Bab 06 — Fondasi Machine Learning

Status: Draft lengkap v0.3 — struktur subbab tanpa penanda unit baca Bagian buku: Level D — Cara Model Belajar dari Data Target pembaca: Pembaca yang sudah memahami vektor, probabilitas, statistik, loss, dan gradient dasar; sekarang siap melihat bagaimana konsep itu menjadi workflow machine learning yang jujur.

Cara membaca bab ini

Mulai Bab 6, pembahasan mengikuti subbab agar setiap topik bisa dibahas sepanjang yang dibutuhkan: ada subbab yang pendek karena konsepnya sederhana, ada yang panjang karena perlu narasi, detail teknis, contoh, dan peringatan.

Janji Bab 6: pembaca memahami bukan hanya “model dilatih dengan data”, tetapi seluruh kebiasaan profesional di balik machine learning: framing, dataset, fitur, label, split, baseline, metrik, leakage, bias-variance, preprocessing, error analysis, reproducibility, dan etika.

Motivasi:
Machine learning yang bagus bukan sekadar algoritma pintar. Ia adalah proses berpikir yang jujur: data apa yang dipakai, jawaban apa yang dipelajari, bagaimana diuji, dan siapa yang terdampak.

Pendalaman awal — sejarah dan formulasi matematika machine learning

Machine learning modern tumbuh dari beberapa jalur: statistik, pattern recognition, teori informasi, optimisasi, dan kecerdasan buatan simbolik. Pada awalnya, banyak sistem “cerdas” dibuat dengan aturan manual. Namun aturan manual sulit merawat variasi dunia nyata. Statistik menawarkan jalan lain: belajar pola dari data. Pattern recognition menambahkan pertanyaan praktis: bagaimana komputer mengenali angka, wajah, suara, atau transaksi mencurigakan? Optimisasi memberi mesin untuk menyesuaikan parameter.

Definisi klasik Tom Mitchell sering diringkas seperti ini: program belajar dari pengalaman E terhadap tugas T dan ukuran performa P jika performanya pada T, diukur oleh P, meningkat dengan pengalaman E. Dalam bahasa proyek ML:

```
E = dataset / pengalaman historis
T = tugas prediksi / keputusan
P = metrik evaluasi
```

Formulasi matematis yang sering muncul adalah empirical risk minimization:

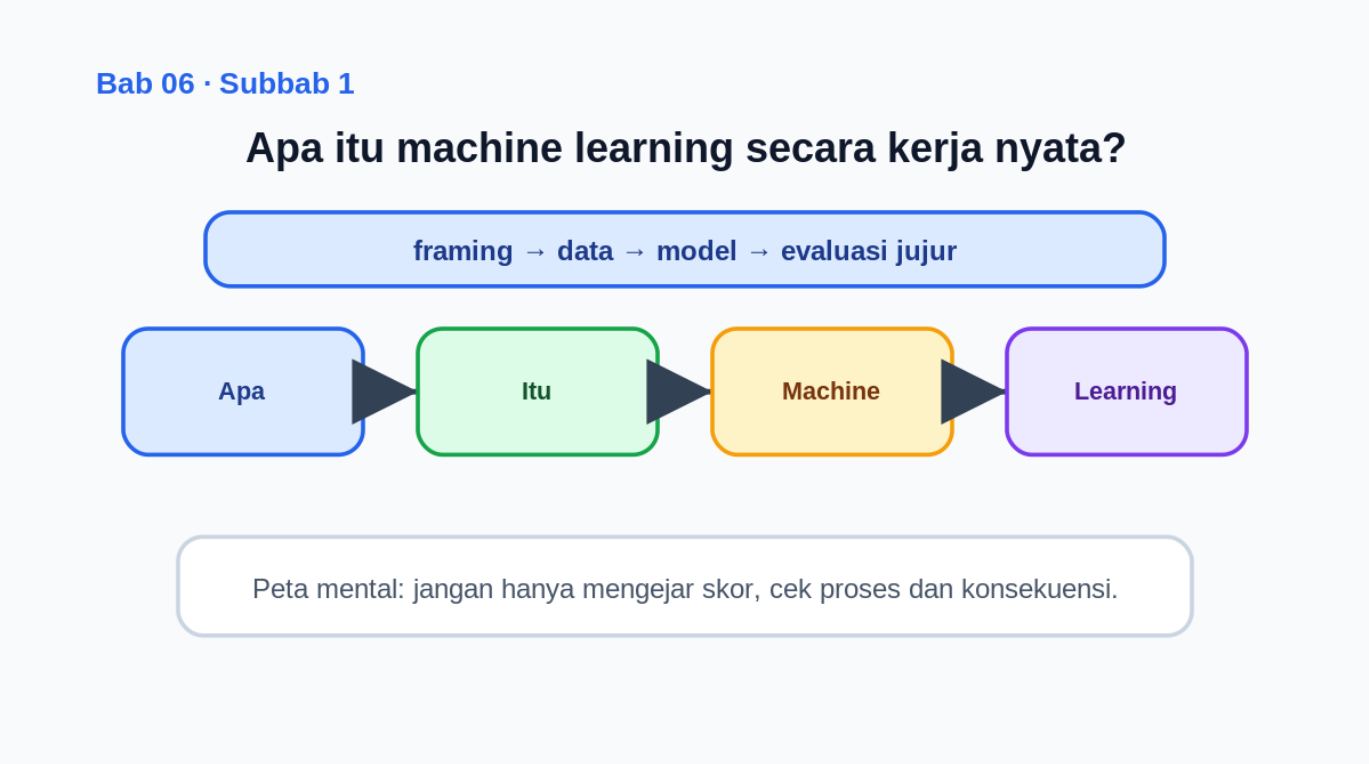
$$R_{emp}(f) = \frac{1}{n} \sum L(f(x_i), y_i)$$
$$f^* = \operatorname{argmin}_f R_{emp}(f)$$

Artinya, kita memilih model f yang meminimalkan rata-rata loss pada data latih. Tetapi tujuan sebenarnya bukan hanya loss train kecil. Tujuan sebenarnya adalah generalisasi: performa baik pada data baru yang belum dilihat.

```
Generalization gap = error_validasi - error_train
```

Jika gap besar, model mungkin overfit. Jika keduanya buruk, model mungkin underfit. Bab 6 adalah fondasi agar pembaca tidak hanya menjalankan algoritma, tetapi memahami apa yang sedang dioptimalkan dan bagaimana menilai kejujurannya.

Subbab 1 — Apa itu machine learning secara kerja nyata?



Apa itu machine learning secara kerja nyata?

Machine learning adalah cara membuat sistem belajar pola dari data sehingga ia bisa membuat prediksi, rekomendasi, atau keputusan pada contoh baru. Kalimat ini sering terdengar sederhana, tetapi di baliknya ada disiplin yang ketat: menentukan masalah, memilih data, membangun fitur, memilih target, membagi data dengan jujur, membuat baseline, mengevaluasi, lalu mengulang dengan hati-hati.

Bayangkan sebuah bimbingan belajar di Indonesia ingin mengetahui siswa mana yang butuh pendampingan tambahan. Guru punya data kehadiran, jumlah tugas selesai, nilai kuis, dan jam belajar. Sistem ML tidak “memahami” siswa seperti guru, tetapi ia dapat belajar pola historis: kombinasi tertentu sering muncul pada siswa yang kemudian butuh bantuan. Dari pola ini, model memberi prediksi awal. Prediksi itu bukan vonis; ia sinyal untuk membantu guru memprioritaskan perhatian.

Perbedaan ML dengan aturan manual ada pada sumber aturan. Aturan manual ditulis langsung manusia, misalnya “jika nilai kuis < 60, beri bantuan”. ML mencari pola dari data, misalnya nilai kuis rendah ditambah absensi tinggi dan tugas tidak lengkap lebih kuat daripada nilai rendah saja. Kelebihannya fleksibel; risikonya pola bisa salah, bias, atau bocor. Karena itu, Bab 6 tidak hanya mengajarkan model, tetapi juga kejujuran evaluasi.

Ide teknis / latihan ketik kecil

```
Aturan manual: manusia menulis aturan.
Machine learning: data membantu menemukan aturan.
```

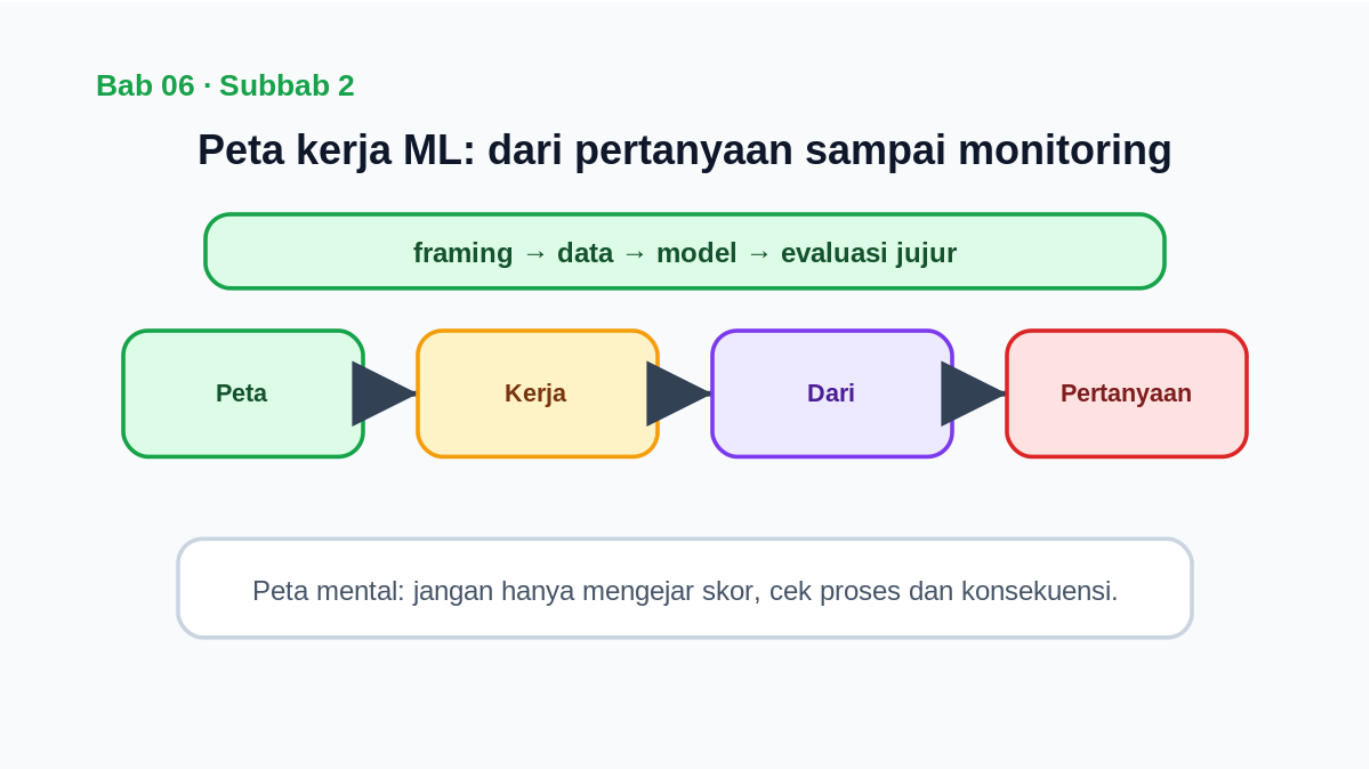
Tes cepat subbab 1

1. Apa perbedaan aturan manual dan machine learning?

2. Mengapa prediksi ML tidak boleh dianggap vonis mutlak?

3. Buat contoh masalah lokal yang cocok dibantu ML tetapi tetap perlu manusia.

Subbab 2 — Peta kerja ML: dari pertanyaan sampai monitoring



Peta kerja ML: dari pertanyaan sampai monitoring

Proyek ML yang sehat tidak dimulai dari memilih algoritma. Ia dimulai dari pertanyaan: keputusan apa yang ingin dibantu? siapa pengguna keputusan itu? data apa yang tersedia? apa konsekuensi jika model salah? Setelah itu baru kita bicara dataset, fitur, label, baseline, training, evaluasi, dan deployment.

Workflow umum: problem framing → data collection → data understanding → split data → baseline → model training → validation → error analysis → test final → dokumentasi → monitoring. Urutan ini penting karena banyak kegagalan ML terjadi sebelum model dilatih. Misalnya target tidak jelas, data tidak mewakili pengguna, atau evaluasi bocor.

Monitoring juga bagian dari fondasi. Model yang bagus hari ini bisa turun kualitas bulan depan karena perilaku pengguna berubah, kurikulum sekolah berubah, harga bahan naik, atau musim liburan memengaruhi pola. ML bukan satu kali sihir; ML adalah sistem yang perlu dirawat.

Ide teknis / latihan ketik kecil

```
Pertanyaan → data → baseline → model → evaluasi → analisis error → monitoring
```

Tes cepat subbab 2

1. Mengapa proyek ML tidak boleh langsung dimulai dari algoritma?

2. Apa yang bisa berubah setelah model dirilis?

3. Tuliskan workflow kecil untuk kasus prediksi stok warung.

Subbab 3 — Problem framing: mengubah niat kabur menjadi target teknis

Problem framing: mengubah niat kabur menjadi target teknis

Problem framing adalah seni mengubah keinginan umum menjadi tugas ML yang jelas. “Buat AI untuk pendidikan” terlalu luas. “Prediksi siswa yang butuh pendampingan tambahan minggu depan berdasarkan data 4 minggu terakhir” jauh lebih jelas. Di sini kita tahu unit prediksi, waktu prediksi, fitur yang boleh dipakai, dan aksi setelah prediksi.

Framing harus menjawab minimal lima hal: apa inputnya, apa outputnya, kapan prediksi dibuat, siapa yang memakai hasilnya, dan kesalahan mana yang lebih mahal. Dalam contoh siswa, false negative berarti siswa yang butuh bantuan terlewat. False positive berarti siswa yang sebenarnya aman tetap ditandai. Keduanya punya biaya berbeda.

Framing juga menentukan jenis tugas: klasifikasi jika output kategori, regresi jika output angka, ranking jika output urutan prioritas, clustering jika ingin menemukan kelompok tanpa label, dan reinforcement learning jika keputusan berurutan dengan reward. Bab ini fokus fondasi supervised ML, karena itu pintu utama ke bab berikutnya.

Ide teknis / latihan ketik kecil

```
Framing baik = input jelas + target jelas + waktu jelas + aksi jelas + risiko jelas
```

Contoh framing terstruktur

Ide kabur: “buat AI untuk membantu guru.”

Ubah menjadi:

```
Unit prediksi: satu siswa per minggu
Input: kehadiran, tugas, nilai kuis, konsultasi 4 minggu terakhir
Target: butuh pendampingan tambahan minggu depan (0/1)
Waktu prediksi: Jumat sore sebelum jadwal minggu depan dibuat
Aksi: guru memilih siswa prioritas untuk sesi tambahan
Metrik utama: recall untuk siswa yang benar-benar butuh bantuan
Risiko: label tidak boleh menjadi stigma permanen
```

Dengan framing ini, model punya batas. Kita tahu fitur apa yang boleh dipakai, label apa yang diprediksi, dan apa konsekuensi salah.

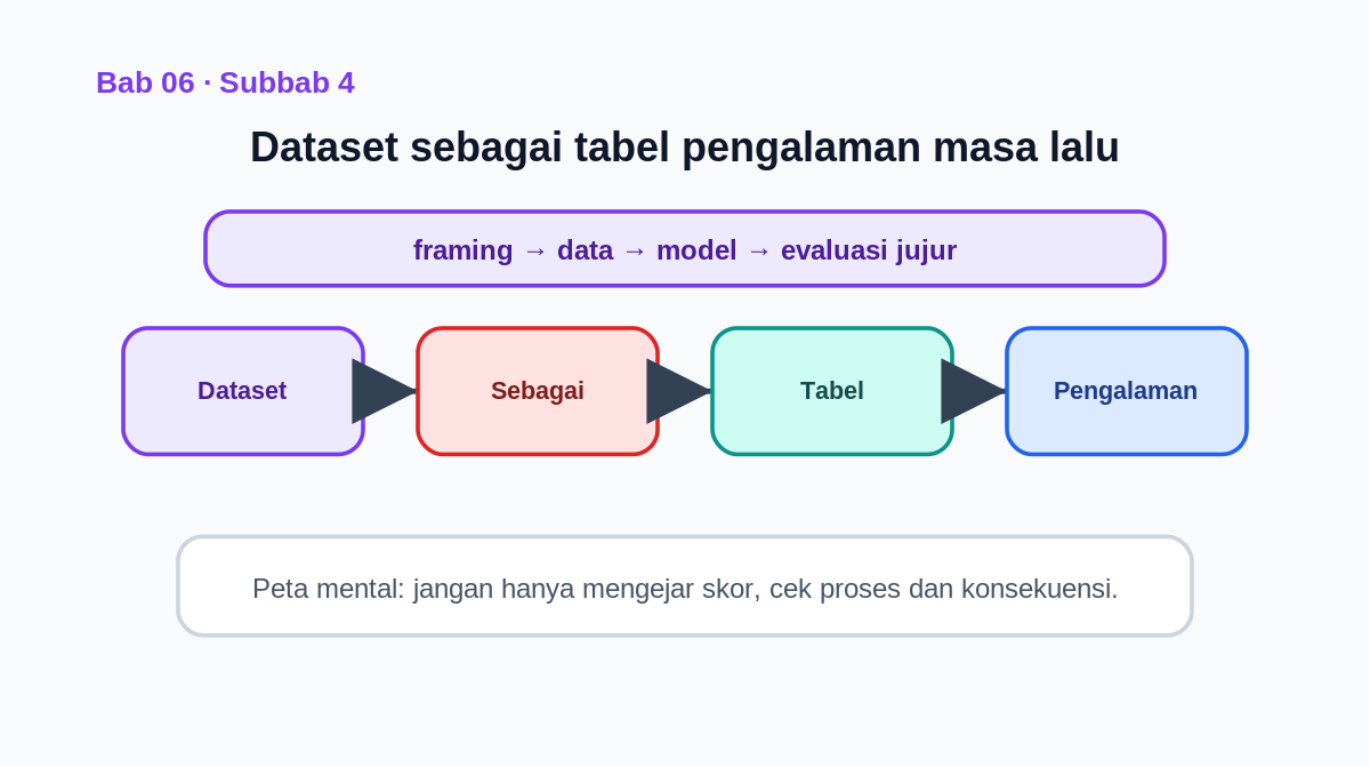
Tes cepat subbab 3

1. Apa lima pertanyaan minimal dalam problem framing?

2. Mengapa “buat AI untuk pendidikan” terlalu kabur?

3. Ubah ide “AI untuk UMKM” menjadi target teknis yang jelas.

Subbab 4 — Dataset sebagai tabel pengalaman masa lalu



Dataset sebagai tabel pengalaman masa lalu

Dataset adalah kumpulan contoh. Dalam data tabular, setiap baris biasanya mewakili satu contoh: satu siswa, satu transaksi, satu produk, satu hari penjualan, atau satu perjalanan. Setiap kolom menyimpan informasi tentang contoh itu. Sebagian kolom menjadi fitur, satu atau beberapa kolom menjadi label/target.

Dataset bukan kebenaran sempurna. Ia adalah jejak pengalaman masa lalu yang dipilih, dicatat, dibersihkan, dan diberi label oleh proses tertentu. Jika proses pencatatan buruk, model belajar dari catatan buruk. Jika hanya kelompok tertentu yang masuk dataset, model cenderung melayani kelompok itu lebih baik.

Sebelum modeling, kita perlu membaca dataset seperti membaca laporan lapangan. Apa arti setiap kolom? satuannya apa? kapan dicatat? apakah ada nilai hilang? apakah ada outlier? apakah label dibuat sebelum atau setelah keputusan? Pertanyaan ini terasa lambat, tetapi menghemat banyak kegagalan.

Ide teknis / latihan ketik kecil

```
Baris = contoh
Kolom = informasi
Dataset = catatan pengalaman, bukan dunia lengkap
```

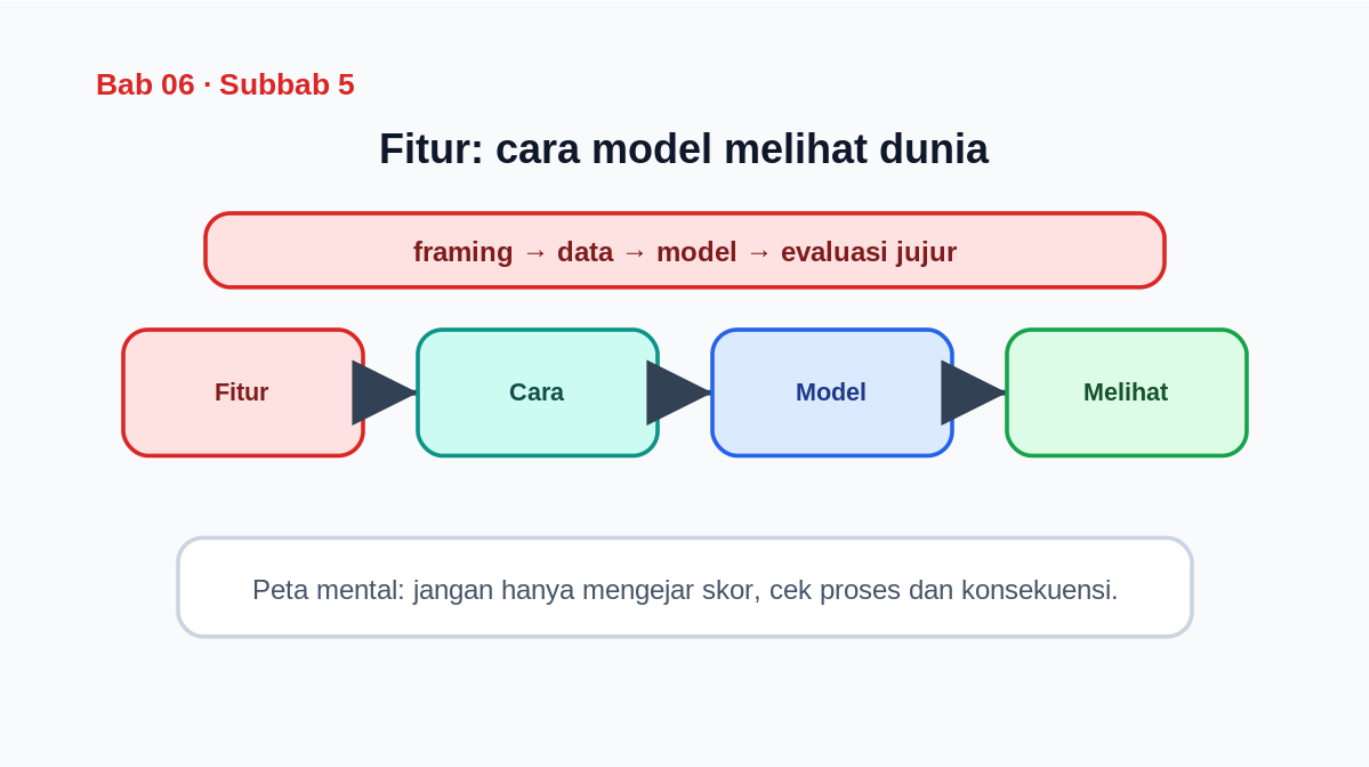
Tes cepat subbab 4

1. Apa arti baris dan kolom dalam dataset tabular?

2. Mengapa dataset tidak boleh dianggap kebenaran sempurna?

3. Sebutkan lima pertanyaan sebelum modeling dataset.

Subbab 5 — Fitur: cara model melihat dunia



Fitur: cara model melihat dunia

Fitur adalah informasi yang diberikan kepada model. Untuk siswa, fitur bisa berupa kehadiran, tugas selesai, jam belajar, nilai kuis, atau jumlah konsultasi. Untuk warung, fitur bisa berupa hari, cuaca, event lokal, stok awal, dan harga. Fitur adalah jendela model; jika jendelanya sempit atau buram, prediksi ikut terbatas.

Fitur harus tersedia pada waktu prediksi. Ini aturan emas. Jika kita ingin memprediksi siswa butuh bantuan minggu depan, kita tidak boleh memakai nilai ujian akhir bulan depan sebagai fitur. Itu bocor dari masa depan.

Fitur juga perlu bermakna. Menambahkan banyak kolom tidak otomatis membuat model lebih baik. Kolom yang noisy, bias, atau tidak stabil bisa merusak. Feature engineering adalah proses membuat representasi yang membantu model melihat pola lebih jelas tanpa membocorkan jawaban.

Ide teknis / latihan ketik kecil

```
siswa = {"kehadiran": 0.75, "tugas_selesai": 0.60, "nilai_kuis": 58}
fitur = [siswa["kehadiran"], siswa["tugas_selesai"], siswa["nilai_kuis"]]
print(fitur)
```

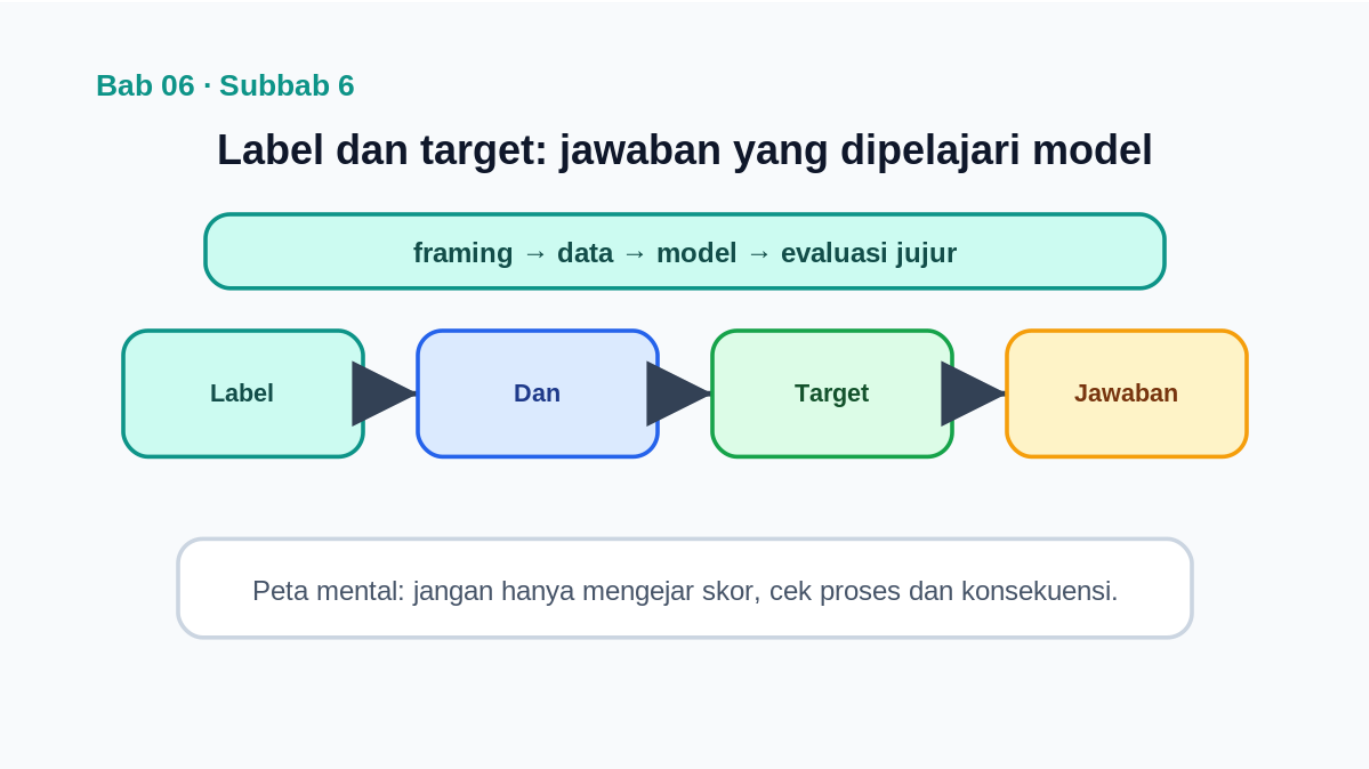
Tes cepat subbab 5

1. Apa itu fitur dalam ML?

2. Mengapa fitur harus tersedia pada waktu prediksi?

3. Buat tiga fitur untuk memprediksi penjualan es teh besok.

Subbab 6 — Label dan target: jawaban yang dipelajari model



Label dan target: jawaban yang dipelajari model

Label adalah jawaban historis yang ingin dipelajari model. Jika tugasnya klasifikasi siswa butuh bantuan, label bisa 1 untuk “butuh bantuan” dan 0 untuk “tidak”. Jika tugasnya regresi stok, target bisa jumlah penjualan besok.

Kualitas label sangat penting. Label bisa salah karena input manusia keliru, definisi tidak konsisten, atau proses masa lalu bias. Misalnya label “siswa bermasalah” mungkin dipengaruhi stigma, bukan kebutuhan akademik. Lebih aman memakai label yang lebih operasional: “butuh pendampingan tambahan berdasarkan rubrik X”.

Label juga harus selaras dengan aksi. Jika aksi kita adalah mengirim tutor, label sebaiknya mengukur kebutuhan tutor, bukan sekadar nilai rendah. Nilai rendah bisa terjadi karena banyak alasan; kebutuhan intervensi adalah target yang lebih dekat dengan keputusan.

Ide teknis / latihan ketik kecil

```
Fitur = informasi untuk menebak
Label/target = jawaban historis yang ingin dipelajari
```

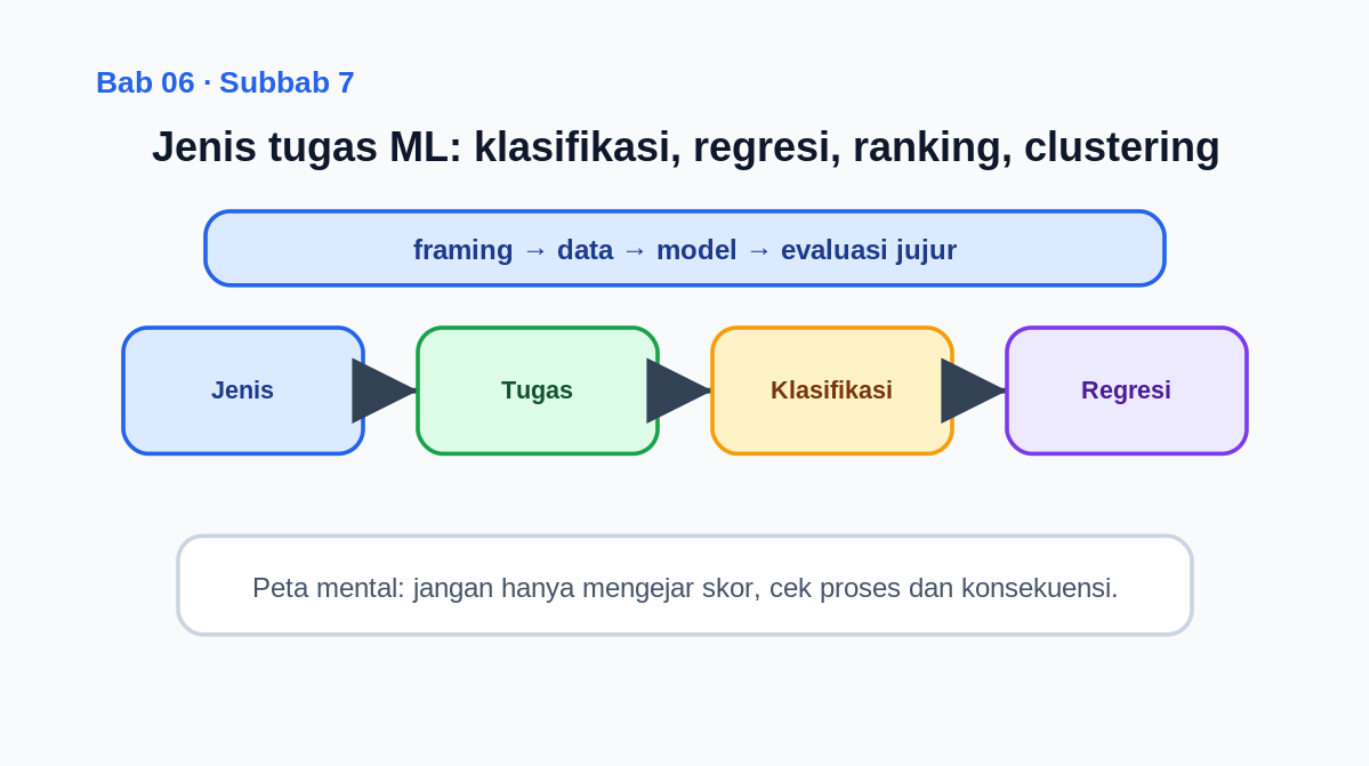
Tes cepat subbab 6

1. Apa bedanya fitur dan label?

2. Mengapa label “siswa bermasalah” berisiko?

3. Buat label yang lebih operasional untuk kasus pendidikan.

Subbab 7 — Jenis tugas ML: klasifikasi, regresi, ranking, clustering



Jenis tugas ML: klasifikasi, regresi, ranking, clustering

Klasifikasi memprediksi kategori: spam/tidak, butuh bantuan/tidak, risiko tinggi/rendah. Regresi memprediksi angka: jumlah penjualan, harga rumah, waktu tunggu. Ranking mengurutkan item: produk mana ditampilkan dulu. Clustering menemukan kelompok tanpa label: segmentasi pelanggan berdasarkan pola belanja.

Kesalahan memilih jenis tugas membuat evaluasi kacau. Jika target sebenarnya angka tetapi dipaksa menjadi kategori, informasi hilang. Jika yang dibutuhkan urutan prioritas tetapi kita hanya membuat klasifikasi ya/tidak, sistem mungkin tidak membantu pengambilan keputusan terbatas.

Dalam buku ini, Bab 7 akan mendalami supervised learning: klasifikasi dan regresi. Bab 8 masuk representation/unsupervised learning. Bab 12 membahas reinforcement learning. Bab 6 memberi peta agar pembaca tahu posisi setiap teknik.

Ide teknis / latihan ketik kecil

```
Klasifikasi: kategori
Regresi: angka
Ranking: urutan
Clustering: kelompok tanpa label
```

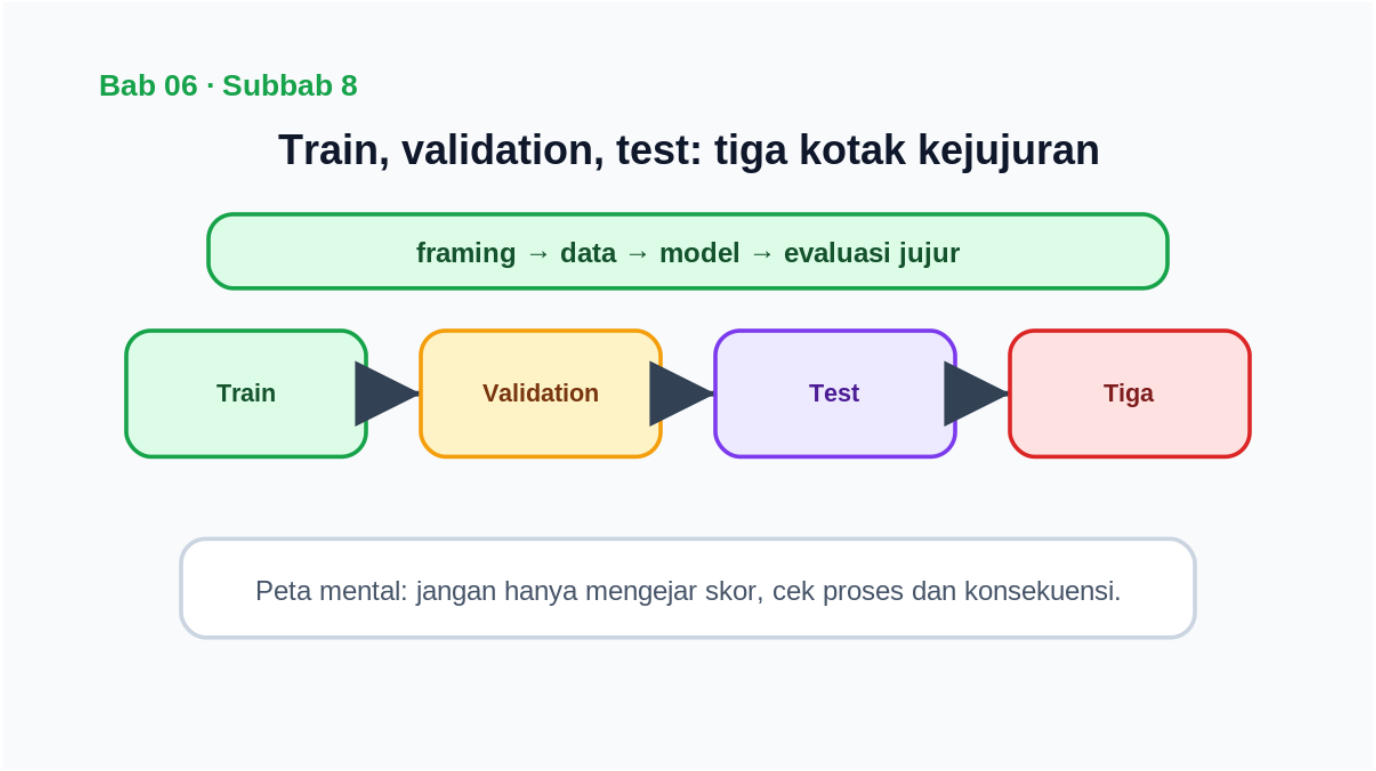
Tes cepat subbab 7

1. Apa perbedaan klasifikasi dan regresi?

2. Kapan ranking lebih tepat daripada klasifikasi?

3. Klasifikasikan tiga masalah lokal ke jenis tugas ML.

Subbab 8 — Train, validation, test: tiga kotak kejujuran



Train, validation, test: tiga kotak kejujuran

Model harus diuji pada data yang belum ia lihat. Karena itu dataset biasanya dibagi menjadi train, validation, dan test. Train dipakai untuk belajar parameter. Validation dipakai untuk memilih model/hyperparameter. Test disimpan sampai akhir untuk estimasi performa yang lebih jujur.

Jika kita melatih dan menilai pada data yang sama, model bisa terlihat hebat karena hafal. Seperti siswa yang ujian dengan soal latihan yang sama persis. Skor tinggi tidak membuktikan pemahaman pada soal baru.

Test set harus dijaga seperti amplop ujian. Jangan berkali-kali mengintip test lalu menyesuaikan model, karena itu membuat test ikut menjadi validation terselubung. Untuk proyek kecil, split sederhana cukup. Untuk proyek serius, validasi silang, time split, atau group split mungkin diperlukan.

Ide teknis / latihan ketik kecil

```
data = list(range(10))
train = data[:6]
valid = data[6:8]
test = data[8:]
print(train, valid, test)
```

Contoh split dengan angka

Jika ada 1000 data:

```
train 60% = 600 data
validation 20% = 200 data
test 20% = 200 data
```

Alur yang benar:

1. Latih model pada train.

2. Pilih threshold/hyperparameter pada validation.

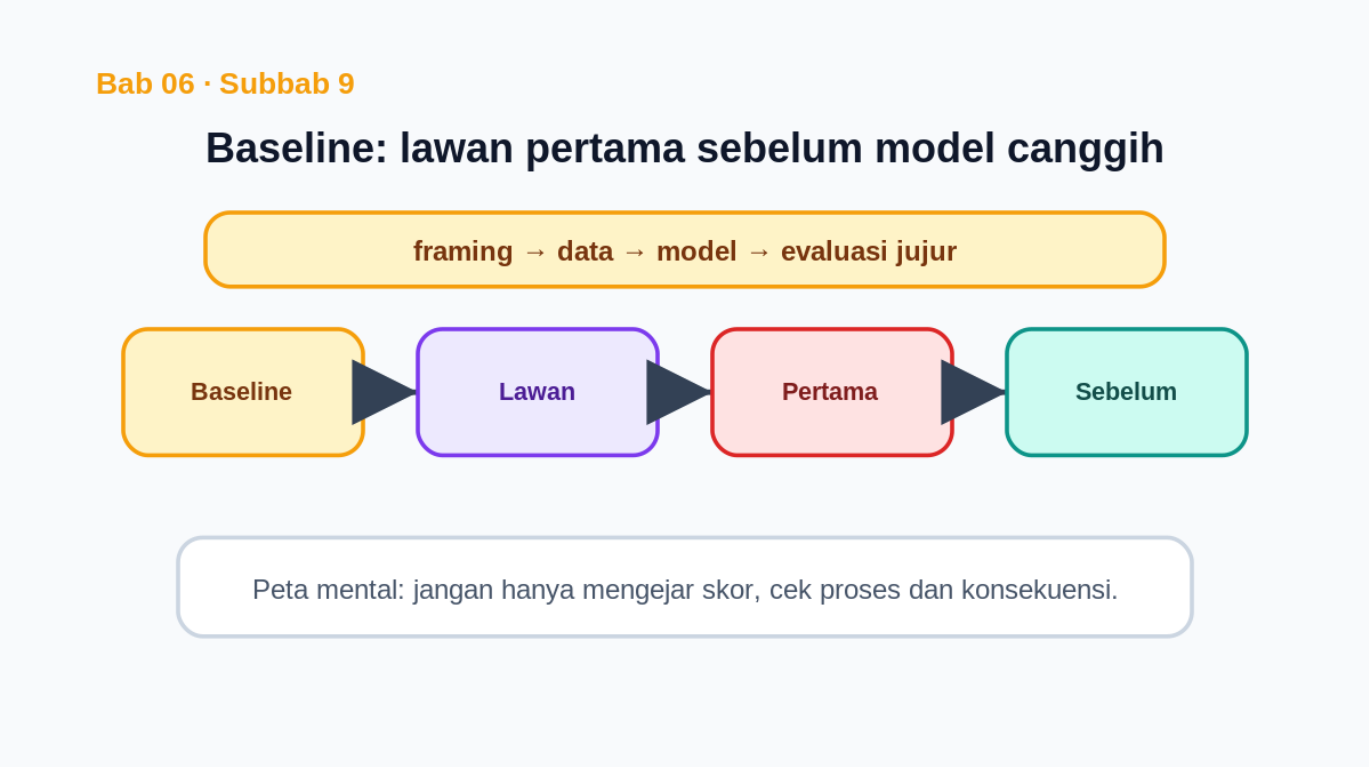
3. Pakai test sekali untuk estimasi akhir.

Jika test dipakai berulang untuk memilih model, maka test berubah menjadi validation kedua dan angka akhirnya terlalu optimis.

Tes cepat subbab 8

1. Apa fungsi train, validation, dan test?
2. Mengapa test set tidak boleh sering diintip?
3. Buat split sederhana untuk 100 data.

Subbab 9 — Baseline: lawan pertama sebelum model canggih



Baseline: lawan pertama sebelum model canggih

Baseline adalah model sederhana sebagai titik pembanding. Untuk klasifikasi, baseline bisa selalu menebak kelas mayoritas. Untuk regresi, baseline bisa selalu menebak rata-rata target. Jika model canggih tidak mengalahkan baseline, model itu belum layak dibanggakan.

Baseline membuat kita jujur. Pada dataset tidak seimbang, model yang selalu menebak “tidak fraud” mungkin punya akurasi 99% jika fraud hanya 1%. Akurasi tinggi itu palsu secara bisnis karena semua fraud terlewat. Baseline membantu kita melihat apakah metrik yang dipakai benar-benar bermakna.

Dalam praktikum Bab 6, pembaca akan membuat majority baseline dan rule-based model kecil. Tujuannya bukan mengejar algoritma rumit, tetapi memahami evaluasi jujur sebelum masuk model supervised yang lebih kuat.

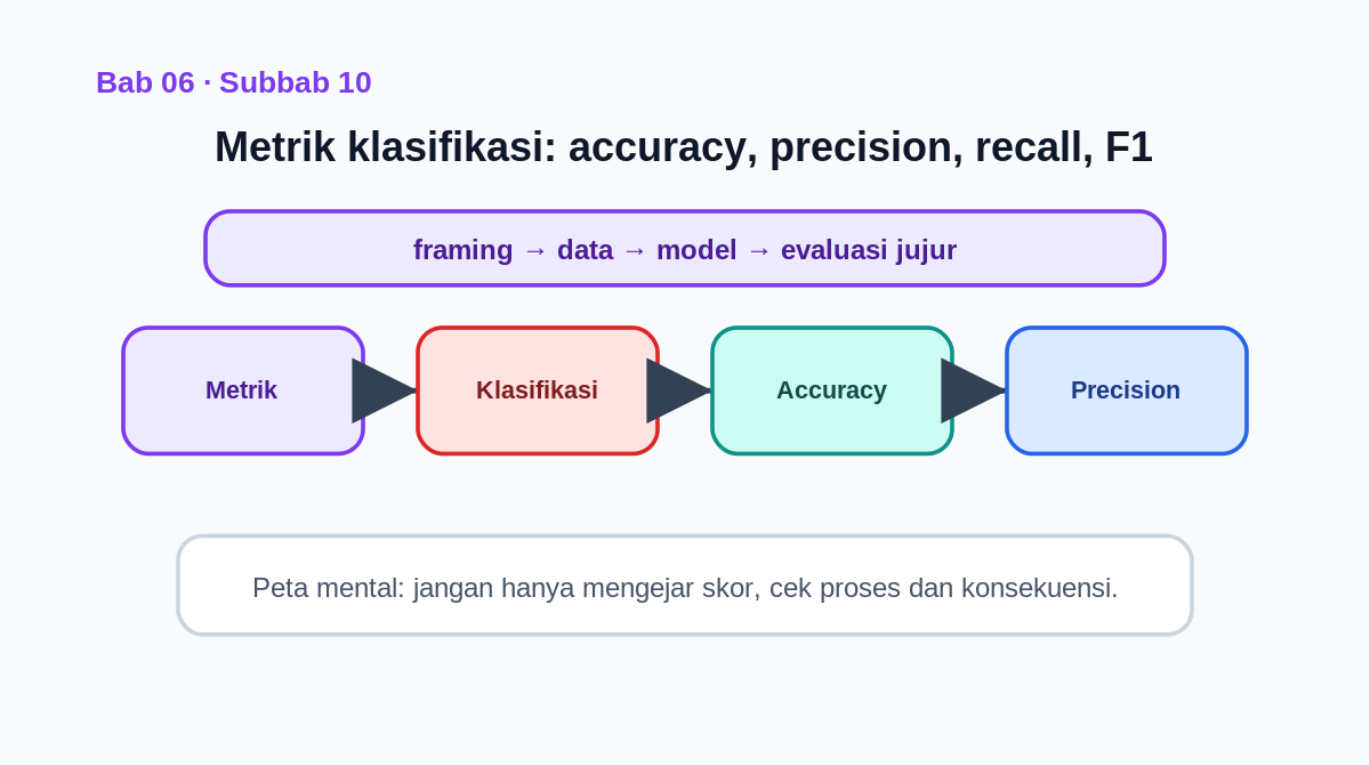
Ide teknis / latihan ketik kecil

```
label_train = [0, 0, 0, 1, 0]
mayoritas = max(set(label_train), key=label_train.count)
print(mayoritas)
```

Tes cepat subbab 9

1. Apa itu baseline?
2. Mengapa akurasi baseline bisa menipu pada data tidak seimbang?
3. Buat baseline untuk prediksi jumlah penjualan.

Subbab 10 — Metrik klasifikasi: accuracy, precision, recall, F1



Metrik klasifikasi: accuracy, precision, recall, F1

Accuracy adalah proporsi prediksi benar. Mudah dipahami, tetapi bisa menipu jika kelas tidak seimbang. Precision menjawab: dari semua yang diprediksi positif, berapa yang benar positif? Recall menjawab: dari semua positif sebenarnya, berapa yang berhasil ditemukan? F1 merangkum precision dan recall dengan harmonic mean.

Contoh pendidikan: positif berarti “butuh bantuan”. Precision rendah berarti banyak siswa ditandai butuh bantuan padahal tidak; tenaga guru bisa terbagi. Recall rendah berarti banyak siswa yang butuh bantuan terlewat; ini mungkin lebih berbahaya. Pilihan metrik harus sesuai konsekuensi.

Tidak ada metrik tunggal yang selalu benar. Untuk skrining kesehatan, recall tinggi mungkin prioritas. Untuk tindakan mahal atau sensitif, precision juga penting. ML bukan hanya hitung skor; ML adalah menyeimbangkan risiko.

Ide teknis / latihan ketik kecil

```
precision = TP / (TP + FP)
recall    = TP / (TP + FN)
F1        = 2PR / (P + R)
```

Contoh hitung metrik lengkap

Misalkan confusion matrix:

```
TP=8, FP=2, TN=7, FN=3
```

Maka:

```
accuracy = (TP+TN)/(TP+FP+TN+FN) = (8+7)/20 = 0,75
precision = TP/(TP+FP) = 8/(8+2) = 0,80
recall = TP/(TP+FN) = 8/(8+3) = 0,727
F1 = 2PR/(P+R) = 2×0,80×0,727/(0,80+0,727) ≈ 0,762
```

Angka ini memberi cerita: model cukup presisi, tetapi masih melewatkan 3 positif.

Tes cepat subbab 10

1. Apa bedanya precision dan recall?

2. Dalam kasus siswa butuh bantuan, mana yang lebih berisiko: FP atau FN?

3. Hitung precision jika TP=8 dan FP=2.

Subbab 11 — Confusion matrix: peta benar-salah yang lebih jujur

Confusion matrix: peta benar-salah yang lebih jujur

Confusion matrix memecah prediksi menjadi empat: true positive, false positive, true negative, false negative. Tabel ini membuat kesalahan terlihat jelas. Daripada hanya berkata akurasi 85%, kita bisa melihat model melewatkan banyak kasus positif atau terlalu sering memberi alarm.

Untuk kasus pendidikan, false negative adalah siswa butuh bantuan tetapi tidak terdeteksi. False positive adalah siswa tidak butuh bantuan tetapi ditandai. Dua kesalahan ini tidak sama secara manusiawi. Confusion matrix memaksa kita membicarakan konsekuensi, bukan hanya angka rata-rata.

Ketika model buruk, jangan langsung ganti algoritma. Lihat confusion matrix per segmen: kelas, wilayah, perangkat, kelompok umur, periode waktu. Bisa jadi performa bagus secara umum tetapi buruk pada kelompok tertentu.

Ide teknis / latihan ketik kecil

```
y_true = [1, 0, 1, 0]
y_pred = [1, 1, 0, 0]
# TP=1, FP=1, FN=1, TN=1
```

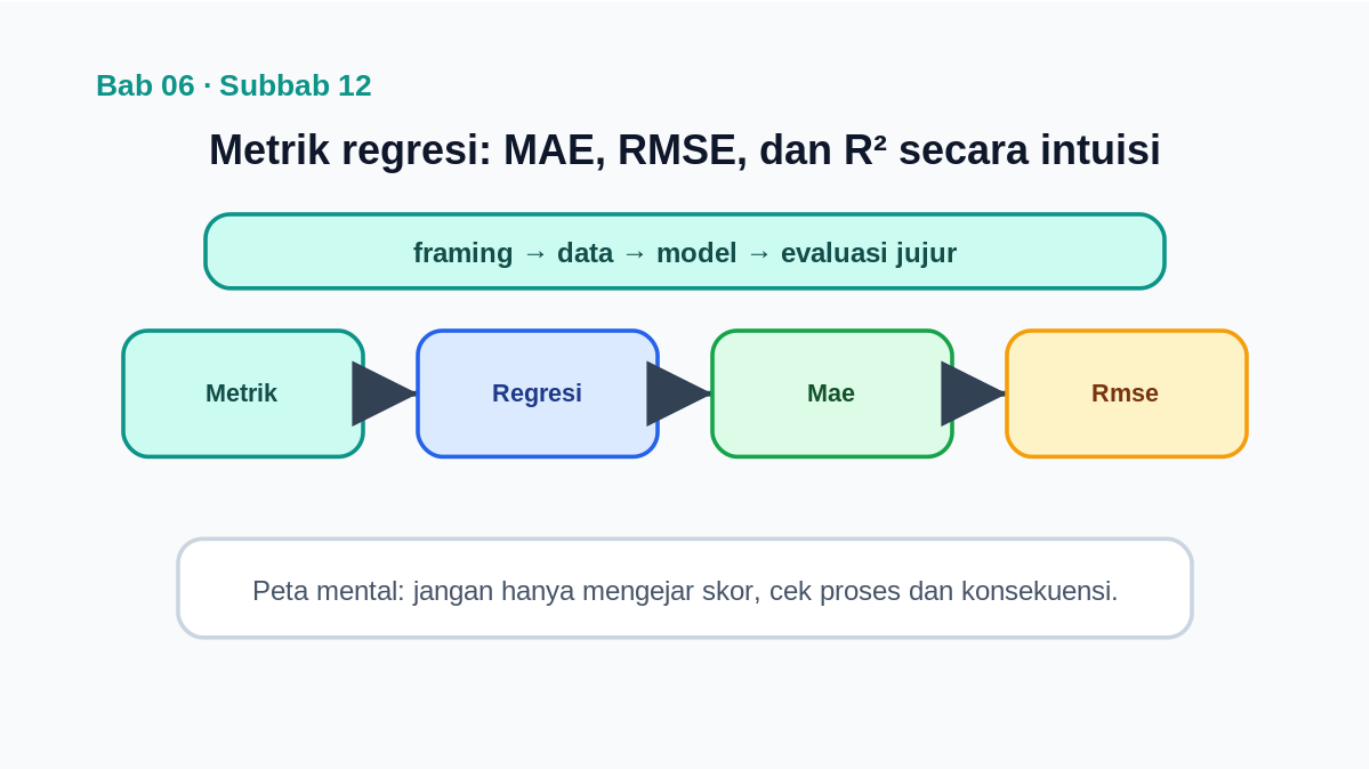
Tes cepat subbab 11

1. Apa empat isi confusion matrix?

2. Mengapa FP dan FN punya konsekuensi berbeda?

3. Buat confusion matrix kecil untuk 6 prediksi.

Subbab 12 — Metrik regresi: MAE, RMSE, dan R^2 secara intuisi



Metrik regresi: MAE, RMSE, dan R² secara intuisi

Untuk target angka, metrik yang umum adalah MAE, RMSE, dan R². MAE menjawab rata-rata meleset berapa unit. RMSE mirip akar MSE, lebih menghukum error besar. R² mengukur seberapa banyak variasi target yang dijelaskan model dibanding baseline rata-rata.

Dalam prediksi stok es teh, MAE mudah dijelaskan: model meleset rata-rata 5 gelas. RMSE berguna jika kesalahan besar sangat merugikan, misalnya kehabisan stok pada event. R² berguna untuk melihat apakah model benar-benar lebih informatif daripada rata-rata, tetapi bisa disalahpahami jika dipakai sendirian.

Metrik regresi harus dikaitkan dengan biaya nyata. Meleset 5 gelas mungkin kecil untuk kafe besar, tetapi besar untuk warung kecil. Angka metrik tidak punya makna penuh tanpa konteks bisnis.

Ide teknis / latihan ketik kecil

```
MAE  = mean(|prediksi - aktual|)
RMSE = sqrt(mean((prediksi - aktual)^2))
```

Contoh hitung RMSE dan baseline

Aktual `[10, 20, 30]`, prediksi `[12, 18, 33]`:

```
error = [2, -2, 3]
MAE = (2+2+3)/3 = 2,33
MSE = (4+4+9)/3 = 5,67
RMSE = sqrt(5,67) ≈ 2,38
```

Jika baseline mean selalu memprediksi 20:

```
error baseline = [10,0,-10]
MAE baseline = 20/3 = 6,67
```

Model mengalahkan baseline mean pada contoh ini.

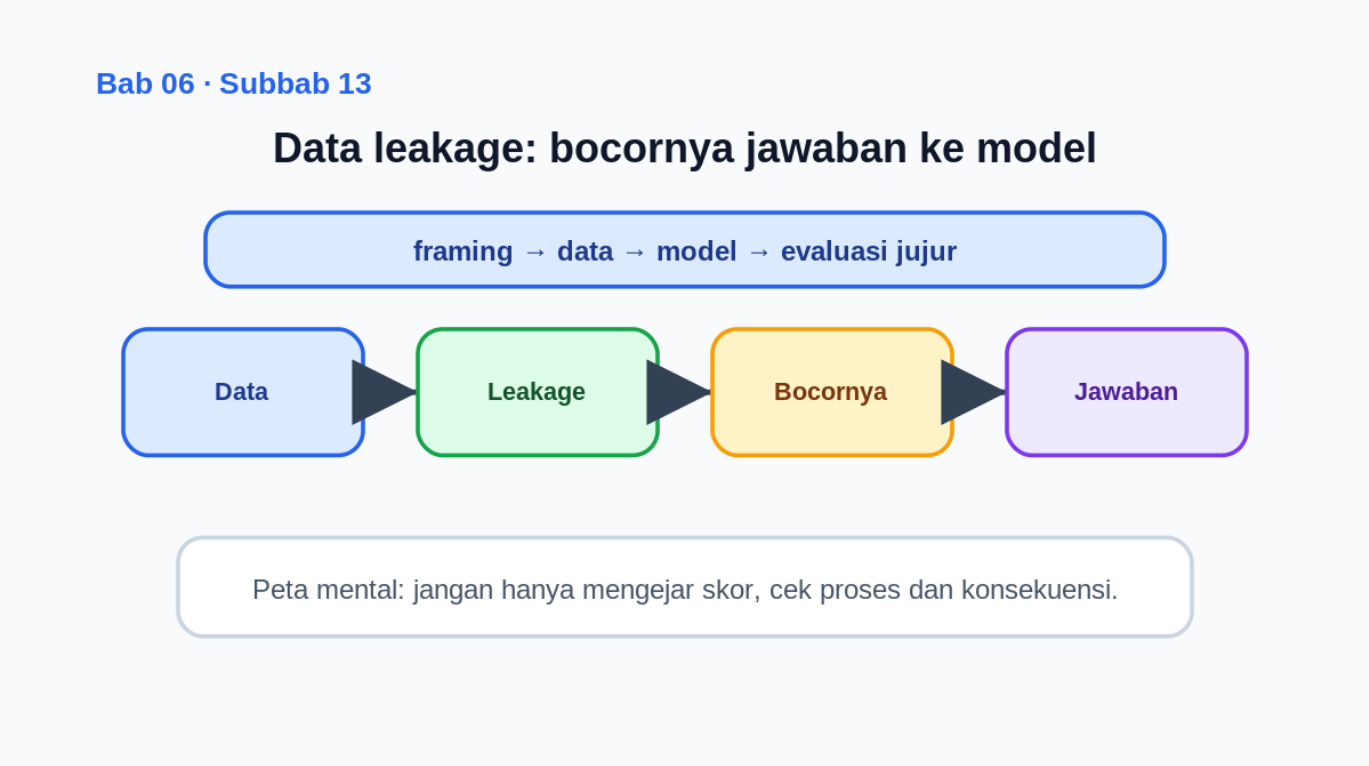
Tes cepat subbab 12

1. Apa arti MAE dalam bahasa bisnis?

2. Mengapa RMSE lebih sensitif terhadap error besar?

3. Buat contoh error prediksi stok dan hitung MAE sederhana.

Subbab 13 — Data leakage: bocornya jawaban ke model



Data leakage: bocornya jawaban ke model

Data leakage terjadi ketika informasi yang tidak tersedia pada waktu prediksi masuk sebagai fitur. Model terlihat sangat akurat, tetapi hanya karena ia mencuri petunjuk dari masa depan atau dari label itu sendiri. Leakage adalah salah satu kesalahan paling berbahaya dalam ML.

Contoh: memprediksi apakah siswa butuh bantuan minggu depan, tetapi fitur berisi “sudah dipanggil tutor minggu depan”. Itu jelas terjadi setelah keputusan. Contoh lain: memprediksi gagal bayar, tetapi fitur berisi status penagihan setelah jatuh tempo.

Tanda leakage: performa terlalu bagus untuk jadi kenyataan, fitur punya hubungan hampir sempurna dengan label, atau fitur baru tersedia setelah outcome terjadi. Cara mencegahnya: tulis waktu prediksi, daftar fitur yang tersedia saat itu, dan audit setiap kolom.

Ide teknis / latihan ketik kecil

```
Pertanyaan anti-leakage: apakah fitur ini sudah diketahui saat prediksi dibuat?
Jika belum, jangan pakai.
```

Timeline anti-leakage

Tulis garis waktu:

```
Minggu 1-4: fitur tersedia
Jumat sore: prediksi dibuat
Minggu 5: intervensi terjadi
Akhir minggu 5: label diketahui
```

Fitur hanya boleh berasal dari sebelum Jumat sore. Jika kolom berasal dari minggu 5 setelah intervensi, kolom itu bocor. Pertanyaan wajib: “apakah nilai fitur ini sudah diketahui saat prediksi dibuat?”

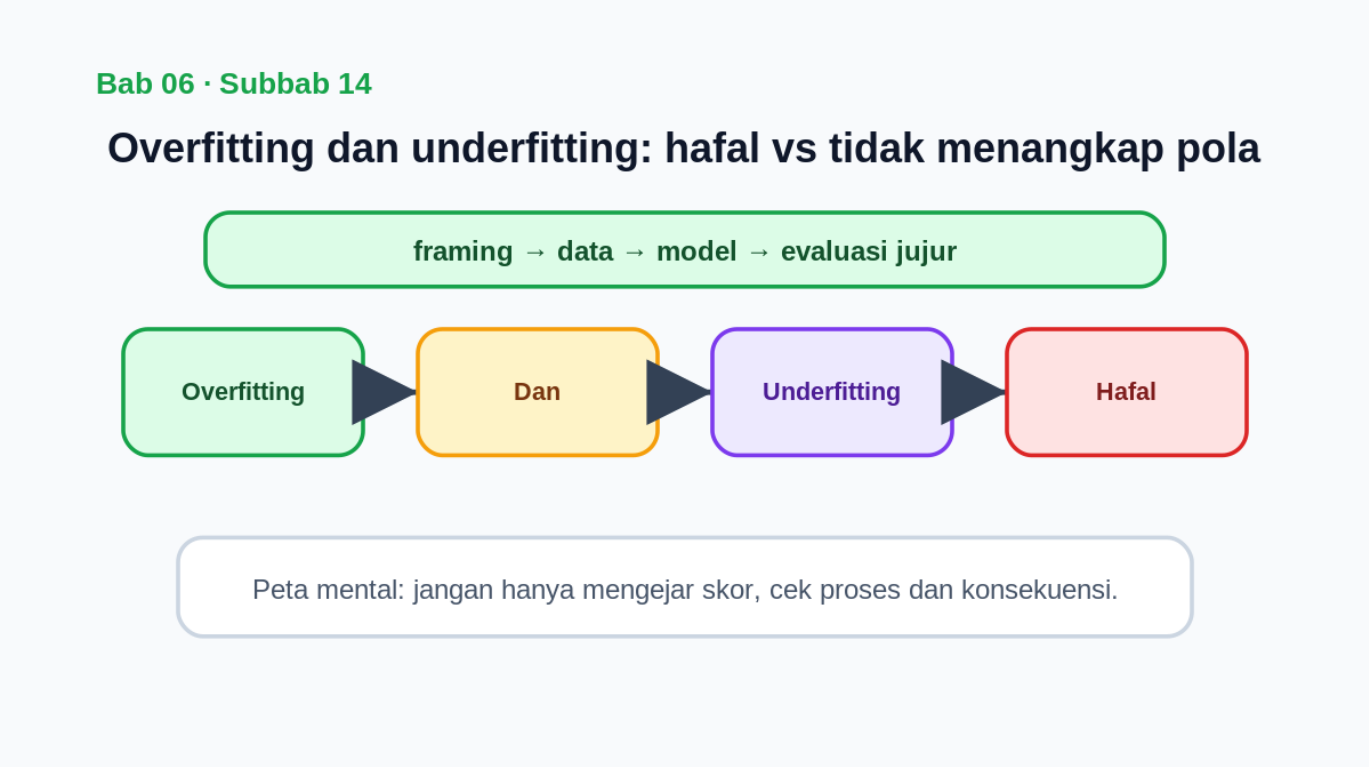
Tes cepat subbab 13

1. Apa itu data leakage?

2. Mengapa performa terlalu bagus bisa mencurigakan?

3. Sebutkan fitur bocor untuk kasus prediksi stok.

Subbab 14 — Overfitting dan underfitting: hafal vs tidak menangkap pola



Overfitting dan underfitting: hafal vs tidak menangkap pola

Underfitting terjadi ketika model terlalu sederhana atau fitur kurang informatif sehingga gagal menangkap pola bahkan pada train data. Overfitting terjadi ketika model terlalu menyesuaikan diri pada train data sehingga gagal pada data baru.

Analogi siswa: underfitting seperti belum belajar materi inti. Overfitting seperti menghafal jawaban latihan tanpa memahami konsep. Saat soal sedikit berubah, performa jatuh.

Gejala praktis: jika train dan validation sama-sama buruk, curigai underfitting. Jika train bagus tetapi validation buruk, curigai overfitting. Solusi bisa berupa fitur lebih baik, model lebih sesuai, regularisasi, data lebih banyak, atau evaluasi split yang lebih sehat.

Ide teknis / latihan ketik kecil

```
Train buruk + valid buruk → underfitting
Train bagus + valid buruk → overfitting
```

Tes cepat subbab 14

1. Apa bedanya overfitting dan underfitting?

2. Bagaimana train/validation loss membantu diagnosis?

3. Buat contoh nilai train/valid yang menunjukkan overfitting.

Subbab 15 — Bias-variance tradeoff dengan bahasa manusia

Bias-variance tradeoff dengan bahasa manusia

Bias adalah kesalahan karena asumsi model terlalu sederhana. Variance adalah sensitivitas model terhadap data latih. Model bias tinggi cenderung underfit. Model variance tinggi cenderung overfit. Tradeoff berarti kita mencari keseimbangan: cukup fleksibel menangkap pola, tetapi tidak terlalu liar mengikuti noise.

Bayangkan menggambar garis tren pada titik data. Garis lurus mungkin terlalu kaku jika pola melengkung: bias tinggi. Garis berliku yang melewati semua titik mungkin terlalu mengikuti noise: variance tinggi. Model baik menangkap pola utama tanpa mengejar setiap kebetulan.

Konsep ini membantu pembaca memahami mengapa model lebih kompleks tidak selalu lebih baik. Banyak proyek nyata menang dengan fitur bersih, baseline kuat, dan evaluasi jujur, bukan algoritma paling glamor.

Ide teknis / latihan ketik kecil

```
Bias tinggi: model terlalu kaku
Variance tinggi: model terlalu sensitif pada data latih
```

Persamaan dekomposisi intuitif

Dalam regresi, error harapan sering dijelaskan secara konseptual sebagai:

$$\text{Expected error} \approx \text{bias}^2 + \text{variance} + \text{irreducible noise}$$

Bias tinggi berarti model terlalu sederhana. Variance tinggi berarti model terlalu sensitif pada data latih. Irreducible noise adalah bagian ketidakpastian yang tidak bisa hilang karena data memang bising atau fitur tidak lengkap.

Contoh: prediksi nilai siswa tidak bisa sempurna hanya dari kehadiran dan kuis karena faktor kesehatan, suasana rumah, dan kualitas tidur tidak tercatat.

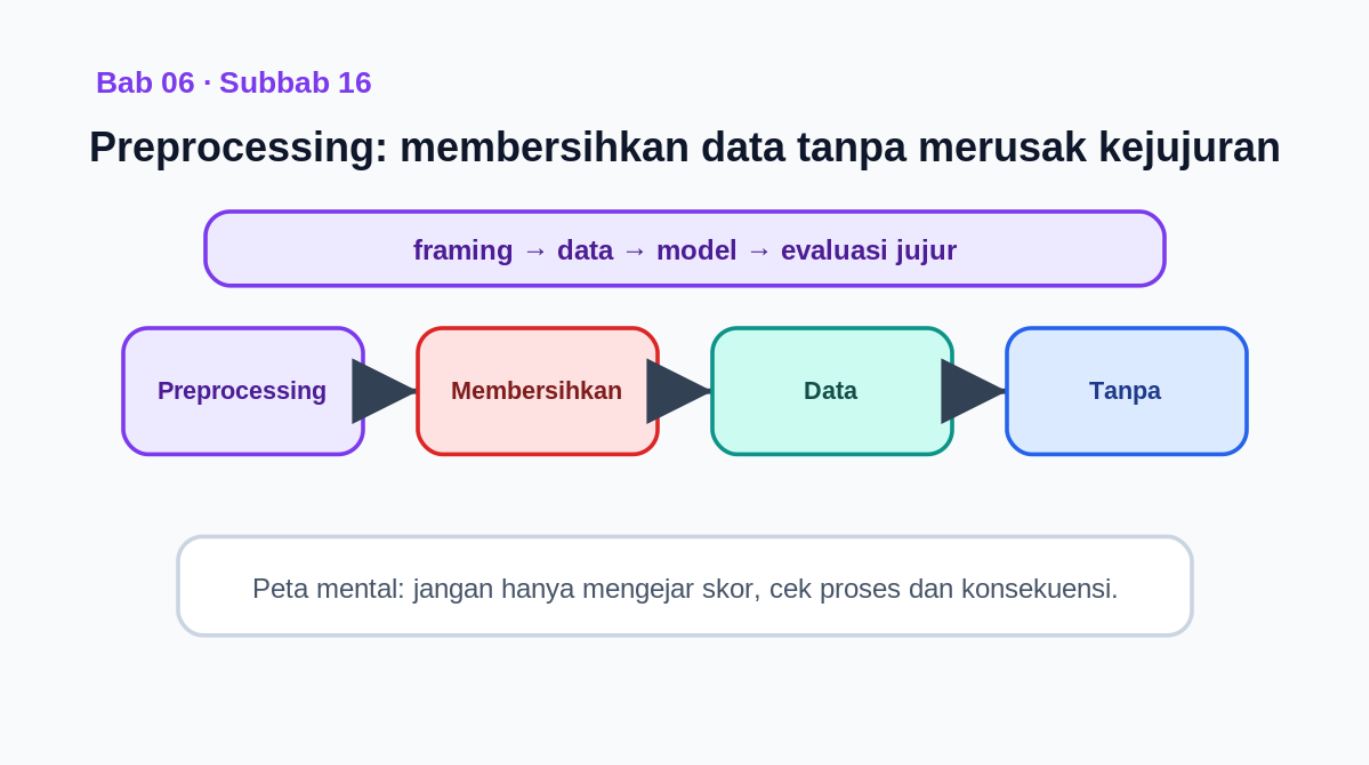
Tes cepat subbab 15

1. Apa arti bias dan variance dalam konteks model?

2. Mengapa model kompleks tidak selalu lebih baik?

3. Berikan analogi bias-variance selain garis tren.

Subbab 16 — Preprocessing: membersihkan data tanpa merusak kejujuran



Preprocessing: membersihkan data tanpa merusak kejujuran

Preprocessing mencakup menangani nilai hilang, outlier, encoding kategori, scaling angka, dan membersihkan format. Langkah ini sering menentukan kualitas model. Data bagus dengan model sederhana sering mengalahkan data berantakan dengan model canggih.

Namun preprocessing juga bisa menyebabkan leakage. Misalnya menghitung rata-rata scaling dari seluruh dataset termasuk test, lalu memakai nilai itu untuk train. Seharusnya parameter preprocessing dipelajari dari train saja, lalu diterapkan ke validation/test. Prinsipnya sama: test harus tetap seperti data baru.

Pembaca perlu membangun kebiasaan pipeline: semua transformasi yang “belajar” dari data harus fit pada train, bukan pada semua data. Ini akan menjadi penting saat memakai scikit-learn pipeline di bab berikutnya.

Ide teknis / latihan ketik kecil

```
Fit preprocessing di train.
Apply transform ke validation/test.
Jangan belajar dari test.
```

Contoh scaling train-only

Train fitur umur akun: `[10, 20, 30]`. Mean train = 20. Test punya nilai `[40]`. Standardisasi test harus memakai mean train, bukan mean gabungan.

```
z_test = (40 - mean_train) / std_train
```

Jika mean dihitung memakai test juga, evaluasi mendapat informasi dari data baru. Ini bentuk leakage preprocessing.

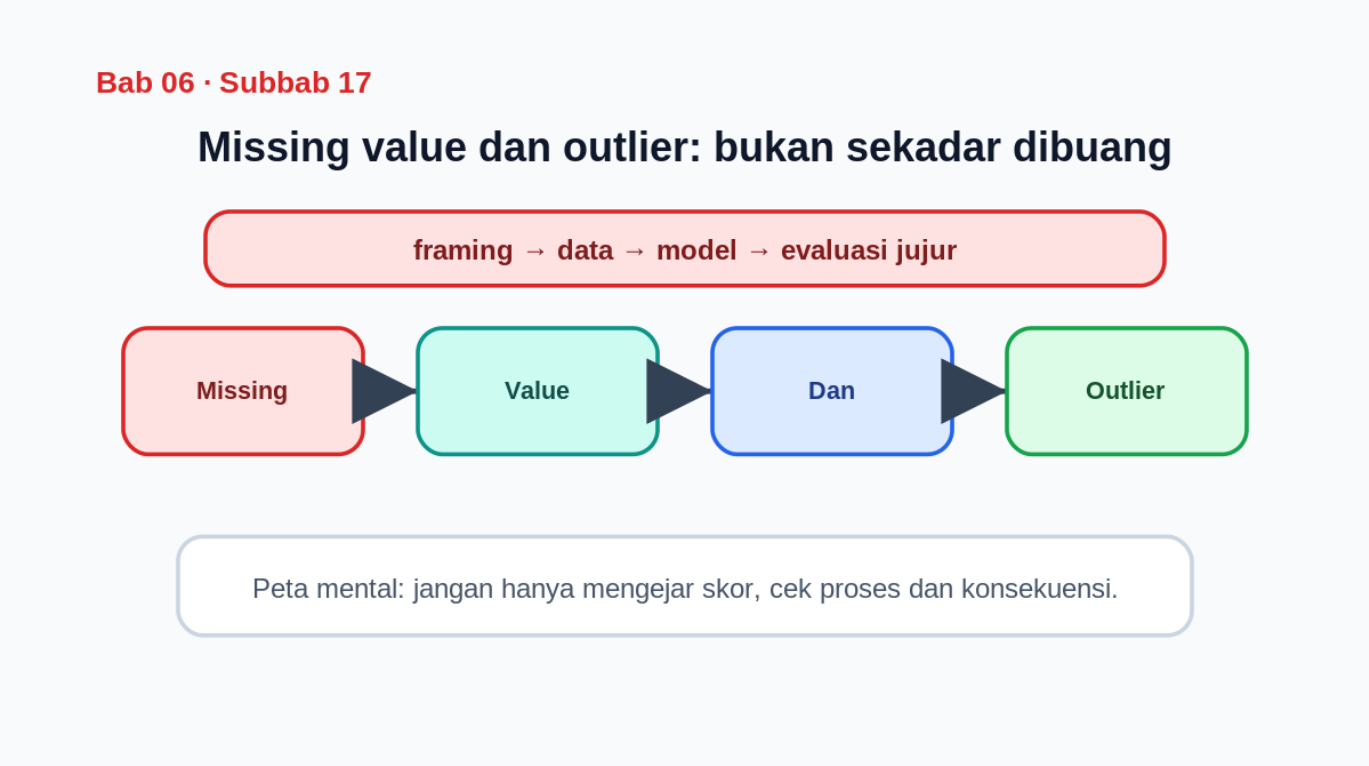
Tes cepat subbab 16

1. Apa saja contoh preprocessing?

2. Bagaimana preprocessing bisa menyebabkan leakage?

3. Mengapa scaling sebaiknya dihitung dari train saja?

Subbab 17 — Missing value dan outlier: bukan sekadar dibuang



Missing value dan outlier: bukan sekadar dibuang

Nilai hilang bisa berarti lupa dicatat, tidak berlaku, pengguna tidak menjawab, sensor mati, atau data sengaja disembunyikan. Cara menanganinya tergantung makna. Kadang diisi median, kadang dibuat kategori “tidak diketahui”, kadang baris dibuang, kadang justru missingness menjadi fitur.

Outlier juga tidak selalu salah. Penjualan 200 gelas mungkin outlier pada hari biasa, tetapi valid saat ada konser dekat warung. Jika outlier valid dan penting, membuangnya membuat model buta terhadap momen ekstrem. Jika outlier akibat salah input, ia perlu diperbaiki.

Prinsipnya: jangan membersihkan data secara otomatis tanpa memahami konteks. Tanya: apakah nilai ini mungkin terjadi? apa penyebabnya? apakah akan muncul saat model dipakai? keputusan apa yang terpengaruh?

Ide teknis / latihan ketik kecil

```
data = [30, 31, None, 29, 200]
bersih_sementara = [x for x in data if x is not None]
print(bersih_sementara)
```

Tes cepat subbab 17

1. Mengapa missing value punya banyak arti?

2. Kapan outlier sebaiknya tidak dibuang?

3. Buat contoh outlier valid pada bisnis lokal.

Subbab 18 — Imbalance: ketika kelas penting justru sedikit

Imbalance: ketika kelas penting justru sedikit

Class imbalance terjadi ketika satu kelas jauh lebih banyak daripada kelas lain. Fraud, penyakit langka, dropout, dan kerusakan mesin sering jarang tetapi penting. Model yang mengejar accuracy bisa mengabaikan kelas minoritas dan tetap terlihat bagus.

Jika 1% transaksi fraud, model yang selalu menebak tidak fraud punya accuracy 99%, tetapi recall fraud 0%. Ini buruk. Untuk kasus imbalance, precision, recall, F1, PR curve, threshold tuning, dan cost-sensitive thinking lebih penting.

Imbalance juga perlu dilihat bersama kapasitas tindakan. Jika hanya ada 10 guru pendamping, model mungkin perlu ranking prioritas, bukan hanya klasifikasi. Metrik harus mengikuti aksi nyata.

Ide teknis / latihan ketik kecil

```
Accuracy tinggi tidak cukup jika kelas penting jarang.
Lihat recall/precision untuk kelas penting.
```

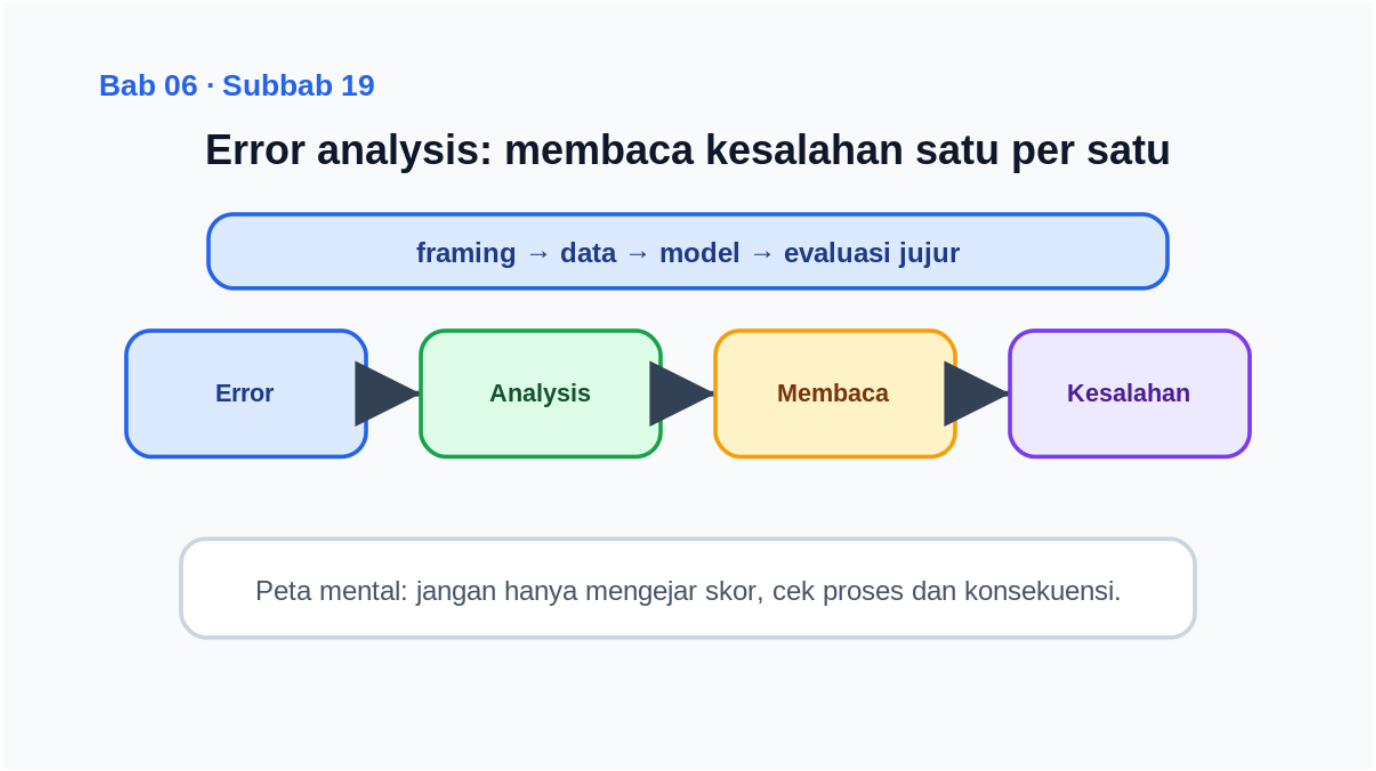
Tes cepat subbab 18

1. Apa itu class imbalance?

2. Mengapa accuracy 99% bisa buruk?

3. Kapan ranking lebih berguna daripada klasifikasi pada data imbalance?

Subbab 19 — Error analysis: membaca kesalahan satu per satu



Error analysis: membaca kesalahan satu per satu

Setelah mendapat skor, buka contoh yang salah. Error analysis adalah kegiatan membaca kasus false positive dan false negative untuk mencari pola. Apakah model sering salah pada siswa dengan data tidak lengkap? Pada wilayah tertentu? Pada hari libur? Pada kelas tertentu?

Error analysis mengubah metrik menjadi arah perbaikan. Jika banyak false negative terjadi pada siswa dengan absensi tinggi tetapi nilai kuis sedang, mungkin fitur interaksi perlu dibuat. Jika false positive banyak pada siswa baru karena data historis pendek, mungkin perlu fitur umur akun atau aturan khusus.

Tanpa error analysis, kita mudah mengganti model secara acak. Dengan error analysis, perbaikan menjadi terarah: data, label, fitur, metrik, threshold, atau proses bisnis.

Ide teknis / latihan ketik kecil

> Skor memberi tahu seberapa buruk.
> Error analysis memberi tahu mengapa buruk.

Template analisis error

ID	y aktual	prediksi	jenis error	fitur mencurigakan	hipotesis
S12	1	0	FN	absensi rendah, nilai sedang	fitur konsultasi tidak cukup
S45	0	1	FP	tugas rendah sementara	ada event khusus minggu itu

Error analysis mengubah angka menjadi tindakan: perbaiki fitur, audit label, ubah threshold, atau buat jalur review manusia.

Tes cepat subbab 19

1. Apa tujuan error analysis?

2. Mengapa melihat contoh salah lebih berguna daripada hanya skor?

3. Sebutkan pola error yang mungkin terjadi pada data pendidikan.

Subbab 20 — Reproducibility: hasil yang bisa diulang

Reproducibility: hasil yang bisa diulang

Reproducibility berarti orang lain, atau diri kita minggu depan, bisa menjalankan eksperimen dan mendapat hasil yang sama atau setidaknya sebanding. Ini membutuhkan seed, versi data, versi kode, parameter, metrik, dan catatan eksperimen.

Dalam pembelajaran, reproducibility membuat praktikum tidak menjadi “kok di laptop saya beda?”. Dalam produksi, reproducibility membuat bug bisa dilacak. Jika model berubah performa, kita perlu tahu apakah penyebabnya data, kode, parameter, atau lingkungan.

Biasakan mencatat: tanggal eksperimen, tujuan, dataset, split, baseline, model, hyperparameter, metrik train/valid/test, dan catatan error. Kebiasaan ini tampak administratif, tetapi sangat profesional.

Ide teknis / latihan ketik kecil

> Catatan minimal: seed + data + split + model + metrik + kesimpulan

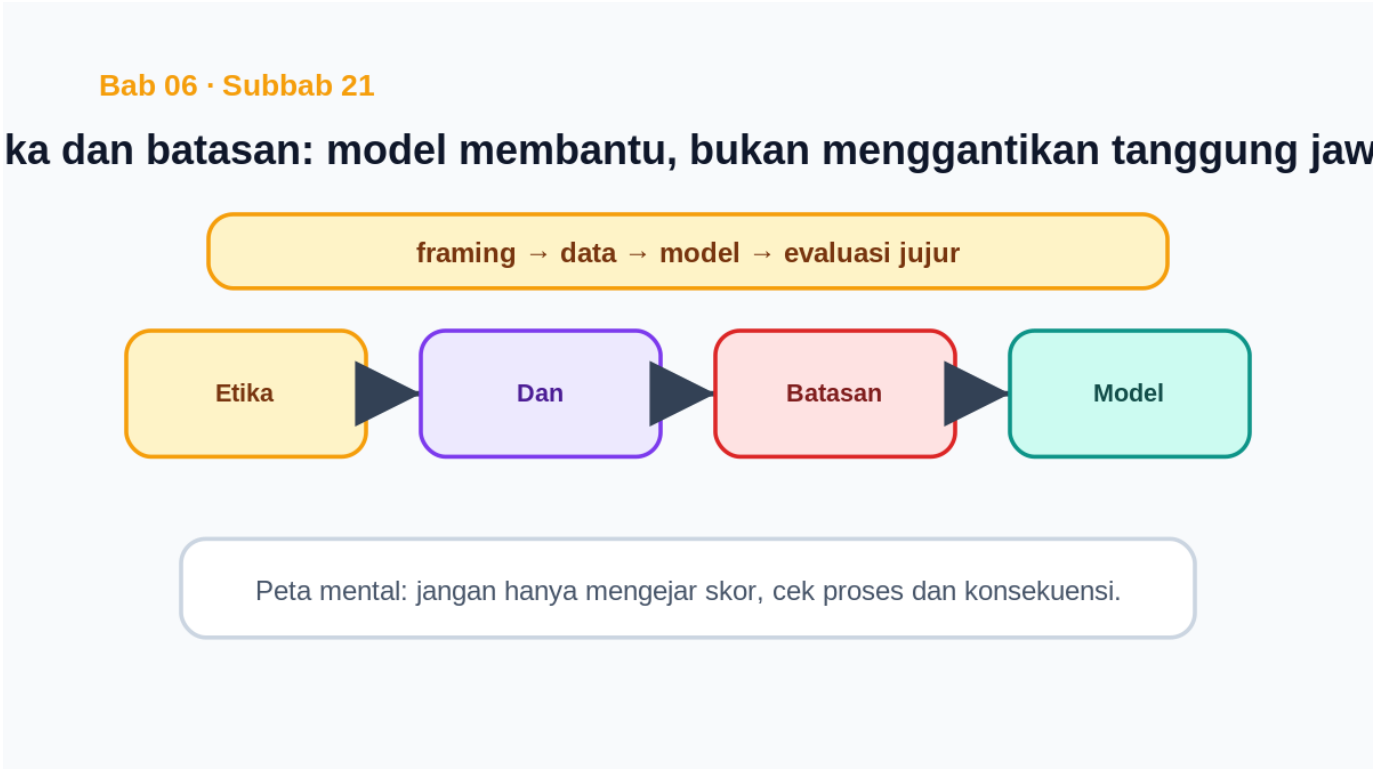
Tes cepat subbab 20

1. Apa itu reproducibility?

2. Mengapa seed penting?

3. Buat template catatan eksperimen kecil.

Subbab 21 — Etika dan batasan: model membantu, bukan menggantikan tanggung jawab



Etika dan batasan: model membantu, bukan menggantikan tanggung jawab

Model ML bisa memengaruhi orang: siapa mendapat bantuan, pinjaman, rekomendasi, pemeriksaan, atau prioritas layanan. Karena itu, fondasi ML harus memuat etika. Pertanyaan penting: siapa yang dirugikan jika model salah? apakah data mewakili kelompok rentan? apakah keputusan bisa dijelaskan? apakah ada jalur banding manusia?

Dalam contoh pendidikan, model prediksi bantuan belajar seharusnya tidak memberi label permanen pada siswa. Ia sebaiknya menjadi alat bantu guru untuk mengalokasikan perhatian, dengan ruang koreksi manusia.

Batasan harus ditulis jujur. Model dilatih pada data tertentu, periode tertentu, definisi label tertentu. Jika dipakai di konteks berbeda, performa bisa berubah. Kejujuran batasan adalah bagian dari kualitas produk, bukan kelemahan.

Ide teknis / latihan ketik kecil

```
Model = alat bantu keputusan
Manusia tetap bertanggung jawab pada dampak keputusan
```

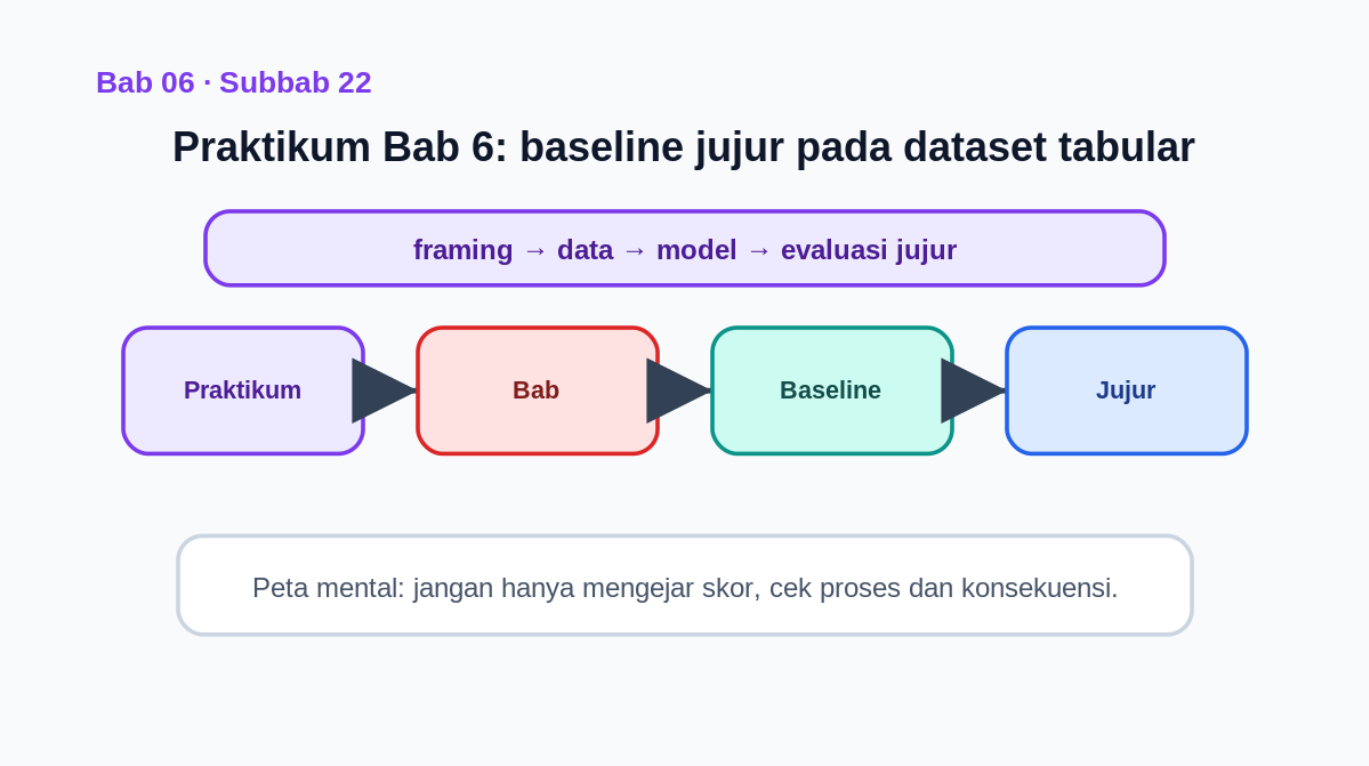
Tes cepat subbab 21

1. Mengapa ML perlu etika sejak fondasi?

2. Apa risiko memberi label permanen pada siswa?

3. Tulis satu batasan model untuk kasus UMKM.

Subbab 22 — Praktikum Bab 6: baseline jujur pada dataset tabular



Praktikum Bab 6: baseline jujur pada dataset tabular

Praktikum Bab 6 membangun dataset kecil tentang kebutuhan pendampingan siswa. Pembaca akan membagi data menjadi train/validation/test, membuat majority baseline, membuat rule-based model sederhana, menghitung confusion matrix dan metrik, lalu melihat contoh leakage.

Tujuannya bukan membuat model tercanggih. Tujuannya membangun kebiasaan ML yang benar: mulai dari baseline, pakai split jujur, pilih metrik sesuai konsekuensi, dan curiga pada performa terlalu sempurna.

File praktikum menggunakan Python standard library agar bisa diketik ulang dan dijalankan di terminal, VS Code, Jupyter, Colab, atau Kaggle. Jika pembaca paham praktikum ini, bab supervised learning berikutnya akan jauh lebih mudah.

Ide teknis / latihan ketik kecil

```
python3 ml_fundamentals_playground.py
```

Tes cepat subbab 22

1. Apa deliverable utama praktikum Bab 6?

2. Mengapa praktikum sengaja memakai baseline sederhana?

3. Sebutkan tiga hal yang harus kamu catat setelah menjalankan praktikum.

Pendalaman teknis tambahan sebelum praktikum

Ada beberapa kebiasaan teknis yang perlu mulai dibangun sejak Bab 6. Pertama, bedakan eksperimen pembelajaran dan evaluasi final. Saat belajar, kita boleh mencoba banyak threshold, banyak fitur, dan banyak versi model pada validation set. Tetapi test set harus dipakai sesedikit mungkin. Jika test set ikut dipakai untuk memilih keputusan, maka angka test tidak lagi mewakili data baru. Ini salah satu bentuk overfitting terhadap evaluasi.

Kedua, biasakan menulis asumsi. Contoh: “label kebutuhan bantuan belajar dibuat berdasarkan rubrik guru”, “fitur hanya memakai data sebelum minggu prediksi”, “model tidak dipakai sebagai keputusan final”, dan “data berasal dari kelas tertentu sehingga belum tentu mewakili sekolah

lain”. Asumsi seperti ini membuat pembaca, reviewer, dan pengguna tahu batas model. Dalam produk komersial, dokumentasi asumsi sering sama pentingnya dengan skor model.

Ketiga, jangan tertipu model yang terlalu sempurna. Dalam praktikum, leakage demo sengaja dibuat agar pembaca melihat model bisa memperoleh skor sempurna dengan cara tidak jujur. Di dunia nyata, leakage bisa lebih halus: kolom tanggal setelah kejadian, status proses yang muncul setelah keputusan, atau agregat yang dihitung memakai masa depan. Setiap fitur harus diaudit dengan pertanyaan waktu: “kapan nilai ini diketahui?”

Keempat, evaluasi harus dikaitkan dengan aksi. Jika output model hanya menjadi daftar prioritas untuk guru, maka ranking dan recall mungkin lebih penting. Jika output memicu intervensi mahal, precision juga penting. Jika prediksi dipakai untuk stok barang, MAE harus diterjemahkan ke biaya: sisa stok, stok habis, atau pelanggan kecewa. Tanpa hubungan ke aksi, metrik hanya angka cantik.

Kelima, error analysis sebaiknya dilakukan dengan empati. Dalam kasus pendidikan, baris data mewakili siswa sungguhan. Kesalahan model bukan sekadar FP atau FN; ada orang yang mungkin terlewat atau mendapat perhatian yang tidak tepat. Karena itu, model fondasi harus dilihat sebagai alat bantu refleksi, bukan mesin pelabel manusia.

Keenam, pisahkan istilah model, sistem, dan produk. Model hanya komponen yang menghasilkan prediksi. Sistem mencakup data pipeline, validasi input, penyimpanan hasil, UI, log, monitoring, dan mekanisme fallback. Produk mencakup pengalaman pengguna, kebijakan operasional, dukungan manusia, dan pertanggungjawaban. Banyak kegagalan AI bukan karena modelnya salah total, tetapi karena sistem di sekeliling model tidak menjaga kualitas data, tidak memberi konteks, atau tidak menyediakan jalur koreksi.

Ketujuh, setiap metrik perlu pembanding waktu. Skor satu kali hanya foto sesaat. Jika model dipakai berbulan-bulan, kita perlu melihat tren: apakah distribusi fitur berubah, apakah proporsi label berubah, apakah precision/recall turun pada segmen tertentu, apakah jumlah kasus yang dikirim ke manusia meningkat. Inilah alasan monitoring sudah diperkenalkan sejak fondasi, walaupun implementasi penuh MLOps ditunda ke fase lain.

Kedelapan, jangan malu memakai model sederhana. Untuk banyak organisasi, baseline yang jelas, rule model yang transparan, dan dashboard error analysis sering lebih berguna daripada model kompleks yang tidak dipahami. Model sederhana juga menjadi alat komunikasi: guru, pemilik warung, manajer UMKM, atau tim operasional dapat memahami alasan awal sebelum mempercayai model yang lebih sulit dijelaskan.

Kesembilan, setiap eksperimen sebaiknya punya keputusan berikutnya. Jika baseline sudah kuat, mungkin fokus berikutnya adalah data dan fitur. Jika recall rendah, coba threshold atau label review. Jika validation bagus tetapi test buruk, periksa split dan distribusi. Jika leakage muncul, hentikan klaim performa dan audit fitur. Dengan cara ini, ML menjadi proses investigasi yang tenang, bukan lomba angka tanpa arah.

Kesepuluh, bangun kebiasaan menulis “model card” sederhana bahkan untuk proyek belajar. Isinya: tujuan model, data yang dipakai, fitur yang digunakan, label, metrik, batasan, risiko, dan kapan model tidak boleh dipakai. Kebiasaan kecil ini membuat pembaca siap bekerja secara profesional, karena proyek AI yang bertanggung jawab harus bisa dijelaskan kepada orang lain, bukan hanya berjalan di laptop sendiri.

Latihan hitung terstruktur Bab 6

1. Dataset 1000 data dibagi 60/20/20. Hitung jumlah train, validation, test.

2. Confusion matrix `TP=12, FP=4, TN=20, FN=6`. Hitung accuracy, precision, recall, dan F1.

3. Aktual regresi `[50, 60, 70]`, prediksi `[55, 58, 65]`. Hitung MAE, MSE, RMSE.

4. Buat timeline prediksi untuk kasus siswa. Tandai fitur mana yang bocor jika diketahui setelah prediksi.

5. Jelaskan apakah kasus berikut overfit atau underfit: train accuracy 0,98, validation accuracy 0,62.

6. Buat model card mini: tujuan model, data, fitur, label, metrik, batasan, risiko.

Latihan ini menjembatani Bab 6 ke Bab 7: sebelum membandingkan algoritma, pembaca harus bisa menghitung metrik dan membaca risiko evaluasi.

Praktikum terpadu Bab 6

File utama:

- `code/ml_fundamentals_playground.py`
- `code/ml_fundamentals_playground.ipynb`

Jalankan dari terminal:

```
cd zero-to-hero-menaklukkan-ai/chapters/06-machine-learning-fundamentals/code
python3 ml_fundamentals_playground.py
```

Eksperimen wajib:

1. Bandingkan majority baseline dan rule-based model.
2. Ubah threshold rule model dan catat precision/recall.
3. Tambahkan fitur bocor dan lihat performa menjadi terlalu sempurna.
4. Hapus fitur bocor dan jelaskan mengapa hasil lebih jujur.
5. Tambahkan data siswa baru dan lihat apakah split/metrics tetap masuk akal.

Ringkasan Bab 6

- ML adalah proses belajar pola dari data untuk membantu prediksi/keputusan pada contoh baru.
- Proyek ML dimulai dari problem framing, bukan algoritma.
- Dataset adalah catatan pengalaman, bukan dunia lengkap.
- Fitur harus tersedia saat prediksi dibuat; label harus selaras dengan aksi.
- Train/validation/test menjaga evaluasi tetap jujur.
- Baseline wajib sebelum model canggih.
- Metrik harus dipilih sesuai konsekuensi, bukan hanya yang terlihat tinggi.
- Confusion matrix membantu membaca jenis kesalahan.
- Leakage membuat model terlihat hebat tetapi tidak jujur.
- Overfitting, underfitting, bias, dan variance adalah bahasa diagnosis model.
- Preprocessing harus dilakukan tanpa belajar dari test set.
- Error analysis, reproducibility, dan etika adalah bagian dari fondasi ML profesional.

Referensi utama bab

[R1] Géron. *Hands-On Machine Learning*. Workflow supervised learning, split, evaluation, leakage awareness; tidak disalin. [R2] James, Witten, Hastie, Tibshirani, Taylor. *An Introduction to Statistical Learning*. Bias-variance, classification/regression, evaluation; tidak disalin. [R3] Mitchell. *Machine Learning*. Definisi klasik machine learning dan konsep generalisasi; tidak disalin. [R4] scikit-learn documentation. Model evaluation, cross-validation, preprocessing pipeline; tidak disalin. [R5] Google Machine Learning Crash Course. Framing ML problems and data preparation; tidak disalin.

Catatan validasi internal v0.3

Aspek	Status	Catatan
Struktur	Baru	Menggunakan subbab, bukan unit baca berbatas tetap.
Kedalaman	Diperluas	22 subbab dengan narasi, detail teknis, kode kecil, dan tes per subbab.
Praktikum	Siap v0.3	Baseline jujur, rule model, metrics, leakage demo.
Risiko	Terbuka	Perlu review teknis ML dan editor manusia sebelum edisi komersial final.