BP801T · B.Pharm Semester VIII · Theory Notes

Ethical Considerations and Translational Applications of AI in Pharmacy

Unit I — AI Lifecycle, Validation & Model Auditing (6 hours)

Unit I (6 hours)

Pills & Python Notes

From-zero teaching standard: every technical term is defined before use.

Syllabus coverage: Unit I (6 hours).

PILLS & PYTHON

Pharmacy • Data • AI • Healthcare

Instagram @pillsandpython · pillsandpython.com · YouTube @PillsAndPython

CHAPTER 1: AI System Lifecycle in Pharmacy

1. Learning objectives

After studying this chapter, you will be able to:

- Define the terms AI system, model, algorithm, training, validation, deployment, and monitoring.
- List the seven stages of the AI system lifecycle in the correct order.
- Describe what happens at each stage, using a pharmacy example (such as an ADR-prediction tool in a hospital).
- Explain why each stage exists and what goes wrong if a stage is skipped.
- State one key takeaway for every stage of the lifecycle.
- Describe how the lifecycle idea protects patients and supports ethical, safe AI use in pharmacy.
- Explain the role of a pharmacist at different stages of the lifecycle.

2. Definitions

Artificial Intelligence (AI). A branch of computer science in which computers are made to perform tasks that normally need human intelligence — for example, recognising patterns, making predictions, or supporting decisions.

AI system. The complete working product: the computer program (model), plus the data it uses, the computers it runs on, the screen or report the user sees, and the rules for how people use it. A model alone is not a system; the system is the model *plus everything around it that makes it usable in real life*.

Machine learning (ML). A type of AI in which the computer is not given fixed rules by a programmer. Instead, it is shown many examples (data) and finds the patterns by itself. Most AI tools in pharmacy today are machine-learning systems.

Algorithm. A step-by-step set of instructions a computer follows. In machine learning, the algorithm is the "learning method" — the recipe the computer uses to learn patterns from data.

Model. The finished result of the learning process: a computer file that has "learnt" the patterns. Think of the algorithm as the student and the model as what the student finally knows after studying. The model is what makes predictions on new cases.

Training. The process of showing examples to the algorithm so that it learns and a model is created. Training always happens on historical (old) data.

Training data. The collection of examples used during training. Each example usually has *inputs* (information about a case) and a *label* (the correct answer for that case).

Label (also called ground truth). The correct, verified answer attached to each training example — for example, "this patient DID develop an adverse drug reaction" or "did NOT". Labels are how the computer checks whether it is learning correctly.

Feature. One piece of input information about a case — for example, patient age, drug dose, or kidney function value. Features are the columns of the data table; each patient is one row.

Prediction. The model's output for a new case — for example, "high risk of ADR" for a newly admitted patient. A prediction is a guess based on learnt patterns, not a certainty.

Validation (of a model). Honest, careful testing of the model on data it has never seen before, to check whether it really works — before it is trusted with real patients. (Chapter 3 explains validation methods in full detail.)

Deployment. Installing the tested AI system into the real workplace (for example, into a hospital's computer system) so that staff can actually use it in daily work. Deployment is "going live".

Monitoring. Watching the deployed AI system continuously after it goes live — checking that it keeps working correctly, that its predictions stay accurate, and that no harm is happening. Monitoring never stops while the system is in use.

Stakeholder. Any person or group affected by the AI system — patients, doctors, pharmacists, nurses, hospital management, and regulators.

Performance. How well the model does its job, measured with numbers (for example, what fraction of its predictions are correct). Good performance on old data does not guarantee good performance on new patients.

Lifecycle. The full journey of an AI system from the first idea ("we have a problem") to the day it is retired ("we stop using it") — including every stage in between.

Retraining. Teaching the model again with newer data when monitoring shows its performance is falling. Retraining creates a new version of the model.

Governance gate (stage gate). A checkpoint between two lifecycle stages where a team reviews the work so far and formally decides: "go ahead", "fix something first", or "stop". Gates prevent a weak system from moving forward.

3. Terminology table

Term	Simple meaning	Pharmacy example
AI system	The whole working product: model + data + software + users + rules	The complete ADR-alert tool running in the hospital, including its screen, database link, and SOP
Model	The "learnt brain" file that makes predictions	The trained file that outputs "high ADR risk" for a patient
Algorithm	The learning method / recipe used during training	The method chosen to learn ADR patterns from past records
Training	Showing old examples to the algorithm so it learns	Feeding 5 years of past prescriptions + ADR reports to the computer
Feature	One input column of information	Patient age, drug name, dose, serum creatinine
Label	The correct answer attached to each example	"ADR occurred: yes/no" confirmed from the ADR register
Prediction	The model's output on a new case	"High risk" flag for a newly admitted patient

Term	Simple meaning	Pharmacy example
Validation	Honest testing on unseen data before real use	Testing the ADR model on last year's records it never trained on
Deployment	Installing the system for real daily use	Connecting the tool to the hospital pharmacy software
Monitoring	Continuous watching after going live	Monthly check: is the tool still catching ADRs correctly?
Stakeholder	Anyone affected by the system	Patients, pharmacists, doctors, hospital management
Retraining	Teaching the model again with fresh data	Re-training every year with the newest ADR data
Governance gate	A "pass/fail" checkpoint between stages	Pharmacy committee approves the tool before hospital-wide launch

4. Explanation

4.1 Why do we need a "lifecycle"?

Technical concept: The AI system lifecycle is the ordered set of stages every AI system passes through, from idea to retirement.

Simple explanation: An AI tool is like a medicine. A medicine is not just a chemical — it goes through a journey: discovery, formulation, testing, manufacturing, quality control, marketing, and then pharmacovigilance (watching for side effects after launch). An AI system has a similar journey. If you skip a stage — for example, you skip testing and directly give the medicine to patients — people can be harmed. The lifecycle is simply the safe, step-by-step path.

Example: A hospital wants an AI tool that warns pharmacists when a prescribed drug combination has a high risk of causing an adverse drug reaction (ADR). If the team jumps straight from "idea" to "install in hospital" without proper testing, the tool may give wrong warnings — either missing real dangers or crying wolf so often that pharmacists stop trusting it.

> **Key takeaway:** *The lifecycle is a safety checklist. Every stage exists to catch a different kind of failure before it reaches the patient.*

4.2 Stage 1 — Problem framing

Technical concept: Problem framing means converting a real-world need into a precise, solvable AI task, with clear users, clear decisions, and clear success criteria.

Simple explanation: Before building anything, ask in plain words: *Whose problem are we solving? What decision will the AI help with? What does "success" look like?* An AI system cannot solve a vague wish like "improve pharmacy". It can only solve a sharp question like "which inpatients are at high risk of an ADR in the next 48 hours, so the pharmacist can review them first?"

Example (ADR tool): The hospital's problem: pharmacists cannot manually review all 500 daily prescriptions in depth, and some high-risk patients are missed. Framed AI task: "Every morning, list the 20 inpatients with the highest predicted ADR risk, so the clinical pharmacist reviews them first." Success = the tool's top-20 list actually contains most patients who later develop ADRs, without too many false alarms.

Key takeaway: *A wrongly framed problem gives a perfectly working solution to the wrong question. Spend time here; it is the cheapest stage to fix mistakes in.*

4.3 Stage 2 — Data collection

Technical concept: Data collection is gathering the historical examples (inputs + labels) the model will learn from, with attention to source, quantity, quality, permissions, and patient privacy.

Simple explanation: Machine learning learns only from examples. No examples = no learning. This stage collects past cases: patient details (features) and what actually happened (labels). It also answers: *Do we have permission to use this data? Is patient identity protected? Is the data enough — or is it biased toward one type of patient?*

Example (ADR tool): The team collects 5 years of inpatient records: age, sex, diagnoses, prescribed drugs, doses, kidney/liver values (features), plus the hospital's ADR register showing which patients actually suffered ADRs (labels). They remove names and ID numbers (de-identification), take ethics-committee permission, and notice a problem: the ADR register is incomplete because busy wards under-report mild ADRs — a data quality issue to handle in the next stage (Chapter 2 covers this fully).

Key takeaway: *The model can only be as good as its examples. Biased or careless data collection creates a biased, unsafe model — no later stage can fully repair it.*

4.4 Stage 3 — Data preparation

Technical concept: Data preparation (also called preprocessing) means cleaning and shaping raw collected data into a reliable table the algorithm can learn from: handling missing values, correcting errors, standardising formats, and encoding information numerically.

Simple explanation: Raw hospital data is messy — like uncleaned vegetables before cooking. Some lab values are missing, drug names are spelled three different ways ("Paracetamol", "PCM", "Crocin"), dates are in different formats, and a few entries are obvious errors (age = 250). Preparation cleans all this: fill or remove missing values sensibly, unify spellings, fix formats, and convert everything into numbers the computer can process.

Example (ADR tool): The team unifies drug names to generic names, converts all doses to mg, fills missing kidney values using careful rules (documented, not hidden), removes duplicate records, and splits free-text doctor notes into structured fields where possible. Every cleaning decision is written down — because the next team must be able to repeat it exactly (reproducibility, Chapter 4).

Key takeaway: *Most of the real work in an AI project (often more than half the time) happens here. Clean data in = trustworthy model out.*

4.5 Stage 4 — Model development

Technical concept: Model development is choosing a learning algorithm, training it on the prepared data, and tuning its settings so it learns the real patterns (not noise) in the data.

Simple explanation: Now the "student" studies. The team picks a learning method suitable for the problem (for yes/no risk prediction, a classification method), feeds the prepared examples, and adjusts settings so the model learns genuine medical patterns (e.g., "elderly + multiple drugs + poor kidney

function = higher ADR risk") rather than memorising coincidences (e.g., "patients admitted on Tuesdays have more ADRs" — pure chance). Trying several methods and comparing them honestly is part of this stage.

Example (ADR tool): The team trains three candidate models, tunes each carefully, and picks the one that best separates high-risk from low-risk patients on data it was *not* trained on. Pharmacists review the model's top risk factors to check they make medical sense (e.g., polypharmacy and renal impairment — sensible; "bed number" — a red flag meaning the model learnt something meaningless).

> **Key takeaway:** *A model that "memorises" training data instead of learning real patterns will fail on new patients. Development must be guided by medical sense, not just by scores.*

4.6 Stage 5 — Validation

Technical concept: Validation is rigorous, independent testing of the finished model on data it has never seen, measuring performance, checking fairness across patient groups, and confirming it is safe enough for real use.

Simple explanation: This is the "examination hall". The model studied (training); now it must answer *new* questions it has never seen, under strict supervision. We check: How many predictions are correct? Does it work equally well for elderly and young patients, men and women? Does it stay stable if the data has small errors? Only if it passes honestly does it move toward real patients. (Chapter 3 explains train/test/validation splits and cross-validation; Chapter 5 explains auditing, which is the formal version of this checking.)

Example (ADR tool): The model is tested on the most recent year's records, which were locked away and never used in training. Result: it correctly flags 8 out of 10 patients who later developed ADRs, with an acceptable false-alarm rate — and it performs similarly for patients above and below 60 years. The pharmacy committee reviews the validation report at a governance gate and approves a limited pilot.

> **Key takeaway:** *Validation is where unsafe models are stopped. Testing on the same data the model trained on is cheating — like giving a student the exam paper a day early.*

4.7 Stage 6 — Deployment

Technical concept: Deployment is integrating the validated model into the real clinical workflow: connecting it to hospital software, designing how users see and act on its outputs, training staff, and writing standard operating procedures (SOPs).

Simple explanation: The model leaves the laboratory and joins the hospital. This means: it appears inside the software pharmacists already use (not a separate app nobody opens), its warnings are clear and explainable, staff are trained on what the tool can and cannot do, and an SOP says who is responsible when the tool and the pharmacist disagree. A common safe approach is "shadow mode" first — the tool runs silently in the background, its predictions are recorded but not shown, so the team can confirm it behaves well in the live environment before anyone acts on it.

Example (ADR tool): The tool is connected to the hospital information system. Each morning it generates the top-20 high-risk list inside the pharmacy dashboard. The SOP states: the pharmacist reviews the list, the final clinical decision always rests with the pharmacist/doctor — the tool *supports*, never *replaces*, professional judgement. Two weeks of shadow mode confirm the live data feed works correctly.

Key takeaway: *The best model in the world is useless if it does not fit the workflow — and dangerous if staff misunderstand or over-trust it. Deployment is about people and process, not just software.*

4.8 Stage 7 — Monitoring

Technical concept: Monitoring is the continuous measurement of the deployed system's performance, data quality, and safety after go-live, with defined triggers for investigation, retraining, or withdrawal.

Simple explanation: After launch, the job is not over — it has just changed. Medicines have pharmacovigilance; AI systems have monitoring. Every month (or continuously), the team checks: Is the tool still accurate? Has the hospital's patient mix changed (new wards, new prescribing patterns) making old patterns outdated? Are users still using it correctly? If performance falls below an agreed level, the team investigates — and may retrain the model or even switch the tool off.

Example (ADR tool): Six months after launch, monitoring shows the tool's accuracy is slowly falling. Investigation finds the hospital started a new oncology unit with chemotherapy protocols the model never saw in training — the world changed, so the old learning is partly outdated (this is called *model drift*; Chapter 3 explains it). The team retrains the model including oncology data, re-validates it, and redeploys version 2. The lifecycle loops back — it is a cycle, not a straight line.

Key takeaway: *Deployment is not the finish line. An unmonitored AI system silently decays and can harm patients long before anyone notices.*

5. Classifications

5.1 AI systems in pharmacy, classified by purpose

Class	What it does	Pharmacy example (illustrative)
Prediction / classification	Puts cases into categories or estimates risk	ADR-risk classifier: high/medium/low risk patient
Regression (number prediction)	Predicts a continuous number	Predicting a patient's likely length of hospital stay
Natural language processing (NLP)	Reads and understands human text	Extracting drug names and doses from discharge summaries
Computer vision	"Sees" and interprets images	Checking tablet appearance for quality defects on a packing line
Recommendation / decision support	Suggests actions to a professional	Suggesting dose adjustment in renal impairment for pharmacist review

5.2 Lifecycle management styles

Style	How it works	Analogy
Linear (waterfall)	Finish one stage fully, then move to the next, rarely going back	Building a house floor by floor
Iterative (build–measure–learn)	Build a small version, test it, improve, repeat in cycles	Cooking: taste, adjust salt, taste again

Modern AI projects are almost always iterative — monitoring feeds back into retraining, which is a new small lifecycle.

5.3 Deployment modes

Mode	Meaning	When used
Shadow mode	Tool runs silently; predictions recorded but not shown to users	First live test — safe
Assisted / human-in-the-loop	Tool shows suggestions; human makes the final decision	Normal clinical use — recommended
Fully automated	Tool acts without human review	Only for low-risk, reversible tasks (e.g., sorting documents) — never for direct patient decisions without strong justification

5.4 What monitoring watches (three streams)

1 **Performance monitoring** — are predictions still correct?

2 **Data monitoring** — is the incoming data still clean and similar to training data?

3 **Usage & safety monitoring** — are staff using the tool as intended? Any patient harm signals?

6. Mechanisms

6.1 How stages connect: outputs become inputs

Each stage produces something the next stage needs:

```
Problem framing ⟶ written task definition ⟶ guides data collection
Data collection ⟶ raw dataset ⟶ feeds data preparation
Data preparation ⟶ clean dataset ⟶ feeds model development
Model development ⟶ trained model ⟶ enters validation
Validation ⟶ validation report ⟶ decision at governance gate
Deployment ⟶ live system + SOPs ⟶ enters monitoring
Monitoring ⟶ performance reports ⟶ trigger retraining (loop back to stage 3/4) or retirement
```

Mechanism in simple words: It is a relay race. Each runner (stage) must hand over a proper baton (output) to the next runner. A dropped baton — for example, undocumented cleaning decisions in stage 3 — trips every later stage.

6.2 The feedback-loop mechanism

Monitoring does not just watch; it *acts*. The mechanism:

1 A metric (e.g., monthly accuracy) is measured.

2 It is compared against a pre-decided threshold (e.g., "must stay above 80%").

3 If it falls below: alert → investigation → root-cause finding → action (fix data feed, retrain model, update SOP, or withdraw the tool).

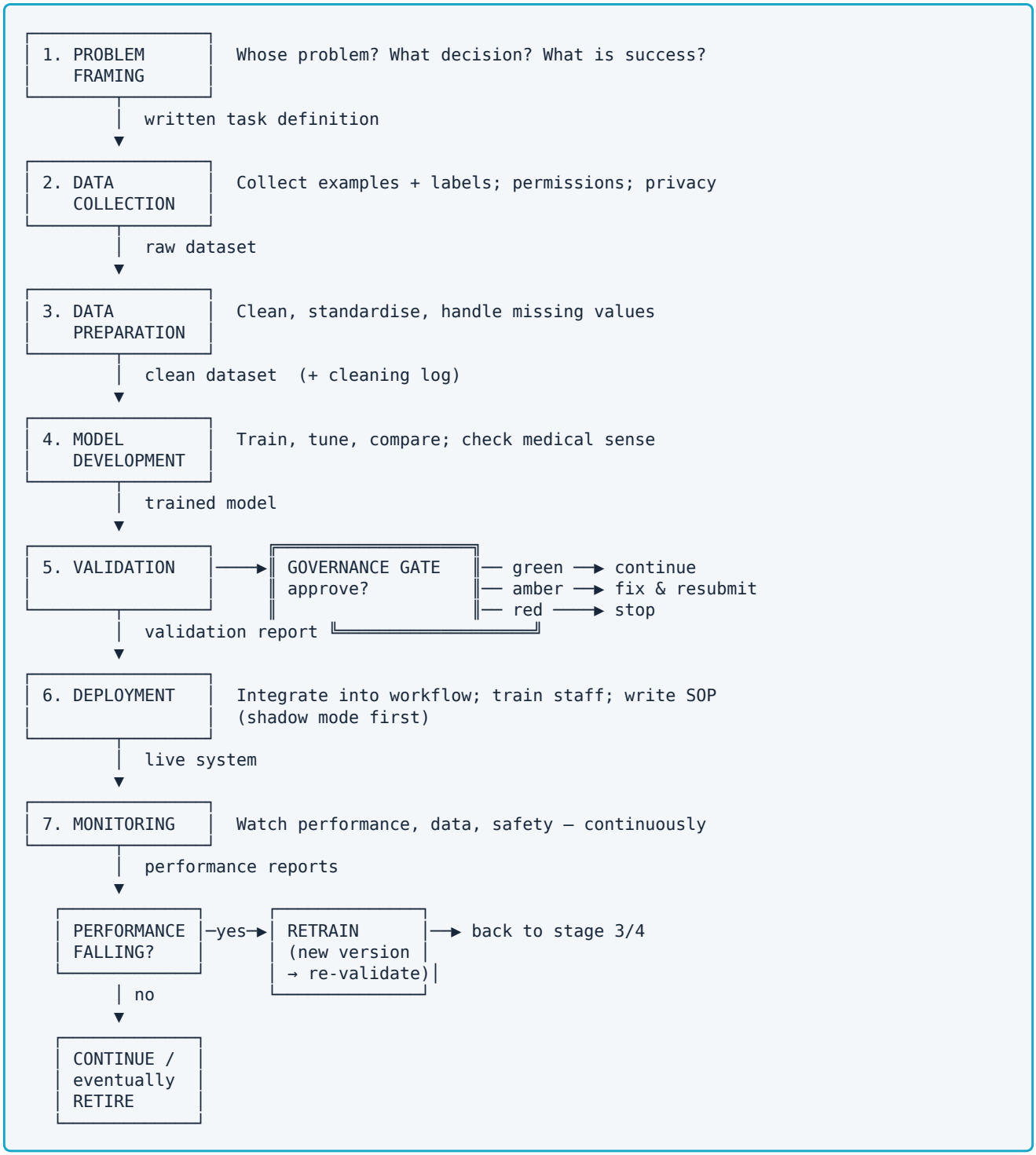
4 Any new model version re-enters validation before redeployment — the cycle repeats.

6.3 Governance gates: the "traffic signal" mechanism

Between critical stages sits a gate review: a small committee (pharmacist, doctor, data scientist, ethics/quality member) checks evidence and gives green (proceed), amber (fix and re-submit), or red (stop). The most important gates are: **before validation** ("is the data good enough to test on?") and **before deployment** ("is the validation honest and the SOP ready?").

7. Diagrams/flowcharts

7.1 The seven-stage lifecycle



7.2 What "skipping a stage" costs

```
Stage skipped          →   Typical consequence
──────────────────────────────────────────────────────────────
Problem framing        →   Solves the wrong problem; nobody uses the tool
Data collection (care) →   Biased model; unfair to some patient groups
Data preparation       →   "Garbage in, garbage out"; nonsense predictions
Model development care →   Memorises data; fails on new patients
Validation             →   Unsafe tool reaches patients
Deployment planning    →   Staff ignore or misuse the tool
Monitoring             →   Silent decay; harm discovered too late
```

8. Infographic concepts

Concept 1 — "The Medicine Factory" conveyor belt. A left-to-right conveyor with seven stations; at each station a pharmacist avatar performs the stage's task (framing = holding a question mark sign; collection = baskets of patient files; preparation = washing vegetables; development = a student studying; validation = an examiner with a red pen; deployment = installing a machine in the hospital; monitoring = a CCTV/watchtower). Under each station, one warning sign shows what happens if the station is skipped. Caption: "An AI system, like a medicine, is only as safe as its journey."

Concept 2 — The cycle wheel. A circular wheel with seven coloured segments and arrows going around, with a small "retrain" arrow looping from Monitoring back to Data Preparation. In the centre: "Patient safety". Around the rim, the three governance gates marked as traffic signals.

Concept 3 — "Shadow → Assisted → Automated" ladder. A three-rung ladder showing deployment modes from safest (shadow) at the bottom to riskiest (fully automated) at the top, with a pharmacist climbing only to the "assisted" rung for clinical decisions, holding a flag labelled "human decides".

9. Tables

9.1 Stage-by-stage summary

#	Stage	Main question answered	Key output	Who is involved
1	Problem framing	What exact decision will the AI support?	Written task definition	Pharmacists, doctors, management
2	Data collection	Do we have enough good examples, legally?	Raw dataset + permissions	Data team, ethics committee, pharmacists
3	Data preparation	Is the data clean and trustworthy?	Clean dataset + cleaning log	Data team, pharmacists (domain checks)
4	Model development	Which learnt patterns are real and useful?	Trained model	Data scientists, pharmacists (sense-check)
5	Validation	Does it work honestly on new data?	Validation report	Independent testers, pharmacy committee
6	Deployment	Does it fit the real workflow safely?	Live system + SOP + trained staff	IT, pharmacists, nurses, doctors
7	Monitoring	Is it still safe and accurate?	Performance reports, alerts	Pharmacists, quality team

9.2 Time and cost of fixing an error, by stage

Error discovered at	Cost/effort to fix	Why
Problem framing	Very low	Change words on paper
Data collection/preparation	Low–moderate	Re-collect or re-clean data
Model development	Moderate	Retrain (takes computer time)
Validation	Moderate	Fix and re-test
Deployment	High	Workflow already changed; staff retraining needed
Monitoring (after harm)	Very high	Patient harm, loss of trust, possible legal action

Lesson: find problems early — it is cheaper and safer.

10. Worked toy examples

ILLUSTRATIVE TOY DATASET (not real patient data). All numbers below are invented for teaching only.

Toy walkthrough: the ADR-prediction tool's lifecycle in miniature.

Imagine a tiny hospital ward with 1,000 past admissions (toy data):

- **Stage 1 (framing):** Task written as: "Each morning, flag the 10% of inpatients at highest ADR risk for pharmacist review. Success = at least 7 of every 10 future ADR cases appear in the flagged group."
- **Stage 2 (collection):** 1,000 records collected; 100 patients had a recorded ADR (labels). Privacy: names removed; ethics permission taken (toy scenario).
- **Stage 3 (preparation):** Found 80 records with missing kidney values → filled using the ward's documented rule; 12 duplicate records removed; drug names unified. Cleaning log written: 3 pages (toy).
- **Stage 4 (development):** Model trained on 700 records; it learns sensible patterns (many drugs + old age + poor kidneys → higher risk).
- **Stage 5 (validation):** Tested on the remaining 300 unseen records: of 30 ADR cases in this set, the model flagged 24 (8 out of 10 — meets the target of 7 out of 10 ✓). False alarms: acceptable. Gate: approved for pilot.
- **Stage 6 (deployment):** Two weeks of shadow mode → live in the pharmacy dashboard with an SOP: "pharmacist decides; tool advises."
- **Stage 7 (monitoring):** Month 4: flagged-group capture falls from 8/10 to 6/10 → investigation → a new antibiotic protocol changed ADR patterns → retrain with new data → re-validate → redeploy version 2.

Teaching point: Notice how every stage produced something the next stage used, and how monitoring looped the project back to stage 3. That loop is why it is called a *cycle*.

11. Analogies

1 **Medicine manufacturing.** API sourcing = data collection; formulation = data preparation; process development = model development; quality control testing = validation; market launch = deployment;

pharmacovigilance = monitoring. A pharmacist already understands this journey — the AI lifecycle is its digital cousin.

2 **Cooking a new dish for a wedding.** Framing = "what should I cook for 200 guests?"; collection = buying ingredients; preparation = washing/chopping; development = trial cooking; validation = tasting by elders before the event; deployment = serving on the wedding day; monitoring = watching guests' plates and asking for feedback for next time.

3 **Clinical trial phases.** Just as a drug passes Phase I → II → III → post-marketing surveillance with gates between phases, an AI system passes development → validation → deployment → monitoring with governance gates.

4 **Teaching a student.** Training = classroom teaching; validation = the final exam with unseen questions; deployment = the student starting a job; monitoring = the employer's probation review. You would never let a student treat patients without an exam — same logic for AI.

12. Pharmacy relevance

- **Pharmacists are domain experts.** At every stage, someone must judge medical sense: Is the framed problem clinically meaningful? Are the labels (ADR yes/no) correct? Do the model's learnt risk factors make pharmacological sense? Only a pharmacist/clinician can answer these — the data scientist cannot.
- **Pharmacovigilance mindset transfers directly.** Pharmacists already do post-marketing surveillance of drugs; monitoring an AI tool uses the same discipline: collect signals, investigate, act, document.
- **Medication-safety tools are the most common pharmacy AI.** ADR prediction, drug-interaction alerts, dose-checking in renal impairment, and antimicrobial-stewardship dashboards all follow this exact lifecycle.
- **The "human-in-the-loop" rule.** In pharmacy, the professional is legally and ethically responsible for the final decision. The lifecycle's deployment stage must encode this: AI advises, pharmacist decides.
- **Documentation habit.** Pharmacy already runs on SOPs, batch records, and audit trails (GMP). The AI lifecycle's demand for written task definitions, cleaning logs, validation reports, and SOPs is the same habit in a new area.

13. India context

- **ABDM and digitisation.** The Ayushman Bharat Digital Mission is pushing health-record digitisation across India. This is good news for data collection (stage 2) — but records are still mixed: big corporate hospitals have structured EHRs, while many smaller hospitals and clinics still run on paper registers and handwritten prescriptions. Lifecycle planning must account for this uneven data landscape.
- **CDSCO oversight.** Software used for clinical decisions may fall under medical-device regulation in India — as per the latest guidance, confirm the current edition with CDSCO. The validation and documentation stages of the lifecycle directly support any regulatory submission.
- **Language and diversity.** Prescriptions and notes may mix English, Hindi, and regional languages; drug brand names vary by state. Data preparation (stage 3) in India must handle this variation — a model trained only on metro-hospital English data may fail in a district hospital.

- **Resource-limited settings.** Many hospitals cannot afford continuous IT support. Monitoring (stage 7) must therefore be designed simply — e.g., a monthly one-page report the chief pharmacist reviews — or it will not happen.
- **Pharmacist-led opportunity.** With doctor shortages in many areas, pharmacist-led medication review supported by simple, well-validated AI tools (built through a careful lifecycle) can genuinely improve safety — provided the lifecycle is followed, not shortcut.

14. Key points

- An **AI system** = model + data + software + users + rules. The model alone is not the system.
- The lifecycle has **seven stages**: problem framing → data collection → data preparation → model development → validation → deployment → monitoring.
- Each stage answers one main question and hands its output to the next stage — like a relay race.
- **Governance gates** between stages stop weak systems from moving forward.
- The lifecycle is a **cycle, not a line**: monitoring feeds back into retraining.
- **Deployment is not the finish line** — monitoring is where long-term safety is protected.
- Finding and fixing errors **early is cheap**; fixing them after deployment (or after harm) is costly.
- In pharmacy, **AI advises and the pharmacist decides** — encode this in the deployment stage.
- Skipping any stage creates a specific, predictable failure (see the table in section 7.2).

15. Common confusions

Confusion	Clarification
"Model" vs "AI system"	The model is the learnt file; the AI system is the model plus data, software, users, and rules. A good model inside a bad system still fails.
"Validation" vs "verification"	Verification = "did we build it right?" (code checks). Validation = "did we build the right thing — does it work for real patients?" This chapter (and Chapter 5) focus on validation.
"Deployment" vs "implementation"	Deployment = installing the technical system. Implementation = the wider change: training staff, changing SOPs, managing the human side. You need both.
"Monitoring" vs "maintenance"	Monitoring = watching and measuring. Maintenance = the fixes you do because monitoring found something. Monitoring comes first.
"The lifecycle ends at deployment"	No — deployment is stage 6 of 7. Monitoring continues for the system's whole life.
"More data always fixes a bad model"	Not if the data is biased or the problem was framed wrongly. Fix the right stage.

16. Common mistakes

1. **Jumping from idea to deployment** — skipping validation; the commonest and most dangerous shortcut.
2. **Treating the model as a finished product** — "we built it, our work is done"; ignoring monitoring until performance collapses.
3. **No pharmacist/clinician in the team** — the model learns medically meaningless patterns nobody catches.
4. **Testing on training data** — gives falsely excellent scores; the model then fails on real patients (Chapter 3 explains why).
5. **Shadow mode skipped** — going straight to live use; the first live failure happens in front of patients.
6. **No SOP for disagreement** — when the tool says "high risk" and the pharmacist disagrees, nobody knows who is responsible.
7. **Monitoring without thresholds** — watching numbers with no pre-decided "act now" level, so warnings are ignored.
8. **Documenting nothing** — cleaning decisions, settings, and validation details live in one person's head; when they leave, the system becomes unmaintainable.

17. Mnemonics

The seven stages — "Please Deliver Prescriptions Daily; Verify Dosage Methodically":

- **P**lease → **P**roblem framing
- **D**eliver → **D**ata collection
- **P**rescriptions → data **P**reparation
- **D**aily → model **D**evelopment
- **V**erify → **V**alidation
- **D**osage → **D**eployment
- **M**ethodically → **M**onitoring

Memory hook: A pharmacist's daily routine — please deliver prescriptions daily, and verify each dosage methodically — is exactly the AI lifecycle order.

Gates: "Green means go, amber means mend, red means end."

18. Exam summary

The AI system lifecycle is the safe, ordered journey of an AI tool from idea to retirement, in seven stages: (1) **problem framing** — converting a real need into a precise AI task with clear success criteria; (2) **data collection** — gathering historical examples with labels, permissions, and privacy protection; (3) **data preparation** — cleaning and standardising raw data, with every decision documented; (4) **model development** — training and tuning the algorithm so it learns real patterns, sense-checked by pharmacists; (5) **validation** — honest testing on unseen data, including fairness across patient groups, followed by a governance gate; (6) **deployment** — fitting the tool into the real workflow with staff training, SOPs, and preferably a shadow-mode trial, keeping the pharmacist as final decision-maker; (7)

monitoring — continuous watching of performance, data, and safety, with pre-decided thresholds that trigger investigation, retraining, or withdrawal. Outputs of each stage feed the next (relay race); monitoring loops back to retraining, making it a cycle. Errors are cheapest to fix early and most expensive after harm. In pharmacy, this lifecycle mirrors drug development and pharmacovigilance — and it is how tools like ADR predictors, interaction checkers, and dose-adjustment supports are built safely, including in Indian hospitals working with ABDM digitisation and mixed paper-digital records.

19. One-minute revision

- AI system = model + data + software + users + rules.
- 7 stages: **P**roblem framing → **D**ata collection → data **P**reparation → model **D**evelopment → **V**alidation → **D**eployment → **M**onitoring ("Please Deliver Prescriptions Daily; Verify Dosage Methodically").
- Each stage hands its output to the next; gates (green/amber/red) sit between stages.
- Monitoring loops back to retraining — it is a cycle, not a straight line.
- Fix errors early: cheap now, costly after harm.
- AI advises; the pharmacist decides. Always.

20. MCQs

Q1. Which of the following best describes an "AI system"? A) Only the trained model file B) The model plus data, software, users, and operating rules C) The computer server it runs on D) The training algorithm alone

Q2. What is the correct order of the AI system lifecycle? A) Deployment → validation → monitoring → training B) Problem framing → data collection → data preparation → model development → validation → deployment → monitoring C) Data collection → deployment → validation → monitoring D) Model development → problem framing → deployment → monitoring

Q3. The main purpose of the "problem framing" stage is to: A) Collect as much data as possible B) Convert a real-world need into a precise, solvable AI task with clear success criteria C) Install the software in the hospital D) Train the model

Q4. A "governance gate" in the lifecycle is: A) A firewall protecting patient data B) A checkpoint review between stages that decides go / fix / stop C) The hospital's main entrance security D) A type of machine-learning algorithm

Q5. Why is "shadow mode" recommended before full deployment? A) It makes the software run faster B) It lets the tool run silently in the live environment so behaviour can be confirmed before anyone acts on its outputs C) It reduces the cost of computers D) It is required for all mobile apps

Q6. Monitoring continues after deployment mainly because: A) The developers need employment B) Real-world data and conditions change over time, so performance can silently decay C) Computers slow down with age D) Regulators enjoy paperwork

Q7. In pharmacy deployment, the "human-in-the-loop" principle means: A) The AI works only when a human watches the screen B) The AI supports the decision, but the pharmacist/doctor makes the final clinical decision C) Humans must type all data manually D) The AI and human take turns weekly

Q8. Which stage typically consumes the most human effort in a real AI project? A) Model development B) Data preparation (cleaning and standardising data) C) Writing the project title D) Deployment

Q9. If monitoring shows falling performance because the hospital opened a new oncology unit with unfamiliar prescribing patterns, the correct lifecycle response is: A) Switch off monitoring B) Investigate, then retrain with new data, re-validate, and redeploy a new version C) Delete the old model immediately without investigation D) Blame the oncology staff

Q10. "The model can only be as good as its examples" best supports which statement? A) Data collection quality limits everything downstream B) Models do not need data C) Validation is unnecessary with big data D) Deployment fixes data problems

Answer key: Q1–B, Q2–B, Q3–B, Q4–B, Q5–B, Q6–B, Q7–B, Q8–B, Q9–B, Q10–A

21. Viva/application questions

1 Define "AI system" and explain how it differs from a "model".

2 List the seven lifecycle stages in order, with one line on each.

3 Why is problem framing called "the cheapest stage to fix mistakes in"?

4 Your hospital wants an AI tool to predict which prescriptions need pharmacist review first. Frame this as a precise AI task (who, what decision, what success looks like).

5 Explain the relay-race idea: what does each stage hand over to the next?

6 What is a governance gate, and name the two most important gates in the lifecycle.

7 A colleague says "our model scored 99% — deploy it tomorrow". Using lifecycle thinking, what would you advise? (Hint: ask what data the 99% came from.)

8 Describe shadow mode and why it is safer than direct live deployment.

9 Monitoring shows the ADR tool's accuracy falling after a new ward opened. Walk through the correct response using the feedback-loop mechanism.

10 How does the AI lifecycle resemble the journey of a medicine from manufacture to pharmacovigilance?

22. References

- WHO guidance on ethics and governance of artificial intelligence for health — as per the latest guidance, confirm the current edition.
- EU AI Act — as per the latest guidance, confirm the current edition (risk-based approach to AI systems).
- CDSCO (Central Drugs Standard Control Organization), India — guidance on software/medical devices — as per the latest guidance, confirm the current edition.
- Ayushman Bharat Digital Mission (ABDM) — public information on health-record digitisation in India.
- Pharmacovigilance Programme of India (PvPI) — public information on ADR reporting in India.
- Standard B.Pharm computer applications / AI-in-pharmacy study material and your university syllabus for BP801T.

CHAPTER 2: Data Quality in Healthcare Datasets

1. Learning objectives

After studying this chapter, you will be able to:

- Define "data quality" and list its five dimensions: accuracy, completeness, consistency, timeliness, and validity.
- Explain each dimension in simple words with a pharmacy example.
- Describe the three big data problems: missing data, biased sampling, and label errors.
- Explain EHR quirks: free-text notes and coding variability.
- Show, with a worked toy example, how missing data can change a result.
- Discuss India-specific challenges: ABDM digitisation, handwriting, and legacy paper records.
- Explain why data quality work belongs mainly to lifecycle stages 2 and 3.

2. Definitions

Data quality. How fit a dataset is for its intended use. High-quality data is correct, complete, uniform, up to date, and in the right format. Low-quality data misleads the model — "garbage in, garbage out".

Dataset. A structured collection of data used in an AI project — usually a table where each row is one case (e.g., one patient admission) and each column is one feature (e.g., age, drug, dose).

Accuracy (data-quality dimension). How close the recorded values are to the true values. *Example:* the recorded dose matches the dose actually prescribed.

Completeness (data-quality dimension). Whether all expected values are present — no missing entries. *Example:* every admission record has a recorded age.

Consistency (data-quality dimension). Whether the same thing is recorded the same way everywhere. *Example:* the drug is always written as "Paracetamol", not sometimes "PCM" or "Crocin".

Timeliness (data-quality dimension). Whether the data is recent enough for the purpose. *Example:* using last month's prescribing data, not data from ten years ago, to predict today's ADR risk.

Validity (data-quality dimension). Whether values follow the expected rules and formats. *Example:* age is a number between 0 and 120, not "abc" or 250.

Missing data. Empty cells in the dataset — information that should exist but was never recorded.

Biased sampling. When the collected examples do not fairly represent the real population the tool will serve — e.g., data from only young patients used to build a tool for all ages.

Label error. When the "correct answer" attached to a training example is itself wrong — e.g., a patient who did have an ADR is labelled "no ADR".

EHR (Electronic Health Record). The digital version of a patient's medical record in a hospital computer system — admissions, prescriptions, lab reports, and notes.

Free-text notes. Doctor/nurse notes typed as ordinary sentences ("patient complained of dizziness after tab..."), not in fixed fields. Computers find these hard to analyse directly.

Coding variability. The same medical concept recorded under different codes or terms by different people or hospitals — e.g., one hospital codes "type 2 diabetes" one way, another hospital a different way.

ABDM (Ayushman Bharat Digital Mission). India's national programme to digitise health records and link them through health IDs. (Public programme; details as per the latest guidance.)

De-identification. Removing names, addresses, phone numbers, and ID numbers from data so individual patients cannot be recognised, while keeping the medical information useful.

3. Terminology table

Term	Simple meaning	Pharmacy example
Data quality	Fitness of data for its purpose	ADR dataset with correct, complete, uniform records
Accuracy	Recorded value = true value	Dose recorded as actually prescribed
Completeness	No missing entries	All 1,000 records have age filled
Consistency	Same format/term everywhere	"Metformin" spelled identically in all rows
Timeliness	Data is recent enough	Using this year's data for this year's tool
Validity	Values follow allowed rules	Age is 0–120, a number, not text
Missing data	Empty cells	Kidney value not recorded for 80 patients
Biased sampling	Examples don't represent everyone	Training only on male patients
Label error	Wrong "correct answer"	Missed ADR marked as "no ADR"
Free-text notes	Ordinary typed sentences	"Pt felt dizzy after medicine" in a notes field
Coding variability	Same disease, different codes	Diabetes coded differently in two hospitals
Garbage in, garbage out	Bad data → bad model, always	Messy records → unreliable ADR predictions

4. Explanation

4.1 The five dimensions of data quality

Technical concept: Data quality is measured along five standard dimensions — accuracy, completeness, consistency, timeliness, validity.

Simple explanation: Think of data quality like the quality of raw materials for a medicine. You check five things:

1. **Accuracy — "Is it true?"** The value in the computer matches reality. If the prescription said 500 mg but someone typed 50 mg, accuracy is broken. An inaccurate dose teaches the model a lie.
2. **Completeness — "Is anything missing?"** Every expected cell is filled. If 30% of patients have no recorded weight, any dose-per-kg calculation is built on guesses.

3 **Consistency — "Is it written the same way everywhere?"** One drug, one spelling, one unit. "Paracetamol 500 mg", "PCM 0.5 g", and "Crocin" in the same dataset look like three different drugs to a computer.

4 **Timeliness — "Is it fresh enough?"** Prescribing patterns change (new antibiotics, new guidelines). Training today's tool on decade-old data is like navigating with an old map.

5 **Validity — "Does it follow the rules?"** Each column has allowed values: age must be a number 0–120; sex must be from a fixed list. "Age = banana" is invalid and must be caught.

Example: A hospital's ADR dataset scores: accuracy 95% (a few typing errors), completeness 70% (many missing lab values), consistency 60% (three spellings per drug), timeliness good (last 2 years), validity 90%. The team knows exactly where to spend cleaning effort — consistency and completeness (stage 3 of the lifecycle).

> **Key takeaway:** *Check all five dimensions separately. A dataset can be accurate but incomplete, or complete but inconsistent — each dimension fails in its own way.*

4.2 Missing data

Technical concept: Missing data means expected values are absent. Common patterns: missing completely by chance, missing because of how the system works, or missing because of the patient's condition.

Simple explanation: Empty cells. Sometimes a nurse forgot to type the weight; sometimes the lab machine was broken that day; sometimes a test was never ordered because the patient looked healthy. The *reason* for missingness matters: if kidney tests are missing mostly for healthy-looking patients, the recorded kidney values come mostly from sicker patients — and the model learns a distorted picture.

Example: In our toy ADR dataset, kidney values are missing for 200 of 1,000 patients — mostly younger patients for whom doctors skipped the test. If the team simply deletes those 200 rows, the model trains only on older/sicker patients and then performs badly on young patients. Better: handle missingness carefully and document the choice (Chapter 1, stage 3).

> **Key takeaway:** *Never silently delete or silently fill missing data. Ask why it is missing first — the reason itself carries information.*

4.3 Biased sampling

Technical concept: Biased sampling occurs when the training examples systematically over- or under-represent some group compared with the population the system will serve.

Simple explanation: If you train a drug-safety tool only on data from one big city hospital, it has never "seen" rural patients, tribal populations, or the elderly in sufficient numbers. When deployed widely, it performs worse for the unseen groups — not because of malice, but because of absence. The model is like a student who only ever solved one type of exam question.

Example: A diabetes-risk model trained on data from a corporate hospital (mostly insured, urban patients) is deployed in a district hospital. It underperforms because diets, co-morbidities, and even the age distribution differ. The sampling was biased toward urban insured patients.

Key takeaway: *Always ask: "Who is missing from this dataset?" The absent groups are exactly where the model will fail.*

4.4 Label errors

Technical concept: A label error is an incorrect ground-truth annotation on a training example, causing the algorithm to learn the wrong lesson.

Simple explanation: Labels are the "answer key". If the answer key itself has wrong answers, the student learns wrong things confidently. In ADR data, label errors happen when ADRs are under-reported (a real ADR labelled "no ADR") or when a symptom is wrongly blamed on a drug (a non-ADR labelled "ADR").

Example: Ward staff report only serious ADRs; mild rashes and nausea often go unrecorded. So the dataset labels many mild-ADR patients as "no ADR". The model then learns that the drug patterns causing mild ADRs are "safe" — a dangerous lesson drawn from wrong labels.

Key takeaway: *Labels deserve the same quality checking as features. In healthcare, labels usually come from busy humans — verify a sample of them by expert review.*

4.5 EHR quirks: free-text notes and coding variability

Technical concept: Electronic Health Records mix structured fields (drop-downs, numbers) with unstructured free text, and different sites code the same concept differently.

Simple explanation: Hospital computers are not as tidy as we imagine. Doctors type quick notes like "pt c/o dizziness ? drug induced" — full of abbreviations, spelling shortcuts, and mixed languages. This is *free text*: rich in meaning, hard for computers. Meanwhile, structured coding varies: Hospital A records diabetes with one code, Hospital B with another (*coding variability*). Merge the two hospitals' data without harmonising codes, and the computer thinks they treat different diseases.

Example: An ADR model needs "dizziness" as a feature. In structured fields it appears rarely; in free-text notes it appears as "dizzy", "dizziness", "chakkar" (Hindi), "giddiness". Unless the team handles these variants in data preparation, the feature is half-missing — a completeness problem wearing a disguise.

Key takeaway: *In healthcare data, the most valuable information often hides in messy text and inconsistent codes. Plan for it in stage 3; do not pretend the data is tidy.*

4.6 Who fixes data quality? (a team effort)

Technical concept: Data-quality work is split between automated checks and human domain judgement.

Simple explanation: Computers are good at *finding* quality problems (counting missing values, listing odd spellings, flagging out-of-range numbers) but poor at *judging* them. Is "chakkar" the same as "dizziness"? Is an unusual dose a typing error or a genuinely sick patient needing a high dose? Software cannot decide — only a clinician or pharmacist can. So the work divides naturally: the data team runs the profiling scans; pharmacists and doctors adjudicate the ambiguous cases; the quality cell verifies the fixes. Nobody can do it alone, and the pharmacist's clinical judgement is the scarcest, most valuable input.

Example: The profiling scan flags 300 records where the dose looks unusual. The data team cannot tell which are errors. A pharmacist reviews a sample: 240 are legitimate high doses for severe infections, 60

are decimal-point typing errors ("5.0 mg" typed as "50 mg"). Only domain knowledge separates the two — a pure software fix would have "corrected" real prescriptions into wrong ones.

```
COMPUTER FINDS            HUMAN JUDGES              TEAM RECORDS
scan: 12% missing    ──►  pharmacist: why?     ──►  cleaning log
scan: 3 spellings    ──►  pharmacist: same?    ──►  mapping table
scan: odd doses      ──►  doctor: real/error?  ──►  fix + note
```

Key takeaway: *Automate the finding, humanise the judging, document everything. Data quality is a team sport, and the pharmacist is the referee.*

5. Classifications

5.1 Types of missing data (by reason)

Type	Meaning	Pharmacy example
Missing by chance	Random slip — anyone could be missing it	Nurse forgot to type weight for a few random patients
Missing by system	The process never captures it	Old paper registers had no column for allergy history
Missing by condition	Absent *because of* the patient's state	Kidney test skipped because the patient looked young and healthy — the healthiest patients are the missing ones

5.2 Common responses to missing data

Response	What it means	Risk
Delete the row	Drop patients with any missing value	Loses data; can create bias (deletes the healthiest)
Fill with average/median	Put a typical value in the empty cell	Hides the fact it was missing; understates uncertainty
Fill with a sensible rule	Use domain logic (e.g., carry forward last value)	Better, but the rule must be documented
Flag as missing	Add a column "kidney value was missing: yes/no"	Lets the model learn that missingness itself is informative

5.3 Sources of label error in pharmacy data

1. **Under-reporting** — mild ADRs never entered in the register.
2. **Misattribution** — a symptom blamed on the wrong drug, or on the drug when the disease caused it.
3. **Copy-paste errors** — "no known allergy" carried forward into every record by default.
4. **Coder disagreement** — two pharmacists judge the same case differently.

5.4 Which checks are automated vs manual

Check	Automated scan finds it	Human must judge it
Missing values per column	✓ counts the empty cells	Why missing? Delete, fill, or flag?
Unique spellings per drug	✓ lists the variants	Which variants mean the same drug?
Out-of-range numbers	✓ flags age = 250	Typing error or a genuinely extreme case?
Duplicate rows	✓ finds identical rows	True duplicates or two similar admissions?
Label correctness	✗ cannot know the truth	✓ pharmacist reviews a sample
Representativeness	~ partial (counts per group)	✓ "who is missing, and does it matter?"

Teaching note: automation scales the finding; only domain experts do the judging.

6. Mechanisms

6.1 How poor data quality propagates through the lifecycle

```
Low accuracy      ⟶ model learns wrong facts      ⟶ wrong predictions
Low completeness  ⟶ model blind to some features  ⟶ fails on those patients
Low consistency   ⟶ one drug looks like three     ⟶ patterns split and weaken
Low timeliness    ⟶ model learns old practice     ⟶ model drift (Chapter 3)
Low validity      ⟶ nonsense values enter maths   ⟶ crashes or absurd outputs
Biased sampling   ⟶ some groups under-learnt      ⟶ unfair performance gaps
Label errors      ⟶ wrong lessons learnt deeply   ⟶ confident wrong predictions
```

Mechanism in simple words: Data quality problems do not stay in the data — they are *converted* into model behaviour problems. Each defect has a signature downstream harm, which is why stages 2–3 (collection, preparation) are the safety-critical stages for data.

6.2 The data-quality audit loop (mechanism)

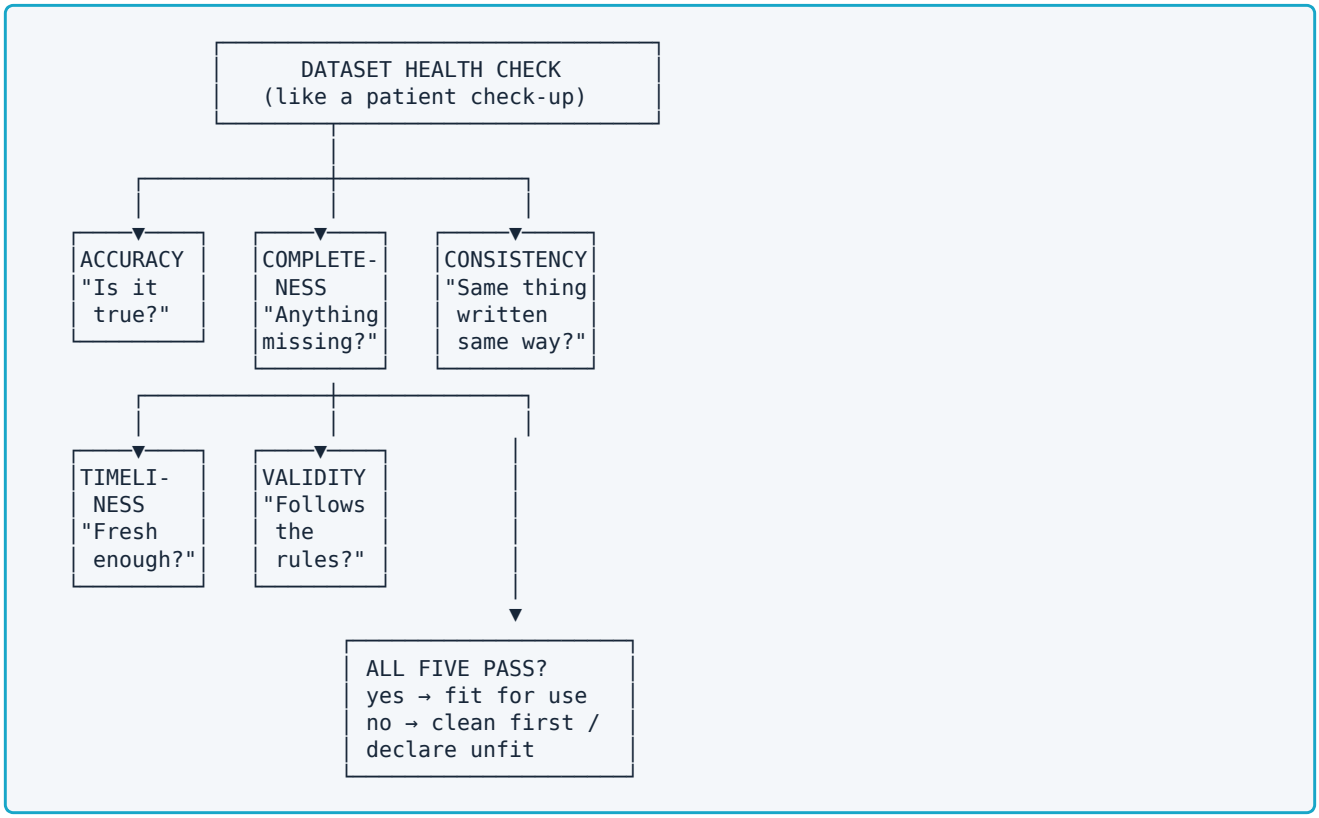
1. **Profile** — automatically scan the dataset: count missing values per column, list unique spellings, check ranges.
2. **Score** — rate each of the five dimensions (e.g., completeness = 72%).
3. **Prioritise** — fix what matters most for the task first (a missing "drug name" matters more than a missing "bed number").
4. **Clean and document** — apply fixes; write every decision in the cleaning log.
5. **Re-profile** — confirm the scores improved; repeat until acceptable or the data is declared unfit.

6.3 How small label errors compound (mechanism)

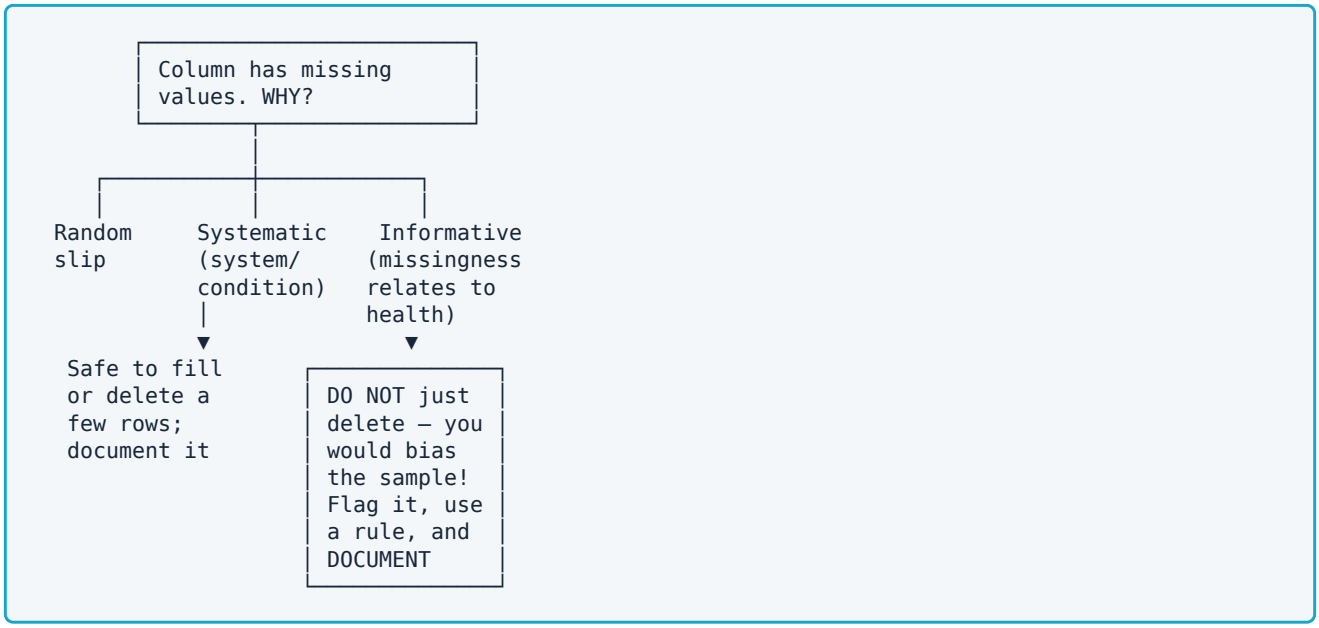
A 5% label-error rate does not cause "5% worse performance" — it can cause much more. Mechanism: the model treats labels as truth, so wrong labels actively *teach wrong patterns*; the model then becomes *confidently* wrong on exactly the cases that matter (e.g., mild ADRs systematically labelled "no ADR" teaches "these drug patterns are safe"). Worse, validation scores computed against the same faulty labels look reassuringly high — the error hides inside the measuring stick itself. This is why this chapter insists on expert verification of a label *sample*: you are checking the ruler, not just the measurement.

7. Diagrams/flowcharts

7.1 The five dimensions — "the data health check-up"



7.2 Missing-data decision flowchart



8. Infographic concepts

Concept 1 — "Five-star data hotel". A hotel with five stars labelled Accuracy, Completeness, Consistency, Timeliness, Validity; each star lights up or stays dark for a sample dataset. Tagline: "Would you admit your model to a 2-star hotel?"

Concept 2 — "The leaky bucket". A bucket (the dataset) with holes labelled "missing values", "wrong labels", "mixed spellings", "old data"; water (useful information) leaking out; a pharmacist patching holes with plasters labelled "cleaning log", "standardise", "expert label review". Caption: "Patch the bucket before you carry the water."

Concept 3 — "Lost in translation" (EHR quirks). A doctor's note shown three ways: handwritten ("Pt c/o chakkar"), typed free text ("pt felt dizzy ?drug"), and structured code — with arrows showing how each must be harmonised into one clean feature, and a warning stamp where information is lost.

9. Tables

9.1 Five dimensions at a glance

Dimension	Question	Pharmacy example of failure	Fix (stage 3)
Accuracy	Is it true?	Dose typed as 50 mg instead of 500 mg	Double-entry check; range alerts; sample verification
Completeness	Anything missing?	Weight missing for 30% of patients	Investigate why; fill by documented rule or flag
Consistency	Same format everywhere?	"Paracetamol"/"PCM"/"Crocin" mixed	Map all variants to generic name
Timeliness	Fresh enough?	10-year-old prescribing data	Collect recent data; refresh regularly
Validity	Follows rules?	Age = "abc" or 250	Reject/flag invalid entries; fix at entry

9.2 Data problems → model harms

Data problem	What the model learns	Patient harm
Missing kidney values (deleted rows)	Trains mostly on sicker patients	Poor predictions for healthier/younger patients
Under-reported mild ADRs (label error)	"These drug patterns are safe"	Misses real ADR risk
Biased sampling (one hospital)	One population's patterns only	Unfair, worse performance elsewhere
Inconsistent drug names	Splits one drug into three weak patterns	Misses interactions and risks
Old data (poor timeliness)	Outdated prescribing patterns	Irrelevant or wrong advice today

9.3 Before/after data preparation — toy ADR dataset profile (illustrative)

Dimension (toy)	Before cleaning	After cleaning
Completeness	72% (kidney values missing in 28%)	100% accounted for (filled by documented rule + missingness flags)
Consistency	3 spellings per drug (average)	1 generic name per drug
Validity	14 impossible values (age 250 etc.)	0 (flagged, corrected/removed with notes)
Accuracy (sample-verified)	~90%	~97% (after fixing sampled errors)
Timeliness	Mixed 2015–2025	Restricted to 2021–2025, documented

Lesson: profiling before and after proves the cleaning worked — and the "after" profile belongs in the datasheet (Chapter 4).

10. Worked toy examples

ILLUSTRATIVE TOY DATASET (not real patient data). The numbers below are invented purely to show the arithmetic of how missing data changes a result.

Toy example: how missing data changes an ADR rate.

A tiny ward dataset of 10 patients (toy). We want a simple result: "What fraction of patients on Drug X had an ADR?"

Full (imaginary complete) data:

Patient	Drug X?	ADR?
1	Yes	Yes
2	Yes	Yes
3	Yes	No
4	Yes	No
5	Yes	No
6	No	No
7	No	No
8	No	No
9	No	Yes
10	No	No

True toy ADR rate for Drug X = 2 out of 5 = **40%**.

Now suppose the ADR column is **missing** for patients 1 and 2 (imagine their ADR forms were lost — and note these were the two ADR cases). A careless analyst deletes rows with missing values and recomputes:

Remaining Drug X patients: 3, 4, 5 → ADR cases among them: 0 → computed rate = 0 out of 3 = **0%**.

The result flipped from 40% to 0% — purely because of *which* rows were missing. The missingness was informative (the lost forms were the ADR cases), and silent deletion produced a dangerously reassuring lie: "Drug X looks perfectly safe."

Teaching point: This is why section 4.2 insists: ask *why* data is missing before you delete or fill. With informative missingness, deletion does not just lose information — it manufactures a false conclusion.

Second toy illustration — inconsistent naming splits a pattern:

Toy counts of "paracetamol" prescriptions: "Paracetamol" 40, "PCM" 25, "Crocin" 15. A computer sees three separate drugs with 40, 25, 15 examples each instead of one drug with 80 examples. A rare interaction visible only at 80 examples stays invisible in all three fragments. Standardising to one generic name in stage 3 reunites the pattern.

Third toy illustration — biased sampling in numbers:

Toy hospital data: 900 urban patients, 100 rural patients (toy). Toy ADR rates: urban 8%, rural 16% (rural patients arrive later and sicker — toy assumption). A model trained on this mix learns mostly urban patterns, because rural patients were only 10% of its "teachers". Deployed in a district hospital serving mostly rural patients, its errors concentrate on rural patients — not because the algorithm is prejudiced, but because it barely studied them. Had the team asked "who is missing?" during collection, they could have deliberately oversampled rural records — or at minimum documented the limitation in the datasheet (Chapter 4) and restricted the tool's intended use until fixed.

11. Analogies

1. **Cooking with dirty vegetables.** Missing data = vegetables never bought; inaccurate data = rotten ones mixed in; inconsistent data = the same vegetable labelled with three names so the cook uses triple the quantity. No cooking skill (algorithm) fixes bad ingredients.
2. **Answer key with wrong answers.** Label errors are exactly this — the student (model) memorises the wrong answers and fails the real exam confidently.
3. **Survey of only one classroom.** Biased sampling = judging the whole school's height by measuring only Class 5. The conclusion will be wrong for other classes.
4. **Old map.** Poor timeliness = navigating a city with a 10-year-old map; new roads (new drugs, new guidelines) are missing and demolished ones still shown.
5. **Hospital case sheet.** A pharmacist already audits prescriptions for legibility, completeness, and correctness — data-quality profiling is the same audit applied to thousands of records at once.

12. Pharmacy relevance

- **Prescription audit skills transfer.** Pharmacists already check prescriptions for missing dose, wrong units, illegible names — the five data-quality dimensions are the same checklist scaled up to datasets.
- **ADR register quality decides model safety.** India's PvPI ADR data is only as useful as its completeness; under-reporting directly becomes label error in any AI trained on it.
- **Drug-name standardisation is a pharmacist's natural job.** Mapping brand names to generics, unifying strengths and units — nobody does this better than a pharmacist in the data-preparation stage.
- **Free-text mining needs domain knowledge.** Deciding whether "chakkar", "dizzy", and "giddiness" mean the same symptom for the model requires clinical judgement, not just software.
- **Ethical duty.** Using poor-quality data to build a tool that guides treatment of real patients is, plainly, unsafe practice. Data quality is a patient-safety issue, not a technical nicety.
- **You are the label-maker.** In most Indian hospitals, ADR labels, allergy records, and diagnosis codes are created by pharmacists and doctors during routine work. Writing them carefully *today* is literally manufacturing the training data of *tomorrow's* AI. Every complete, accurate case sheet is a brick in a future safe model.

13. India context

- **ABDM digitisation is underway but uneven.** As per the latest public information on ABDM, health IDs and digital records are expanding — but many district hospitals, PHCs, and private clinics still run on paper registers. Datasets built today will mix digital EHR rows with digitised paper rows of lower completeness and consistency.
- **Handwriting and legacy records.** Decades of handwritten case sheets exist. When digitised (by typing or scanning), handwriting misreads become accuracy errors ("mg" read as "mcg" is a 1000-fold error). Any AI project using legacy data must budget for expert verification of samples.
- **Multilingual free text.** Notes mix English with Hindi/regional terms ("chakkar", "bukhar"). Data-preparation pipelines built only for English will silently lose this information — a completeness and fairness issue for non-English-speaking patients.
- **Coding variability across states and systems.** Different hospitals use different drug formularies, brand names, and local codes. Merging data across sites without harmonisation creates false patterns.
- **Practical advice for Indian B.Pharm students.** When you one day evaluate an AI tool, first ask: "Which hospitals' data trained this? Were district/rural patients included? Was the data mostly English?" The answers reveal the sampling bias before you see any performance number.
- **What you can practise today.** You do not need a computer to build data-quality sense. Take 20 prescriptions during your hospital visit and audit them on the five dimensions: is the dose accurate and legible, is anything missing (age? weight?), are drug names consistent, is the prescription current, are values valid? That exercise is exactly what data profiling feels like at scale — and it makes you the domain expert every AI team needs.

14. Key points

- Data quality = fitness for purpose, judged on **five dimensions**: accuracy, completeness, consistency, timeliness, validity.
- The three big problems: **missing data**, **biased sampling**, **label errors** — each harms the model differently.
- Always ask *why* data is missing before deleting or filling; informative missingness makes deletion dangerous.
- "Who is missing from this dataset?" is the single most important bias question.
- Labels are made by busy humans — verify a sample by expert review.
- EHR quirks (free-text notes, coding variability) hide valuable information in messy form; plan for them in data preparation.
- Data-quality work lives in lifecycle stages 2–3, and defects there become model-behaviour defects later.
- **Garbage in, garbage out** — no algorithm rescues bad data.

15. Common confusions

Confusion	Clarification
"Accuracy" (data) vs "accuracy" (model)	Data accuracy = recorded values match reality. Model accuracy = predictions match reality. Related but different — you need both.
"Complete" vs "accurate"	A dataset can be 100% complete (no missing cells) yet full of wrong values — complete but inaccurate. Check each dimension separately.
"Deleting missing rows is the clean solution"	It is the *easy* solution, and often the *wrong* one — it can bias the sample, as the toy example showed.
"More data beats better data"	A million biased rows teach bias very confidently. Quality first, then quantity.
"Free text is useless for AI"	It is hard to use, not useless — it often holds the richest clinical detail. It needs NLP handling in stage 3.
"De-identification removes all usefulness"	Done well, it removes identity while keeping medical patterns. Done carelessly (e.g., deleting all dates), it can destroy timeliness information.

16. Common mistakes

1. **Silent deletion of missing rows** — no note of how many rows were dropped or why; bias sneaks in unnoticed.
2. **Filling missing values with the average and forgetting it** — the dataset then pretends to be more complete and certain than it is.
3. **Ignoring *why* data is missing** — treating informative missingness as random.
4. **Merging multi-hospital data without harmonising codes and drug names** — one concept becomes many fragments.
5. **Trusting labels blindly** — never having a pharmacist verify a sample of ADR labels.
6. **Using decade-old data for today's tool** — timeliness failure; prescribing practice has moved on.
7. **No cleaning log** — fixes applied but not written down; nobody can reproduce or audit the dataset (links to Chapter 4).
8. **Judging data by size alone** — "we have 2 million rows" with no quality dimensions reported.

17. Mnemonics

Five dimensions — "A Cat Can Talk Vividly":

- **A**ccuracy — *is it true?*
- **C**ompleteness — *anything missing?*
- **C**onsistency — *same way everywhere?*
- **T**alk = **T**imeliness — *fresh enough?*
- **V**ividly = **V**alidity — *follows the rules?*

Memory hook: Picture a cat inspecting your dataset like a drug inspector, talking vividly about each quality dimension.

Three big problems — "MiB-L": "Missing, Biased, Labels" — the three villains of data quality.

Garbage in, garbage out → "GIGO": say it as "guy-go" — bad data guy goes in, bad predictions go out.

18. Exam summary

Data quality is the fitness of a dataset for its purpose, measured on five dimensions: **accuracy** (values match reality), **completeness** (no missing entries), **consistency** (uniform formats/terms), **timeliness** (data recent enough), and **validity** (values follow allowed rules). Three major problems threaten it: **missing data** (empty cells — always ask *why* before deleting or filling, since informative missingness makes deletion manufacture false conclusions, as the toy ADR-rate example showed: 40% true rate became 0% after careless deletion), **biased sampling** (examples not representing the served population — always ask "who is missing?"), and **label errors** (wrong "correct answers", e.g., under-reported ADRs teaching the model that risky patterns are safe). Healthcare data adds **EHR quirks**: free-text notes (rich but messy, often multilingual) and coding variability (same concept, different codes across sites). Quality is profiled, scored, fixed, and documented in lifecycle stages 2–3 via the audit loop (profile → score → prioritise → clean + document → re-profile). In India, ABDM digitisation is expanding but records remain mixed paper-digital, multilingual, and uneven across settings — so sampling-bias and consistency questions are essential before trusting any tool. The governing law: **garbage in, garbage out**.

19. One-minute revision

- 5 dimensions: **A**ccuracy, **C**ompleteness, **C**onsistency, **T**imeliness, **V**alidity ("A Cat Can Talk Vividly").
- 3 villains: **Mi**ssing data, **B**iased sampling, **L**abel errors.
- Missing? Ask WHY first — deletion can flip results (toy: 40% → 0%).
- Bias? Ask WHO IS MISSING?
- Labels? Verify a sample — humans make them.
- EHR quirks: free text + coding variability → harmonise in stage 3.
- GIGO: no algorithm rescues bad data.

20. MCQs

Q1. Which data-quality dimension asks "is the recorded value true?" A) Completeness B) Accuracy C) Timeliness D) Validity

Q2. "Paracetamol", "PCM", and "Crocin" appearing for the same drug in one dataset is a failure of: A) Accuracy B) Completeness C) Consistency D) Timeliness

Q3. In the toy ADR example, deleting rows with missing ADR values changed the computed ADR rate from 40% to 0% because: A) The computer made an arithmetic error B) The missingness was informative — the lost records were the ADR cases C) 40% is always wrong D) Drug X is perfectly safe

Q4. Biased sampling means: A) The dataset is too large B) The examples systematically over- or under-represent some groups versus the served population C) The labels are wrong D) The data is too old

Q5. A patient who suffered an ADR but is recorded as "no ADR" is an example of: A) Missing data B) Label error C) Coding variability D) Poor timeliness

Q6. Free-text clinical notes are challenging for AI mainly because: A) They contain too little information B) They are unstructured — abbreviations, mixed languages, and varied phrasing C) Doctors never write notes D) Computers cannot store text

Q7. Coding variability refers to: A) Different computer programming languages B) The same medical concept recorded under different codes/terms across people or hospitals C) Secret codes in prescriptions D) Variable drug prices

Q8. The correct first step when you find missing values in a column is to: A) Immediately delete all such rows B) Ask why the values are missing C) Fill them with zeros D) Ignore them

Q9. Which response to missing data lets the model learn that "missingness itself is informative"? A) Deleting the row B) Adding a flag column like "kidney value missing: yes/no" C) Filling with the average silently D) Sorting the table

Q10. "Garbage in, garbage out" (GIGO) means: A) Computers produce waste heat B) Poor-quality data inevitably produces unreliable models, regardless of algorithm C) Old computers should be discarded D) Data entry operators are at fault

Answer key: Q1–B, Q2–C, Q3–B, Q4–B, Q5–B, Q6–B, Q7–B, Q8–B, Q9–B, Q10–B

21. Viva/application questions

1 Define data quality and name its five dimensions, with one pharmacy example for each.

2 Explain the difference between data accuracy and model accuracy.

3 Reproduce the toy ADR-rate example: how did careless deletion change 40% to 0%? What was the root cause?

4 Give two reasons why kidney-function values might be missing, and explain why the reason matters.

5 What is biased sampling? Give an India-relevant example (e.g., metro-hospital data used in a district hospital).

6 List three sources of label error in ADR datasets and suggest one fix for each.

7 Explain "free-text notes" and "coding variability" with examples a pharmacist would recognise.

8 A dataset has 2 million rows but ADR labels were never verified. Is it fit for training an ADR predictor? Justify using the five dimensions.

9 Describe the data-quality audit loop (profile → score → prioritise → clean + document → re-profile).

10 Why is drug-name standardisation (brand → generic) a natural pharmacist's task in data preparation?

11 A profiling scan flags 300 records with unusual doses. Who should judge whether they are errors or genuine cases, and why can't software alone decide?

12 Explain how biased sampling in data collection becomes a fairness failure discovered later in a model audit (link Chapters 2 and 5).

22. References

- WHO guidance on ethics and governance of artificial intelligence for health — as per the latest guidance, confirm the current edition (data quality and representativeness).
- Ayushman Bharat Digital Mission (ABDM) — public information on health-record digitisation in India.
- Pharmacovigilance Programme of India (PvPI) — public information on ADR reporting in India.
- EU AI Act — as per the latest guidance, confirm the current edition (data governance requirements for high-risk AI).
- Standard B.Pharm computer applications / AI-in-pharmacy study material and your university syllabus for BP801T.

CHAPTER 3: Train vs Test vs Validation, Cross-Validation, Model Drift, Data Leakage

1. Learning objectives

After studying this chapter, you will be able to:

- Define training set, validation set, and test set, and explain why each exists.
- Explain overfitting from absolute zero, with a pharmacy example.
- Describe k-fold cross-validation conceptually and work through a small toy example.
- Distinguish model drift from data drift (concept drift vs data drift) with a pharmacy example.
- Define data leakage and explain two concrete types: target leakage and train-test contamination.
- State how to avoid data leakage in practice.
- Explain where these ideas sit in the AI lifecycle (stages 4–5 and 7).

2. Definitions

Training set. The portion of the dataset the model learns from. The model sees these examples repeatedly during training and adjusts itself to fit them.

Validation set (tuning set). A separate portion of data, never used in training, used to check the model *during development* — for example, to choose between candidate models or adjust settings. Think of it as practice exams while studying.

Test set. A final portion of data, locked away and used only *once*, at the very end, to give an honest estimate of how the model will perform on completely new patients. Think of it as the final board exam.

Overfitting. When a model learns the training data *too* well — including its random quirks and noise — so it performs excellently on training data but poorly on new data. Like a student who memorises the textbook word-for-word but cannot answer a slightly rephrased question.

Underfitting. The opposite problem: the model learns too little, performing poorly even on the training data — like a student who barely opened the book.

Generalisation. The ability of a model to perform well on new, unseen cases. Good generalisation is the entire goal of machine learning.

Hyperparameter. A setting of the learning algorithm chosen by the human developer before training (for example, how strictly the model should avoid memorisation). Not learnt from data — set by hand, usually tuned using the validation set.

K-fold cross-validation. A technique that splits the data into K equal parts ("folds"), trains on K−1 parts and tests on the remaining part, and repeats this K times so every part serves as the test part once. The K scores are averaged for a more reliable performance estimate.

Model drift. A fall in the model's real-world performance over time because the world changed after training. Two flavours: *data drift* and *concept drift* (defined below).

Data drift. The incoming data slowly changes its character compared with the training data — e.g., patients get older on average, or a new drug enters the formulary. The model's learnt patterns still hold, but it sees unfamiliar inputs.

Concept drift. The *relationship* between inputs and the correct answer itself changes — e.g., a new guideline changes which drug combinations are considered risky. The old learnt rules are now partly wrong.

Data leakage. When information from outside the training set — especially information about the correct answer, or data from the test set — improperly "leaks" into training, making the model look better in testing than it really is. Leakage produces falsely excellent scores and real-world failure.

Target leakage. A type of data leakage where a feature used in training secretly contains information about the label (the answer) that would not be available at prediction time — e.g., using "antidote prescribed" to predict "ADR occurred".

Train-test contamination. A type of data leakage where test-set information contaminates training — e.g., the same patient's records appear in both sets, or cleaning statistics were computed on the whole dataset including the test set.

3. Terminology table

Term	Simple meaning	Pharmacy example
Training set	Examples the model studies	700 past admissions used for learning
Validation set	Practice-exam data for tuning	150 admissions to compare candidate models
Test set	Final unseen exam, used once	150 locked-away admissions for the final score
Overfitting	Memorised the textbook; fails new questions	99% on training, 60% on new patients
Generalisation	Doing well on new cases	Works on next year's patients too
K-fold cross-validation	Rotate which part is the test part; average scores	5-fold: five train/test rotations
Data drift	Input data slowly changes character	New oncology ward changes patient mix

Term	Simple meaning	Pharmacy example
Concept drift	The rules themselves change	New guideline redefines risky combinations
Target leakage	Feature secretly contains the answer	"Antidote given" used to predict "ADR happened"
Train-test contamination	Test info leaks into training	Same patient in both sets

4. Explanation

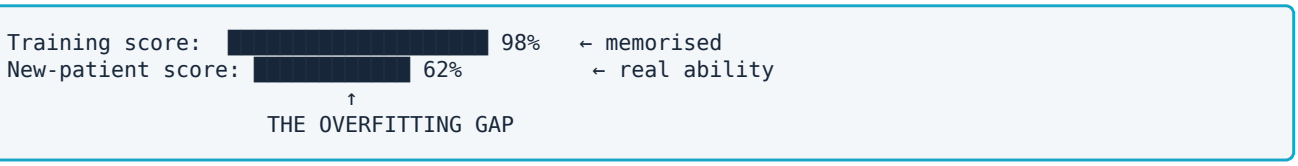
Sequence used below for each concept: **technical concept → simple explanation → example → diagram → key takeaway.**

4.1 Overfitting — the idea behind all the splits

Technical concept: Overfitting is the gap between training performance and real-world performance caused by the model fitting noise (random quirks) in the training data rather than true patterns.

Simple explanation: Imagine a student preparing for a pharmacology exam. Instead of understanding *why* beta-blockers lower heart rate, she memorises the exact 100 practice questions and their answers. On those 100 questions she scores 100%. But in the real exam, where questions are rephrased, she fails — she memorised, she did not understand. A model does exactly the same thing when it is too flexible and the training data too small: it memorises quirks ("patients in bed 12 had more ADRs") instead of learning real medical patterns.

Example: An ADR model scores 98% correct on its training data. The team celebrates — until validation on new patients gives 62%. Investigation: the model had learnt that "admission on a Tuesday" predicted ADRs (a chance quirk in the training sample). It memorised noise. This is overfitting.



Key takeaway: *A high training score proves nothing by itself. The only score that matters is on data the model has never seen — which is why we split data into train/validation/test.*

4.2 Training set — where learning happens

Technical concept: The training set is the subset of data used to fit the model's internal parameters.

Simple explanation: This is the textbook the student studies. The model reads these examples again and again, adjusting itself to reduce its mistakes on them. Typically the largest split (often around 60–80% of the data in toy illustrations — the exact ratio is a design choice, not a law).

Example: Of 1,000 toy ADR records, 700 form the training set. The model learns patterns like "elderly + polypharmacy + poor kidneys → higher ADR risk" from these 700.

Key takeaway: *Training data is for learning only — never for honest evaluation. Judging a student by the questions she practised on is not an exam.*

4.3 Validation set — the practice exams

Technical concept: The validation set is held-out data used during development to compare models and tune hyperparameters, without touching the test set.

Simple explanation: While studying, the student takes practice tests to check progress and adjust her study method. Crucially, practice tests are *not* the final board exam — if she takes the same practice test 50 times and keeps tweaking until she scores 100%, she has started memorising the practice test too. That is why we keep a *separate* final test set.

Example: The team trains three candidate ADR models on the 700 training records, then checks each on 150 validation records. Model B wins. They adjust one setting, re-check on the validation set — better. But they never touch the final 150 test records during all this tuning.

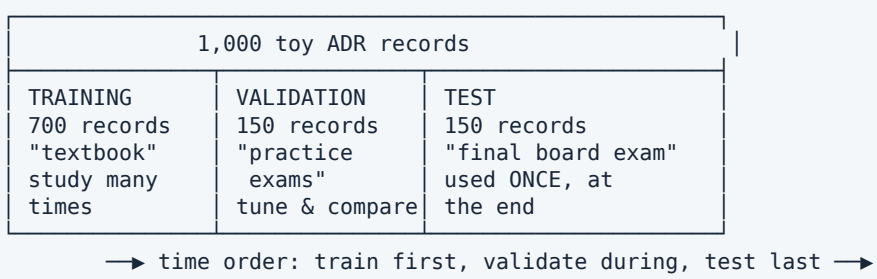
Key takeaway: *The validation set guides development choices. Because we peek at it repeatedly, its score becomes slightly optimistic — so we still need the untouched test set for the final verdict.*

4.4 Test set — the final board exam

Technical concept: The test set is held-out data used exactly once, after all development is finished, to estimate real-world performance.

Simple explanation: The final board exam: unseen questions, taken once, no retakes, no tweaking afterwards. If the model scores well here, we have genuine evidence it will work on future patients. If someone "adjusts the model and re-tests" repeatedly on the test set, it stops being a test set — it becomes another validation set, and the honesty is lost.

Example: After all tuning, the winning ADR model is run once on the locked-away 150 test records: it flags 8 out of 10 future ADR cases. This 8/10 is the honest number reported to the pharmacy committee and written in the validation report (lifecycle stage 5).



Key takeaway: *Test once, report honestly, never tune on it. The test set is the foundation of trustworthy validation.*

4.5 K-fold cross-validation — rotating the exam

Technical concept: K-fold cross-validation partitions data into K equal folds; each fold takes a turn as the validation portion while the model trains on the other K−1 folds; the K resulting scores are averaged.

Simple explanation: One practice exam might be accidentally easy or hard. So instead, divide the class notes into 5 equal bundles. Study bundles 1–4, test on bundle 5. Then study bundles 1,2,3,5 and test on bundle 4. Repeat until every bundle has been the test once. Average the five scores — this average is far more reliable than any single test, especially when data is limited.

Toy worked example (illustrative toy dataset): 100 toy patient records, 5-fold cross-validation (each fold = 20 records):

Round	Trains on folds	Tests on fold	Toy score
1	2,3,4,5	1	80%
2	1,3,4,5	2	76%
3	1,2,4,5	3	82%
4	1,2,3,5	4	78%
5	1,2,3,4	5	79%

Average = (80+76+82+78+79)/5 = **79%**. The team reports "about 79%, stable across folds" instead of trusting one lucky 82% or one unlucky 76%.

```
Fold:   [ 1  ][ 2  ][ 3  ][ 4  ][ 5  ]
Round1:  TEST  train train train train
Round2: train  TEST  train train train
Round3: train train  TEST  train train
Round4: train train train  TEST  train
Round5: train train train train  TEST
                  ↓
         Average the 5 scores → reliable estimate
```

Key takeaway: *Cross-validation squeezes reliable evidence out of limited data by rotating the test role. It is the standard honest practice when datasets are small — common in pharmacy.*

4.6 Model drift — when the world moves on

Technical concept: Model drift is performance decay over time because the live environment no longer matches the training environment — either the inputs changed (data drift) or the input→answer relationship changed (concept drift).

Simple explanation: A student who topped the 2020 syllabus may struggle with the 2026 syllabus — either the *questions* changed style (data drift) or the *correct answers* changed because guidelines were revised (concept drift). The student did not become duller; the world moved.

Example — data drift: The ADR model was trained when the hospital had no oncology unit. A new oncology ward opens; chemotherapy patients with very different drug patterns arrive. The model's old rules still hold in principle, but it keeps seeing unfamiliar inputs and its accuracy slides from 80% to 68%. *The data changed.*

Example — concept drift: A new national guideline declares that a previously "safe" drug combination is now high-risk in the elderly, and prescribing practice changes. The model's learnt rule "this combination = low risk" is now medically wrong. *The relationship between inputs and the correct answer changed.*

```
DATA DRIFT:              CONCEPT DRIFT:
inputs ⟶ changed        rules ⟶ changed
"new kinds of            "what counts as
 patients arriving"       correct changed"

Both ⟶ performance falls ⟶ monitoring (stage 7) must catch it
       ⟶ fix = retrain on new data + re-validate
```

Key takeaway: *Drift is inevitable — medicine changes. That is why lifecycle stage 7 (monitoring) exists, and why models need scheduled re-validation, like periodic calibration of a lab instrument.*

4.7 Data leakage — the silent score-inflater

Technical concept: Data leakage is the improper presence of test/answer information in training, producing inflated evaluation scores and real-world failure.

Simple explanation: Leakage is cheating that the cheater often does not notice. Two classic forms:

Leakage type 1 — Target leakage (the answer hidden in the question). A feature contains information about the label that would not exist at prediction time. *Example:* To predict "will this patient develop an ADR?", the team includes the feature "antidote administered during stay". But antidotes are given *because* an ADR happened — the feature is a consequence of the answer, not a predictor of it. In training, the model "discovers" the perfect rule: antidote given → ADR occurred (100% training score!). In real use, at admission time, no one knows yet whether an antidote will be given — the feature does not exist when the prediction is needed. The model is useless, despite its perfect score.

Leakage type 2 — Train-test contamination (the exam paper leaked). Test information contaminates training. *Example A:* The same patient's two admissions appear once in the training set and once in the test set. The model partly memorises that patient and scores artificially high on "unseen" data that is not really unseen. *Example B:* The team computes cleaning statistics (e.g., average kidney value used for filling missing cells) on the *entire* dataset before splitting. Test-set information has now leaked into the training data through those statistics. The correct procedure: split first, compute statistics on the training portion only, then apply them to validation/test.

```
TARGET LEAKAGE:                    TRAIN-TEST CONTAMINATION:
feature ⟶ secretly IS              test data ⟶ secretly IN
the answer                         the training

"antidote given"                   same patient in both sets;
predicts "ADR happened"            stats computed before splitting
— perfect score, zero              — inflated score,
real-world value                   false confidence
```

How to avoid leakage:

1. **Timeline rule:** For every feature, ask — "would this information exist at the moment we make the prediction?" If not, drop it.
2. **Split first:** Split data into train/validation/test *before* any cleaning statistics, feature selection, or tuning.
3. **Patient-level splitting:** Keep all records of one patient in a single split — never spread across train and test.

4 **Pharmacist review:** Have a domain expert scan the feature list for "too-good-to-be-true" predictors (a classic leakage smell).

Key takeaway: *If a model's score looks unbelievably good, suspect leakage before celebrating. Leakage is the commonest cause of "99% in the lab, failure in the ward".*

5. Classifications

5.1 Split roles

Split	Used for	Touched during tuning?	Used how many times?
Training	Learning patterns	Constantly	Many
Validation	Comparing models, tuning settings	Yes (repeatedly)	Many
Test	Final honest estimate	Never	Once

5.2 Fitting problems

Problem	Training score	New-data score	Cause	Fix
Overfitting	High	Low	Memorised noise	More data, simpler model, cross-validation
Underfitting	Low	Low	Learnt too little	More informative features, more flexible model
Good fit	High	High (close)	Learnt real patterns	Deploy with monitoring

5.3 Drift types

Type	What changed	Pharmacy example	Detection clue
Data drift	Input characteristics	New oncology ward; new formulary drug	Feature distributions shift in monitoring
Concept drift	Input→answer relationship	Guideline redefines a risky combination	Accuracy falls though inputs look similar

5.4 Leakage types

Type	Mechanism	Smell (warning sign)	Prevention
Target leakage	Feature contains the answer	Suspiciously perfect score; feature is a consequence of the outcome	Timeline rule: feature must exist at prediction time
Train-test contamination	Test info in training	Score drops sharply on truly new data	Split first; patient-level splits

6. Mechanisms

6.1 The honest-evaluation mechanism (why splits work)

1 The model adjusts itself to reduce errors on the **training** set — so training error is *biased downward* (optimistic).

2 The **validation** set is unseen during training but seen repeatedly during tuning — so validation error is *slightly optimistic*.

3 The **test** set is unseen during both — so test error is the *least biased* estimate of future performance.

4 Each split removes one layer of optimism. Skipping a split = accepting that layer of self-deception.

6.2 Cross-validation mechanism

Repeated train/test rotation converts "one possibly lucky/unlucky split" into "K experiments whose average is stable". Mechanism: every data point serves exactly once as unseen evaluation data, so no point is wasted and no single split dominates the verdict.

6.3 Drift-detection mechanism (monitoring, stage 7)

1 **Track** feature distributions and performance metrics on a schedule (e.g., monthly).

2 **Compare** against the training baseline: are inputs shifting (data drift)? Is accuracy falling (either drift)?

3 **Investigate** the cause: new ward? new guideline? new formulary?

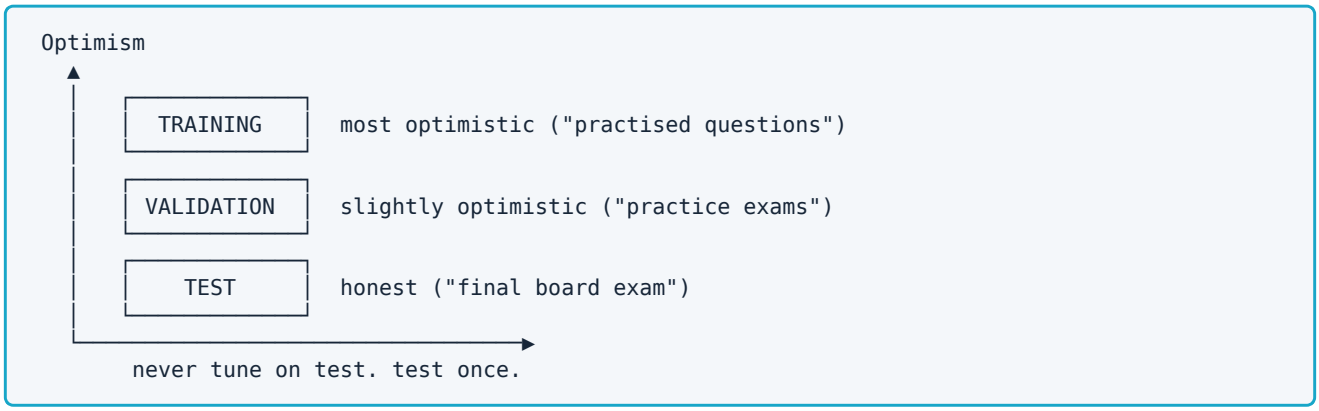
4 **Act**: retrain on recent data → re-validate → redeploy new version (lifecycle loop), or withdraw.

6.4 Leakage-prevention mechanism

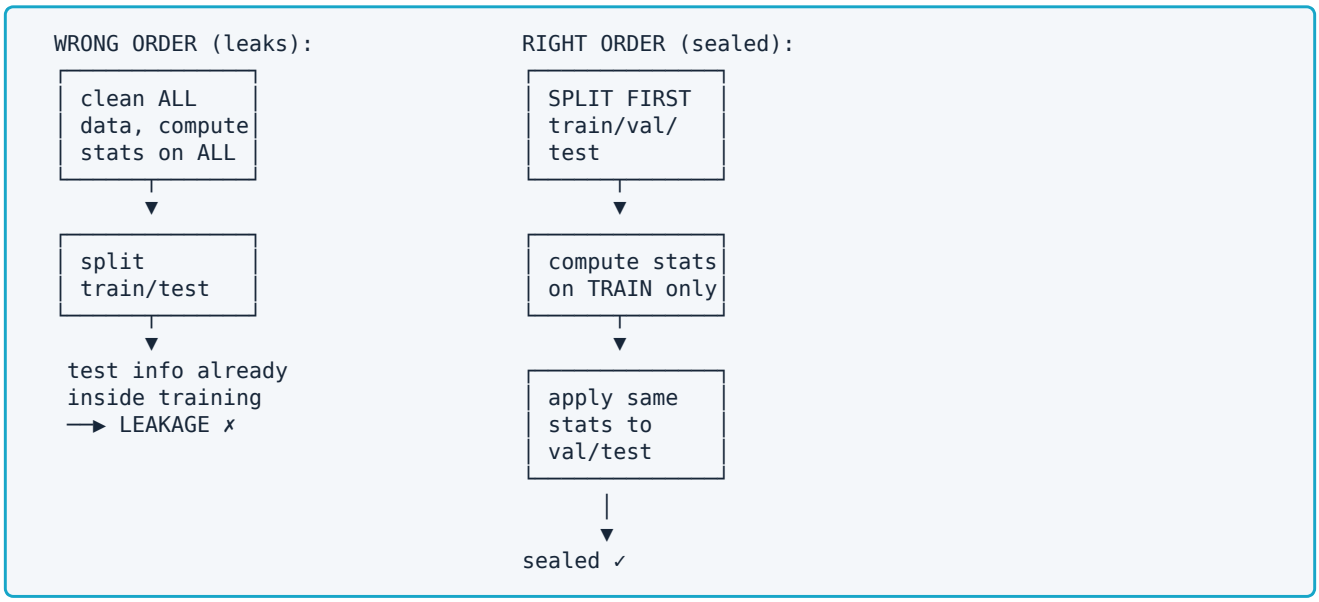
Leakage is prevented by *ordering discipline*: split → compute statistics on train only → train → validate → test once. Any step that looks at test data before the final test (even "just for cleaning") breaks the seal.

7. Diagrams/flowcharts

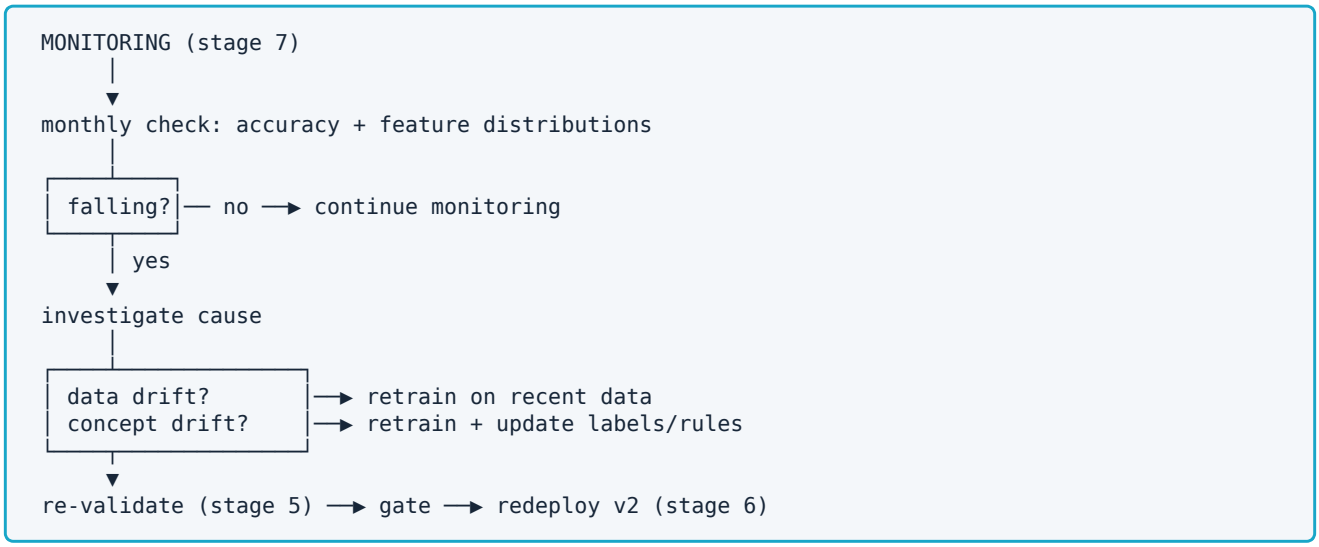
7.1 The three splits and their honesty levels



7.2 Leakage-prevention ordering



7.3 Drift response loop



8. Infographic concepts

Concept 1 — "Three exam halls". Three doors labelled Training (student studying at a desk, books open), Validation (practice test, teacher watching), Test (sealed exam hall, single attempt, "do not enter until final day"). A banner: "Judge the student only by the sealed hall."

Concept 2 — "The memoriser vs the understander". Two students: one with a photographic head full of exact question-answer pairs (overfitting — panics when the question is rephrased), one with a concept map of drug mechanisms (good fit — answers rephrased questions calmly). Caption: "Memorising noise vs learning patterns."

Concept 3 — "The leaking pipe". A pipe labelled "training pipeline" with two leaks dripping answer-key liquid into it: one leak labelled "target leakage (antidote feature)", one labelled "test data leaking back". A pharmacist tightening the joints with wrenches labelled "split first" and "timeline rule".

Concept 4 — "The shifting river". A bridge (the model) built over a river; over time the river changes course (drift) and the bridge no longer spans the water. A monitoring watchtower spots it; engineers

rebuild (retrain). Caption: "The world moves — recalibrate."

9. Tables

9.1 Splits master table

Aspect	Training set	Validation set	Test set
Purpose	Learn patterns	Tune & compare models	Final honest score
Sees the model during training?	Yes	No	No
Used during tuning?	Indirectly (it is what is tuned)	Yes, repeatedly	Never
Number of uses	Many	Many	Once
Score optimism	High	Slight	Minimal
Analogy	Textbook	Practice exams	Board exam

9.2 Overfitting vs underfitting vs good fit (toy numbers, illustrative)

Situation	Toy training score	Toy new-data score	Diagnosis
Model A	98%	62%	Overfitting — memorised noise
Model B	55%	52%	Underfitting — learnt too little
Model C	84%	81%	Good fit — learnt real patterns ✓

9.3 Drift vs leakage — do not confuse

	Drift	Leakage
When	After deployment, over time	During development
Cause	World changed	Test/answer info entered training
Symptom	Performance falls gradually in monitoring	Score suspiciously high in testing, then fails live
Fix	Retrain on new data, re-validate	Remove leaking feature; re-split properly; re-evaluate

10. Worked toy examples

ILLUSTRATIVE TOY DATASET (not real patient data). All numbers below are invented for teaching.

Toy example 1 — overfitting exposed by the test set.

A toy ADR model trained on 200 records:

- Training score: 196/200 correct = **98%**
- Validation score (unseen 50 records): 38/50 = **76%**
- Test score (locked 50 records, used once): 31/50 = **62%**

Interpretation: the 98% was memorisation. The honest estimate is ~62%, not 98%. Decision: do not deploy; simplify the model, get more data, and re-validate. *Reporting the 98% to the committee would have been dishonest — even if unintentional.*

Toy example 2 — k-fold cross-validation (5-fold, 100 toy records).

Computed earlier in section 4.5: fold scores 80%, 76%, 82%, 78%, 79% → average **79%**. The team reports "79% ± stable across folds" and notes the model is consistent — no single fold disagrees wildly, which increases trust.

Toy example 3 — target leakage arithmetic.

Toy data, 100 admissions, 10 real ADRs. Feature: "antidote given during stay" (given only when ADR occurred):

- Rule learnt: antidote = yes → predict ADR. Training score: 100/100 = **100%** (perfect — and perfectly useless).
- At real prediction time (admission), "antidote given" is unknown for every patient → the model's perfect rule cannot fire → real performance collapses to guessing.

Teaching point: a 100% score with a consequence-of-outcome feature is the classic leakage smell.

Toy example 4 — drift arithmetic.

Toy ADR model, monthly monitoring of "fraction of flagged patients who truly develop ADRs":

Month	Toy value
1 (launch)	80%
2	79%
3	76%
4	70%
5	66%

Steady fall → investigation → new antibiotic protocol introduced in month 3 changed ADR patterns (concept drift + data drift mix) → action: retrain including months 3–5 data, re-validate, redeploy v2. Without monitoring, the decay would have continued silently.

11. Analogies

1. **Student and exams** (used throughout): training = studying the textbook; validation = practice tests; test = the final board exam taken once; overfitting = memorising without understanding; leakage = getting the exam paper early; drift = the syllabus changing after you graduated.
2. **Cricket net practice vs match.** Training in the nets against the same bowler (training set) makes you look like a champion; the real match against unknown bowlers (test set) reveals true skill. Practising only one bowler's deliveries = overfitting.
3. **Weighing scale calibration.** A model is like a weighing scale: validated once at the factory (test set), but it drifts with time and temperature — so pharmacies recalibrate scales periodically. Model monitoring and retraining is recalibration.

4 **Leaking exam paper.** Target leakage = the answer key printed at the bottom of the question paper; train-test contamination = the examiner accidentally giving the same paper for practice and finals. Both inflate marks meaninglessly.

5 **River and bridge** (drift): the bridge (model) was perfect for the old river course; the river shifted, and now the bridge lands on dry land. Rebuilding = retraining.

12. Pharmacy relevance

- **Dose-prediction and ADR tools live or die by honest validation.** A pharmacist must be able to ask: "Was this tested on unseen patients? Was the test set truly locked?" — the two questions that expose most weak claims.
- **Small pharmacy datasets make cross-validation essential.** A single hospital rarely has lakhs of ADR cases; k-fold cross-validation extracts trustworthy evidence from modest data instead of gambling on one split.
- **Formularies and guidelines change constantly** — new drugs, revised STGs (standard treatment guidelines), antibiotic policy updates. Concept drift is not theoretical in pharmacy; it is routine. Any deployed tool needs a re-validation calendar.
- **Leakage smells need pharmacists.** Only a domain expert recognises that "antidote prescribed" or "ADR treatment started" as a *feature* is a consequence of the outcome. Data scientists alone often miss this — the pharmacist is the leakage detector.
- **Patient-level splitting matters in pharmacy data.** One patient may have many admissions; spreading them across train and test lets the model "recognise" the patient instead of learning medicine.

13. India context

- **Small, single-centre datasets.** Many Indian studies and tools are built on one hospital's data. Cross-validation is especially important here — a single lucky split on small data is dangerously misleading.
- **Guideline churn.** Indian standard treatment guidelines, essential-medicine lists, and antibiotic policies are updated periodically; each update is a potential concept-drift event for deployed tools. Build re-validation into the SOP — as per the latest guidance, confirm the current edition of relevant guidelines.
- **Multi-site data, one model.** A model trained on metro corporate-hospital data and deployed in district hospitals faces immediate data drift (different age mix, different formularies, different languages in notes). Validate on the deployment population, not just the training population.
- **ABDM-era data growth.** As digitisation expands, datasets will grow and shift in composition (more rural records entering). Expect data drift as the data itself becomes more representative — a good change that still requires re-validation.
- **Exam-hall honesty as culture.** The train/test discipline mirrors examination ethics every Indian student knows: you do not set the paper from the textbook exercises alone. Bring that same ethic to model evaluation.

14. Key points

- **Overfitting** = memorising training quirks; high training score, low new-data score. The whole split system exists to expose it.
- **Training set** learns; **validation set** tunes/compares (practice exams); **test set** gives the final honest score — used once, never tuned on.
- **K-fold cross-validation** rotates the test role across K folds and averages — reliable estimates from limited data.
- **Data drift** = inputs changed; **concept drift** = the rules changed. Both decay performance; monitoring catches them; retraining + re-validation fixes them.
- **Data leakage** = test/answer information in training. Two types: **target leakage** (feature contains the answer) and **train-test contamination** (test data in training).
- Prevent leakage: **timeline rule** (feature must exist at prediction time), **split first**, **patient-level splits**, pharmacist review of features.
- A suspiciously perfect score is a leakage smell, not a triumph.

15. Common confusions

Confusion	Clarification
Validation set vs test set	Both are "unseen during training", but validation is used repeatedly for tuning (slightly optimistic), test is used once at the end (honest). They are not interchangeable.
Cross-validation vs "testing many times on the test set"	Cross-validation rotates held-out folds *during development*. Repeatedly re-testing on the *test* set destroys its honesty.
Overfitting vs "the model is wrong"	An overfit model is not "wrong" on training data — it is brilliantly right on the wrong things (noise). The failure appears only on new data.
Data drift vs concept drift	Data drift: new kinds of inputs (new ward). Concept drift: same inputs, new correct answers (new guideline). Both need retraining, but investigation differs.
Leakage vs overfitting	Overfitting = learnt noise too well (fix: simplify, more data). Leakage = cheated via answer info (fix: remove leak, re-split). A 99% score suggests leakage first.
"More folds always better"	More folds = more computation and more overlap between training sets; 5 or 10 folds is the usual practical choice (illustrative convention, not a law).

16. Common mistakes

1. **Reporting the training score as the model's performance** — the original sin of model evaluation.
2. **Tuning on the test set** — peeking repeatedly until the score looks good, then claiming an honest result.
3. **Computing cleaning statistics before splitting** — silent train-test contamination (e.g., averages from the full dataset).

4 **Splitting by row instead of by patient** — one patient's admissions land in both train and test.

5 **Including consequence-features** — "antidote given", "ADR treatment started", "length of stay" used to predict events that cause them (target leakage).

6 **Ignoring drift** — deploying and never monitoring; the model silently decays as guidelines and formularies change.

7 **Trusting one lucky split** — especially with small pharmacy datasets; use cross-validation.

8 **Celebrating a 99–100% score** instead of hunting for the leak that produced it.

17. Mnemonics

The three splits — "TVT: Textbook, Viva-practice, Terminal exam":

- **T**raining = **T**extbook (study)
- **V**alidation = **V**iva-practice (practice tests)
- **T**est = **T**erminal exam (once, sealed)

Overfitting — "Memorised, not Mastered": if training ≫ new-data score, the model *memorised*; it did not *master*.

Drift — "Data changed or Definitions changed": data drift = the *data* changed; concept drift = the *definitions/rules* changed.

Leakage prevention — "SPLIT":

- **S**plit first, before any statistics
- **P**atient-level splits (one patient, one split)
- **L**ook at features with the timeline rule
- **I**nvestigate perfect scores (leakage smell)
- **T**est once, never tune on it

18. Exam summary

Honest evaluation needs three data splits because models can **overfit** — memorise training quirks (high training score, e.g., toy 98%) while failing new data (toy 62%), like a student who memorises practice questions. The **training set** is the textbook (learn), the **validation set** holds practice exams (compare models, tune settings — slightly optimistic because reused), and the **test set** is the sealed final exam used once for the honest score. **K-fold cross-validation** rotates K equal folds through the test role and averages the K scores (toy 5-fold: 80/76/82/78/79 → 79%), giving reliable estimates from limited pharmacy data. After deployment, **model drift** decays performance: **data drift** (inputs change — new oncology ward) vs **concept drift** (rules change — new guideline redefines risk); monitoring detects the fall and retraining + re-validation fixes it. **Data leakage** — test/answer information entering training — inflates scores and causes live failure: **target leakage** (feature contains the answer, e.g., "antidote given" predicting ADR; timeline rule prevents it) and **train-test contamination** (test data in training, e.g., same patient in both sets or statistics computed before splitting; prevented by split-first discipline and

patient-level splits). A suspiciously perfect score is a leakage smell. Pharmacists are essential leakage detectors and drift watchers, especially with India's small single-centre datasets, changing guidelines, and metro-to-district deployment shifts.

19. One-minute revision

- Overfitting = memorised, not mastered (train 98%, new data 62%).
- Splits: **T**extbook (train) → **V**iva-practice (validation) → **T**erminal exam (test, once, sealed).
- K-fold: rotate the test fold K times, average the scores (toy: 79%).
- Drift: **data** changed (new inputs) vs **concept** changed (new rules) → monitor → retrain → re-validate.
- Leakage: answer info in training. Target leakage (antidote feature) vs train-test contamination (same patient both sides).
- Prevent: **SPLIT** — split first, patient-level, timeline rule, investigate perfect scores, test once.
- 99% score? Hunt the leak, don't celebrate.

20. MCQs

Q1. Overfitting means: A) The model is too simple B) The model memorised training quirks, scoring high on training data but low on new data C) The computer overheated during training D) The dataset is too large

Q2. The validation set is used to: A) Give the final honest performance score B) Train the model C) Compare candidate models and tune settings during development D) Replace the test set

Q3. The test set must be used: A) Repeatedly until the score looks good B) Once, at the end, with no tuning based on it C) Before training begins D) Only for training

Q4. In 5-fold cross-validation with 100 records, each round: A) Trains on 20 and tests on 80 records B) Trains on 80 and tests on 20 records, rotating the test fold each round C) Uses all 100 records for training only D) Tests on the same 20 records every round

Q5. Data drift refers to: A) The model's code being deleted B) Incoming data gradually changing character versus training data (e.g., new ward, new patient mix) C) The correct answers changing due to new guidelines D) Electricity fluctuations

Q6. A new guideline redefining which drug combinations are "high risk" causes: A) Data drift B) Concept drift C) Overfitting D) Underfitting

Q7. Using "antidote administered" as a feature to predict "ADR occurred" is: A) Good feature engineering B) Target leakage — the feature is a consequence of the answer and won't exist at prediction time C) Data drift D) Cross-validation

Q8. Computing the average kidney value on the full dataset (before splitting) and using it to fill missing values is: A) Best practice B) Train-test contamination — test information leaked into training via the statistic C) Target leakage D) Concept drift

Q9. The correct order of operations to prevent leakage is: A) Clean all data → compute stats → split → train B) Split first → compute statistics on training data only → train → validate → test once C) Test first → train later D) Train → split → clean

Q10. A model scores 100% on evaluation. The first thing an experienced pharmacist should suspect is: A) A brilliant model ready for deployment B) Data leakage inflating the score C) Underfitting D) Data drift

Answer key: Q1–B, Q2–C, Q3–B, Q4–B, Q5–B, Q6–B, Q7–B, Q8–B, Q9–B, Q10–B

21. Viva/application questions

1 Explain overfitting using the student-and-exam analogy. Give toy numbers.

2 Why can't we evaluate a model on its training data? What does each of the three splits do?

3 Work through a 4-fold cross-validation on 80 toy records: how many records per fold, and what happens in round 3?

4 Distinguish data drift from concept drift with one pharmacy example of each.

5 Monitoring shows a steady monthly fall in a deployed ADR tool's performance. Walk through detection → investigation → action.

6 Define target leakage and explain why "antidote given" is a leaking feature for ADR prediction, using the timeline rule.

7 Give two examples of train-test contamination and state how to prevent each.

8 A vendor reports "99.5% accuracy" for their ADR predictor. List five questions you would ask before believing it.

9 Why must splitting be done at the patient level rather than the record level in pharmacy data?

10 Where do train/test/validation, cross-validation, drift, and leakage sit in the seven-stage AI lifecycle?

22. References

- WHO guidance on ethics and governance of artificial intelligence for health — as per the latest guidance, confirm the current edition (validation and monitoring expectations).
- EU AI Act — as per the latest guidance, confirm the current edition (accuracy, robustness, and monitoring for high-risk AI).
- Standard machine-learning and biostatistics study material (train/test methodology, cross-validation) and your university syllabus for BP801T.
- Pharmacovigilance Programme of India (PvPI) — public information on ADR data in India.

CHAPTER 4: Documentation Practices & Reproducibility

1. Learning objectives

After studying this chapter, you will be able to:

- Define documentation, reproducibility, model card, datasheet for datasets, version control, and audit trail.
- Describe what goes into a model card and into a dataset datasheet, from zero.
- Explain version control using a simple analogy.
- Explain what an audit trail is and why regulators and hospitals need one.
- Describe the "reproducibility crisis" concept and why it matters in healthcare AI.
- Connect AI documentation to the GMP-style documentation mindset pharmacists already know.
- Read and interpret a filled toy model card for an illustrative ADR classifier.

2. Definitions

Documentation. Written records describing what was done, how, by whom, and why — so that another competent person can understand, repeat, check, or take over the work later. In AI: task definitions, data descriptions, cleaning logs, model settings, validation reports, SOPs.

Reproducibility. The ability of an independent team, using the documented methods and (where permitted) the same data, to obtain essentially the same results. If a result cannot be reproduced, it cannot be trusted.

Model card. A short, standardised document (usually 1–3 pages) that describes one trained model: what it is for, what data trained it, how it was evaluated, how well it performed (including across patient subgroups), its limitations, and its intended and out-of-scope uses. Like the "label/pack insert" of a model.

Datasheet for datasets. A standardised document describing one dataset: why it was created, what it contains, how it was collected, how it was cleaned, who is represented (and who is missing), privacy measures, and known limitations. Like the "certificate of analysis" of data.

Version control. A system that saves every changed version of files (code, documents, datasets, models) with a number/name, date, author, and note of what changed — so any earlier version can be retrieved and changes can be compared. Nothing is ever silently overwritten.

Audit trail. A chronological, tamper-evident record of who did what, when, in a system — e.g., who edited a record, who approved a model version, who changed a threshold. Regulators inspect audit trails to verify that proper process was followed.

Reproducibility crisis. The widely discussed observation that many published scientific/AI results cannot be reproduced by other teams — often because methods, data, code, or settings were incompletely documented or shared. In healthcare, an unreproducible model is an unsafe model.

Intended use. The specific purpose, population, and setting a model is designed and validated for. Using it outside the intended use is off-label use of AI.

Lineage (data/model lineage). The recorded history of where data came from and what transformations it went through — "which raw file → which cleaning steps → which dataset version → which model version".

3. Terminology table

Term	Simple meaning	Pharmacy example
Documentation	Written record of what/how/who/why	Cleaning log, validation report, SOP
Reproducibility	Another team can repeat and get the same result	Re-training the ADR model gives matching scores
Model card	1–3 page "label" of a model	Card for ADR-risk classifier v2.1
Datasheet for datasets	Standard "certificate" of a dataset	Sheet for "City Hospital ADR dataset 2021–25"
Version control	Numbered, dated history of every file change	model v1.0 → v1.1 → v2.0 with change notes
Audit trail	Who-did-what-when record	"Dr. Rao approved deployment, 12-Mar, 10:30"
Reproducibility crisis	Many published results can't be repeated	A claimed 95% model no one can rebuild
Intended use	Approved purpose + population + setting	Adult inpatients, City Hospital, pharmacist review
Lineage	History trail of data → model	Raw EHR extract → cleaned v3 → model v2.1

4. Explanation

4.1 Documentation — writing down the work

Technical concept: Documentation is the complete, organised written record of an AI project's decisions, data, methods, and results across the lifecycle.

Simple explanation: Every pharmacist knows: "if it is not documented, it did not happen." A batch of tablets is not released on memory — it has batch manufacturing records, QC reports, and signatures. AI work needs the same discipline. Documentation answers, for every stage: *What did we do? Why this choice and not another? Who did it and when? What were the results?* Six months later, a new team member — or an auditor, or a regulator — must be able to reconstruct the whole story from the documents alone.

Example: The ADR project documents: the framed task (1 page), data sources and permissions (ethics letter filed), the cleaning log (what was filled/deleted and why), model settings tried, validation results per subgroup, the deployment SOP, and the monitoring plan. When the chief pharmacist is transferred, the new chief reads the file and understands everything — no knowledge walks out the door.

> **Key takeaway:** *Documentation is how an AI project survives staff changes, audits, and time. Undocumented work is unfinished work.*

4.2 Reproducibility — "show me again"

Technical concept: A result is reproducible if an independent party following the documented procedure obtains consistent results.

Simple explanation: In pharmacy QC, if one analyst reports a tablet assay of 99% and a second analyst following the same method gets 72%, the method — or the documentation — is broken, and the batch is not released. Same for AI: if the vendor claims "our model scores 90%" but no independent team can rebuild and re-test it to ~90%, the claim is not evidence — it is advertising. Reproducibility needs three ingredients: the data (or equivalent data), the code/settings, and complete documentation of every step.

Example: A journal paper claims an ADR predictor with 95% accuracy. An independent hospital team requests the dataset description, the cleaning steps, and the model settings — and receives only vague paragraphs. They cannot reproduce 95%; their rebuild scores 70%. The original claim is now doubtful, and no responsible hospital would deploy based on it.

> **Key takeaway:** *Trustworthy AI is reproducible AI. "Trust us, it works" is not documentation.*

4.3 The reproducibility crisis

Technical concept: The reproducibility crisis is the recognised pattern across science — including AI for health — where a substantial share of published findings cannot be independently reproduced, often due to incomplete documentation, data leakage (Chapter 3), selective reporting of the best run, or code that is never shared.

Simple explanation: Imagine 100 published "AI detects disease X" papers. Independent teams try to redo 100 of them and succeed with only a fraction — the rest fail because the original data had leaks, the reported score was the luckiest of 50 attempts, or key steps were never written down. For pharmacy, the lesson is sharp: a publication or vendor brochure is the *start* of evaluation, not the end. Always demand the model card, the datasheet, and independent validation on your own data before trusting a tool with patients.

> **Key takeaway:** *Published performance numbers are claims, not facts, until reproduced. Your hospital's own validation (lifecycle stage 5) is the only score that counts for your patients.*

4.4 Model cards — the "pack insert" of a model

Technical concept: A model card is a standardised short document accompanying a trained model, summarising its identity, purpose, data, performance, limitations, and usage guidance.

Simple explanation: Every medicine strip has a pack insert: what it is, what it is for, dosage, warnings, "do not use if...". A model card is the same idea for a model — anyone receiving the model reads 1–3 pages and knows exactly what it is and is not safe for. Standard sections:

1. **Model details** — name, version, date, developers, contact.
2. **Intended use** — what task, which population, which setting; and **out-of-scope uses** (what it must NOT be used for).
3. **Training data** — which dataset (with its datasheet), size, time period, population.
4. **Evaluation data & results** — which test set, overall scores, and scores **broken down by subgroup** (age, sex, ward) — because a single overall number can hide unfairness (Chapter 5 builds on this).
5. **Limitations & risks** — known weaknesses, e.g., "not validated for paediatric patients", "performance falls if prescribing patterns change".
6. **Ethical considerations & recommendations** — e.g., "human-in-the-loop required; pharmacist makes the final decision".

Example: The ADR classifier's model card states: intended for adult inpatients at City Hospital for pharmacist triage; NOT for paediatric or ICU patients (not validated there); test score 8/10 ADR capture overall, 7/10 in patients over 80 (slightly lower — flagged honestly); limitation: "retrain if formulary

changes significantly".

> **Key takeaway:** *A model without a model card is like a medicine strip without a label — you cannot use it safely because you do not know what it is.*

4.5 Datasheets for datasets — the "certificate of analysis" of data

Technical concept: A datasheet for datasets documents a dataset's motivation, composition, collection, cleaning, and limitations in a standard format.

Simple explanation: Raw material for medicines comes with a certificate of analysis: identity, purity, source, test results. A dataset datasheet is the same for data. Standard sections:

1. **Motivation** — why was this dataset created? For which task?
2. **Composition** — what is in it? How many records, which features, which time period, which population?
3. **Collection** — how gathered? From which systems/hospitals? With what permissions and consent? How de-identified?
4. **Cleaning (preprocessing)** — what was fixed, filled, or removed — referencing the cleaning log.
5. **Representation & gaps** — who is well represented? Who is missing or under-represented? (Links to biased sampling, Chapter 2.)
6. **Limitations & recommended uses** — e.g., "mild ADRs under-reported; not suitable for training mild-ADR detectors without correction".

Example: The datasheet for "City Hospital ADR dataset 2021–25" notes: 48,000 admissions, adult only, mild ADRs under-reported (ward staff report mainly serious events), paediatric cases excluded, de-identified by removing 18 identifier types, cleaning log v3 attached. A team reading this instantly knows the dataset's strengths and blind spots.

> **Key takeaway:** *The datasheet forces honest admission of what the data does NOT contain — and what is missing shapes the model more than what is present.*

4.6 Version control — never silently overwrite

Technical concept: Version control records every version of project files with identifiers, timestamps, authorship, and change notes, preserving full history.

Simple explanation: Think of a prescription register where corrections are made by striking through with a single line, dating, and initialling — never by erasing. Version control is the digital equivalent for all project files: dataset v1.0, v1.1, v2.0; model v1.0, v2.0; SOP v3. Each version notes *what changed and why*. If version 2.0 performs worse, you can instantly go back to v1.3. If an auditor asks "which model was live in June?", the record answers exactly.

Example: ADR model history: v1.0 (initial, validated Jan) → v1.1 (threshold adjusted after pharmacist feedback, Mar) → v2.0 (retrained with oncology data, Jul). Monitoring alert in August? The team checks exactly which version was live each month — no guessing.

```
v1.0 ──Jan──▶ validated, deployed (shadow)
 │  change note: "threshold 0.5→0.4 after pharmacist feedback"
 ▼
v1.1 ──Mar──▶ live, ward-wide
 │  change note: "retrained incl. oncology data (drift fix)"
 ▼
v2.0 ──Jul──▶ re-validated, redeployed
     every version retrievable; nothing overwritten
```

Key takeaway: *"Final_v2_REAL_final.xlsx" chaos is not version control. Numbered versions with dated change notes are — and auditors will ask for them.*

4.7 Audit trails — the who-did-what-when record

Technical concept: An audit trail is a chronological, tamper-evident log of actions and decisions in a system or project.

Simple explanation: In a pharmacy, the controlled-drug register records every tablet in and out with date, signature, and balance — so any discrepancy can be traced. An AI audit trail does the same for the project and the live system: who approved the validation report, who deployed version 2.0, who changed the risk threshold, which pharmacist overrode the tool's alert and why. When something goes wrong — a missed ADR — the trail shows whether process was followed or broken.

Example: A patient suffers an ADR the tool did not flag. The audit trail shows: model v2.0 was live (current), the patient was scored "low risk" at 09:14, the pharmacist reviewed and agreed at 09:40. Investigation focuses on the model's miss (a drift/retraining issue) rather than on human error — because the trail proves the human followed the SOP.

Key takeaway: *Audit trails protect both patients and staff: they turn "who is to blame?" into "what in the process failed?" — which is the only question that prevents recurrence.*

5. Classifications

5.1 Documentation across the lifecycle

Lifecycle stage	Key documents produced
1. Problem framing	Task definition, success criteria, stakeholder list
2. Data collection	Datasheet for datasets, permissions/ethics letters, de-identification record
3. Data preparation	Cleaning log (every decision + reason), dataset version history
4. Model development	Model card (draft), settings tried, candidate comparison
5. Validation	Validation report (scores overall + by subgroup), gate approval record
6. Deployment	Final model card, SOP, staff training records, shadow-mode report
7. Monitoring	Monitoring logs, drift reports, retraining records, audit trail

5.2 What must be version-controlled

Item	Example versions	Why
Datasets	ADR-data v1.0 → v2.0	Know exactly what each model trained on
Code & settings	training-script v3 → v4	Reproduce any result
Models	ADR-model v1.0 → v2.0	Roll back if a new version fails
Documents	SOP v2 → v3, model card v1 → v2	Auditors check the version in force at an incident date
Thresholds/config	risk threshold 0.5 → 0.4	Small changes, big effects — record them

5.3 Documentation audiences

Audience	What they need	Document
Pharmacist user	How to use it, limits, what to do on disagreement	SOP + model card (intended use, limitations)
Hospital management	Is it safe, legal, worth it?	Validation report, gate approvals
Auditor / regulator	Was proper process followed?	Audit trail, version history, all stage documents
Future developers	How to maintain/retrain it	Code, settings, cleaning log, lineage
Patients/public	Is it fair and safe?	Plain-language summary of purpose and safeguards

6. Mechanisms

6.1 How documentation creates trust (mechanism)

```
Complete documentation
   → independent team can reproduce results
      → claims become verifiable facts
         → auditors/regulators can check process
            → hospital management can approve with confidence
               → pharmacists use the tool correctly (SOP)
                  → patients are protected
Each arrow breaks if documentation is missing at that step.
```

6.2 The lineage mechanism

Every model version points to the exact dataset version that trained it; every dataset version points to the raw source plus the cleaning log. Mechanism: when a defect is found (e.g., a labelling error in raw data), lineage lets the team trace *forward* — "which models and deployments are affected?" — instead of guessing. This is exactly like a drug recall tracing batch numbers to distribution lists.

6.3 Version-control mechanism (how it works)

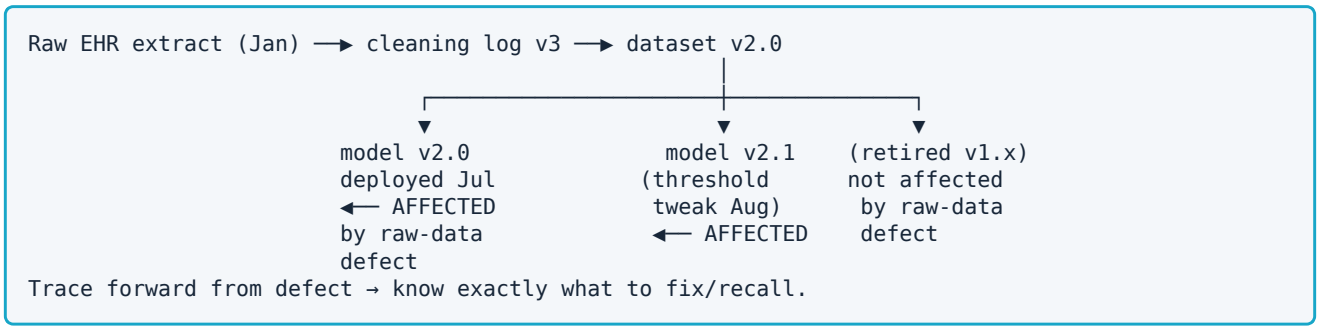
1. Every change is **committed** with a message ("threshold 0.5→0.4: pharmacist feedback, Mar").
2. Versions are **numbered** (major.minor: new retraining = major; small tweak = minor).
3. Any version is **retrievable** — the live system records which version is deployed.
4. Changes are **comparable** — diff v1.1 vs v2.0 shows exactly what changed.

7. Diagrams/flowcharts

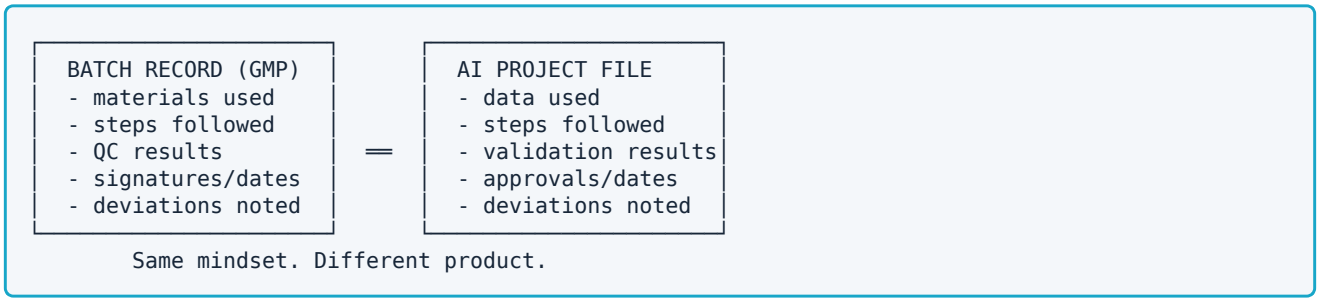
7.1 Documents across the lifecycle (the paper trail)



7.2 Lineage trace (recall-style)



7.3 "If it is not documented, it did not happen"



8. Infographic concepts

Concept 1 — "The model's pack insert". A medicine strip beside a document shaped like a pack insert; callouts map insert sections (uses, dosage, warnings, "do not use if") to model-card sections (intended use, thresholds, limitations, out-of-scope). Caption: "No label, no sale — no model card, no deployment."

Concept 2 — "The time machine" (version control). A timeline with version flags v1.0 → v1.1 → v2.0; a pharmacist turning a dial back to v1.1 when v2.0 misbehaves, with the change notes floating above each flag. Caption: "Every version kept. Every change explained. Any version retrievable."

Concept 3 — "The detective's board" (audit trail + lineage). A corkboard with photos/documents connected by red string: raw data → cleaning → dataset v2 → model v2.0 → deployment approval signature → monitoring log, with a magnifier on one signature. Caption: "When something goes wrong, follow the

string."

Concept 4 — "Recipe card vs secret recipe" (reproducibility). Two chefs: one with a detailed recipe card (ingredients, quantities, timings) whose dish a second chef reproduces perfectly; one guarding a "secret recipe" whose dish nobody can copy. Caption: "Science is the recipe card."

9. Tables

9.1 Model card sections — what goes in each

Section	Contents	Pharmacy parallel
Model details	Name, version, date, developers, contact	Manufacturer + batch no.
Intended use	Task, population, setting	Indications
Out-of-scope uses	What it must NOT be used for	Contraindications
Training data	Dataset, size, period, population	Composition/ingredients
Evaluation	Test set, overall + subgroup scores	Clinical trial results
Limitations & risks	Known weaknesses	Warnings & precautions
Ethical notes	Human-in-the-loop, fairness notes	"Use under medical supervision"

9.2 Datasheet sections — what goes in each

Section	Contents	Why it matters
Motivation	Why created, for which task	Prevents misuse for other tasks
Composition	Size, features, time period	Know what the model "saw"
Collection	Sources, permissions, de-identification	Legal and ethical validity
Cleaning	Fixes applied (links cleaning log)	Reproducibility
Representation & gaps	Who is included/missing	Fairness; biased-sampling check
Limitations	Known defects, e.g., under-reported mild ADRs	Honest expectations

9.3 GMP vs AI documentation — the mindset transfer

GMP habit (you know this)	AI equivalent (this chapter)
Batch manufacturing record	Cleaning log + training log
Certificate of analysis (raw material)	Datasheet for datasets
Pack insert / label	Model card
Controlled-drug register (who/when)	Audit trail
Change control with versions	Version control
Deviation reporting	Monitoring alerts + investigation records
"If not documented, not done"	Same — exactly the same

10. Worked toy examples

ILLUSTRATIVE TOY EXAMPLE (not real data, not a real product). The model card below is a teaching mock-up with invented numbers.

Toy model card — "ADR-Triage v2.0" (illustrative)

1. Model details. Name: ADR-Triage; version: 2.0; date: July (toy year); developers: City Hospital pharmacy + data team (toy); contact: pharmacy AI committee (toy); type: classification model (high/low ADR risk).

2. Intended use. Each morning, rank adult inpatients by predicted ADR risk so the clinical pharmacist reviews the highest-risk patients first. Setting: general wards of one hospital (toy). Human-in-the-loop: the pharmacist makes all final clinical decisions; the tool only prioritises the review queue.

3. Out-of-scope uses. NOT for paediatric patients, NOT for ICU patients, NOT as a substitute for clinical judgement, NOT for hospitals other than the one it was validated in.

4. Training data. "City Hospital ADR dataset 2021–25" (see toy datasheet): 48,000 adult admissions (toy number); features: age, sex, diagnoses, drug list, doses, kidney/liver values; labels: ADR yes/no from the hospital ADR register. Known gap: mild ADRs under-reported.

5. Evaluation. Test set: 5,000 locked admissions from the most recent 6 months (toy), never used in training. Overall: flagged 8 of every 10 patients who developed ADRs (toy). By subgroup (toy): age <60: 8.2/10; age ≥60: 7.8/10; male: 8.1/10; female: 7.9/10 — similar across groups ✓. False-alarm rate: about 3 false flags per 10 flags (toy).

6. Limitations & risks. Not validated for children or ICU; performance may fall if the formulary changes substantially (monitor monthly); depends on complete medication lists — missing drugs weaken predictions.

7. Ethical considerations & recommendations. Requires pharmacist review of every flagged case; do not use the score alone to withhold or change therapy; re-validate before use in any new ward or hospital; audit trail of overrides reviewed monthly.

Teaching point: Notice what the card does — it tells the truth about weaknesses (elderly slightly lower, mild ADRs under-reported, ICU excluded). That honesty is the whole purpose. A card that lists no limitations is a marketing sheet, not a model card.

Toy datasheet extract — "City Hospital ADR dataset 2021–25" (illustrative)

- **Motivation:** created to train and validate the ADR-Triage prioritisation tool.
- **Composition (toy):** 48,000 adult inpatient admissions; columns: age, sex, diagnosis codes, drug names (standardised to generic), doses, kidney/liver values, ADR label (yes/no).
- **Collection:** extracted from the hospital EHR and ADR register; ethics-committee permission (toy ref); de-identified (names, IDs, addresses removed).
- **Cleaning:** drug names unified to generics; 6% duplicate rows removed; missing kidney values flagged (not silently filled); full cleaning log v3.
- **Representation & gaps:** adults 18+ only; 55% male; mild ADRs under-reported (ward reporting bias); paediatric and ICU cases excluded.
- **Limitations:** not suitable for paediatric modelling; mild-ADR labels unreliable; single-centre — may not represent other hospitals.

11. Analogies

1. **Pack insert.** The model card is the pack insert; the datasheet is the certificate of analysis; the audit trail is the controlled-drug register. You already know these documents — only the product changed.
2. **Recipe card vs secret recipe.** Reproducible science is a recipe card anyone can follow; irreproducible claims are a "secret recipe" that conveniently cannot be checked.
3. **Prescription register corrections.** Version control = strike-through, date, initial — never erase. Every old version retrievable, every change explained.
4. **Batch recall.** Data/model lineage works like batch-number traceability: a defect in raw material → find every batch (model version, deployment) affected → targeted recall, not panic.
5. **CCTV in the pharmacy.** The audit trail is CCTV footage for decisions: boring 99% of the time, priceless on the day something goes wrong.

12. Pharmacy relevance

- **You already have the mindset.** GMP, GLP, prescription documentation, controlled-drug registers — pharmacy is a documentation profession. AI documentation is the same ethic applied to data and models. This is a genuine career advantage for pharmacists in AI teams.
- **Regulatory readiness.** Any clinical AI tool in India may face CDSCO scrutiny — as per the latest guidance, confirm the current edition. Complete lifecycle documentation is the backbone of any submission or inspection.
- **Medico-legal protection.** If an AI-assisted decision is ever questioned, the model card (intended use, limitations), the SOP, and the audit trail are the pharmacist's evidence of diligent, responsible practice.
- **Handover safety.** Hospital pharmacists transfer frequently. Documentation is what keeps the AI tool safe across staff changes.
- **Formulary and PTC committees.** Pharmacy and Therapeutics Committees already evaluate drugs with dossiers; model cards + datasheets + validation reports are the equivalent dossier for evaluating AI tools.

13. India context

- **CDSCO and documentation.** As digitisation grows, expect increasing regulatory attention to clinical software in India — as per the latest guidance, confirm the current edition with CDSCO. Teams that document from day one will be ready; teams that "document later" will struggle.
- **ABDM and data documentation.** As ABDM-linked records accumulate, datasheets for datasets become essential: which hospitals, which years, which populations, what de-identification — otherwise no one can judge what a national-scale dataset actually represents.
- **Small teams, big turnover.** Many Indian hospital IT/pharmacy teams are small with frequent transfers. Without version control and written SOPs, the AI tool becomes one person's personal project — and dies when they leave.

- **Language of documentation.** Write model cards and SOPs in simple English (and key safety points in the local language where needed) so ward pharmacists — not just data scientists — can actually use them.
- **Low-cost tools suffice.** Version control and audit trails do not need expensive software — even disciplined dated folders with change-note files are a valid start; the mindset matters more than the tool.

14. Key points

- **Documentation** = written what/how/who/why across all lifecycle stages. "If not documented, not done."
- **Reproducibility** = an independent team can repeat the work and get consistent results; it needs data + code/settings + complete documentation.
- **Reproducibility crisis** = many published AI-health claims cannot be reproduced; treat performance claims as claims until independently validated on your data.
- **Model card** = the model's pack insert (details, intended use, out-of-scope, data, evaluation by subgroup, limitations, ethics).
- **Datasheet for datasets** = the data's certificate of analysis (motivation, composition, collection, cleaning, gaps, limitations).
- **Version control** = numbered, dated, noted history of every file; any version retrievable; nothing silently overwritten.
- **Audit trail** = who-did-what-when; turns blame into process improvement.
- **Lineage** = traceability from raw data → dataset version → model version → deployment (recall-style).
- The **GMP documentation mindset transfers directly** — pharmacists already think this way.

15. Common confusions

Confusion	Clarification
Model card vs validation report	The validation report is the detailed test evidence; the model card is the short standardised summary label. The card points to the report.
Datasheet vs cleaning log	The datasheet summarises the dataset (what it is); the cleaning log details every fix (what was done). The datasheet references the log.
Version control vs backup	A backup saves copies against loss; version control saves *history with reasons* — numbered, dated, noted, comparable, retrievable by version.
Audit trail vs monitoring log	Monitoring logs track *system performance*; audit trails track *human/system actions* (who approved, who changed, who overrode). Both needed.
"Documentation slows us down"	Undocumented speed creates unmaintainable, unauditable systems — the most expensive kind. Documentation is part of the work, not overhead.

Confusion	Clarification
Reproducibility vs repeatability	Repeatability = same team, same setup, same result. Reproducibility = *independent* team, documented method, consistent result. Reproducibility is the stronger claim.

16. Common mistakes

1. **"We'll document at the end"** — decisions and reasons are forgotten within weeks; document as you go.
2. **Model deployed with no model card** — users cannot know its limits; misuse follows.
3. **Dataset shared with no datasheet** — next team inherits unknown biases and gaps.
4. **"Final_final_v2.xlsx" versioning** — no numbering, no change notes, no retrievability.
5. **Silent overwriting** — editing the live SOP or threshold without a new version and approval record.
6. **No audit trail on overrides** — pharmacists overrule the tool but nobody records why; the learning opportunity is lost and accountability is unclear.
7. **Publishing only the best score** — selective reporting that fuels the reproducibility crisis; report the honest, pre-registered evaluation.
8. **Code/settings kept on one laptop** — the project dies with the laptop or its owner.

17. Mnemonics

The two key documents — "CM-D D": "Card for the Model, Datasheet for the Data":

- **M**odel **C**ard = model's pack insert (intended use, limits, subgroup scores)
- **D**atasheet = data's certificate of analysis (source, gaps, cleaning)

Version control — "NDN-R": Numbered, Dated, Noted, Retrievable. Every version: a Number, a Date, a Note of what changed, and Retrievable any time.

Audit trail — "WWW": Who did What, When.

Reproducibility needs — "D-C-D": Data, Code/settings, Documentation. Missing any one of the three, and nobody can reproduce your result.

18. Exam summary

Documentation is the written what/how/who/why of an AI project across all lifecycle stages — "if not documented, not done" — and it is what lets projects survive staff changes, audits, and time. **Reproducibility** (an independent team getting consistent results from documented methods + data + code) is the test of trustworthy claims; the **reproducibility crisis** — many published AI-health results being unreproducible due to leaks, selective reporting, or missing methods — means performance claims must be treated as claims until validated on your own data. Two standard documents carry the core information: the **model card** (the model's pack insert — details, intended use, out-of-scope uses, training data, evaluation overall and by subgroup, limitations, ethical notes) and the **datasheet for datasets** (the data's certificate of analysis — motivation, composition, collection/permissions, cleaning, representation

gaps, limitations). **Version control** keeps every file version numbered, dated, noted, and retrievable (never silently overwrite); the **audit trail** records who did what when, turning incidents into process learning; **lineage** traces raw data → dataset version → model version → deployment, like batch recall. Pharmacists already own this mindset through GMP, batch records, and controlled-drug registers — a direct career advantage in AI teams, and essential for CDSCO readiness and medico-legal protection in India.

19. One-minute revision

- Document as you go: "if not documented, not done."
- Reproducibility = independent team, same method, consistent result (needs Data + Code + Documentation).
- Crisis = many published claims unreproducible → validate on YOUR data.
- **Model card** = model's pack insert; **Datasheet** = data's certificate of analysis.
- Version control: **N**umbered, **D**ated, **N**oted, **R**etrievable.
- Audit trail: **W**ho did **W**hat, **W**hen.
- Lineage = batch-recall traceability for data → model.
- GMP mindset transfers directly — pharmacists already think this way.

20. MCQs

Q1. A model card is best described as: A) A business card of the data scientist B) A standardised 1–3 page "pack insert" for a model: intended use, data, evaluation, limitations C) A computer graphics card D) A patient ID card

Q2. A datasheet for datasets documents: A) Only the file size B) Motivation, composition, collection, cleaning, representation gaps, and limitations of a dataset C) The price of the data D) The programmer's resume

Q3. Reproducibility requires which three ingredients? A) Data, code/settings, and complete documentation B) A fast computer, big data, and luck C) Publication, press coverage, and funding D) Training data only

Q4. The "reproducibility crisis" refers to: A) A shortage of computers B) Many published scientific/AI results not being independently reproducible, often due to incomplete documentation or leakage C) A crisis in paper manufacturing D) Models reproducing too quickly

Q5. Version control means: A) Controlling which version of Windows is used B) Numbered, dated, noted history of every file version, with any version retrievable C) Backing up once a year D) Deleting old files

Q6. An audit trail records: A) Only system errors B) Who did what, when — a chronological record of actions and decisions C) Patient diagnoses D) Electricity usage

Q7. Data/model lineage is most similar to: A) A family tree of programmers B) Pharmaceutical batch traceability — tracing a defect from raw material forward to all affected batches C) A hospital map D) A drug price list

Q8. Which model-card section is most like a medicine's "contraindications"? A) Model details B) Out-of-scope uses C) Training data D) Contact information

Q9. The strongest reason a hospital should do its own validation instead of trusting a vendor's brochure is: A) Brochures are expensive B) Published/vendor claims may be unreproducible; only local validation reflects the hospital's own patients and data C) Vendors are always dishonest D) Validation is fun

Q10. "If it is not documented, it did not happen" in AI work mainly protects: A) Only the data scientist's job B) Continuity across staff changes, auditability, reproducibility, and patient safety C) Computer speed D) Nothing — it is just paperwork

Answer key: Q1–B, Q2–B, Q3–A, Q4–B, Q5–B, Q6–B, Q7–B, Q8–B, Q9–B, Q10–B

21. Viva/application questions

1 Define documentation and reproducibility. Why does reproducibility need data + code + documentation?

2 Explain the reproducibility crisis and its practical lesson for a hospital considering a vendor's AI tool.

3 List the standard sections of a model card and explain each in one line.

4 List the standard sections of a dataset datasheet and explain why "representation & gaps" matters for fairness.

5 Read the toy model card for ADR-Triage v2.0: state its intended use, two out-of-scope uses, and two limitations.

6 Explain version control using the prescription-register correction analogy. What does "NDN-R" stand for?

7 What is an audit trail? Walk through how it helps investigate a missed ADR alert.

8 Explain data/model lineage and its parallel with pharmaceutical batch recall.

9 Map five GMP documentation habits to their AI equivalents.

10 A start-up offers your hospital an ADR predictor but provides no model card or datasheet. What do you ask for, and what do you refuse to do until you receive it?

22. References

- WHO guidance on ethics and governance of artificial intelligence for health — as per the latest guidance, confirm the current edition (transparency and documentation).
- EU AI Act — as per the latest guidance, confirm the current edition (technical documentation for high-risk AI systems).
- CDSCO (Central Drugs Standard Control Organization), India — as per the latest guidance, confirm the current edition.
- Ayushman Bharat Digital Mission (ABDM) — public information on health-data digitisation in India.
- Standard B.Pharm pharmaceutical jurisprudence / GMP study material (documentation, batch records, audit trails) and your university syllabus for BP801T.

CHAPTER 5: Model Auditing Basics

1. Learning objectives

After studying this chapter, you will be able to:

- Define a model audit and explain its purpose in simple words.
- Distinguish internal audit from external audit, with advantages of each.
- List and explain the four-part audit checklist: data, performance across subgroups, robustness, documentation.
- Carry out a basic bias/fairness check across patient subgroups using a worked toy example.
- Describe the standard structure of an audit report.
- State when a model must be re-audited.
- Explain the pharmacist's role in auditing AI tools.

2. Definitions

Audit. A systematic, independent examination of a system against defined criteria, producing documented findings and recommendations. In pharmacy you know financial audits and GMP audits; a *model audit* applies the same discipline to an AI system.

Model audit. A structured review of an AI system — its data, performance, fairness, robustness, documentation, and real-world behaviour — to answer: "Is this system safe, fair, and fit for its intended use?"

Internal audit. An audit conducted by people inside the organisation (but independent of the team that built the system) — e.g., the hospital's own quality cell reviewing the pharmacy's ADR tool.

External audit. An audit conducted by an outside party — e.g., an independent agency, academic group, or regulator — with no stake in the system's success.

Fairness (in AI). The property that a model's performance and errors do not systematically disadvantage any patient subgroup (by age, sex, language, socioeconomic background, or other characteristics).

Bias (in AI). A systematic error or unfair tendency in a model's behaviour toward particular groups — usually inherited from biased data (Chapter 2) or flawed design, not from intention.

Subgroup. A defined subset of the population — e.g., patients aged ≥60, female patients, rural patients. Fairness is checked *subgroup by subgroup*, because a good overall score can hide poor performance for one group.

Robustness. The ability of a model to keep performing correctly when inputs are slightly imperfect — missing values, typing errors, unusual but legitimate cases — without collapsing or behaving erratically.

Adversarial test (simple sense). Deliberately testing the model with tricky, edge-case, or slightly corrupted inputs to see whether it fails gracefully or dangerously.

Audit criteria. The pre-decided standards the system is judged against — e.g., "ADR capture ≥7/10 overall AND ≥7/10 in every age subgroup; no subgroup gap larger than 1/10".

Finding. One documented audit observation — what was checked, what was found, how serious it is.

Recommendation. The auditor's advised action for each finding — fix, monitor, restrict use, or withdraw.

Re-audit. A repeat audit conducted after a defined trigger — time elapsed, model retrained, data changed, or an incident occurred.

3. Terminology table

Term	Simple meaning	Pharmacy example
Model audit	Systematic independent check: safe, fair, fit for use?	Annual review of the ADR-triage tool
Internal audit	Done by the organisation's own independent team	Hospital quality cell audits pharmacy AI
External audit	Done by an outside independent party	University team audits the hospital's tool
Fairness	No subgroup systematically disadvantaged	Works equally for elderly and young
Bias	Systematic unfair tendency toward a group	Model misses ADRs in elderly women
Subgroup	A defined population subset	Age ≥60, female, rural patients
Robustness	Steady under imperfect inputs	Handles a missing lab value gracefully
Audit criteria	Pre-decided pass/fail standards	"≥7/10 capture in every subgroup"
Finding / Recommendation	Observation / advised action	"Gap found in ≥80 group → retrain with more elderly data"
Re-audit	Repeat audit after a trigger	After v2.0 retraining, or yearly

4. Explanation

4.1 What is a model audit?

Technical concept: A model audit is a systematic, independent, evidence-based examination of an AI system against pre-defined criteria, covering data, performance, fairness, robustness, and documentation.

Simple explanation: You know drug audits: an inspector visits a manufacturing plant with a checklist — premises, documents, QC, records — and reports what passes and what must improve. A model audit is the same idea for an AI tool. An auditor (who did *not* build the tool) works through a checklist: Is the data sound? Does it perform well — for *everyone*, not just on average? Does it survive messy real-world inputs? Is everything documented? The output is a written report with findings and recommendations — not a casual opinion.

Example: City Hospital's ADR-triage tool has run for a year. The hospital quality cell (internal auditors, independent of the pharmacy AI team) audits it: they re-check the datasheet, re-run the test set, break scores down by age/sex/ward, feed it records with missing values to test robustness, and read the model

card and SOP. They find it works well overall but underperforms for patients over 80 — a fairness finding with a recommendation.

Key takeaway: *An audit is how an organisation proves — to itself, to regulators, and to patients — that its AI tool deserves continued trust. Trust without checking is hope, not governance.*

4.2 Internal vs external audit

Technical concept: Internal audits are performed by the organisation's own independent function; external audits by outside parties. Both have distinct strengths.

Simple explanation:

- **Internal audit** = the hospital checking its own tool. Advantages: cheaper, faster, auditors understand local workflows and data, can be done frequently (even quarterly). Limitation: the auditors ultimately work for the same organisation — there is a risk, however well managed, of soft-pedalling bad news.
- **External audit** = inviting outside experts. Advantages: genuine independence, fresh eyes, credibility with regulators and the public, benchmarking against other hospitals. Limitations: costlier, slower, outsiders need time to understand local context.

Example: The hospital quality cell does an internal audit every 6 months (cheap, routine). Before expanding the ADR tool to two sister hospitals, management commissions an external audit by a university health-informatics group — because a wider rollout deserves independent credibility.

Key takeaway: *Use both: internal audits for routine hygiene, external audits for big decisions (rollout, regulatory submission, after a serious incident). Independence must be real — auditors must not audit their own work.*

4.3 The four-part audit checklist

Technical concept: A basic model audit examines four domains: (1) data, (2) performance across subgroups, (3) robustness, (4) documentation.

Simple explanation: The auditor's four questions:

Check 1 — Data: "Was the system built on sound data?" Re-examine the datasheet: sources, permissions, de-identification, cleaning log, representation gaps (Chapter 2). Re-verify a sample of labels. Ask: has the data changed since validation (new wards, new formulary)? Red flags: undocumented cleaning, unverified labels, unacknowledged sampling bias.

Check 2 — Performance across subgroups: "Does it work for everyone?" Re-run evaluation and break every score down by subgroup (age bands, sex, ward, language). An overall 8/10 can hide 9/10 for the young and 5/10 for the elderly. Compare against pre-decided criteria. This is the fairness check (detailed in 4.4).

Check 3 — Robustness: "Does it survive the real world?" Feed the model imperfect inputs: missing lab values, misspelled drug names, unusual-but-legal doses, duplicate records. A robust model degrades gracefully (flags "uncertain — needs human review"); a brittle model gives confident nonsense. Also check: does it behave sanely at the edges (very old patients, extreme lab values)?

Check 4 — Documentation: "Can we prove all of this?" Model card current? Datasheet matches the data actually used? Version control complete — which model version is live *right now*? Audit trail of overrides and approvals intact? SOP followed in practice (not just on paper)? Missing documents = failed check, because unverifiable claims cannot support patient care.

Example: The ADR tool audit scores: Data ✓ (datasheet honest, labels spot-verified); Subgroups ⚠ (≥80 group at 6/10 vs criterion 7/10 — finding); Robustness ✓ (handles missing values by flagging uncertainty); Documentation ⚠ (model card still says v1.1 while v2.0 is live — finding). Two findings, two recommendations — a useful audit, not a rubber stamp.

Key takeaway: *The checklist turns "we think it's fine" into checkable evidence across four domains. A pass in three domains does not excuse a fail in the fourth.*

4.4 Bias and fairness checks across subgroups

Technical concept: A fairness check measures the model's performance separately for each relevant patient subgroup and compares them against each other and against pre-set criteria, to detect systematic disadvantage.

Simple explanation: An overall score is an average — and averages hide. A tool can score 85% overall while scoring 95% for men and 70% for women; the average looks fine, the reality is unfair. The fairness check is simple arithmetic with moral weight:

1. Choose relevant subgroups (age bands, sex, and any locally important group — e.g., rural vs urban, language).
2. Compute the key performance number *within each subgroup* on unseen data.
3. Compare: biggest gap between subgroups? Does every subgroup meet the minimum criterion?
4. If a gap is large or a subgroup fails the criterion → **finding** → investigate cause (often: that subgroup was under-represented in training — Chapter 2's biased sampling, now visible as model behaviour) → recommend fix (collect more data, retrain, or restrict the tool's use for that subgroup until fixed).

Worked toy fairness example — ILLUSTRATIVE TOY DATASET (not real patient data):

Toy test set: 1,000 patients, 100 of whom developed ADRs. The tool flags patients as high risk. "Capture" = fraction of true ADR cases the tool flagged.

Subgroup (toy)	ADR cases	Flagged (captured)	Toy capture
Age <60	40	34	34/40 = **85%**
Age 60–79	35	28	28/35 = **80%**
Age ≥80	25	15	15/25 = **60%**
Male	55	45	45/55 = **82%**
Female	45	32	32/45 = **71%**
Overall	100	77	**77%**

Audit criteria (toy, pre-decided): overall capture ≥75% AND every subgroup ≥75% AND no subgroup gap >10 percentage points.

- Overall 77% ✓ (passes).

- But: age ≥80 at 60% ✗ (fails the 75% minimum); female 71% ✗ (fails); gap between <60 (85%) and ≥80 (60%) = 25 points ✗ (exceeds 10).

Findings: (1) The tool systematically under-serves the oldest patients — the group most vulnerable to ADRs. (2) A sex gap of 11 points needs investigation. Likely root cause (to verify): training data under-represented the very elderly and women (biased sampling → learnt patterns weaker for them).

Recommendations: (1) Do NOT expand use to geriatric wards until fixed. (2) Collect more elderly/female ADR cases, retrain, re-validate with subgroup criteria, then re-audit. (3) Meanwhile, add an SOP warning: "tool less reliable for age ≥80 — pharmacist to apply extra clinical vigilance."

> **Key takeaway:** *Always break scores down by subgroup. The overall number is the beginning of the fairness question, never the answer.*

4.5 Audit report structure

Technical concept: The audit report is the formal written output: scope, method, evidence, findings, recommendations, and overall opinion.

Simple explanation: Like a drug inspection report, it must let a reader who was not present understand exactly what was checked and concluded. Standard structure:

1. **Title & scope** — which system, which version, which period, what was (and was not) examined.
2. **Auditors & independence** — who audited, their relationship to the project (internal/external), dates.
3. **Criteria** — the pre-decided standards used (e.g., subgroup thresholds).
4. **Method & evidence** — what data re-examined, what tests re-run, whom interviewed, which documents read.
5. **Findings** — numbered, each with: what was checked, evidence, severity (critical/major/minor).
6. **Recommendations** — numbered, linked to findings, with suggested timelines and owners.
7. **Overall opinion** — e.g., "fit for continued use with corrective actions" / "use restricted until findings closed" / "not fit — withdraw".
8. **Follow-up plan** — when re-audit happens and who verifies the fixes.

Example (toy): "Audit of ADR-Triage v2.0, City Hospital, Jan–Jun (toy period). Overall opinion: fit for continued use in general wards WITH corrective actions; use in geriatric ward SUSPENDED until Finding 1 closed. Re-audit in 6 months or after retraining, whichever is earlier."

> **Key takeaway:** *A good audit report is specific, evidence-linked, and ends with an unambiguous overall opinion — not vague praise.*

4.6 When to re-audit

Technical concept: Re-audit triggers: scheduled intervals, model changes, data/environment changes, incidents, and scope expansion.

Simple explanation: An audit is a photograph, not a permanent certificate. Re-audit when:

1. **Time passes** — e.g., every 6–12 months (toy cadence; set your own), because drift is gradual.
2. **The model changes** — any retraining or version bump (v2.0 → v3.0) invalidates the old audit.
3. **The data or environment changes** — new ward, new formulary, new guideline, merged datasets.
4. **An incident occurs** — a missed ADR, a complaint, a monitoring alert that revealed a real problem.
5. **Scope expands** — new hospital, new patient group (e.g., adding paediatrics), new use case.

Key takeaway: *Diary the next audit at the end of every audit. "Audit once" is as useful as "validate once" — i.e., not at all.*

5. Classifications

5.1 Audit types

Type	Who	Strengths	Limits	Typical timing
Internal	Own organisation, independent of builders	Cheap, fast, local knowledge, frequent	Less independence	Every 6–12 months (illustrative)
External	Outside experts/regulators	Real independence, credibility, fresh eyes	Costly, slower, needs onboarding	Before rollout, submission, or after incidents

5.2 Finding severity

Severity	Meaning	Example (toy)
Critical	Unsafe now — restrict/withdraw until fixed	Tool used in ICU though never validated there
Major	Must fix on a deadline; may restrict scope meanwhile	≥80 subgroup at 60% vs 75% criterion
Minor	Improve; no immediate safety impact	Model card version number outdated

5.3 The four checklist domains

Domain	Question	Evidence examined
Data	Built on sound data?	Datasheet, cleaning log, label sample, permissions
Subgroup performance	Works for everyone?	Test scores broken down by subgroup vs criteria
Robustness	Survives real-world mess?	Imperfect-input tests, edge-case behaviour
Documentation	Provable?	Model card, version control, audit trail, SOP compliance

5.4 Re-audit triggers

Trigger	Example
Calendar	6–12 monthly routine (illustrative cadence)

Trigger	Example
Model change	Retrained v3.0 deployed
Data/environment change	New oncology ward; formulary revision
Incident	Missed ADR flagged by monitoring
Scope expansion	Rollout to sister hospital; adding paediatrics

6. Mechanisms

6.1 How an audit produces safety (mechanism)

```
Pre-set criteria
   → independent evidence gathering (4 domains)
      → findings (graded by severity)
         → recommendations (owner + deadline)
            → management decision (continue / restrict / withdraw)
               → fixes implemented
                  → verification + RE-AUDIT
The loop closes only when fixes are verified — a report alone changes nothing.
```

6.2 How bias becomes visible (mechanism)

```
Biased sampling (stage 2)          Subgroup fairness check (audit)
under-represented group     ──►    low subgroup score
        │                                  │
        ▼                                  ▼
model learns group weakly           gap vs other subgroups
        │                                  │
        └──────── root cause ──► fix: more data + retrain ──┘
The audit makes the invisible (sampling bias) visible (score gap)
and actionable (targeted data collection).
```

6.3 Independence mechanism

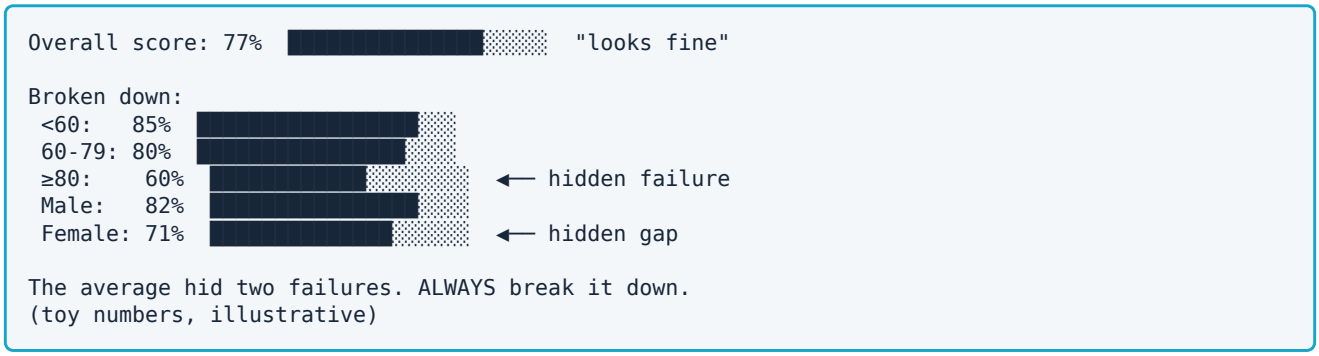
Auditors must not audit their own work: the builder cannot be the judge. Mechanism: separate reporting lines (quality cell reports to management, not to the AI team), documented conflicts of interest, and — for external audits — contractual independence. Without this, audits become self-congratulation.

7. Diagrams/flowcharts

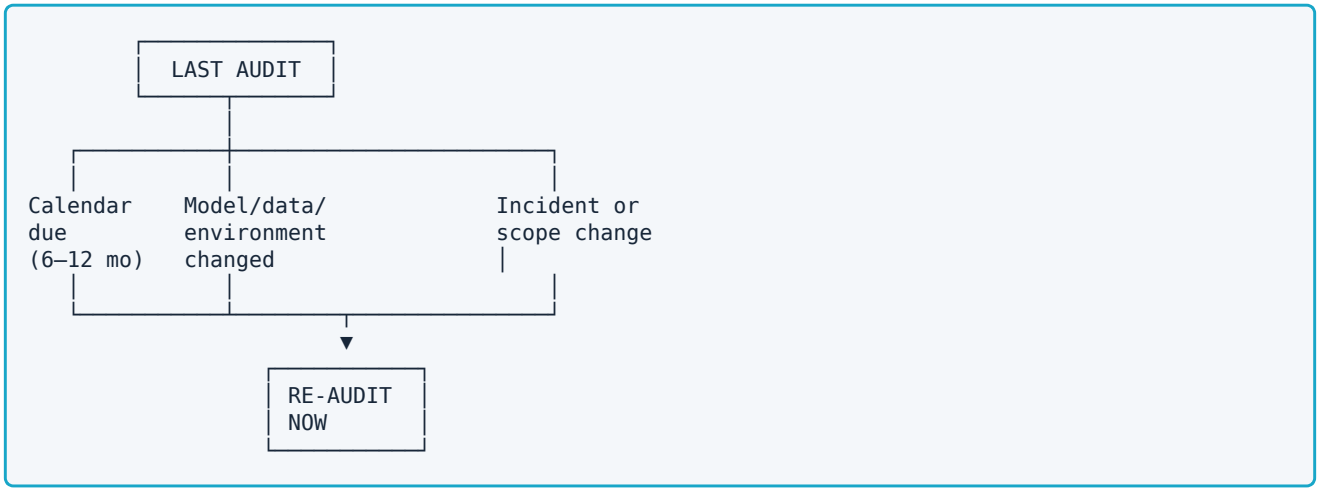
7.1 The audit flow



7.2 The hidden-subgroup problem (why overall scores mislead)



7.3 Re-audit triggers



8. Infographic concepts

Concept 1 — "The four-legged stool". A stool with four legs labelled Data, Subgroups, Robustness, Documentation; a pharmacist sitting on it representing patient safety. One short leg (a failed domain) = the stool wobbles. Caption: "Safety stands on all four."

Concept 2 — "The average lie". A large "77% ✓" badge cracking open to reveal subgroup bars, with the ≥80 bar glowing red at 60%. Caption: "The average hid the failure. Break it down."

Concept 3 — "Inspector's clipboard". A GMP-style inspector (hairnet, clipboard) inspecting not a factory but a computer showing the ADR tool, with checklist ticks: data ✓, subgroups ⚠, robustness ✓, documents ⚠. Caption: "Same discipline, new machine."

Concept 4 — "Audit calendar wheel". A yearly wheel with audit flags, plus lightning-bolt stickers for triggers (new version, new ward, incident, expansion). Caption: "Audits are photographs — retake them when the scene changes."

9. Tables

9.1 Internal vs external audit

Aspect	Internal audit	External audit
Conducted by	Own organisation's independent team	Outside party
Independence	Moderate (same employer)	High
Cost/speed	Low / fast	Higher / slower
Local knowledge	High	Must be learnt
Credibility	Good internally	High externally (regulators, public)
Best for	Routine hygiene (6–12 monthly)	Rollouts, submissions, incidents

9.2 Toy fairness audit — results vs criteria

Subgroup (toy)	Toy capture	Criterion	Verdict
Overall	77%	≥75%	✓ Pass
Age <60	85%	≥75%	✓ Pass
Age 60–79	80%	≥75%	✓ Pass
Age ≥80	60%	≥75%	✗ **Fail (major finding)**
Male	82%	≥75%	✓ Pass
Female	71%	≥75%	✗ **Fail (major finding)**
Max subgroup gap	25 pts (<60 vs ≥80)	≤10 pts	✗ **Fail**

9.3 Audit report structure at a glance

Section	One-line purpose
Title & scope	What system/version/period was examined
Auditors & independence	Who, and how independence was ensured
Criteria	Pre-decided pass/fail standards
Method & evidence	What was checked and how

Section	One-line purpose
Findings	Numbered observations with severity
Recommendations	Action + owner + deadline per finding
Overall opinion	Fit / restricted / withdraw
Follow-up plan	When re-audit happens

10. Worked toy examples

ILLUSTRATIVE TOY DATASET (not real patient data). All numbers below are invented for teaching.

Toy example 1 — the subgroup fairness check (full walk-through).

Setup: toy ADR-triage tool, toy test set of 1,000 patients with 100 true ADR cases (never used in training). Pre-decided criteria: overall capture ≥75%, every subgroup ≥75%, max gap ≤10 points.

Step 1 — compute overall: 77/100 = 77% ✓. Step 2 — break down by age and sex (table in section 4.4): ≥80 = 60% ✗, female = 71% ✗, gap 25 points ✗. Step 3 — findings: F1 (major): age ≥80 capture 60% < 75% criterion — oldest, most vulnerable patients worst served. F2 (major): female capture 71% < 75%; sex gap 11 points. F3 (minor): model card lists v1.1 though v2.0 is live. Step 4 — root-cause hypothesis: training set had few ≥80 and female ADR cases (sampling bias) — verify against the datasheet's representation table. Step 5 — recommendations: R1: suspend geriatric-ward use; collect targeted elderly data; retrain; re-validate against subgroup criteria. R2: investigate sex gap (check for label bias — were women's ADRs under-reported?). R3: update model card within 7 days (toy deadline). Step 6 — overall opinion: "Fit for continued use in general adult wards WITH corrective actions; geriatric use suspended; re-audit after retraining or in 6 months, whichever is earlier."

Teaching point: the audit did not just find low numbers — it converted them into *who is harmed, why, and what to do*.

Toy example 2 — robustness spot-test (illustrative).

Auditor feeds 50 toy records with defects: 20 with missing kidney values, 15 with drug-name misspellings ("Paracetmol"), 10 with duplicate entries, 5 with extreme-but-legal values (age 102).

- Result (toy): 44/50 handled gracefully (flagged "uncertain — pharmacist review"); 4 misspelled-drug records silently treated the drug as absent (no warning!) — **finding (major)**: misspellings cause silent feature loss; 2 extreme-age records scored normally ✓.
- Recommendation: add a spelling-tolerance dictionary and an explicit "unrecognised drug name" warning; re-test.

Teaching point: robustness testing finds the failures that clean test sets never show — because real wards are messy.

11. Analogies

1 **GMP/drug inspection.** The model audit is a drug-plant inspection for software: checklist, evidence, graded findings, corrective actions, re-inspection. Pharmacists have seen this film before.

2 **School report card with subject-wise marks.** Overall 77% is the aggregate; subgroup scores are subject-wise marks. A student failing maths while averaging 77% still needs help in maths — averages hide failures.

3 **Financial audit.** Just as money needs independent verification against books, model claims need independent verification against data. "Trust, but verify" — the auditor's motto.

4 **Vehicle fitness test.** A car needs periodic fitness certification; between tests, breakdowns or modifications trigger re-testing. Models need the same: calendar + triggers.

5 **Cricket umpire.** The auditor is the umpire — independent of both teams, judging against fixed rules, with the authority to stop play when it is unsafe.

12. Pharmacy relevance

- **Pharmacists are natural auditors.** Drug-utilisation review, prescription audit, and GMP inspection already train pharmacists in checklist thinking, evidence grading, and corrective-action follow-up. Model auditing is adjacent skill, not alien skill.
- **Subgroup fairness is a clinical issue.** Elderly polypharmacy patients — the group most at risk of ADRs — are exactly the group toy audits keep catching as under-served. A pharmacist knows clinically *why* that gap is dangerous, which makes the finding land with proper urgency.
- **PTC and ethics committees.** Pharmacy & Therapeutics Committees and institutional ethics committees are the natural homes for internal AI audits in hospitals — they already govern drug safety.
- **PvPI linkage.** Audit findings about under-detected ADRs should feed back into ADR reporting culture — a weak ADR register (Chapter 2) becomes visible through audit, creating pressure to improve reporting.
- **Career note.** "AI auditor for clinical tools" is a plausible near-future role for pharmacists who understand both medicines and this chapter's checklist — document-minded clinicians are rare and valuable.

13. India context

- **Who audits?** Large hospitals can form internal audit cells via the PTC/ethics committee; smaller hospitals can pool resources — e.g., a university pharmacy department acting as external auditor for district hospitals (a workable, low-cost model).
- **Geriatric and rural gaps are the likely findings.** Indian training datasets skew urban and younger; expect fairness audits to surface exactly the ≥80 and rural gaps from the toy example — plan data collection accordingly rather than being surprised.
- **Language subgroups matter.** A tool validated on English notes may underperform where notes mix Hindi/regional terms — include language/region as an audit subgroup where relevant.
- **CDSCO direction.** As clinical AI oversight evolves in India — as per the latest guidance, confirm the current edition with CDSCO — documented audits (internal and external) are the most likely first regulatory expectation. Start the habit before it is mandated.

- **ABDM-scale fairness.** As datasets grow to state/national scale under ABDM, subgroup audits must scale too: state, rural/urban, and socioeconomic subgroups become auditable — and findings there have policy-level importance.

14. Key points

- A **model audit** = systematic, independent check against pre-set criteria: safe, fair, fit for intended use?
- **Internal** audits: cheap, frequent, routine hygiene. **External** audits: independent, credible, for big decisions. Use both; auditors must never audit their own work.
- The **four-domain checklist**: data → performance across subgroups → robustness → documentation.
- **Fairness** is checked subgroup-by-subgroup; overall scores hide subgroup failures (toy: 77% overall hid 60% for ≥80).
- **Robustness** = graceful behaviour on messy real-world inputs; test it deliberately.
- Findings are graded (**critical/major/minor**) and paired with recommendations (action + owner + deadline).
- The report ends with an unambiguous **overall opinion**: fit / fit-with-actions / restricted / withdraw.
- **Re-audit triggers**: calendar due, model retrained, data/environment changed, incident, scope expansion.
- An audit report alone changes nothing — the loop closes when fixes are verified.

15. Common confusions

Confusion	Clarification
Audit vs validation	Validation (stage 5) tests the model before deployment; audit is the wider, periodic, independent review of the whole system (data, fairness, robustness, documents) during its life. Validation is a subset of what audits re-examine.
Internal audit vs self-review	Self-review = the builders checking themselves (useful, but not an audit). Internal audit = a *different, independent* team inside the organisation. Independence is the defining feature.
Fairness vs equal scores	Fairness does not always mean identical scores for all groups (some medical differences are real); it means no *systematic, unjustified* disadvantage — judged against pre-set, clinically reasoned criteria.
Bias (AI) vs bias (statistics)	Related: statistical bias = systematic error in estimation; AI bias = that systematic error producing unfair disadvantage for a group. In this subject they point the same way.
"Passed overall" = fair	No — the toy 77% overall passed while two subgroups failed. Overall pass is necessary, not sufficient.
Robustness vs accuracy	Accuracy = correct on clean test data. Robustness = stays sensible on messy real inputs. A model can be accurate yet brittle.

16. Common mistakes

1. **Auditing your own work** — the builders "audit" themselves; independence theatre.
2. **No pre-set criteria** — deciding pass/fail *after* seeing the scores (moving the goalposts).
3. **Reporting only overall scores** — the subgroup failures stay hidden; fairness never checked.
4. **One audit, forever** — treating the audit certificate as permanent; ignoring re-audit triggers.
5. **Findings without owners and deadlines** — recommendations evaporate; nothing is fixed.
6. **Skipping robustness testing** — clean test data only; the first messy ward record breaks the tool live.
7. **Documentation checked last or never** — unverifiable system passes on trust.
8. **Ignoring the root cause** — retraining on the same biased data and expecting different subgroup scores.

17. Mnemonics

The four audit domains — "D-P-R-D": "Data, People-groups, Robustness, Documents":

- **D**ata — sound?
- **P**eople-groups (subgroups) — fair for everyone?
- **R**obustness — survives mess?
- **D**ocuments — provable?

Memory hook: "Doctor Prescribes Rational Drugs" — an audit ensures the AI, like a good doctor, is **D**ata-sound, fair to all **P**atients, **R**obust, and **D**ocumented.

Finding severity — "CMM": Critical (stop now), Major (fix by deadline), Minor (improve).

Re-audit triggers — "TIMIS": Time elapsed, model retrained (version change), data/environment changed, Incident, Scope expanded. *(Pronounce "tim-iss" — "time is" up for re-audit.)*

Independence — "Never audit your own homework."

18. Exam summary

A **model audit** is a systematic, independent examination of an AI system against pre-decided criteria, answering: safe, fair, and fit for intended use? **Internal audits** (own organisation's independent team) are cheap and frequent — routine hygiene; **external audits** (outside party) bring real independence and credibility for rollouts, regulatory submissions, and incidents. The auditor works through four domains: **(1) data** — datasheet honesty, cleaning log, label verification, representation gaps; **(2) performance across subgroups** — every key score broken down by age, sex, ward (an overall 77% can hide 60% for patients ≥80 — the toy fairness example — so criteria must be per-subgroup with a max-gap rule); **(3) robustness** — deliberate testing with missing values, misspellings, edge cases, expecting graceful degradation; **(4) documentation** — current model card, version control, audit trail, SOP compliance. Findings are graded critical/major/minor, each paired with a recommendation (action + owner + deadline), and the report closes with an unambiguous overall opinion (fit / fit-with-actions / restricted / withdraw). **Re-audit** is triggered by calendar (6–12 months, illustrative), retraining, data/environment change, incidents, or scope expansion — because an audit is a photograph, not a permanent certificate. Pharmacists, with their GMP-audit and prescription-review training, are natural AI auditors, and documented audits are the

likeliest first regulatory expectation as CDSCO oversight of clinical AI evolves in India.

19. One-minute revision

- Audit = independent check vs pre-set criteria: safe, fair, fit for use?
- Internal (routine, cheap) + external (independent, credible) — never audit your own work.
- 4 domains: **D**ata, **P**eople-groups, **R**obustness, **D**ocuments ("Doctor Prescribes Rational Drugs").
- Fairness: break scores by subgroup — overall 77% hid 60% for ≥80 (toy).
- Findings: **C**ritical / **M**ajor / **M**inor + recommendation (action, owner, deadline).
- Report ends with overall opinion: fit / restricted / withdraw.
- Re-audit triggers: **TIMIS** — Time, version change, env change, Incident, Scope expansion.

20. MCQs

Q1. A model audit is: A) A casual opinion about the software B) A systematic, independent examination of an AI system against pre-defined criteria C) A financial review of the IT budget D) A training session for users

Q2. The key difference between internal and external audit is: A) Internal audits are always better B) Who conducts it — the organisation's own independent team vs an outside party — affecting independence and credibility C) The number of pages in the report D) External audits are never needed

Q3. Which is NOT one of the four audit checklist domains? A) Data B) Performance across subgroups C) Robustness D) Marketing and sales

Q4. In the toy fairness example, the overall capture was 77% (criterion ≥75%) but the audit still produced major findings because: A) 77% is too low B) The ≥80 subgroup scored 60% and females 71%, failing the per-subgroup criterion, with a 25-point gap C) The auditor disliked the tool D) Toy examples don't count

Q5. A "critical" finding means: A) The report was well written B) Unsafe now — restrict or withdraw use until fixed C) A minor spelling error D) The audit took too long

Q6. Robustness testing involves: A) Only checking the overall accuracy again B) Deliberately feeding imperfect inputs (missing values, misspellings, edge cases) to check graceful behaviour C) Making the computer run faster D) Re-training the model

Q7. When must a model be re-audited? (Choose the best answer) A) Never — one audit is permanent B) On calendar schedule, after retraining/version change, data or environment change, incidents, or scope expansion C) Only when the auditor feels like it D) Only after 10 years

Q8. The most likely root cause when one subgroup scores much lower than others is: A) The subgroup's patients are uncooperative B) Under-representation of that subgroup in training data (sampling bias) — to be verified against the datasheet C) The computer dislikes them D) Random chance, always

Q9. An audit report must end with: A) Vague praise B) An unambiguous overall opinion (fit / fit-with-actions / restricted / withdraw) plus a follow-up plan C) A blank page D) An advertisement

Q10. "Never audit your own homework" means: A) Students should not study B) Auditors must be independent of the team that built the system C) Homework is useless D) Only external audits count

Answer key: Q1–B, Q2–B, Q3–D, Q4–B, Q5–B, Q6–B, Q7–B, Q8–B, Q9–B, Q10–B

21. Viva/application questions

1 Define model audit. How does it differ from model validation (lifecycle stage 5)?

2 Compare internal and external audits: strengths, limits, and when to use each.

3 Explain the four-domain audit checklist with one checkable question per domain.

4 Walk through the toy fairness example: compute the subgroup captures, apply the criteria, and state the findings.

5 Why can an overall score of 77% still mean the tool is unfair? What is the correct reporting practice?

6 Design three robustness tests for an ADR-triage tool and state what "graceful" behaviour looks like for each.

7 A finding reads: "Model card v1.1 live while v2.0 deployed." Grade its severity and write the recommendation (action + owner + deadline).

8 List five re-audit triggers with a pharmacy example for each.

9 Sketch the audit report structure and explain what goes in "overall opinion".

10 Your hospital's PTC asks you to plan the first internal audit of a deployed dose-checking tool. Outline scope, team (with independence), criteria, and timeline.

22. References

- WHO guidance on ethics and governance of artificial intelligence for health — as per the latest guidance, confirm the current edition (accountability and oversight).
- EU AI Act — as per the latest guidance, confirm the current edition (conformity assessment and monitoring for high-risk AI).
- CDSCO (Central Drugs Standard Control Organization), India — as per the latest guidance, confirm the current edition.
- Pharmacovigilance Programme of India (PvPI) — public information on ADR reporting in India.
- Standard B.Pharm pharmaceutical jurisprudence / GMP study material (audits, inspection, documentation) and your university syllabus for BP801T.

End of PART A — Unit I (chapters 1–5).

PART B — Units II–III: Regulatory Framework & Explainable AI; AI in Pharmacy Automation & Supply Chain (chapters 6–9)