WHAT COUNTS AS A SUCCESSFUL RESULT?

The evaluation contract

A convincing demonstration is a starting point.
An evaluation contract defines the evidence needed to accept a design.

Today: turn requirements into cases, scores, and release decisions.

A GOOD-LOOKING ANSWER IS INCOMPLETE EVIDENCE

An answer can sound clear while using the wrong policy, exposing another customer's data, or arriving too late.

Evaluation compares behavior with the product's requirements.

Software checks still matter. AI quality checks add evidence about predictions, retrieved knowledge, generated content, and model-directed actions.

One fluent response faces four independent checks: policy, permission, timing, and usefulness.

POLICY

Does it follow the right policy?

PERMISSION

Does it expose another customer's data?

TIMING

Is it on time for the task?

USEFULNESS

Does it actually help the user?

GIVE EACH CASE A COMPLETE CONTRACT

A useful case records:

Input and caller context	
Available data and tool results	
Expected behavior	
Scoring rule	
System and dataset versions	

Without context, “Was this answer correct?” can be ambiguous. The same question can need different answers under different policies or permissions.

DEFINE SUCCESS FOR THE USER'S JOB

POLICY QUESTION

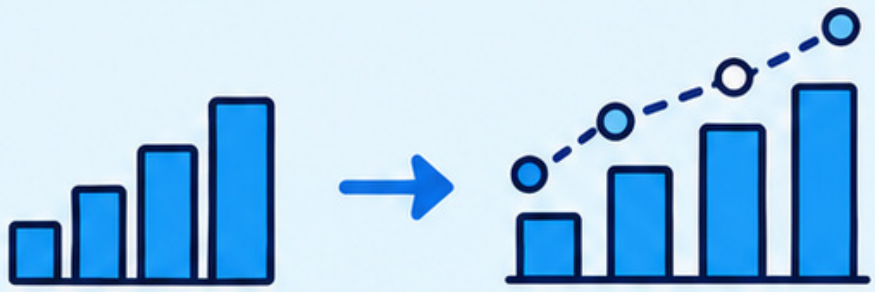
For a policy question, success can require an answer that addresses the request, matches the applicable document, and cites its source.

User asks a policy question

Expected outcome: helpful answer that matches the document and cites its source.

DEMAND FORECAST

For a demand forecast, success needs a comparison with later observed demand.



Model produces a forecast

Expected outcome: a comparison with later observed demand.

The evaluation contract describes the result the user needs, then chooses measurements that represent it.

SOMETIMES SUCCESS MEANS STOPPING

Case 1: Missing Evidence

Suppose the supplied documents do not cover an opened-item exception.

What is the policy for an opened-item exception?

I don’t have enough information in the provided documents. Let me hand this off to a specialist.

This handoff response passes.

The expected response can be clarification or handoff. A confident invented rule fails this case.

Case 2: Complete Evidence

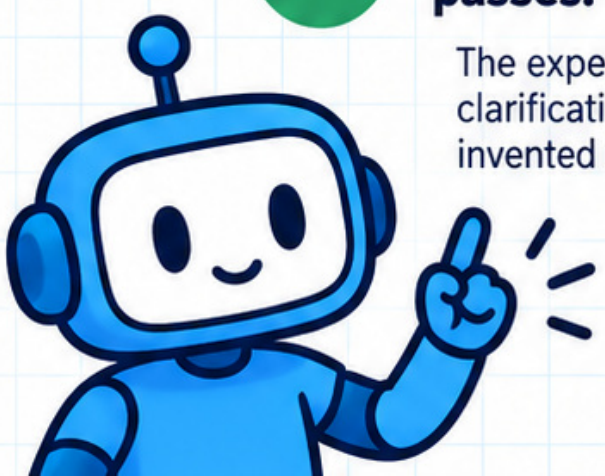
The supplied documents cover the question and provide the needed information.

What is the policy for an opened-item exception?

I don’t have enough information in the provided documents. Let me hand this off to a specialist.

This handoff response fails.

Stopping is successful only when the case expects it. An assistant that hands off every answerable question does not meet the product's purpose.



CHOOSE CASES THAT EXPOSE DIFFERENT BEHAVIOR

Ordinary requests

"Summarize this article."

Difficult requests

"Solve this complex problem."

Missing evidence

"Answer with limited information."

Ambiguous wording

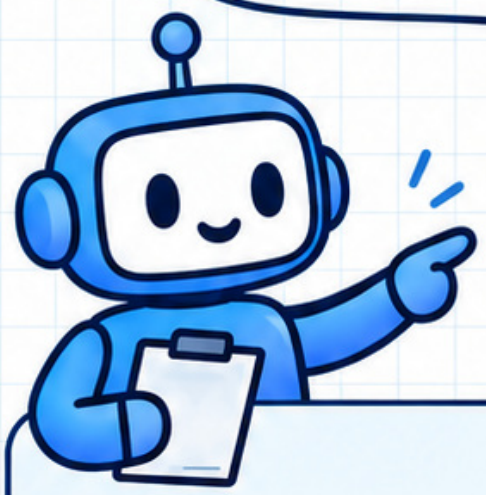
"What does 'it' refer to here?"

Denied access

"I don't have permission to access this."

Dependency failures

"A required tool is unavailable."



EVALUATION SET

RESULTS (GROUPED BY CATEGORY)

Ordinary requests

Difficult requests

Missing evidence

Ambiguous wording

Denied access

Dependency failures

Include ordinary requests, difficult requests, missing evidence, ambiguous wording, denied access, and dependency failures.

Use cases that resemble the intended users and data.

Keep the categories visible in the results. A strong overall score can hide weak performance on the requests that matter most.

USE DIRECT CHECKS FOR CLEAR RULES

Some expectations have a direct pass or fail answer:

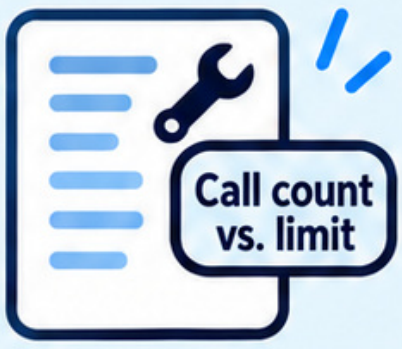
Does the response match its schema?

Reads the response output (e.g., JSON).

Did an unauthorized record enter context?

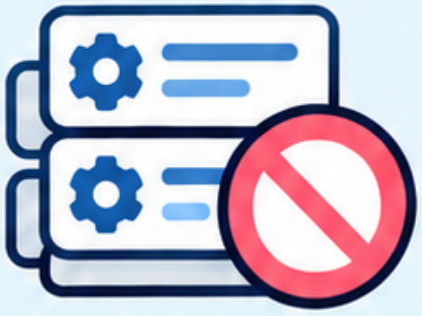
Reads the retrieved context (e.g., documents).

Did the run exceed its tool-call limit?

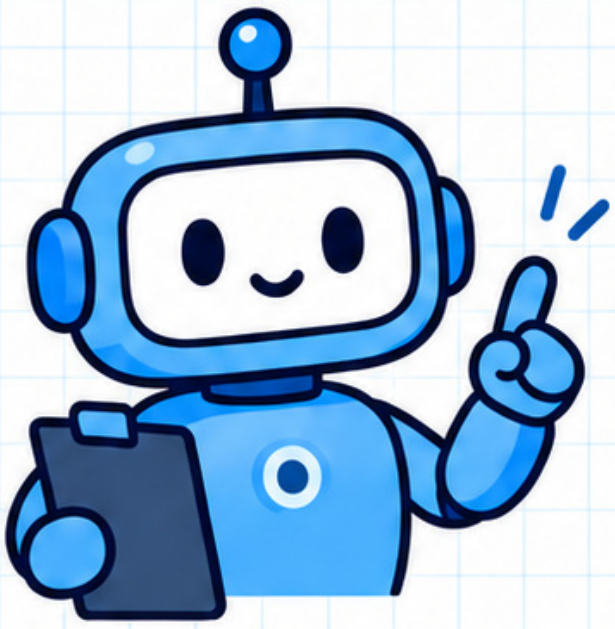


Reads the tool-call log (e.g., calls made).

Did one request create duplicate actions?



Reads the action records (e.g., events).

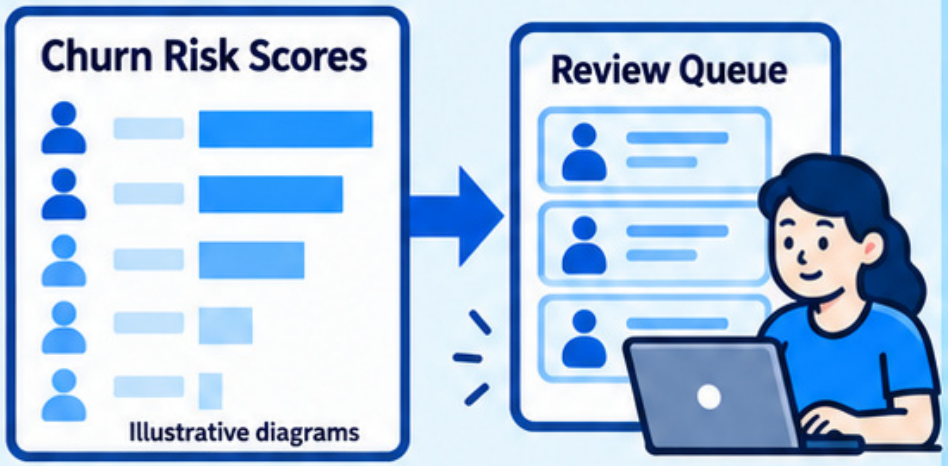


Use application assertions for these rules instead of asking a language model to judge everything.

@learn.machinelearning

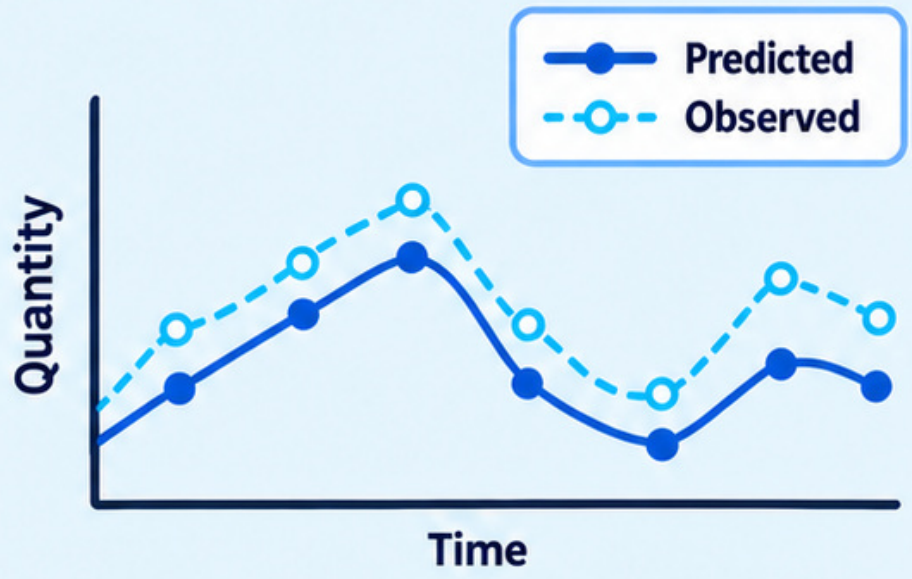
MATCH ML METRICS TO THE DECISION

A churn model can rank customers for a limited review team. Precision and recall at that review capacity are useful measurements.

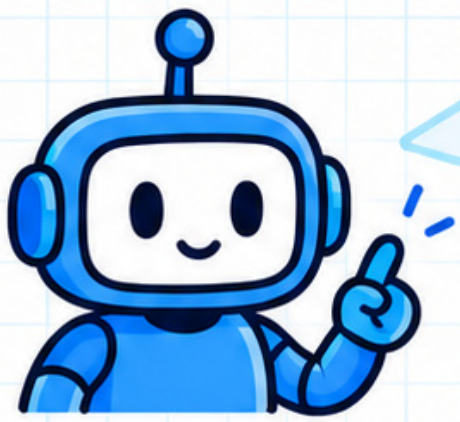


Precision and recall

A demand forecast predicts quantities. Error magnitude matters.



Forecast error



Choose the metric from the decision and its consequences. A single accuracy percentage does not describe every ML task.

GIVE GENERATED ANSWERS A SPECIFIC RUBRIC

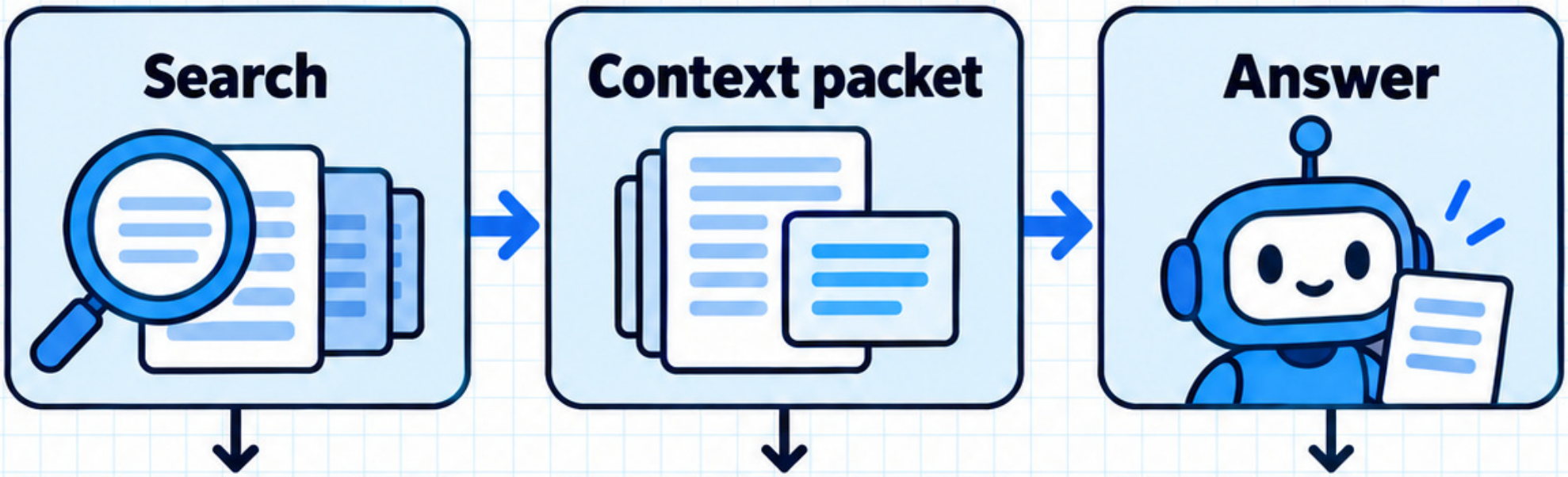
Based on this document

For each answer, assess:

- Does it address the question?
- Are its claims supported by the supplied evidence?
- Does it use the applicable policy version?
- Do the citations point to the supporting source?

Separate these criteria.
Evidence support alone does not prove that the source is current or correct.

EVALUATE COMPONENTS TO LOCATE FAILURES



Did search return the required document?

Did context assembly keep the relevant exception?

Did generation follow that evidence?

If the answer uses the wrong policy, inspect retrieval before changing the prompt.

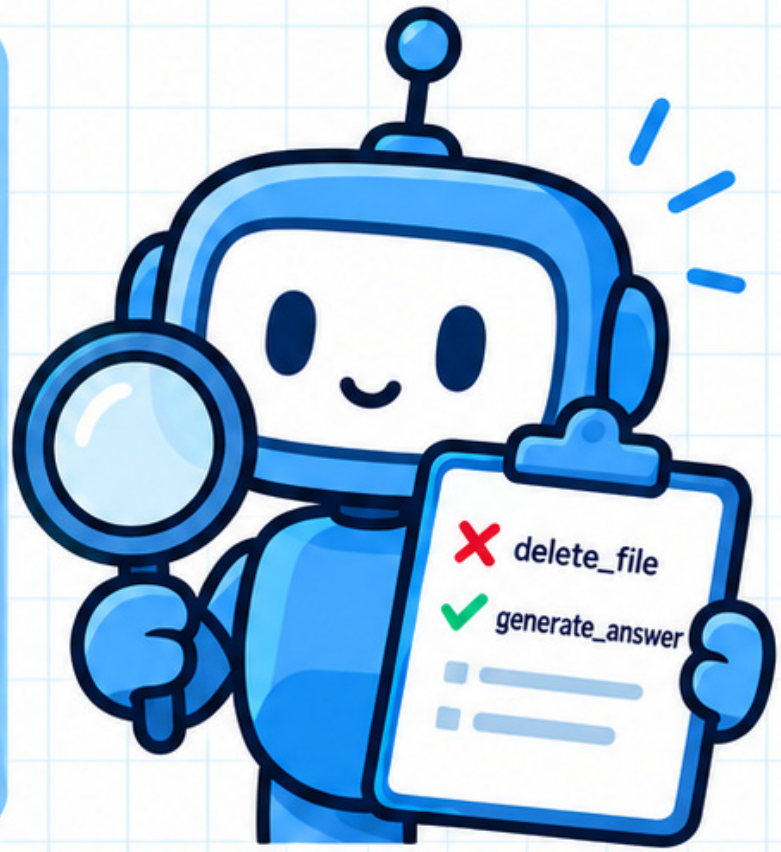
Component checks explain where to investigate. The final task evaluation still determines whether the whole result is useful.

ACTIONS NEED A TRAJECTORY CHECK

Fictional agent trace

1.	**tool_call** search_docs	`arguments: { … }`
2.	**tool_call** delete_file	Permission: not granted Executed anyway **State: file deleted** ✗
3.	**tool_call** generate_answer	`arguments: { … }`
4.	**final_message**	"Here is the answer: …"

Overall case: FAIL



An agent can produce a correct final message after taking an unnecessary or disallowed action.

Inspect the tool names, arguments, permission decisions, and resulting state.

The final text is only part of the evidence. A successful action task must also respect the allowed path and its execution limits.

READ THE SCORE WITH ITS DENOMINATOR

FICTIONAL TEST RUN

Case 1

PASS

fictional test run

Case 2

PASS

fictional test run

Case 3

PASS

fictional test run

Case 4

PASS

fictional test run

Case 5

FAIL

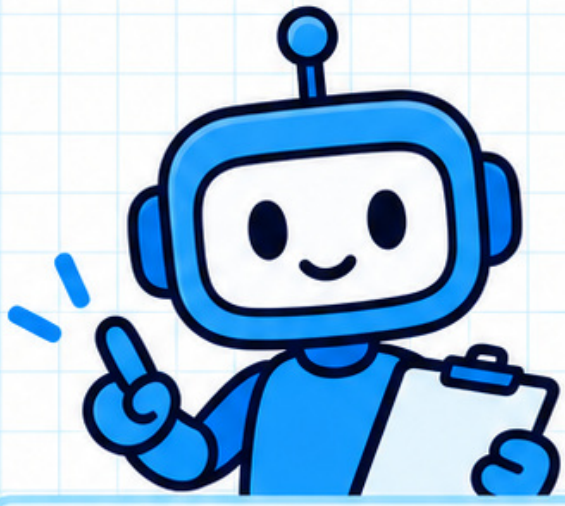
fictional test run

Case 6

FAIL

Protected data

fictional test run



Teaching example: six cases run, and four meet their expected behavior.

4 ÷ 6 = 66.7%

case pass rate.

Now inspect the two failures. If one exposes protected data, the overall average cannot cancel that failure. Keep required safety checks visible beside the aggregate result.

MEASURE QUALITY, TIME, AND COST TOGETHER

Record request duration and spend for the same evaluated workload.

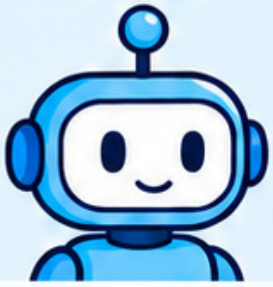
Report quality beside latency and cost per successful task. State which expenses and failed attempts the cost calculation includes.

A comparison is incomplete when a faster design silently changes the task set, response quality, or amount of work.

EVALUATED WORKLOAD (SAME FOR BOTH)

Task description:

CANDIDATE A



Configuration notes:

COMPARISON SCORECARD

Metric	Candidate A	Candidate B
Quality (whole workload)		
Latency (request duration)		
Cost (per successful task)		
Declared scope (expenses and failed attempts included)		

CANDIDATE B

Configuration notes:

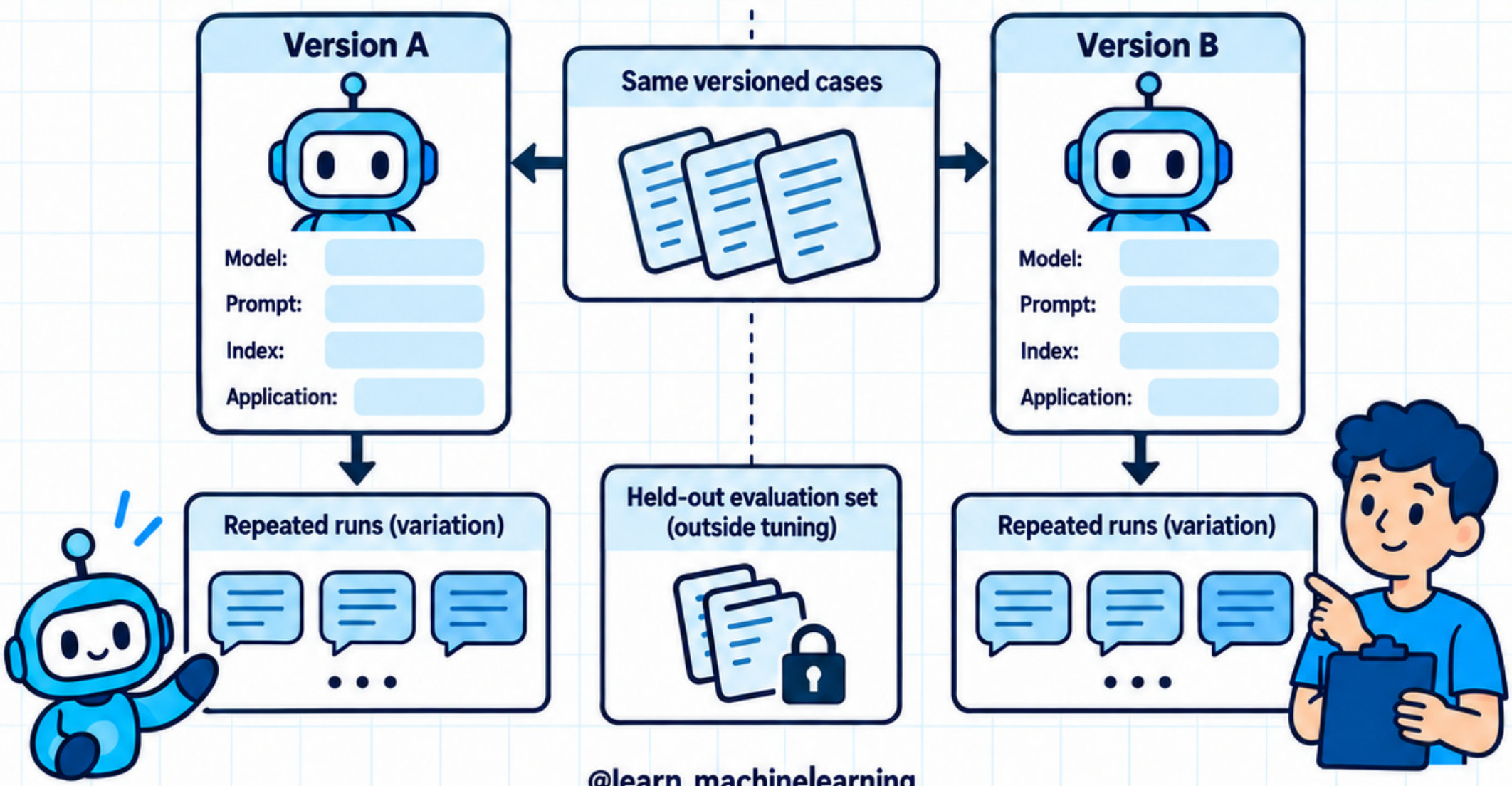
A comparison is incomplete when a faster design silently changes the task set, response quality, or amount of work.

COMPARE VERSIONS UNDER CONTROLLED CONDITIONS

Keep cases, source snapshots, and scoring rules fixed for a comparison. Record the model, prompt, index, and application versions.

Model outputs can vary across repeated runs. Record that variation rather than treating one small difference as proof.

Keep a separate evaluation set out of tuning for decisions that need independent evidence.



AI SYSTEM DESIGN • DAY 6

15/18

CALIBRATE AUTOMATED JUDGMENT WITH PEOPLE

A model judge can help review many answers, but it can also prefer longer wording or miss subtle errors.

Use a clear rubric and compare its decisions with human-reviewed examples.

Disagreements are useful evidence. Inspect them before treating the judge's score as a reliable measurement.

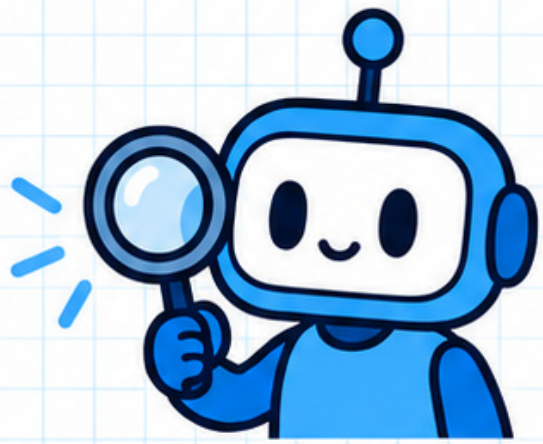
Human Judgment

Illustrative judgment comparison

Answer	Human	Model	Agreement?
	✓	✓	Yes
	✓	✗	No
	✗	✗	Yes
	✓	✗	No
...	...	...	...

Disagreements

Review disagreements



@learn.machinelearning

ASSIGN TOOLS TO EVIDENCE COLLECTION

Python or pytest:
check defined behavior.

→ **Assertions**

scikit-learn:
calculate prediction metrics.

→ **Metrics**

PostgreSQL:
store versioned cases and outcomes.

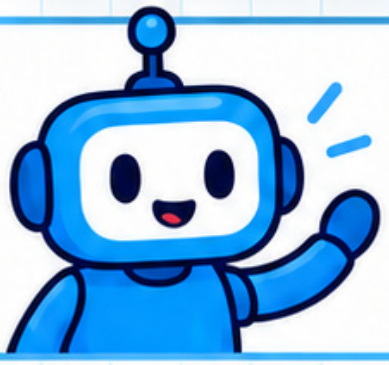
→ **Case records**

OpenTelemetry:
capture request timing and steps.

→ **Timings**

Phoenix:
inspect AI traces and compare evaluations.

→ **Trace comparisons**



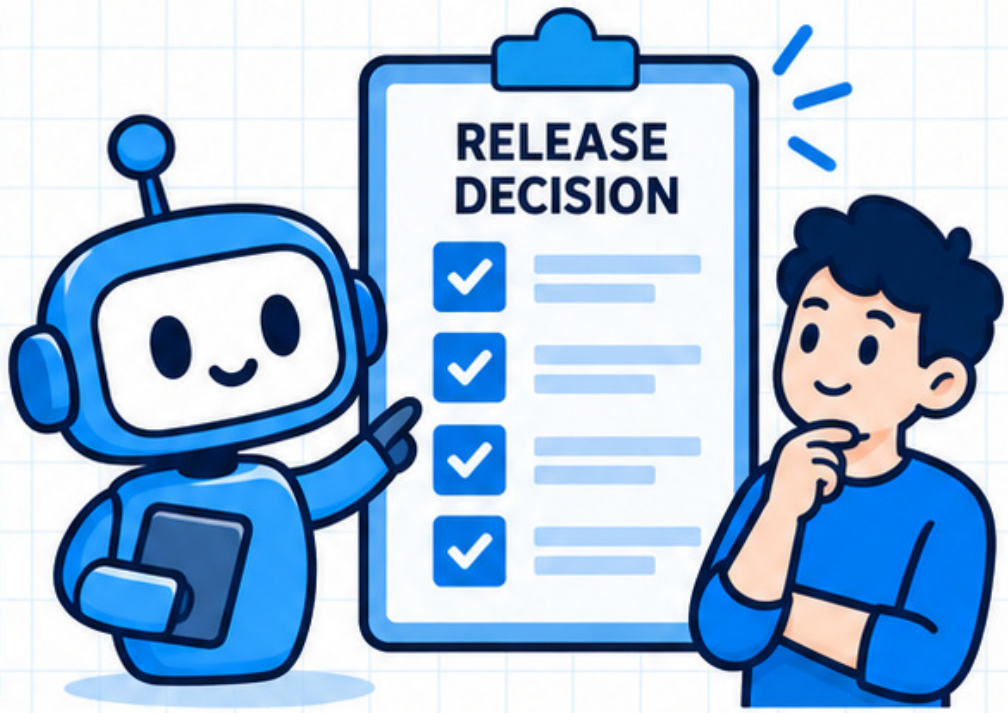
The tools collect and organize evidence.

The evaluation contract defines what that evidence must demonstrate.

MAKE ACCEPTANCE RULES EXPLICIT

A release decision can require:

1. All access-boundary tests pass.
2. Task quality meets the declared target on the named dataset.
3. Latency stays within the stated workload budget.
4. Cost stays within its declared scope.



Rule	Target	Observed Evidence	Decision
All access-boundary tests pass.	—	—	—
Task quality meets the declared target on the named dataset.	—	—	—
Latency stays within the stated workload budget.	—	—	—
Cost stays within its declared scope.	—	—	—

Each rule needs a target, measurement method, and observed result.
Passing a small test suite is not a guarantee about every future request.

YOUR TURN

Create six evaluation cases for a policy assistant or a demand forecast.
For each case, write the input, context, expected behavior, and scoring rule.
Include an ordinary case, a difficult case, and a failure case.

Case 1: Ordinary Case

Input:
Context:
Expected behavior:
Scoring rule:

Case 2

Input:
Context:
Expected behavior:
Scoring rule:

Case 3

Input:
Context:
Expected behavior:
Scoring rule:

Evaluation Case Template

EVALUATION CASE TEMPLATE

Input:
Context:
Expected behavior:
Scoring rule:

Case 4

Input:
Context:
Expected behavior:
Scoring rule:

Case 5

Input:
Context:
Expected behavior:
Scoring rule:

Case 6: Failure Case

Input:
Context:
Expected behavior:
Scoring rule:

EVALUATION RESULT

ARCHITECTURE DECISION

DAY 7: Labels and observation windows

Design the evidence before optimizing the system.