

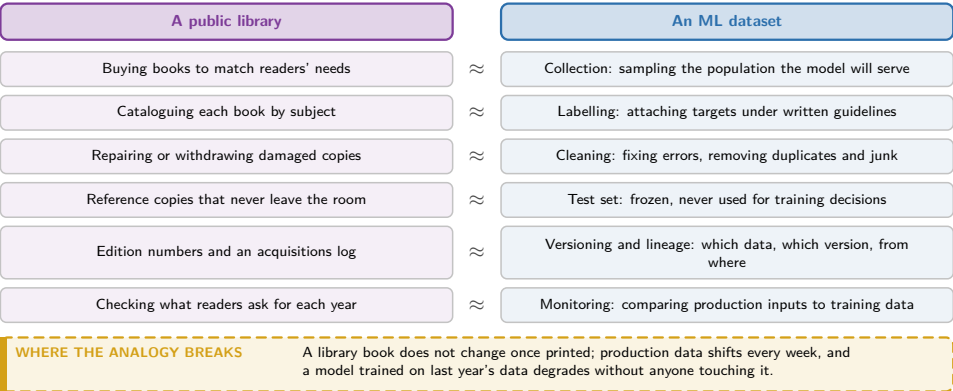


The Data Lifecycle for ML

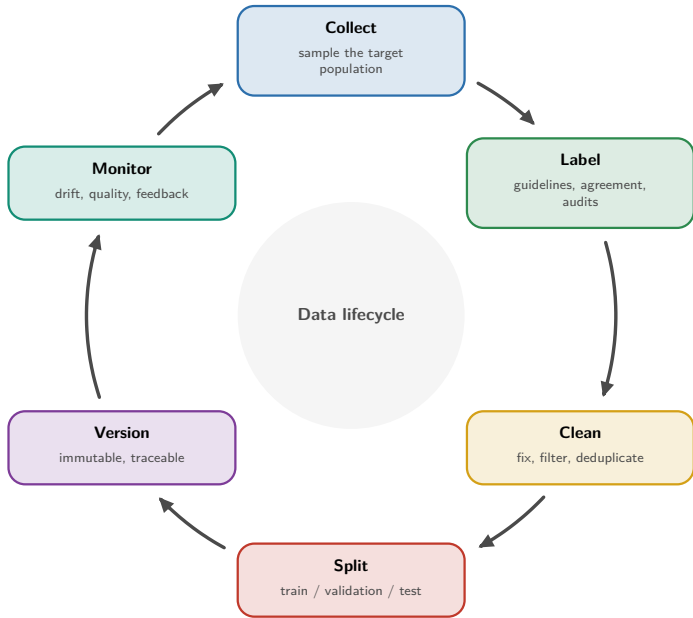
Data folder | Collection, labelling, cleaning, splitting, versioning, monitoring; data-centric AI; datasheets for datasets (Gebru et al.), data cards and Croissant metadata

#Data #Lifecycle #DataCentricAI #Datasheets

The lifecycle at a glance



A dataset is built like a library collection: the same jobs, with the same failure if one is skipped



The six stages loop: monitoring findings send you back to collection or labelling. Every stage is a place where quality is won or lost

Stage	Output	Typical failure	Check
Collection	raw records with provenance and licence	sampling bias: data from one hospital, one country, one season	compare collected vs target population on key attributes
Labelling	labels + guideline version	ambiguous guideline, tired or unskilled labellers	inter-annotator agreement, golden-set audits (sheet 04)
Cleaning	validated table or corpus	silent drops, units mixed (kg vs lb), duplicates	row counts per step, validation rules (sheet 02)
Splitting	train / validation / test	same entity or near-duplicate in train and test	group or time split; overlap check by key and by hash
Versioning	immutable snapshot + id	'final_v3_fixed.csv' overwritten in place	content hash recorded with each model (sheet 07)
Monitoring	drift and quality reports	upstream schema change breaks features silently	schema checks and drift metrics on live inputs (mlops folder)

What each stage produces, the typical failure, and the check that catches it

© Data-centric vs model-centric AI

Model-centric	vs	Data-centric
architecture, hyper-parameters, loss	What changes	labels, coverage, cleaning, augmentation ✓
small once a good baseline exists	Typical gain on small data	often larger: label noise is the bottleneck ✓
GPU hours, experiments	Effort shape	people hours, tooling, review
a better model helps one task	Reuse	a better dataset helps every future model ✓
overfitting the benchmark	Risk	fixing labels to match the model's errors

Two ways to improve a system. The term data-centric AI was popularised by Andrew Ng's 2021 campaign and competition, where the model was fixed and only the data could change

Systematic data improvement — Look at errors by slice (class, source, language), fix the slice's labels or collect more of it, retrain, and measure on a frozen test set.

Error analysis — Reading a sample of misclassified examples and tagging the cause (label wrong, input ambiguous, rare case, model weakness) before deciding what to fix.

Slice — A subset of data defined by a condition (night-time images, French tickets, users over 65) whose performance is tracked separately.

✗ Fixing test labels with the model's help, then reporting the new test score

If you relabel only the test examples the model got wrong, the test set now agrees with the model and the score rises without any real improvement. Relabel with blind reviewers, on a random sample or the whole set, and keep the old version to compare.

≡ Splitting without fooling yourself

Conditions	R1	R2	R3	R4
Prediction is about the future	Y	N	N	N
Several rows per entity (patient, user)	--	Y	N	N
Rare classes (under about 5 %)	--	--	Y	N
Split to use				
Temporal split (train past, test later)	X			
Group split by entity id		X		
Stratified random split			X	
Plain random split				X
Notes	also keep entities apart if both apply	GroupKFold, GroupShuffleSplit	StratifiedKFold, stratify=y	fine for i.i.d. rows only

Which split strategy to use. Leakage audits and cross-validation are taught in machine-learning; here only the data side: what goes where

PROBLEM 12,000 chest X-rays from 300 patients (40 images each on average), 18 % show pneumonia. Make a 70 / 15 / 15 split that tests on unseen patients.

- GIVEN**
- Unit of the split: the patient, not the image (images of one patient are near-duplicates)
 - Target shares: train 70 %, validation 15 %, test 15 %
 - Positive rate in the whole set: 18 %

1 Split the patient ids

`300 x 0.70 = 210; 300 x 0.15 = 45; 45`

Why: every image follows its patient

2 Count images per set

`train approx. 210 x 40 = 8,400; val approx. 1,800; test approx. 1,800`

Why: patients have different image counts, so totals vary

3 Check class balance

`e.g. test positive rate 15.9 % vs 18 % overall`

Why: a gap over 2-3 points distorts metrics

4 Rebalance if needed

`StratifiedGroupKFold` or re-draw with a fixed seed

Why: keeps groups whole and classes balanced

5 Record and freeze

`patient ids per set + seed + data version hash`

Why: the test set must be the same for every future model

ANSWER 210 / 45 / 45 patients, about 8,400 / 1,800 / 1,800 images, positive rate within 2 points of 18 % in each set

CHECK ✓ No patient id appears in two sets: `len(set(train_ids) & set(test_ids)) == 0`

A group split by patient, with the class balance checked afterwards

Resized copies of one image, a reposted article, the same customer under two ids: an exact-key split misses them. Hash or embed the content and check train/test overlap before trusting a score (methods in sheets 02 and 05).

Datasheet: chest X-ray pneumonia set v2.1	
Owner: imaging data team	Version: 2.1 (sha256 3f9a...)
Licence: internal, research use	Updated: 2026-09-30
1. Motivation	
<i>Name the task it was built for</i>	
Why was it created, by whom, funded by whom?	
2. Composition *	
<i>Include counts per class and per site</i>	
What is an instance, how many, which labels, missing data, sensitive attributes?	
3. Collection process	
<i>Say who is NOT represented</i>	
How, when, by whom; sampling strategy; consent and ethics review	
4. Preprocessing, cleaning, labelling	
<i>Link the labelling guideline version</i>	
What was removed or transformed; is raw data kept?	
5. Uses *	
<i>Known limitations go here</i>	
Tasks it suits, tasks it must not be used for	
6. Distribution	
<i>State the licence explicitly</i>	
Who can get it, licence, export or access limits	
7. Maintenance	
<i>Give a contact and an update cadence</i>	
Who maintains it, how errors are reported, versioning policy	

Format	Who / when	What it is	Use it for
Datasheet for datasets	Gebru et al., 2018	free-text answers to about 50 questions in 7 sections	internal and published datasets; audits
Data Card	Google, 2022	structured summary cards for stakeholders	product and review documentation
Hugging Face dataset card	Hugging Face	README.md with YAML metadata (licence, languages, tasks)	datasets on the Hub
Croissant	MLCommons, 2024	JSON-LD file describing files, record sets, fields, licence	machine-readable loading and search
Model card	Mitchell et al., 2019	the model-side counterpart (responsible-ai folder)	link the datasheet from it

🔥 Why documentation matters: ImageNet person categories (2019)

Researchers auditing ImageNet's 'person' subtree found offensive and non-visual labels inherited from WordNet categories. In 2019-2020 the maintainers removed many of these categories and later released a version with faces blurred. A datasheet's 'composition' and 'sensitive attributes' questions are designed to surface such issues before release, not years later.

MYTH

- ✗ *Collect everything, decide later*

FACT

- ✔ Collect for a stated task and population

Unfocused collection brings privacy risk (GDPR data minimisation) and still misses the rare cases you need

✖ A random split is always fine

- Random splits leak when rows are grouped or time-ordered

Use group or temporal splits for patients, users, sessions and forecasts.

✖ Once cleaned, the dataset is done

- ✓ Every new batch and every relabel creates a new version

Version, document and re-run quality checks each time.

✓ baseline / mandatory ☐ recommended – verify in your context

- ✓ Write the target population and the task before collecting
- ✓ Keep raw data read-only; transform into new versions
- ✓ Freeze a test set by entity or time and never train on it
- ✓ Store the data version hash with every trained model
- ☐ **MUST** Write a datasheet (or dataset card) before sharing
- ☐ **SHOULD** Track metrics per slice, not only overall
- ☐ **SHOULD** Keep a golden set to audit labellers and drift

Tick a box only when you could explain the point to someone else



Six stages, one loop

collect, label, clean, split, version, monitor



Split by the unit of generalisation

entity or time, then check class balance



Document with a datasheet

seven sections, including who is missing



Data-centric first on small data

fix labels and coverage before tuning



Version everything immutably

a model is traceable only to a frozen dataset

Q1 Name the six lifecycle stages in order.

A collection, labelling, cleaning, splitting, versioning, monitoring (then back to collection)

Q2 What does a data-centric approach keep fixed?

A The model code; only the data changes (labels, coverage, cleaning, augmentation).

Q3 When do you need a group split?

A When several rows belong to one entity (patient, user, device): all its rows go to one set.

Q4 When do you need a temporal split?

A When the model predicts the future: train on earlier periods, test on later ones.

Q5 The seven sections of a datasheet for datasets?

A motivation, composition, collection process, preprocessing/cleaning/labelling, uses, distribution, maintenance

Q6 What is Croissant?

A MLCommons' JSON-LD metadata format (2024) that makes a dataset's files, fields and licence machine-readable.

Q7 Why is relabelling only the test errors wrong?

A It aligns the test set with the model's predictions and inflates the score without real improvement.

Q8 What must be stored with every trained model?

A The exact data version (content hash or snapshot id), the code version and the split definition.

Q9 Give one check for sampling bias.

A Compare the distribution of key attributes (site, age, region, season) in the collected data with the target population.