

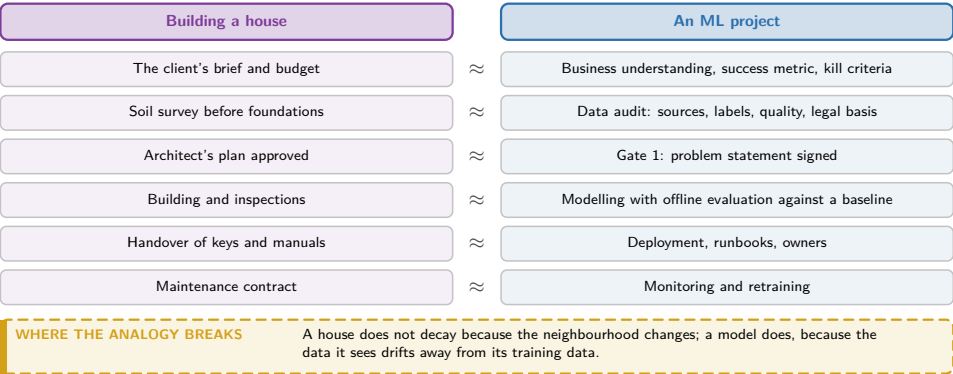


The ML Project Lifecycle

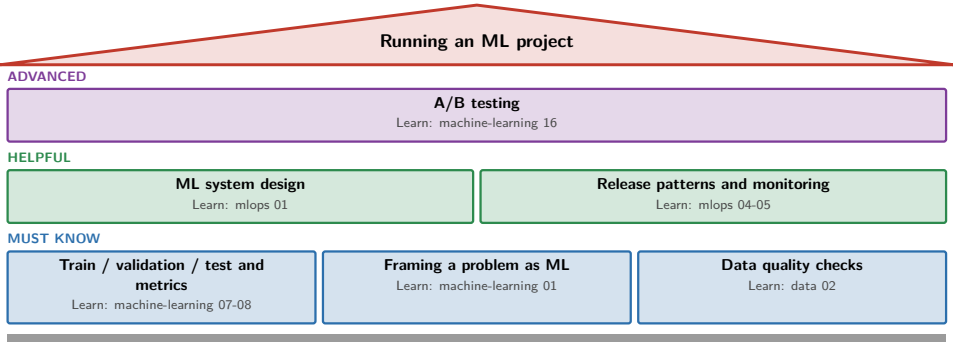
Workflows 01 | From business question to monitoring: CRISP-DM, Microsoft TDSP, CRISP-ML(Q), Google's Rules of ML principles; phases, go/no-go gates, deliverables per phase, roles, and a churn project walked through every gate

#Workflows #CRISP-DM #TDSP #Lifecycle

The job in one picture

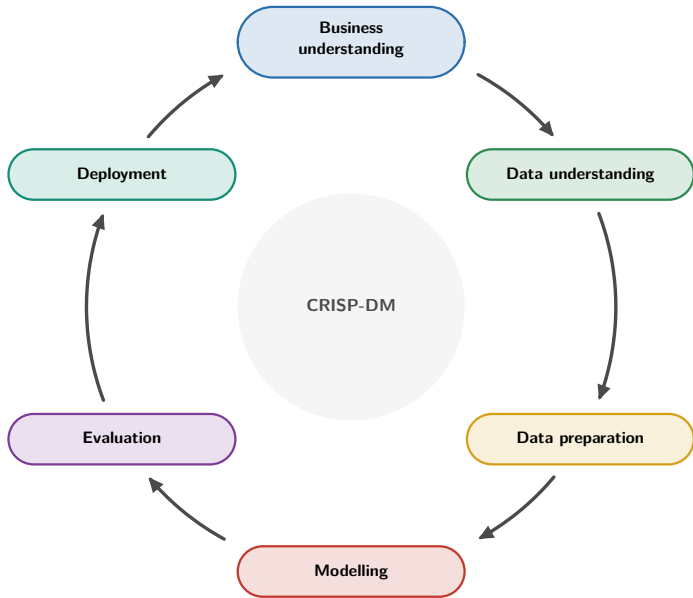


An ML project behaves like building a house, with one twist



What to know before running a project through these gates

Lifecycle models compared



CRISP-DM: six phases on a loop. Evaluation can send you back to business understanding; deployment feeds new questions

Model	Phases	Adds	Use when
CRISP-DM (2000)	business understanding, data understanding, data preparation, modelling, evaluation, deployment	a neutral, tool-free vocabulary; explicit loops	any analytics or ML project
SEMMA (SAS)	sample, explore, modify, model, assess	focus on the modelling core	rarely today; no business or deployment phase
TDSP (Microsoft)	business understanding, data acquisition and understanding, modelling, deployment, customer acceptance	roles, repository template, document templates, acceptance sign-off	team projects with a client
CRISP-ML(Q) (2021)	business and data understanding, data preparation, model engineering, model evaluation, deployment, monitoring and maintenance	a quality-assurance step per phase, risk analysis, monitoring phase	production ML with regulatory or safety stakes
Google MLOps levels 0-2	manual → automated training pipeline → automated CI/CD of pipelines	automation maturity, not phases	planning how much to automate (mlops 03)

The main lifecycle models and what each adds. They share the same spine; pick the vocabulary your organisation already uses

Rule 1 — Don't be afraid to launch a product without machine learning: a heuristic gets you to market and becomes the baseline.

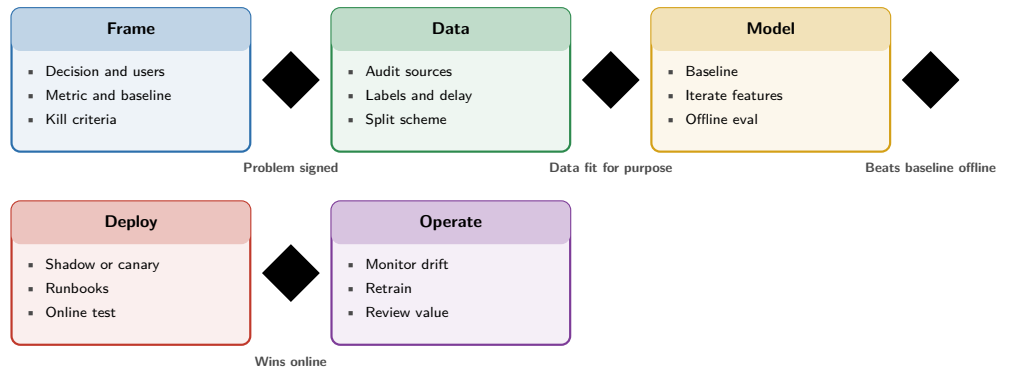
Rule 2 — First, design and implement metrics: log what the system does before you have a model to improve.

Rule 3 — Choose machine learning over a complex heuristic: once rules become unmaintainable, a model is easier to update.

Rule 4 — Keep the first model simple and get the infrastructure right: most early gains come from a correct pipeline and features.

Train like you serve — Log the features exactly as computed at serving time and train on those logs; reuse code between training and serving to avoid skew.

🚩 **Phases, gates and deliverables**



The lifecycle used in every playbook of this folder: five phases, four gates. A failed gate loops back, it does not stop by default

Gate	Pass criteria (examples)	Deliverable	Typical loop-back
G1 Problem signed	decision, users, north-star and proxy metric, baseline, cost of errors, kill criteria written	1-page problem statement	metric cannot be measured → instrument first (Rule 2)
G2 Data fit	labels exist or can be bought; enough rows per class; legal basis and access confirmed; no target leakage found	data audit report, datasheet, split definition	labels too late or biased → change the target or the scope
G3 Offline win	beats the rule baseline on the frozen test set by the agreed margin, per key segment, within latency and cost budget	evaluation report, model card draft	no lift → better features or data, not a bigger model
G4 Online win	A/B or canary shows the north-star metric moves; guardrail metrics stable	launch decision record, runbooks, alerts	offline/online gap → check training-serving skew
Ongoing	drift, performance with delayed labels, cost, incidents reviewed monthly	monitoring dashboard, retraining log	value gone → retire the model

Gate criteria and deliverables. Write the criteria at kick-off, before anyone sees a result

Task	Product owner	Data scientist	ML engineer	Data engineer	Risk / legal
Problem statement and metric	A	R	C		C
Data audit and access		A		R	C
Baseline and modelling	I	R A	C		
Serving and pipelines		C	R A	C	
Online test and launch	A	R	R		C
Monitoring and retraining	I	R	R A		

R Responsible A Accountable C Consulted I Informed

Who does what (RACI: Responsible does, Accountable decides, Consulted, Informed). One A per row

✓ **Phase checklists**

Frame and data (before any training)

- ✓ baseline / mandatory ☐ recommended – verify in your context
- ✓ Write the decision the prediction drives and who acts on it
- ✓ Pick one north-star metric and one or two proxy model metrics for example churn rate; recall at top 10 % of scores
- ✓ Build the non-ML baseline (rule, average, last value) Rule 1: it is also the fallback
- ✓ State latency, cost, privacy and explainability constraints
- ✓ Fix the label definition and its delay (for example churn = no purchase for 90 days)
- ✓ Define the split before exploring: time-based if the future is predicted machine-learning 08
- ☐ **SHOULD** Write kill criteria with a date
- ☐ **COULD** Draft a datasheet for the training data data 01

Model, deploy, operate

- ✓ baseline / mandatory ☐ recommended – verify in your context
- ✓ Track every run: code commit, data version, config, metrics mlops 02
- ✓ Evaluate per segment, not only overall
- ✓ Freeze the test set; touch it only at the gate
- ✓ Release in shadow or canary with a rollback mlops 04
- ✓ Alert on input drift and on the proxy metric with delayed labels mlops 05
- ☐ **MUST** Write runbooks: who is paged, how to roll back, how to retrain
- ☐ **SHOULD** Review business value every quarter; retire models that no longer pay

★ Worked example: a churn project through the gates

PROBLEM

Should we deploy a churn model that picks which customers get a retention call each week?

1. G1 Frame

⇒ decision = top 5,000 to call; north-star = churners saved;
proxy = precision at 5,000
why: capacity fixes k; precision at k is the metric that matches the decision

2. G2 Data

⇒ label = churn within 60 days of the score date; time split by month
why: a random split would leak future behaviour

3. Baseline score

⇒ rule precision at 5,000 = 0.08 → 400 churners reached
why: measured on the last 3 months of history

4. Model

⇒ GBM precision at 5,000 = 0.20 → 1,000 churners reached
why: usage drop and complaints features carry the signal

5. G3 Offline

⇒ saved per week: rule $0.2 \times 400 = 80$; model $0.2 \times 1,000 = 200$
why: gate criterion was 'at least 1.5x the rule'

6. G4 Online

⇒ 4-week A/B: half the call list from the rule, half from the model
why: offline precision can be inflated by skew or label delay

7. Operate

⇒ monthly drift report; retrain quarterly; compare saved churners to the rule arm kept at 10 %
why: a small holdout keeps measuring value

GIVEN

- 1,000,000 customers; monthly churn 2 %
- Call-centre capacity: 5,000 calls per week
- Baseline rule: call customers whose contract ends within 30 days
- A call saves 20 % of the churners it reaches

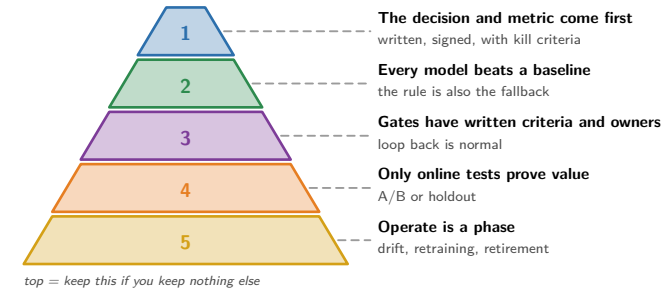
ANSWER

Deploy if the A/B confirms about 2.5x more saved churners than the rule (200 vs 80 per week offline)

Check: Ratio $1,000 / 400 = 2.5 \geq$ the 1.5x gate threshold; the 20 % save rate cancels out of the ratio

⚠ Pitfalls by phase

- Frame: optimising AUC when the decision uses only the top k scores; measure at k.
- Data: label leakage through features computed after the prediction date.
- Model: tuning on the test set by looking at it at every iteration.
- Deploy: features computed differently online and offline (training-serving skew).
- Operate: no holdout group, so nobody can show the model still adds value.



What to remember, most important on top

✓ Self-test

Q1 Name the six CRISP-DM phases in order.

- A Business understanding, data understanding, data preparation, modelling, evaluation, deployment.

Q2 What does TDSP add that CRISP-DM lacks?

- A Defined roles, repository and document templates, and a customer-acceptance phase.

Q3 What does CRISP-ML(Q) add?

- A A quality-assurance step with risk analysis in every phase and an explicit monitoring and maintenance phase.

Q4 Rule 1 of Zinkevich's Rules of ML?

- A Don't be afraid to launch a product without machine learning.

Q5 What must the problem statement contain?

- A Decision and users, north-star and proxy metrics, baseline, constraints, cost of errors, kill criteria.

Q6 Why write gate criteria at kick-off?

- A So results are judged against targets fixed before anyone sees them, not moved afterwards.

Q7 In the churn example, why precision at 5,000?

- A Call capacity fixes the list size; only the top 5,000 scores drive the decision.

A telecom wants to cut churn with retention offers. Figures are illustrative assumptions stated at kick-off

YOU MAY HEAR

"CRISP-DM is a waterfall: finish each phase, then move on"

BUSTED

✓ ACTUALLY

Its arrows go back

Evaluation regularly sends teams back to business understanding; plan two or three loops.

YOU MAY HEAR

"A better offline metric guarantees business impact"

BUSTED

✓ ACTUALLY

Only the online test proves value

Skew, delayed labels and user reactions break the link between proxy and north-star metrics.

YOU MAY HEAR

"Deployment ends the data scientist's job"

BUSTED

✓ ACTUALLY

Operate is a phase with owners

CRISP-ML(Q) and TDSP both add monitoring or acceptance; unowned models decay silently.

YOU MAY HEAR

"ML is always better than rules"

BUSTED

✓ ACTUALLY

Rules first, ML when rules get complex

Zinkevich's Rule 1 and Rule 3: launch with heuristics, switch when they become unmaintainable.

Lifecycle beliefs that cost projects months

Q8 Offline metric up, online metric flat: first suspect?

A Training-serving skew or label leakage; then a weak link between proxy and north-star metric.

Q9 Why keep a small holdout after launch?

A To keep measuring the model's value against the baseline as data drifts.