan

Mean (Rataan): Jumlah seluruh nilai dibagi banyaknya data.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Median: Nilai tengah dari data yang sudah diurutkan.
Modus: Nilai yang paling sering muncul.
Contoh: Diberikan data tinggi badan (dalam cm):

$$\{160, 165, 170, 165, 180, 170, 165\}$$

- Rataan = $\frac{160+165+170+165+180+170+165}{7} = \frac{1175}{7} \approx 167.86$
- Median = 165 (data tengah dari urutan: 160, 165, 165, **165**, 170, 170, 180)
- Modus = 165 (muncul 3 kali)

3.3.2 Ukuran Penyebaran

Range: Selisih nilai maksimum dan minimum.
Variansi (s^2): Rata-rata kuadrat selisih antara nilai data dan mean-nya.

$$s^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \bar{x})^2$$

Standar Deviasi (s): Akar kuadrat dari variansi.
Contoh: Data nilai ujian:

$$\{80, 85, 90, 95, 100\}$$

- Rataan = 90
- Variansi = $\frac{(80-90)^2+(85-90)^2+(90-90)^2+\ldots}{5} = \frac{250}{5} = 50$
- Standar deviasi = $\sqrt{50} \approx 7.07$

3.3.3 Ukuran Posisi

Kuartil: Membagi data menjadi empat bagian sama besar. - Q1 (kuartil pertama): batas 25- Q2 = median (50- Q3: batas 75

Interquartile Range (IQR): Selisih antara Q3 dan Q1, mengukur rentang tengah 50

$$\text{IQR} = Q3 - Q1$$

Persentil: Membagi data menjadi 100 bagian. Persentil ke-k menunjukkan batas bawah dari $k\%$ data terendah.
Contoh: Data pendapatan bulanan (juta rupiah):

$$\{3, 3, 4, 5, 6, 7, 9, 10, 11\}$$

- ◇ Median (Q2) = 6
- ◇ Q1 = 4, Q3 = 9
- ◇ IQR = 9 - 4 = 5

3.3.4 Kegunaan dalam Machine Learning

Statistika deskriptif digunakan untuk:

- ◇ Mengetahui apakah data mengandung pencilan (outlier)
- ◇ Membandingkan distribusi antar fitur
- ◇ Melakukan normalisasi atau scaling (berbasis rataan dan deviasi)
- ◇ Menyaring fitur yang tidak relevan (misalnya, fitur dengan variansi sangat kecil)

3.4 Outlier (Pencilan)

Outlier adalah nilai yang secara signifikan berbeda dari sebagian besar data lainnya. Outlier dapat muncul karena kesalahan pengukuran, variasi alami, atau fenomena langka. Dalam analisis data dan machine learning, penting untuk mengidentifikasi outlier karena mereka dapat:

- ◇ Mempengaruhi rataan dan standar deviasi secara ekstrem
- ◇ Mempengaruhi distribusi data dan hasil model regresi
- ◇ Menyebabkan model overfitting pada kasus yang tidak representatif

3.4.1 Contoh Sederhana

Misalkan data tinggi badan (dalam cm):

$$\{160, 162, 161, 159, 158, 163, \underline{185}\}$$

Mayoritas data berkisar antara 158–163 cm, tetapi 185 cm tampak jauh lebih tinggi dari yang lain — kemungkinan merupakan outlier.

3.4.2 Metode Deteksi Outlier

1. Z-Score (Standar Deviasi)

Z-score mengukur berapa banyak standar deviasi sebuah nilai berbeda dari rataan.

$$z = \frac{x - \bar{x}}{s}$$

Jika $|z| > 2$ atau 3, maka nilai tersebut sering dianggap sebagai outlier (tergantung konteks).

Contoh:

Untuk data:

$$\{10, 12, 11, 13, 90\}$$

- Rataan = 27.2, Standar deviasi = 32 - Z-score dari 90: $\approx (90 - 27.2)/32 = 1.96$

Tidak terlalu ekstrem, tapi dalam konteks data kecil ini 90 mungkin tetap dipertimbangkan sebagai pencilan.

2. Metode IQR (Interquartile Range)

Outlier bisa juga dideteksi dengan rumus:

$$\text{Batas bawah} = Q1 - 1.5 \cdot \text{IQR}, \quad \text{Batas atas} = Q3 + 1.5 \cdot \text{IQR}$$

Nilai di luar rentang ini dianggap sebagai outlier.

Contoh:

Data pendapatan (juta rupiah):

$$\{3, 4, 5, 6, 7, 8, 20\}$$

- Q1 = 4.5, Q3 = 7.5, IQR = 3 - Batas bawah = 4.5 - 4.5 = 0 - Batas atas = 7.5 + 4.5 = 12

Maka $20 > 12 \rightarrow$ dianggap outlier.

3.4.3 Penanganan Outlier

Beberapa pendekatan:

- **Dihapus:** Jika diyakini sebagai kesalahan atau sangat langka.
- **Diubah (capping/winsorizing):** Dibatasi ke nilai ambang tertentu.
- **Dianalisis terpisah:** Jika relevan untuk insight atau anomaly detection.
- **Dipertahankan:** Jika memang mewakili fenomena nyata yang penting (misalnya transaksi besar dalam data keuangan).

3.4.4 Outlier (pencilan) dalam Machine Learning

Dalam machine learning, outlier (pencilan) dapat:

- Menurunkan akurasi model regresi
- Memengaruhi hasil clustering dan PCA
- Menyebabkan konvergensi training lebih lambat atau tidak stabil

Karena itu, deteksi dan penanganan outlier merupakan bagian penting dari proses eksplorasi dan praproses data.

3.5 Distribusi Variabel Acak

Distribusi probabilitas menggambarkan bagaimana nilai-nilai dari sebuah variabel acak tersebar. Dalam statistika dan machine learning, memahami bentuk distribusi sangat penting untuk memilih model, teknik evaluasi, dan strategi transformasi data.

Distribusi dibagi menjadi dua jenis utama:

- **Diskret:** Variabel acak mengambil nilai terbatas atau terhitung (contoh: jumlah pelanggan, jumlah keberhasilan)
- **Kontinu:** Variabel acak dapat mengambil nilai dari rentang kontinu (contoh: tinggi badan, suhu)

3.5.1 Distribusi Uniform (Seragam)

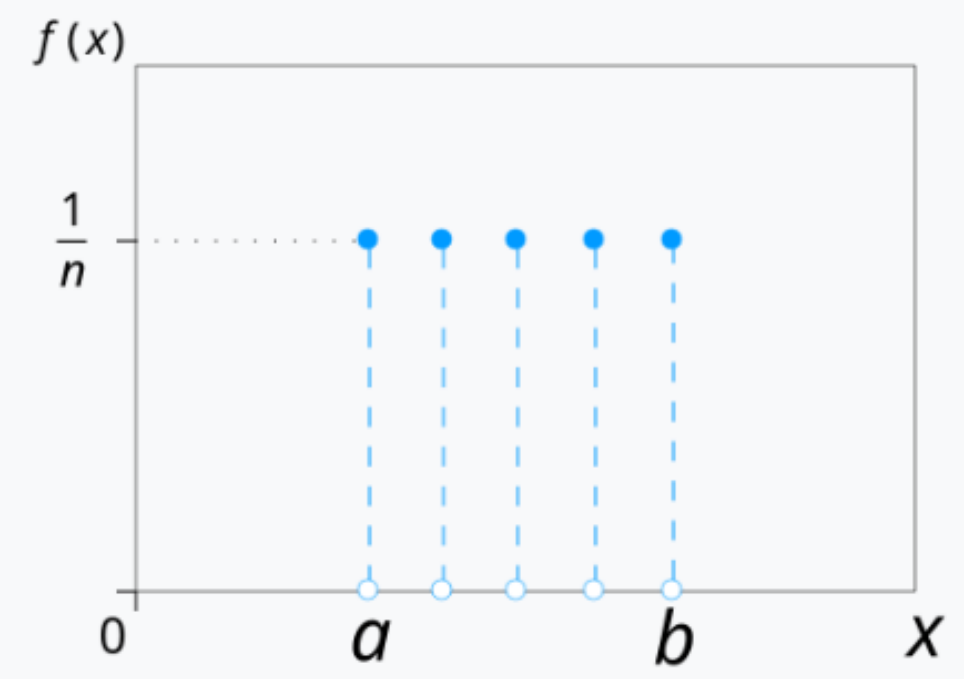
Definisi: Semua nilai dalam rentang tertentu memiliki peluang yang sama.

- Diskret: peluang melempar dadu (1–6) $\rightarrow$ semua hasil punya $P = \frac{1}{6}$
- Kontinu: nilai acak antara 0 dan 1, dengan densitas konstan

Ciri khas: - Tidak ada nilai yang lebih "umum" daripada yang lain. - Bentuk grafiknya datar (konstan).
Contoh:

- Random sampling: memilih angka acak antara 0 dan 1.
- Pemodelan noise atau ketidakpastian awal.

Distribusi seragam untuk variabel acak diskret dapat digambarkan pada Gambar 3.1



Gambar 3.1. Distribusi uniform (seragam) diskret

3.5.2 Distribusi Normal (Gaussian)

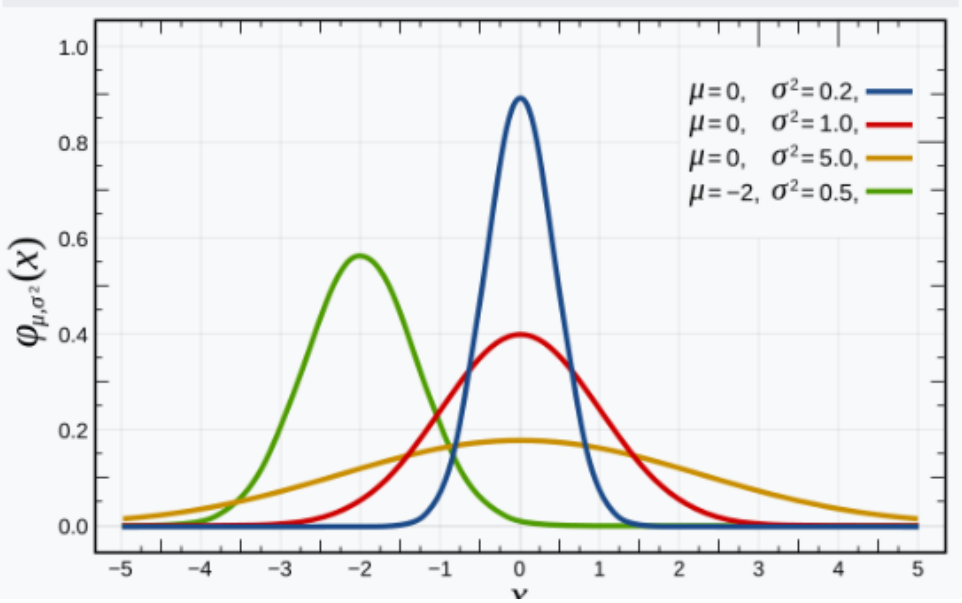
Definisi: Nilai terdistribusi secara simetris di sekitar rataan. Paling sering dijumpai dalam fenomena alam dan sosial.

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Ciri khas: - Simetris, berbentuk lonceng (bell-shaped curve) - Rataan = median = modus - Sekitar 68% data berada dalam 1 standar deviasi dari mean, 95% dalam 2 sd
Contoh:

- Tinggi badan manusia
- Error dalam pengukuran
- Noise dalam model regresi

Kurva distribusi normal dengan berbagai kemungkinan nilai parameternya ditunjukkan pada Gambar 3.2.



Gambar 3.2. Distribusi normal kontinyu

Catatan: Banyak algoritma machine learning (misalnya linear regression, naive Bayes) mengasumsikan data berdistribusi normal.

3.5.3 Distribusi Binomial

Definisi: Mengukur jumlah keberhasilan dalam n percobaan Bernoulli (ya/tidak) yang identik, dengan peluang sukses p pada setiap percobaan.

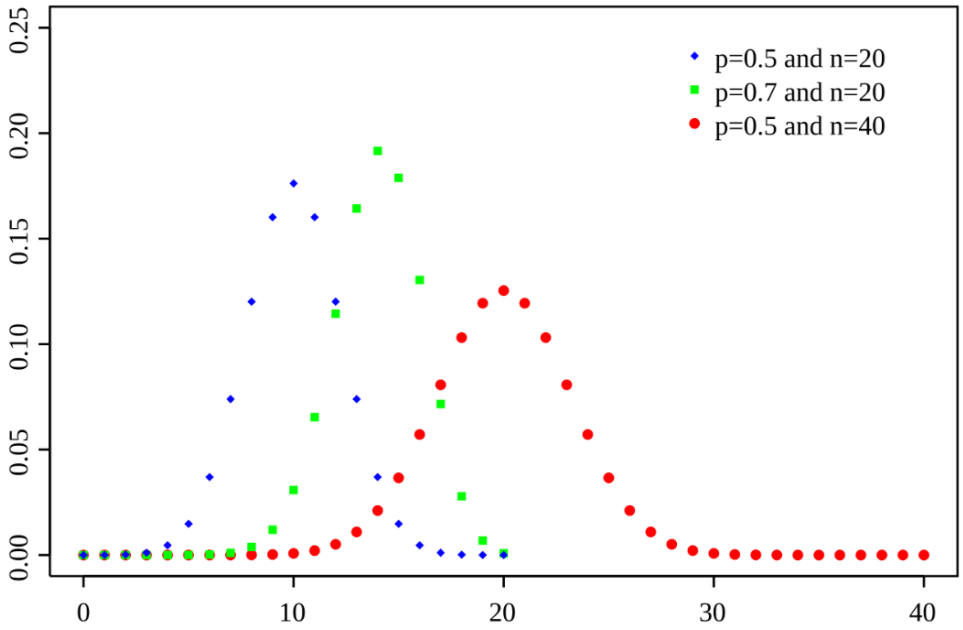
$$P(k \text{ sukses}) = \binom{n}{k} p^k (1-p)^{n-k}$$

Ciri khas: - Diskret - Cocok untuk percobaan dengan dua hasil: sukses/gagal
Contoh:

- Peluang mendapat 3 kepala dari 5 lemparan koin.

- Peluang 7 dari 10 orang menjawab benar dalam kuis multiple choice.

Kurva distribusi binomial dengan berbagai kemungkinan nilai p dan n ditunjukkan pada Gambar 3.3.



Gambar 3.3. Distribusi normal kontinyu

3.5.4 Distribusi Poisson

Definisi: Mengukur jumlah kejadian dalam suatu interval waktu atau ruang, jika kejadian tersebut terjadi secara acak namun dengan rata-rata tetap λ.

$$P\left(k \text{ kejadian}\right) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Ciri khas: - Diskret - Digunakan untuk kejadian langka dalam interval tetap - Tidak ada batas maksimum, meskipun nilai besar jadi makin jarang

Contoh:

- Jumlah pelanggan yang masuk ke toko per jam.
- Jumlah error dalam 1 halaman cetak.
- Jumlah gempa bumi kecil di satu daerah dalam sebulan.

3.5.5 Perbandingan Distribusi Diskret vs Kontinu

- Binomial dan Poisson adalah distribusi **diskret**
- Uniform dan Gaussian biasanya digunakan dalam bentuk **kontinu**
- Banyak distribusi diskret menjadi semakin mirip distribusi normal jika jumlah percobaan/sampel data meningkat (misalnya: binomial $\longrightarrow$ normal untuk n besar)

3.5.6 Distribusi Sampling dan Inferensi Sederhana

Distribusi Sampling

Distribusi sampling adalah distribusi probabilitas dari suatu statistik (misalnya rataan) yang dihitung dari banyak sampel acak dari suatu populasi.

Misalnya:

- Kita memiliki populasi siswa SMA se-Indonesia.
- Kita ambil 100 sampel acak, masing-masing berisi 30 siswa.
- Dari setiap sampel, kita hitung rata-rata nilai matematika.
- Distribusi dari rata-rata nilai tersebut (100 buah) disebut **distribusi sampling**.

Distribusi sampling sangat penting untuk:

- Mengestimasi parameter populasi dari sampel terbatas.
- Menghitung seberapa besar ketidakpastian dari estimasi tersebut.

Standard Error (SE)

Standard error adalah ukuran seberapa menyebar rataan-rataan sampel dari distribusi sampling.

$$\text{SE} = \frac{\sigma}{\sqrt{n}}$$

di mana:

- σ = standar deviasi populasi
- n = ukuran sampel

Jika σ tidak diketahui, kita gunakan standar deviasi sampel s sebagai aproksimasi.

Contoh: Diketahui bahwa nilai ujian siswa dalam populasi memiliki standar deviasi $\sigma = 10$. Jika kita mengambil sampel sebanyak $n = 25$, maka:

$$\text{SE} = \frac{10}{\sqrt{25}} = 2$$

Ini berarti, rata-rata nilai sampel kita memiliki simpangan sekitar 2 dari nilai rata-rata populasi.

Confidence Interval (Interval Kepercayaan)

Kita dapat membangun rentang estimasi untuk parameter populasi, misalnya rata-rata μ.

Untuk distribusi normal:

$$\text{CI}_{95\%} = \bar{x} \pm 1.96 \cdot \text{SE}$$

Artinya, kita 95% yakin bahwa rata-rata populasi μ berada dalam rentang tersebut.

Contoh: Sebuah sampel menghasilkan $\bar{x} = 75$, $s = 12$, $n = 36$:

$$\text{SE} = \frac{12}{\sqrt{36}} = 2$$

$$\text{CI}_{95\%} = 75 \pm 1.96 \cdot 2 = 75 \pm 3.92 \Rightarrow [71.08, 78.92]$$

Aplikasi dalam Evaluasi Model Machine Learning

Dalam machine learning, kita sering membagi data menjadi *train/test split* atau menggunakan *cross-validation*. Setiap pengujian dapat dianggap sebagai **sampel** dari performa model.

Distribusi dari skor akurasi/kerugian pada banyak eksperimen tersebut membentuk distribusi sampling dari metrik model.

Contoh: Akurasi model pada 5 fold cross-validation:

$$[0.78, 0.82, 0.80, 0.79, 0.81]$$

Rata-rata: $\bar{x} = 0.80$, Standar deviasi sampel $s = 0.0158$, maka:

$$\text{SE} = \frac{0.0158}{\sqrt{5}} \approx 0.00707$$

$$\text{CI}_{95\%} = 0.80 \pm 1.96 \cdot 0.00707 \approx [0.786, 0.814]$$

Dengan demikian, kita tidak hanya melaporkan **nilai akurasi**, tetapi juga **ketidakpastian dari estimasi tersebut**.

3.6 Korelasi dan Hubungan Antar Variabel

Dalam analisis data, kita sering tertarik untuk mengetahui apakah dua variabel memiliki hubungan. Salah satu cara mengukurnya adalah dengan **korelasi**, yaitu sejauh mana dua variabel berubah secara bersamaan.

3.6.1 Apa itu Korelasi?

Korelasi adalah ukuran statistik yang menyatakan kekuatan dan arah hubungan linear antara dua variabel numerik.

- Jika dua variabel cenderung meningkat atau menurun bersama, disebut **korelasi positif**.
- Jika satu meningkat sementara yang lain menurun, disebut **korelasi negatif**.
- Jika tidak ada pola linier yang jelas, dikatakan **tidak berkorelasi** (korelasi mendekati nol).

3.6.2 Koefisien Korelasi Pearson

Koefisien korelasi Pearson (r) mengukur kekuatan hubungan linear antara dua variabel numerik.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \cdot \sqrt{\sum (y_i - \bar{y})^2}}$$

Nilai r selalu berada antara -1 dan +1:

- $r = +1$: korelasi positif sempurna
- $r = -1$: korelasi negatif sempurna
- $r = 0$: tidak ada korelasi linier

Contoh:
Misalkan data tinggi badan dan berat badan:

$$x = \{160, 165, 170\}, \quad y = \{55, 60, 65\}$$

Tinggi dan berat naik bersama $\rightarrow$ r mendekati $+1$.

3.6.3 Visualisasi Korelasi

Hubungan antar dua variabel dapat divisualisasikan dengan:

- **Scatter plot**: titik-titik data akan membentuk pola diagonal jika berkorelasi.
- **Heatmap korelasi**: digunakan untuk banyak variabel sekaligus.

Gambar 3.4 berikut memberikan contoh beberapa nilai korelasi dan visualisasi data yang menggambarkan korelasi tersebut.

3.6.4 Korelasi vs Kausalitas

Penting untuk diingat bahwa:

$$\textbf{Korelasi} \not\Rightarrow \textbf{Kausalitas}$$

Contoh:

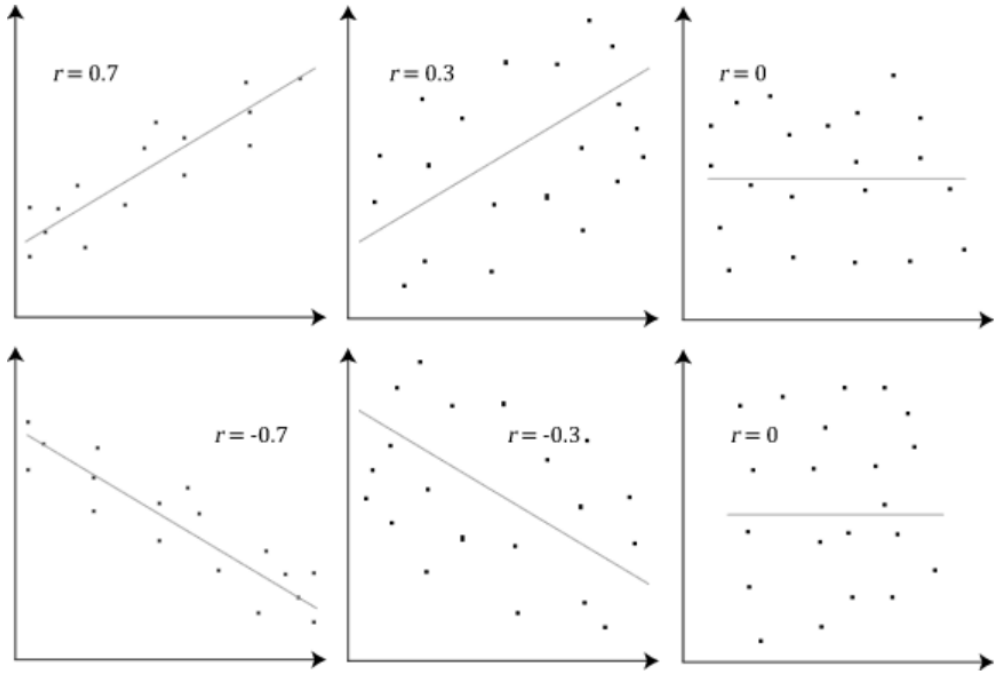
- Korelasi antara konsumsi es krim dan tenggelam di laut: bukan sebab-akibat, melainkan faktor musiman (cuaca panas).

Dalam machine learning dan statistik, kita harus berhati-hati agar tidak menyimpulkan bahwa satu variabel menyebabkan yang lain hanya karena mereka berkorelasi.

3.6.5 Korelasi dalam Machine Learning

Beberapa penggunaan dan perhatian dalam ML:

- **Seleksi fitur:** fitur yang sangat berkorelasi bisa menyebabkan informasi berulang.
- **Multikolinearitas:** dua atau lebih fitur berkorelasi tinggi $\rightarrow$ bisa menyebabkan model regresi tidak stabil.



Gambar 3.4. Scatter plot korelasi

- **Diagnosis data:** korelasi awal membantu menentukan apakah transformasi data diperlukan.

Contoh Praktis:

- Dalam regresi linier, dua fitur "umur" dan "pengalaman kerja" mungkin berkorelasi tinggi → bisa mengganggu interpretasi koefisien.
- Dalam NLP, korelasi antar fitur token (misalnya jumlah huruf kapital vs panjang kata) bisa mengungkap informasi tersembunyi.

3.7 Analisis Data Eksploratif (EDA)

Analisis Data Eksploratif (Exploratory Data Analysis / EDA) adalah proses sistematis untuk memahami struktur data, pola, anomali, dan hubungan antar variabel sebelum membangun model prediktif.

EDA sangat penting dalam machine learning karena:

- Membantu mengidentifikasi fitur yang relevan atau tidak relevan
- Mengungkap outlier dan data yang hilang (missing values)
- Memahami distribusi dan korelasi antar variabel
- Menentukan transformasi data yang dibutuhkan (scaling, encoding, dll.)

3.7.1 Tahapan EDA Umum

1. **Memahami struktur data:**
    - ◇ Jumlah baris dan kolom
    - ◇ Tipe data setiap kolom
    - ◇ Jumlah nilai unik per fitur
2. **Menyajikan statistik deskriptif:**
    - ◇ Mean, median, modus
    - ◇ Variansi, standar deviasi, rentang
3. **Deteksi nilai yang hilang:**
    - ◇ Jumlah dan persentase missing values
    - ◇ Pola kemunculan nilai hilang
4. **Identifikasi pencilan (outlier):**
    - ◇ Menggunakan IQR atau z-score
5. **Analisis hubungan antar variabel:**
    - ◇ Korelasi numerik
    - ◇ Frekuensi kategorikal
    - ◇ Visualisasi interaksi fitur

3.7.2 Contoh Pertanyaan yang Dijawab dalam EDA

- ◇ Fitur mana yang memiliki sebaran nilai paling bervariasi?
- ◇ Adakah hubungan antara fitur A dan target?
- ◇ Apakah ada fitur dengan dominasi satu nilai saja?
- ◇ Apakah terdapat nilai yang tidak logis atau ekstrem?
- ◇ Apakah distribusi data mendekati normal?

3.8 Visualisasi Data dalam EDA

Visualisasi data adalah bagian penting dalam proses eksplorasi data. Dengan visualisasi, kita dapat lebih mudah mengenali pola, anomali, hubungan antar variabel, serta distribusi data. Setiap jenis plot memiliki kegunaan dan keunggulan masing-masing, tergantung pada jenis data dan tujuan analisis.

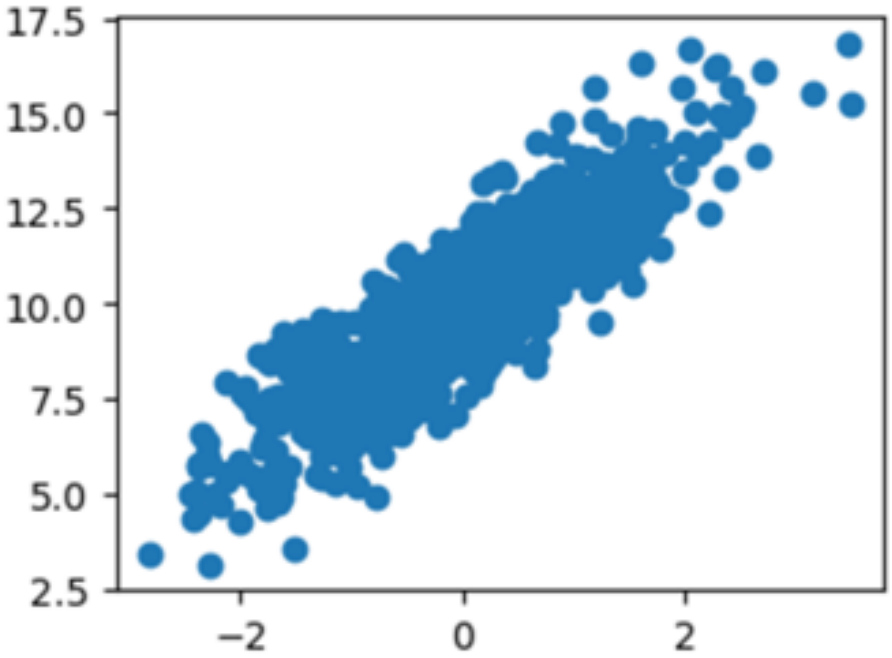
3.8.1 Scatter Plot

Scatter plot digunakan untuk melihat hubungan antara dua variabel numerik. Masing-masing titik mewakili satu observasi dengan koordinat (x, y).

Fungsi:

- Menilai korelasi antara dua fitur numerik
- Mendeteksi pola linier, non-linier, atau kelompok (clustering)
- Menemukan outlier

Contoh:



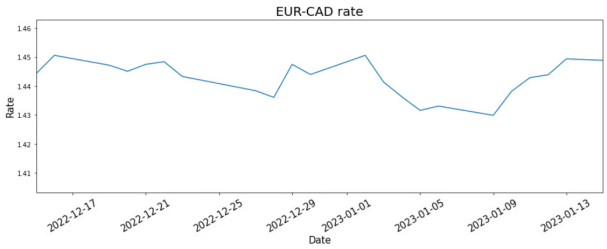
3.8.2 Line Plot

Line plot digunakan untuk menunjukkan perubahan nilai terhadap waktu atau urutan tertentu. Sering digunakan dalam data time series.

Fungsi:

- Menyajikan tren atau pergerakan data seiring waktu
- Mengamati fluktuasi atau siklus musiman

Contoh:



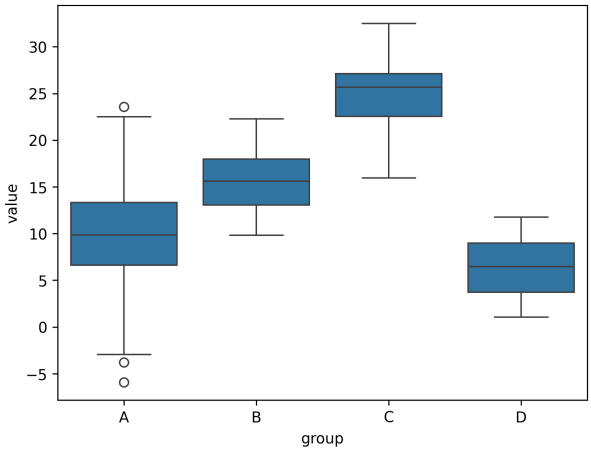
3.8.3 Box Plot

Box plot (juga dikenal sebagai box-and-whisker plot) menunjukkan ringkasan statistik seperti median, kuartil, dan pencilan.

Fungsi:

- Menyajikan penyebaran data dan outlier
- Membandingkan distribusi beberapa kelompok
- Menyederhanakan visualisasi data yang tidak normal

Contoh:



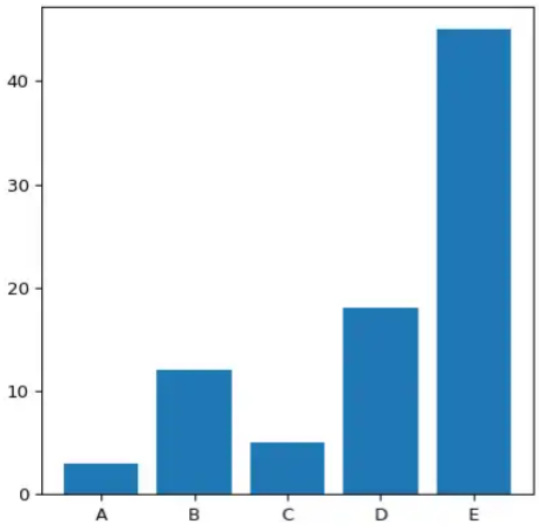
3.8.4 Bar Chart

Bar chart cocok untuk menyajikan data kategorikal (diskrit) dan membandingkan jumlah atau proporsi antar kategori.

Fungsi:

- Visualisasi frekuensi atau proporsi kategori
- Membandingkan performa antar kelas atau kelompok

Contoh:



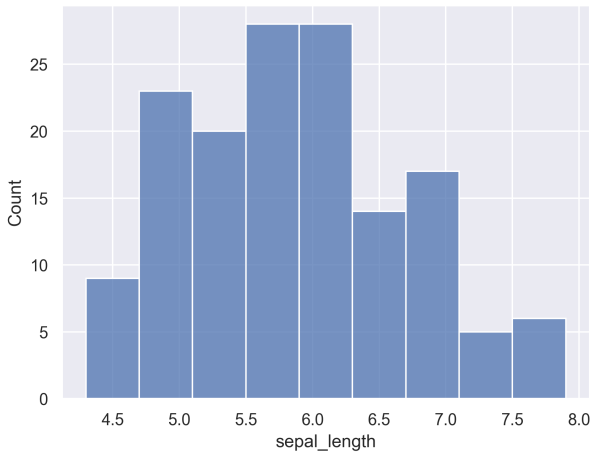
3.8.5 Histogram

Histogram adalah sejenis *barchart* yang menyajikan distribusi frekuensi dari sebuah variabel numerik dengan membaginya menjadi interval (bin).

Fungsi:

- Mengetahui bentuk distribusi data (normal, miring, dll.)
- Mendeteksi skewness, kurtosis, dan pencilan
- Digunakan sebagai dasar untuk scaling atau transformasi

Contoh:



3.8.6 Heatmap Korelasi

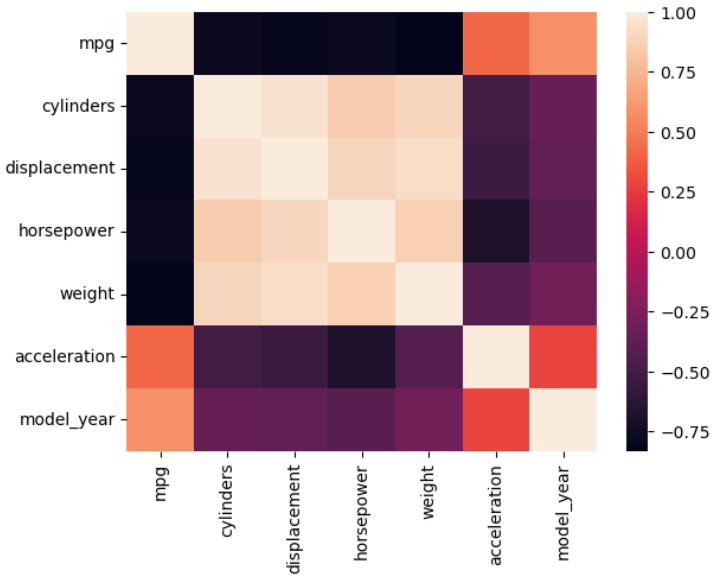
Heatmap korelasi digunakan untuk menampilkan korelasi antara banyak variabel numerik dalam bentuk matriks berwarna.

Fungsi:

- Mengidentifikasi fitur yang berkorelasi tinggi

- Deteksi multikolinearitas sebelum modeling

Contoh:



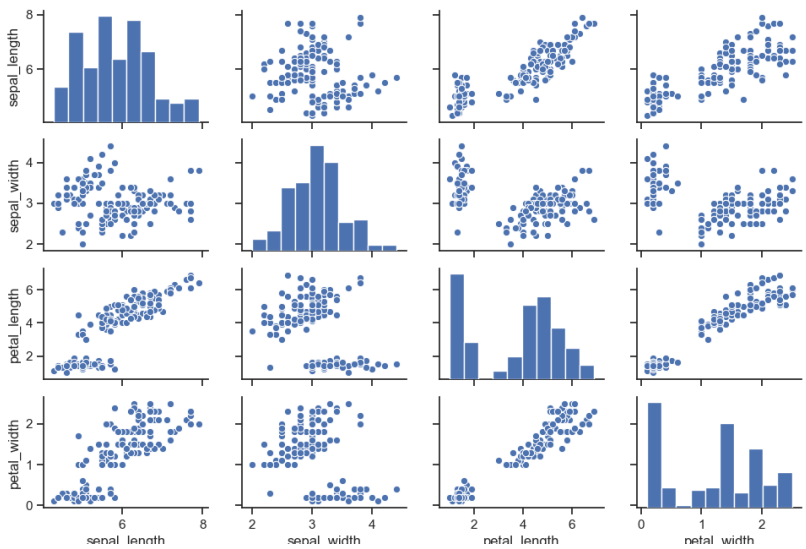
3.8.7 Pairplot (Multiplot)

Pairplot adalah kumpulan scatter plot antar semua pasangan fitur dalam dataset numerik, sering dipakai untuk eksplorasi awal.

Fungsi:

- Memahami interaksi antar fitur sekaligus
- Digunakan untuk dataset kecil sampai sedang

Contoh:



Referensi dan Bahan Bacaan Lanjutan

Untuk memahami peluang, distribusi, dan statistika deskriptif lebih lanjut, pembaca dapat merujuk pada:

- Diez, D. M., Barr, C. D., & Cetinkaya-Rundel, M. (2019). *OpenIntro Statistics*. [DBCR19]
- Illowsky, B., & Dean, S. (2017). *Introductory Statistics*. OpenStax. [ID17]
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning*. Springer. [JWHT13]
- Wasserman, L. (2004). *All of Statistics: A Concise Course in Statistical Inference*. Springer. [Was04]

BAB 4

Pengantar Pemrograman Python

Python adalah bahasa pemrograman yang populer di bidang data science dan AI karena sintaksnya yang mudah dipahami, ekosistem library yang lengkap (seperti NumPy, pandas, Matplotlib), dan dukungan komunitas yang luas. Berikut ini ringkasan dasar-dasar Python yang diperlukan untuk memahami materi selanjutnya.

4.1 Lingkungan Pengembangan Python

Untuk mulai menulis dan menjalankan kode Python, peserta dapat memilih berbagai jenis lingkungan pengembangan yang sesuai dengan kebutuhan dan kenyamanan masing-masing:

- **Instalasi Lokal Python + Editor Teks:** Python dapat diunduh dan diinstal dari situs resmi (https://www.python.org/). Setelah terinstal, peserta dapat menulis kode Python menggunakan editor teks seperti *Visual Studio Code* atau *Sublime Text*, dan menjalankannya melalui terminal atau command prompt.
- **Jupyter Notebook:** Merupakan platform interaktif berbasis web yang sangat populer dalam data science. Jupyter memungkinkan pengguna menulis kode, menampilkan grafik, dan menambahkan catatan atau penjelasan dalam satu dokumen. Dapat diinstal melalui distribusi Anaconda (https://www.anaconda.com/).
- **Google Colab:** Alternatif yang sangat mudah digunakan tanpa perlu instalasi. Google Colab (https://colab.research.google.com/) adalah platform berbasis cloud yang mendukung Jupyter Notebook. Peserta hanya perlu akun Google untuk mulai menulis dan menjalankan kode Python, bahkan dengan dukungan GPU jika diperlukan.

Semua pendekatan di atas mendukung pembelajaran Python untuk keperluan AI dan data science. Bagi pemula, Google Colab menjadi pilihan yang praktis karena langsung bisa digunakan di browser tanpa instalasi apa pun.

4.2 Dasar Sintaks dan Tipe Data

Python memiliki beberapa tipe data dasar yang sering digunakan dalam pemrograman, seperti bilangan bulat (integer), bilangan desimal (float), teks (string), dan nilai logika

(boolean). Setiap variabel di Python tidak perlu dideklarasikan tipenya secara eksplisit, sehingga membuat sintaks menjadi ringkas dan mudah dibaca.

```
# Variabel dan tipe dasar
x = 10            # integer
y = 3.14          # float
nama = "Fafa"     # string
flag = True       # boolean
```

4.3 Struktur Data: List, Tuple, Dictionary

Struktur data adalah cara menyimpan dan mengorganisasi data dalam program. Python menyediakan struktur data built-in yang sangat fleksibel: list untuk data urutan yang bisa diubah, tuple untuk data urutan yang tetap, dan dictionary untuk pasangan kunci-nilai.

```
# List
angka = [1, 2, 3, 4]

# Tuple (immutable)
koordinat = (10.0, 20.0)

# Dictionary (key-value)
orang = {"nama": "Fafa", "umur": 17}
```

4.4 Kontrol Alur: Pengkondisian dan Perulangan

Kontrol alur memungkinkan kita menentukan bagaimana program berjalan berdasarkan kondisi tertentu (percabangan) atau mengulang blok kode tertentu (perulangan). Python menggunakan kata kunci seperti `if`, `else`,`for`, dan `while` untuk mengatur logika ini.

```
# If-else
if x % 2 == 0:
    print("Genap")
else:
    print("Ganjil")

# For loop
for v in angka:
    print(v * 2)

# While loop
i = 0
while i < 3:
    print(i)
    i += 1
```

4.5 Fungsi dan Input/Ouput Sederhana

Fungsi adalah blok kode yang bisa digunakan kembali untuk menjalankan tugas tertentu. Python mendukung pembuatan fungsi dengan kata kunci `def`. Fungsi dapat menerima parameter dan mengembalikan hasil. Untuk mencetak keluaran, digunakan fungsi `print()`.

```
# Definisi fungsi
def kuadrat(a):
    return a * a

print(kuadrat(5))  # Output: 25
```

4.6 Fungsi Lanjutan dan Teknik Python Efisien

Setelah memahami dasar-dasar fungsi, tipe data, dan kontrol alur dalam Python, ada beberapa fitur lanjutan yang sangat berguna dalam praktik eksplorasi dan pemrosesan data. Subbab ini memperkenalkan teknik-teknik Python yang efisien dan umum digunakan dalam tugas-tugas machine learning dan analisis data.

List dan Dictionary Comprehension

Comprehension adalah cara singkat dan ekspresif untuk membuat list atau dictionary baru dari iterable yang ada.

```
# List comprehension: kuadrat dari bilangan genap 0..9
squares = [x**2 for x in range(10) if x % 2 == 0]

# Dictionary comprehension: peta angka dan kuadratnya
square_map = {x: x**2 for x in range(5)}
```

Lambda Function (Fungsi Tak Bernama)

Fungsi lambda digunakan untuk membuat fungsi sederhana dalam satu baris.

```
# Fungsi lambda untuk menghitung kuadrat
square = lambda x: x**2

# Digunakan dengan map()
numbers = [1, 2, 3, 4]
squared = list(map(lambda x: x**2, numbers))
```

Fungsi Sebagai Objek dan Nested Function

Dalam Python, fungsi dapat disimpan dalam variabel, dikirim sebagai argumen, atau dikembalikan sebagai hasil.

```
def multiplier(factor):
```

```
    def multiply(x):
        return x * factor
    return multiply

double = multiplier(2)
print(double(5))  # Output: 10
```

*args dan **kwargs

Kadang kita ingin fungsi menerima jumlah argumen yang fleksibel.

```
def total(*args):
    return sum(args)

def describe_person(name, **kwargs):
    print(f"Name: {name}")
    for key, value in kwargs.items():
        print(f"{key}: {value}")

describe_person("Alya", age=17, hobby="chess")
```

Enumerate dan Zip

`enumerate()` memberikan indeks saat iterasi, dan `zip()` menggabungkan dua list secara berpasangan.

```
names = ['Ali', 'Budi', 'Cici']
scores = [85, 90, 78]

for i, name in enumerate(names):
    print(i, name)

for name, score in zip(names, scores):
    print(f"{name} scored {score}")
```

Latihan Mini

- ◇ Buatlah list comprehension untuk menghasilkan daftar bilangan kubik dari 1 sampai 10.
- ◇ Gunakan `zip()` untuk menggabungkan dua list dan menghasilkan dictionary.
- ◇ Buat fungsi `make_power(n)` yang menghasilkan fungsi untuk mempangkatkan bilangan dengan n.

Fitur-fitur ini membuat kode Python lebih ringkas, ekspresif, dan efisien — suatu hal yang penting ketika menyelesaikan soal dengan batasan waktu seperti dalam IOAI.

4.7 Pengenalan NumPy

NumPy (Numerical Python) adalah library penting yang digunakan untuk komputasi numerik dan manipulasi array. NumPy memperkenalkan struktur array multidimensi yang efisien dan fungsi matematis tingkat lanjut untuk operasi linier, statistik, dan lainnya.

Array NumPy mirip dengan list di Python, tetapi memiliki kemampuan matematis yang jauh lebih efisien dan optimal. Berikut adalah beberapa contoh operasi dasar yang sering digunakan dalam analisis data:

```
import numpy as np

# Membuat array dari list
arr = np.array([1, 2, 3])
print(arr * 2)       # [2 4 6]
print(arr.mean())    # Rata-rata: 2.0
```

NumPy juga menyediakan berbagai fungsi untuk membuat array khusus dan melakukan manipulasi data:

```
# Membuat array berisi nol, satuan, atau bilangan acak
zeros = np.zeros((2, 3))        # array 2x3 berisi nol
ones = np.ones((2, 2))          # array 2x2 berisi satu
randoms = np.random.rand(3, 3)  # array 3x3 bilangan acak [0, 1)

# Membuat array dengan range tertentu
arr_range = np.arange(0, 10, 2)   # [0 2 4 6 8]
linspace = np.linspace(0, 1, 5)   # [0.   0.25 0.5  0.75 1.  ]

# Operasi matriks
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])

print(A + B)            # Penjumlahan elemen-wise
print(A @ B)            # Perkalian matriks
print(A.T)              # Transpos
print(np.linalg.inv(A)) # Invers matriks
```

Dengan fungsi-fungsi ini, kita dapat dengan cepat melakukan komputasi yang kompleks dan efisien, yang sangat berguna dalam proses machine learning dan data science.

Vectorization dan Broadcasting di NumPy

Salah satu kekuatan utama `NumPy` dibandingkan Python murni adalah kemampuan untuk melakukan operasi vektor dan matriks secara langsung (vectorized operation) tanpa menggunakan perulangan eksplisit (`for` loop).

Vectorization

Vectorization berarti menulis operasi dalam bentuk ekspresi aljabar, sehingga NumPy dapat memprosesnya lebih efisien menggunakan optimasi internal dan SIMD (*Single Instruction, Multiple Data*).

```
import numpy as np

x = np.array([1, 2, 3, 4])
y = np.array([10, 20, 30, 40])

# Tanpa loop
z = x + y   # [11, 22, 33, 44]
```

Dibandingkan dengan Python biasa:

```
z = []
for i in range(len(x)):
    z.append(x[i] + y[i])
```

Selain penulisan lebih ringkas, operasi vectorized juga jauh lebih cepat secara runtime.

Broadcasting

Broadcasting adalah mekanisme di mana NumPy secara otomatis menyesuaikan ukuran array yang berbeda agar operasi antar array tetap bisa dilakukan.

Contoh:

```
x = np.array([1, 2, 3])
b = 10

# b akan "diperluas" menjadi array [10, 10, 10] secara implisit
z = x + b   # [11, 12, 13]
```

Broadcasting dua array:

```
A = np.array([[1], [2], [3]])   # shape (3,1)
B = np.array([10, 20, 30])      # shape (3,)

# A akan di-broadcast ke shape (3,3)
# B akan di-broadcast ke shape (3,3)
result = A + B

# Output:
# [[11 21 31]
#  [12 22 32]
#  [13 23 33]]
```

Prinsip Broadcasting (aturan):

1. Jika dua array memiliki jumlah dimensi berbeda, array dengan lebih sedikit dimensi akan ditambahkan dimensi 1 di sebelah kiri.

2. Kemudian, dimensi dibandingkan dari belakang:

   - ◇ Jika ukurannya sama → valid
   - ◇ Jika salah satunya adalah 1 → akan diulang
   - ◇ Jika berbeda dan bukan 1 → error

Latihan Singkat

- ◇ Buatlah dua array acak ukuran (1000,) dan hitung hasil penjumlahan elemen demi elemen menggunakan:
  - – `for` loop biasa
  - – operasi vectorized
  - – Bandingkan waktu eksekusinya
- ◇ Gunakan `np.arange` untuk membuat array 2D bentuk (5, 1), lalu tambahkan dengan array (3,) dan amati hasil broadcasting-nya.

Dengan menggunakan vectorization dan broadcasting, kita dapat menulis kode yang:

- ◇ Lebih ringkas dan mudah dibaca
- ◇ Jauh lebih cepat (hingga ratusan kali lipat pada array besar)
- ◇ Lebih mirip notasi matematika aljabar linear

4.8 Pengenalan Matplotlib

`Matplotlib` adalah library visualisasi di Python yang memungkinkan kita membuat grafik dan plot data dengan mudah. Library ini sangat berguna untuk menganalisis data secara visual, melihat pola, distribusi, dan hubungan antar variabel.

Dalam praktik AI dan data science, visualisasi digunakan untuk: - Melihat distribusi data - Mengetahui hubungan antar variabel - Memantau proses pelatihan model

Berikut adalah contoh alur lengkap: mulai dari membuat data dengan NumPy, memanipulasinya, lalu memvisualisasikannya dengan Matplotlib.

Contoh: Membuat dan Menampilkan Grafik Data

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

# 1. Buat data menggunakan NumPy
x = np.linspace(0, 10, 100)             # 100 angka dari 0 sampai 10
y = np.sin(x)                           # nilai y = sin(x)
```

```
noise = np.random.normal(0, 0.1, size=x.shape)  # noise acak
y_noisy = y + noise                      # y yang diberi gangguan

# 2. Plot data
plt.figure(figsize=(8, 4))
plt.plot(x, y, label='Asli: sin(x)', linewidth=2)
plt.plot(x, y_noisy, label='Dengan noise', linestyle='--')
plt.title("Plot Fungsi sin(x) dan Versi Bising")
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.grid(True)
plt.show()
```

Plot ini menunjukkan bagaimana fungsi `sin(x)` terlihat ideal, sedangkan data hasil pengamatan di dunia nyata sering kali mengandung *noise* atau gangguan. Visualisasi seperti ini sangat penting untuk memahami sifat data sebelum digunakan dalam model machine learning.

Contoh Visualisasi Lain: Histogram dan Scatter Plot

```
# Histogram data acak
data = np.random.normal(loc=0, scale=1, size=1000)
plt.hist(data, bins=30, color='skyblue')
plt.title("Histogram Data Normal")
plt.xlabel("Nilai")
plt.ylabel("Frekuensi")
plt.show()

# Scatter plot
x = np.random.rand(50)
y = 2 * x + 0.5 + np.random.normal(0, 0.1, size=50)
plt.scatter(x, y)
plt.title("Contoh Scatter Plot")
plt.xlabel("x")
plt.ylabel("y")
plt.show()
```

Visualisasi di atas menggambarkan: - **Histogram** untuk melihat distribusi nilai dalam data (misalnya distribusi normal). - **Scatter plot** untuk melihat hubungan antara dua variabel.

Kesimpulan

Dengan kombinasi NumPy dan Matplotlib, kita bisa: - Menghasilkan data numerik - Melakukan manipulasi sederhana - Menyajikan grafik informatif untuk memahami pola dan karakteristik data

Kemampuan ini sangat penting sebelum membangun atau melatih model AI.

Referensi dan Bahan Bacaan Lanjutan

Berikut adalah beberapa sumber online yang cocok untuk belajar Python secara lebih mendalam:

- *Learn Python* di Tutorialspoint — panduan sintaks dan dasar bahasa secara lengkap
- *Automate the Boring Stuff with Python* (Al Sweigart) — buku online gratis untuk pemula automatetheboringstuff.com
- *Python for Data Science Handbook* (Jake VanderPlas) — referensi penggunaan NumPy, pandas, dan Matplotlib jakevdp.github.io
- Dokumentasi resmi Python — docs.python.org
- **Tutorial NumPy Resmi** `https://numpy.org/learn/` Panduan belajar dan dokumentasi NumPy langsung dari situs resminya.
- **Matplotlib Pyplot Tutorial** `https://matplotlib.org/stable/tutorials/introductory/pyplot.html` Panduan resmi untuk mempelajari fungsi-fungsi plotting dasar.
- **Matplotlib Gallery (Galeri Contoh Plot)** `https://matplotlib.org/stable/gallery/index.html` Kumpulan contoh visualisasi menarik dengan kode sumber lengkap.

Bibliografi

[DBCR19] David M Diez, Christopher D Barr, and Mine Cetinkaya-Rundel, *Openintro statistics*, 3rd ed., OpenIntro, 2019.

[Dev23] Google Developers, *Machine learning crash course*, https://developers.google.com/machine-learning/crash-course, 2023.

[DFO20] Marc Peter Deisenroth, A Aldo Faisal, and Cheng Soon Ong, *Mathematics for machine learning*, Cambridge University Press, 2020.

[ID17] Barbara Illowsky and Susan Dean, *Introductory statistics*, OpenStax, 2017.

[JWHT13] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, *An introduction to statistical learning*, Springer, 2013.

[Uni23] Duke University, *Data science math skills*, https://www.coursera.org/learn/datasciencemathskills, 2023.

[Was04] Larry Wasserman, *All of statistics: A concise course in statistical inference*, Springer, 2004.