BP801T · B.Pharm Semester VIII · Theory Notes

Ethical Considerations and Translational Applications of AI in Pharmacy

Unit V — Guided Project: Translational AI in Pharmacy (6 hours)

Unit V (6 hours)

Pills & Python Notes

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

Syllabus coverage: Unit V (6 hours).

PILLS & PYTHON

Pharmacy • Data • AI • Healthcare

Instagram @pillsandpython · pillsandpython.com · YouTube @PillsAndPython

CHAPTER 12: Guided Project Framework — Implementing Supervised ML on Pharmacy Data

About this chapter: This is a complete, do-it-yourself project guide. You will not build a real medical AI here — you will build a small, honest **toy project** on invented data, following the same steps professionals follow. Every dataset and result in this chapter is labelled illustrative.

1. Learning Objectives

After studying this chapter, you will be able to:

- Choose a suitable supervised-ML mini-project from six pharmacy options.
- Follow the seven-stage project framework: define problem → collect/prepare toy data → choose model → train/validate → evaluate → document → present.
- Explain train/test split and cross-validation in simple terms.
- Name the basic performance metrics (accuracy, precision, recall) at concept level.
- Judge which **risk tier** your project model would fall in and what documentation it needs.
- Identify ethical risks (bias, privacy) and apply a bias-check checklist.
- Fill a structured **AI implementation plan** template (illustrative example provided).
- Avoid the most common student project pitfalls.
- Understand how your project will be evaluated (rubric concept).

2. Definitions

- **Project:** A planned piece of work with a goal, steps, and a deadline — here, building and documenting a small ML model.
- **Supervised ML (supervised machine learning):** Teaching a computer using examples that already have the correct answers ("labels"). Like showing a child 100 photos labelled "tablet" or "capsule" until the child can label a new photo alone.
- **Label:** The correct answer attached to each training example (e.g., "will have side effect: yes/no").
- **Feature:** One input column the model learns from (e.g., age, dose, number of medicines).
- **Training:** The phase where the model studies labelled examples and adjusts itself to get answers right.
- **Validation:** Checking the model's skill on examples it has NOT studied, to see if it truly learned or just memorised.
- **Train/test split:** Dividing your data into two parts — one for training, one for honest testing (typically 70–80% train, 20–30% test).

- **Cross-validation:** Repeating train/test several times on different slices of data and averaging the scores, so the result is not a lucky accident of one split.
- **Accuracy:** Fraction of all predictions the model got right.
- **Precision:** Of all the cases the model flagged "yes," how many were truly "yes"? (Avoids false alarms.)
- **Recall:** Of all the truly "yes" cases, how many did the model catch? (Avoids missing real cases.)
- **Risk tier:** A category describing how much harm an AI system could cause if wrong — higher tier = stricter rules.
- **Bias-check checklist:** A fixed list of questions asked before accepting a model's results, to catch unfairness.
- **Implementation plan:** A written document describing what the AI will do, on what data, with what safeguards, before it is built or used.
- **Toy data (in this chapter):** Small invented datasets you create yourself for learning — never real patient records.

3. Terminology Table

Term	Short meaning	Think of it as
Supervised ML	Learning from labelled examples	A student studying solved question papers
Label	Correct answer on training data	The answer key
Feature	Input column used for learning	The clues in a detective case
Train/test split	Separate study material from exam paper	Study notes vs. final exam
Cross-validation	Multiple practice exams, averaged	Best-of-five match series
Accuracy	Overall % correct	Total marks percentage
Precision	Correctness of "yes" alarms	How often the fire alarm means real fire
Recall	Completeness of catching "yes" cases	How many real fires the alarm caught
Risk tier	Harm category of the AI	Medicine schedule (H, X) — stricter for riskier
Implementation plan	Written "what/why/how/safeguards"	A prescription before dispensing

4. Explanation (from absolute zero)

4.1 What is this project about?

By the end of this course you should not only *know about* AI — you should have *built* something small with it. This chapter guides you through a **supervised ML mini-project on pharmacy data**: you pick a problem, create a small toy dataset (invented by you, e.g., 50–200 rows in a spreadsheet), train a simple model using a no-code or low-code tool, test it honestly, and write it up like a professional.

You are **not** expected to write complex code from scratch. Spreadsheet tools, free no-code ML platforms, or a few lines of guided Python — any honest route is acceptable. What matters is the **thinking**: clean problem definition, honest testing, documented limits, ethical checks.

4.2 The six project options

Choose ONE. Each is described with its goal, toy data idea, and what "success" looks like:

Option 1 — ADR prediction (adverse drug reaction). Goal: predict which patients are more likely to experience a side effect. Toy data: 100 invented patient rows — age, number of medicines, kidney function category, and label "ADR occurred: yes/no." Success: a model that flags high-risk combinations better than guessing.

Option 2 — Drug stability forecasting. Goal: predict whether a formulation stays stable (passes quality tests) under given storage conditions. Toy data: 80 invented batches — temperature, humidity, months stored, packaging type, label "stable: yes/no." Success: model identifies which storage factor matters most.

Option 3 — Medicine demand prediction. Goal: predict next month's demand for a medicine at a pharmacy. Toy data: 24 months of invented monthly sales plus festival-season flags. Success: a forecast closer to actual toy sales than "same as last month."

Option 4 — QC failure prediction. Goal: predict which production batches are likely to fail quality-control tests. Toy data: 120 invented batches — raw-material supplier grade, mixing time, operator shift, label "QC pass/fail." Success: model catches most failing batches (high recall) without too many false alarms.

Option 5 — Disease-risk scoring. Goal: build a simple risk score for a condition (e.g., diabetes risk: low/medium/high). Toy data: 150 invented persons — age, BMI category, family history, activity level, label "risk band." Success: sensible risk bands that a health worker could explain.

Option 6 — Medication adherence prediction. Goal: predict which patients are likely to miss doses. Toy data: 100 invented patients — age, distance to pharmacy, number of daily doses, side-effect history, label "adherent: yes/no." Success: model identifies patients needing counselling reminders.

How to choose: Pick the option whose data you can most realistically invent *sensibly* (you understand the domain), and whose result you could explain to a pharmacist. Interest beats complexity — a simple, honest project beats a fancy, confused one.

4.3 The seven-stage framework (overview)

```
1. DEFINE the problem      → one clear question + who decides what with the answer
2. COLLECT & PREPARE toy data → spreadsheet, clean it, label it
3. CHOOSE the model         → start simple (decision tree / logistic regression concept)
4. TRAIN & VALIDATE         → train/test split, cross-validation concept
5. EVALUATE                 → accuracy, precision, recall; read the mistakes
6. DOCUMENT                 → implementation plan + limits + ethics
7. PRESENT                  → explain to classmates like a professional
```

Each stage is detailed below. Follow them in order — skipping stage 1 or 6 is the most common reason projects fail.

4.4 Stage 1 — Define the problem (the most important stage)

Write three sentences:

1. **Question:** "Can we predict X for Y?" (e.g., "Can we predict which batches fail QC, to prioritise testing?")
2. **User:** Who will use the answer? (e.g., "the QC in-charge")
3. **Action:** What will they do differently with the model's output? (e.g., "test flagged batches first, investigate their process parameters")

If you cannot write sentence 3, your problem is not ready — a model nobody acts on is a toy with no purpose. (Your data may be toy, but your *purpose* must be real.)

4.5 Stage 2 — Collect and prepare toy data

- **Create 50–200 rows** in a spreadsheet. Columns = features + one label column.
- **Invent sensibly:** use your pharmacy knowledge. Example: in ADR data, older age + more medicines → more "yes" labels. The data must carry a *learnable pattern*; random labels teach nothing.
- **Clean:** no empty cells, consistent spellings ("Yes" vs "yes" must match), numbers as numbers.
- **Document your invention:** write one paragraph: "This is toy data I invented for learning. It does not come from real patients."
- **Never use real patient data** for a classroom project — privacy rules forbid it, and toy data teaches the same skills.

What a toy dataset looks like (ILLUSTRATIVE TOY DATASET — Option 1, ADR prediction, first 5 of 100 rows):

Age	Medicines count	Kidney function	ADR occurred (label)
34	2	Normal	No
67	6	Reduced	Yes
45	3	Normal	No
72	8	Reduced	Yes
52	4	Normal	No

Notice the *pattern*: older age + more medicines + reduced kidney function → "Yes." You invent the pattern deliberately (it mirrors real clinical logic), then the model must *discover* it. That is the whole exercise in miniature.

4.6 Stage 3 — Choose the model (start simple)

For a first project, simple models win:

- **Decision tree (concept):** a flowchart the computer learns from data — "if age > 60 and medicines > 5 → high ADR risk." Easy to draw and explain.
- **Logistic regression (concept):** the classification cousin of the regression in Chapter 10 — weighs each feature and outputs a probability.

Rule: **start with the simplest model that could work.** You can mention fancier models in "future work." Examiners reward a simple model fully understood over a complex one copied blindly.

4.7 Stage 4 — Train and validate

- **Split:** randomly keep 20–30% of rows aside as the **test set** — the model never sees these during training. This is your honest exam.
- **Train** on the remaining 70–80%.

- **Cross-validation (concept):** instead of one split, rotate which slice is the test slice 5 times and average the scores. This guards against a lucky/unlucky single split. With tiny toy data, even 3-fold cross-validation is fine to describe.
- **Golden rule:** never test on training data — that is like writing an exam from your own notes with the answers visible. Scores will look perfect and mean nothing.

4.8 Stage 5 — Evaluate (read the mistakes, not just the score)

Compute on the test set:

- **Accuracy:** overall % correct.
- **Precision:** of flagged "yes," how many truly "yes."
- **Recall:** of truly "yes," how many caught.

Then build a tiny **confusion table** (toy example):

	Actually YES	Actually NO
Predicted YES	14 (caught)	6 (false alarm)
Predicted NO	4 (missed)	76 (correctly clear)

Accuracy = (14+76)/100 = 90%. Precision = 14/(14+6) = 70%. Recall = 14/(14+4) ≈ 78%.

Which metric matters depends on the problem: for QC failure, **recall** matters most (missing a bad batch is dangerous); for adherence reminders, **precision** matters (don't harass adherent patients with false alarms). Say *which* metric you prioritise and *why* — that sentence shows professional judgement.

Also list 2–3 specific mistakes the model made and hypothesise why ("it missed batch #42 because the supplier-grade column was ambiguous").

4.9 Stage 6 — Document (the implementation plan)

Write the structured implementation plan (template + filled example in section 10). Documentation is what separates a student experiment from a professional deliverable: it lets someone else reproduce, audit, and challenge your work.

4.10 Stage 7 — Present

A 5–8 minute presentation: problem → toy data → method → result → limitation → ethical check → what you'd do next. Use one diagram (your framework flowchart) and one table (your confusion table). End with your biggest limitation — honesty impresses examiners more than big accuracy numbers.

5. Classifications

5.1 Classification of the six project options

```
SUPERVISED ML MINI-PROJECTS (pharmacy)
├── Patient-safety focused
│   ├── 1. ADR prediction          (who is at risk of side effects?)
│   └── 6. Adherence prediction    (who may miss doses?)
├── Quality/manufacturing focused
│   ├── 2. Stability forecasting    (will the batch stay stable?)
│   └── 4. QC failure prediction    (which batch may fail tests?)
└── Planning focused
    ├── 3. Demand prediction        (how much stock next month?)
    └── 5. Disease-risk scoring     (who needs screening first?)
```

5.2 Classification of risk tiers for project models

Tier	Meaning	Example project model	Documentation needed
Low	Supports learning/planning; no direct patient decision	Demand prediction, stability toy model	Problem statement, data description, metrics, limits
Medium	Could influence care if misused; needs human review	Adherence prediction, risk scoring	Above + bias check, human-in-the-loop statement, review plan
High	Would directly drive diagnosis/treatment if deployed	ADR prediction used as a clinical alert	Above + clinical validation plan, expert sign-off, monitoring plan, regulatory consultation — as per the latest guidance, confirm the current edition

Key takeaway: Your classroom toy model stays in the Low tier *because it is a toy*. The tier table teaches you what a *real* deployment would demand — never deploy a classroom model on real patients.

5.3 Classification of validation methods

Method	What it does	When to use
Single train/test split	One honest exam	Minimum for every project
Cross-validation (3–5 fold)	Multiple exams, averaged	When data is small or you want stable scores
External check (concept)	Test on data from another source/time	Real deployments; beyond classroom scope

6. Mechanisms

6.1 Mechanism — how a decision tree learns (concept)

1. Look at all training rows; find the single feature + cutoff that best separates yes/no (e.g., "medicines > 5").
2. Split rows into two branches; repeat inside each branch ("age > 60?").
3. Stop when branches are small or pure enough.
4. The result is a flowchart; a new case just walks down the branches.

Key takeaway: A decision tree is a flowchart learned from data — you can draw it, explain it, and argue with it. That explainability is why it suits first projects.

6.2 Mechanism — how train/test split prevents self-deception

1 Hide the test rows before training (lock them in a drawer).

2 Train only on the rest.

3 Open the drawer; score the model on unseen rows.

4 The score now measures *learning*, not *memory*.

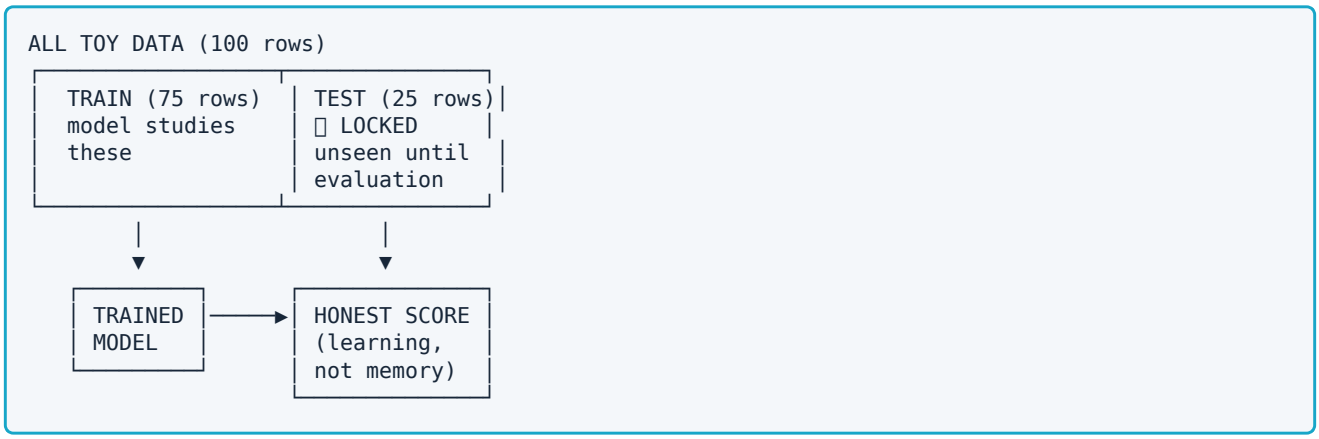
Key takeaway: Testing on training data measures memory; testing on unseen data measures learning. Only the second counts.

7. Diagrams / Flowcharts

Diagram 12A — The seven-stage project framework



Diagram 12B — Train/test split (the locked drawer)



8. Infographic Concepts

Infographic 1 — "The locked drawer." A student studying at a desk (training data, books open) next to a locked drawer labelled "test set — do not open until exam day." Caption: "Study on the training set. Prove on the test set."

Infographic 2 — "Three lenses on performance." Three magnifying glasses over the same toy results: one labelled Accuracy ("overall % right"), one Precision ("how many alarms were real"), one Recall ("how many real cases caught"). Caption: "One score never tells the whole story — look through all three lenses."

Infographic 3 — "Project mountain." A mountain climb with seven base camps labelled 1–7 (Define → Present), a flag at the summit labelled "Documented, ethical, honest project." A side path marked "shortcut: skip documentation" leads off a cliff. Caption: "There are no shortcuts past documentation."

9. Tables

Table 12.1 — Choosing your project option (quick guide)

Option	Needs what domain knowledge	Easiest toy data to invent	Watch out for
1. ADR prediction	Side-effect basics	Patient rows	Don't invent implausible drug–ADR pairs
2. Stability forecasting	Storage, packaging	Batch rows	Keep factor effects sensible (heat ↑ → fail ↑)
3. Demand prediction	Seasons, festivals	Monthly sales	Add a festival flag; don't make sales random
4. QC failure prediction	Manufacturing basics	Batch rows	Balance pass/fail (not 119 pass, 1 fail)
5. Disease-risk scoring	Risk factors	Person rows	Risk bands must be explainable
6. Adherence prediction	Patient behaviour	Patient rows	Avoid blaming language in labels

Table 12.2 — Metric priority by project option

Option	Prioritise	Why
1. ADR prediction	Recall	Missing a real ADR risk harms patients

Option	Prioritise	Why
2. Stability forecasting	Precision	False "unstable" alarms waste good batches
3. Demand prediction	(Error size, concept)	Being close matters more than yes/no
4. QC failure prediction	Recall	A missed bad batch reaches patients
5. Disease-risk scoring	Recall at high-risk band	Don't miss people needing screening
6. Adherence prediction	Precision	Don't wrongly label adherent patients

Table 12.3 — Bias-check checklist (ask before accepting results)

#	Check question	If "yes," do this
1	Does my toy data miss an important group (elderly, rural, women)?	Note it as a limitation; don't claim general use
2	Could a feature be a proxy for something unfair (e.g., pincode ≈ income)?	Remove or justify the feature
3	Is one label far rarer than the other (<10%)?	Report metrics per group, not just overall accuracy
4	Would errors hurt one group more than another?	Prioritise the metric protecting that group
5	Am I using any real patient data?	Stop. Replace with toy data immediately
6	Could someone act on this model without expert review?	Add "human-in-the-loop required" to the plan
7	Have I written down all limitations honestly?	If no, the documentation is incomplete

10. Worked Toy Examples

Toy Example 12.1 — Mini confusion-table calculation (ILLUSTRATIVE TOY DATASET)

A toy QC-failure model tested on 100 unseen batches:

	Actually FAIL	Actually PASS
Predicted FAIL	14	6
Predicted PASS	4	76

- Accuracy = (14 + 76) ÷ 100 = **90%**
- Precision = 14 ÷ (14 + 6) = **70%** (7 of 10 alarms real)
- Recall = 14 ÷ (14 + 4) ≈ **78%** (caught ~8 of 10 real failures)

Interpretation for the QC in-charge: "The model catches most bad batches but raises some false alarms — use it to *prioritise* testing order, not to skip testing." All numbers invented for teaching.

Toy Example 12.2 — Filled implementation-plan template (ILLUSTRATIVE)

STRUCTURED AI IMPLEMENTATION PLAN — filled example (illustrative toy project)

1. Project title: Toy QC-failure prioritisation model for a teaching exercise

2. Problem & user: Predict which tablet batches are likely to fail QC, so the QC in-charge can test high-risk batches first. *User:* QC in-charge (in a real plant). *Action:* reorder the testing queue.

3. Data: Toy dataset of 120 invented batches (features: supplier grade A/B/C, mixing time minutes, shift day/night; label: QC pass/fail). Source: invented by the student for learning — no real patient or company data. *Status: illustrative toy dataset.*

4. Model: Decision tree (chosen for explainability — the tree can be drawn and discussed).

5. Validation: 75% train / 25% locked test split; 3-fold cross-validation described; metrics: accuracy, precision, recall — recall prioritised (missing a bad batch is the costliest error).

6. Toy results: Accuracy 90%, precision 70%, recall ~78% on the toy test set (illustrative).

7. Limitations: Toy data invented by one student; only 120 rows; no real-world validation; supplier grades simplified; cannot be used for real decisions.

8. Risk tier: Low as a classroom toy. If ever deployed for real: High tier — would need clinical/quality expert sign-off, prospective validation, and monitoring, as per the latest guidance — confirm the current edition.

9. Ethical checks (bias checklist): No real data used ✓; night-shift feature checked as possible proxy for staffing, not worker blame ✓; errors affect testing order only, human does final testing ✓; limitations documented ✓.

10. Human-in-the-loop: The model only *reorders* testing; a qualified QC analyst still performs every test and makes the pass/fail decision.

11. Next steps: Collect larger, real, consented data under proper approvals; compare with logistic regression; add calibration.

Toy Example 12.3 — Choosing the wrong metric (ILLUSTRATIVE TOY DATASET)

Two toy adherence models, each 90% accurate on toy data:

- Model X: precision 60%, recall 95% → catches nearly all non-adherent patients but falsely flags many adherent ones (annoying reminders to the wrong people).
- Model Y: precision 95%, recall 60% → reminders go only to the truly non-adherent, but 40% of needy patients are missed.

Lesson: identical accuracy, opposite behaviour. **Always report precision and recall separately and choose based on the cost of each error type.** (Toy comparison for teaching.)

11. Analogies

1 **Supervised ML = studying solved question papers.** Labels are the answer key; the exam is the unseen test set.

2 **Train/test split = practice matches vs. final match.** You train in practice; the trophy depends on the final you haven't played.

3 **Testing on training data = copying in the exam.** Full marks, zero learning — and everyone knows it.

4 **Cross-validation = best-of-five series.** One lucky win proves little; winning the series proves skill.

5 **Precision vs recall = smoke alarm tuning.** Too sensitive (high recall): alarms at burnt toast. Too strict (high precision): silent during a real fire. Tune to the cost of each mistake.

6 **Implementation plan = prescription before dispensing.** No pharmacist dispenses without a prescription; no AI should be built without a plan.

7 **Risk tiers = drug schedules.** Like Schedule H/X restricts riskier drugs, higher AI risk tiers demand stricter controls.

12. Pharmacy Relevance

This chapter is the bridge from *learning about* AI to *doing* AI as a pharmacist:

- **You already own the data:** dispensing logs, stock registers, QC records, vaccination registers — pharmacists generate the raw material of every project option.
- **You already own the judgement:** knowing which metric matters (recall for QC, precision for adherence) is pharmacy judgement, not coding skill.
- **You will be the human-in-the-loop:** in every real deployment, the pharmacist reviews, counsels, and takes responsibility — the model only advises.
- **Career edge:** a B.Pharm graduate who has actually built, validated, and documented a small ML project — honestly, with limits stated — stands out in industry, hospital, and regulatory roles alike.

13. India Context

- **Language and access:** If your project involves patient-facing outputs (adherence reminders, risk messages), plan for Indian linguistic diversity and low digital literacy — a model is only as useful as its last-mile communication.
- **Data reality:** Indian pharmacy data is often paper-based or in non-standard formats. Your toy project should note this: "real deployment would first need digitisation and standardisation (ABDM-aligned records, as per the latest guidance — confirm the current edition)."
- **Regulatory path:** Any AI that influences diagnosis, treatment, or quality decisions in India would face regulatory scrutiny — as per the latest guidance, confirm the current edition. Your implementation plan's risk-tier section exists precisely to show you understand this.
- **Equity:** Bias checks matter especially in India — models trained on urban, private-hospital data must not be assumed to work for rural PHC populations. State this limitation explicitly in every project.
- **Opportunity:** Pharmacists in India are often the most accessible health professionals; adherence and demand-prediction projects built with pharmacists' data have genuine last-mile value.
- **Documentation habit:** Indian regulators increasingly expect documented, auditable processes (as per the latest guidance — confirm the current edition). Practising implementation plans now builds the exact habit industry and hospital QA departments require later.

14. Key Points

- Six project options: ADR prediction, stability forecasting, demand prediction, QC failure prediction, disease-risk scoring, adherence prediction.
- Seven stages: define → collect/prepare → choose model → train/validate → evaluate → document → present. Never skip 1 or 6.
- Supervised ML learns from labelled examples; start with simple, explainable models (decision tree).
- Train/test split: lock 20–30% away for honest testing; cross-validation averages multiple splits.
- Report accuracy, precision, AND recall; prioritise the metric matching the costliest error.
- Risk tiers (low/medium/high) decide documentation strictness; classroom toys stay low tier — never deploy on real patients.
- Bias-check checklist: missing groups, proxy features, rare labels, unequal harm, real data, human-in-the-loop, honest limits.
- Implementation plan = the professional deliverable (filled illustrative example in section 10).
- Common pitfalls: testing on training data, quoting toy results as facts, skipping documentation, ignoring bias.

15. Common Confusions

1 **"Higher accuracy = better model."** Not always. A 95%-accurate model can be useless if it misses every rare but critical case (e.g., all QC failures). Always check precision and recall.

2 **"The model is objective, so it can't be biased."** Wrong. A model learns the biases hidden in its training data. Objectivity must be *checked*, not assumed — hence the bias checklist.

3 **"Cross-validation means the model is validated for real use."** No — it only means the score is stable on *your* data. Real-world validation needs new, independent data.

4 **"Toy project = worthless project."** Wrong. A toy project done honestly teaches the complete professional workflow; a "real-data" project done carelessly teaches bad habits.

5 **"Documentation is just formality."** Documentation is the product. In regulated healthcare, an undocumented model is an unusable model.

16. Common Mistakes

1 **Testing on the training data** — perfect scores that mean nothing. Lock the test set away.

2 **Inventing labels randomly** — the data must carry a learnable pattern, or the model learns noise.

3 **Using real patient data** for a classroom project — a privacy violation; always use toy data.

4 **Quoting toy results as findings** ("my model achieved 90% accuracy" without adding "on my own invented toy dataset").

5 **Choosing the fanciest model** instead of the most explainable one.

6 **Skipping the bias checklist** — "my data is small so bias doesn't matter" is false; small data can be more biased.

7 **No human-in-the-loop statement** — every healthcare AI plan must say who reviews the output.

8 **Presenting without stating limitations** — examiners trust students who name their weaknesses.

9 **Imbalanced toy data** — e.g., 95 "pass" rows and 5 "fail" rows; the model can score 95% by always predicting "pass." Keep classes reasonably balanced or say explicitly how you handled imbalance.

10 **Feature leakage** — accidentally including a column that *contains* the answer (e.g., a "QC inspector's final remark" column when predicting QC outcome). The model scores 100% and learns nothing. Ask: "Would this information exist *before* the outcome in real life?" If not, delete the column.

17. Mnemonics

- **Seven stages — "Please Clean Clothes, Then Examine, Dress, Present"**: **P**roblem define, **C**ollect & prepare, **C**hoose model, **T**rain & validate, **E**valuate, **D**ocument, **P**resent. (Or simply: **D**efine, **C**ollect, **C**hoose, **T**rain, **E**valuate, **D**ocument, **P**resent — "Do Careful Checks To Earn Distinction, Pharmacist!")
- **Three metrics — "APR"**: **A**ccuracy (overall), **P**recision (alarm quality), **R**ecall (catch completeness).
- **Bias checklist — "GROUPS"**: **G**roups missing? p**R**oxy features? **O**utcome rare? **U**nequal harm? **P**rivate real data? **S**upervision (human-in-the-loop)?
- **Golden rule — "Lock the drawer"**: test set stays locked until evaluation day.

18. Exam Summary

Chapter 12 is a complete guide to a supervised-ML mini-project on pharmacy data using toy datasets. Six project options are offered: ADR prediction, drug stability forecasting, medicine demand prediction, QC failure prediction, disease-risk scoring, and medication adherence prediction. The seven-stage framework runs: define the problem (question + user + action), collect and prepare 50–200 rows of sensibly invented toy data, choose a simple explainable model (decision tree / logistic regression concept), train and validate with a 70–80/20–30 train/test split plus cross-validation concept, evaluate with accuracy, precision, and recall plus a confusion table (prioritising the metric that matches the costliest error), document via a structured AI implementation plan, and present in 5–8 minutes ending with the biggest limitation. Risk tiers (low/medium/high) determine documentation strictness — classroom toy models stay low tier and must never be deployed on real patients; real deployments need expert sign-off and regulatory consultation as per the latest guidance. The bias-check checklist (GROUPS: missing groups, proxy features, rare outcomes, unequal harm, private real data, human supervision) guards against unfair models. Common pitfalls include testing on training data, quoting toy results as facts, and skipping documentation. Evaluation rewards honest problem definition, clean methodology, correct validation, ethical checks, and clearly stated limitations over raw accuracy numbers.

19. One-Minute Revision

Pick one of six options (ADR, stability, demand, QC, risk score, adherence). Seven stages: **D**efine, **C**ollect, **C**hoose, **T**rain, **E**valuate, **D**ocument, **P**resent — "Do Careful Checks To Earn Distinction, Pharmacist!" Lock 20–30% as test set ("lock the drawer"); report **APR** (Accuracy, Precision, Recall); prioritise the metric matching the costliest error. Risk tiers decide paperwork; toys stay low tier. Bias check = **GROUPS**. Never use real patient data; never quote toy results as facts; always name your biggest limitation.

20. MCQs (with Answer Key)

Q1. In supervised ML, a "label" is: a) The name of the software used b) The correct answer attached to each training example c) A barcode on a medicine strip d) The title of the project report **Answer: b**

Q2. The first stage of the project framework is: a) Choose the model b) Define the problem (question + user + action) c) Train the model d) Present the results **Answer: b**

Q3. A train/test split is used to: a) Make training faster b) Test the model on unseen data so the score measures learning, not memory c) Double the dataset size d) Encrypt the data **Answer: b**

Q4. Cross-validation means: a) Validating with a foreign dataset b) Repeating train/test on different data slices and averaging scores c) Crossing out invalid rows d) Training two models at once **Answer: b**

Q5. For QC-failure prediction, the most important metric is usually: a) Accuracy only b) Recall — missing a bad batch is the costliest error c) Training speed d) Model file size **Answer: b**

Q6. Precision answers: a) How many real cases were caught b) Of all "yes" alarms, how many were truly "yes" c) How fast the model runs d) How much data was used **Answer: b**

Q7. A classroom toy ADR-prediction model belongs to which risk tier, and why? a) High — because ADR is serious b) Low — because it is a toy for learning, never deployed on real patients c) Medium — because students built it d) No tier applies to student work **Answer: b**

Q8. Which bias-check question asks about pincode acting as a proxy for income? a) Groups missing? b) Proxy features? c) Unequal harm? d) Private real data? **Answer: b**

Q9. The "human-in-the-loop" statement in an implementation plan says: a) Which human wrote the code b) Who reviews the model's output before any action is taken c) How many humans are in the dataset d) That humans are better than AI **Answer: b**

Q10. The most common reason student ML projects fail is: a) Using too little mathematics b) Skipping problem definition or documentation c) Choosing pharmacy data d) Using a decision tree **Answer: b**

21. Viva / Application Questions

1 Explain supervised ML to a classmate using the "solved question papers" analogy.

2 Why must the test set be "locked" before training? What goes wrong otherwise?

3 Calculate accuracy, precision, and recall from a confusion table I give you.

4 Your toy model shows 98% accuracy but 40% recall on QC failures — is it good? Why?

5 Which project option would you choose and why? Define its problem in three sentences.

6 Walk through the bias-check checklist (GROUPS) for your chosen option.

7 What risk tier would your project fall in if deployed for real? What extra documentation would it need?

8 Explain the difference between precision and recall using the smoke-alarm analogy.

9 Why is a decision tree a good first model for a student project?

10 Present your implementation plan's "limitations" section — what is your project's biggest weakness?

11 A classmate used real hospital prescriptions for their project. What do you tell them?

12 How does the pharmacist act as "human-in-the-loop" in a real deployment of your model?

22. References

- Standard machine-learning and data-science textbooks on supervised learning, model validation, and performance metrics — as per the latest editions, confirm the current edition.
- Regulatory guidance on AI/ML-based medical devices and clinical decision support — as per the latest guidance, confirm the current edition (national regulator and international guidance documents).
- World Health Organization. Ethics and governance of artificial intelligence for health — as per the latest guidance, confirm the current edition.
- All datasets, results, and the filled implementation-plan example in this chapter are illustrative toy examples created for teaching; they are not real findings and must not be cited as evidence.