

FREE SAMPLE: LESSON 1 OF 6

A COURSE IN SIX LESSONS

Test it before you trust it

Evaluating an AI assistant in six lessons

Seven Sky Consulting

Free sample of the course book. Edition of 7 October 2026.

This sample holds lesson 1 in full. Its notebook and the data it reads come with it.



Lesson 1, in full

This is the first of six lessons from "Test it before you trust it", a course on how to test an AI assistant that answers questions from documents, before you decide to launch it.

The course is for an analyst or a team lead who can read Python. Each lesson is a chapter and one Jupyter notebook. This sample has both for lesson 1: the chapter that follows, and the notebook

`lesson_1_build_the_test_set.ipynb` with the three data files it reads. The notebook runs with no network and no API key.

Everything in the case is made up for teaching. The Quillhaven Lending Library does not exist. Its documents and all 150 test questions were written by the course author. The numbers say nothing about any real product.

The case

A library has put an AI assistant on eight of its documents: the membership guide, the borrowing rules, the fee schedule and so on, 40 short passages in all. Before anyone looks at a single answer, the team has to decide what it will ask the assistant, and which of those questions it will keep unseen until the day of the test. That is the whole of lesson 1.

To run the notebook

1. Unzip the sample. Keep the `data` folder next to the `notebooks` folder.
2. You need Python 3.10 or later with `numpy`, `pandas`, `matplotlib` and Jupyter.
3. Open `notebooks/lesson_1_build_the_test_set.ipynb` and choose "Restart and run all".

The notebook is saved with its outputs, so you can also read it without running it.

LESSON 1

Write the questions before you look at the answers

A test is only as good as its questions, and questions are only fair if they were written before anyone saw how the assistant answers them. This lesson builds the test set and puts two thirds of it out of reach.

Notebook: `lesson_1_build_the_test_set.ipynb`. It reads `documents.csv`, `passages.csv` and `questions.csv`.

Why the order matters

Suppose you try the assistant first. You ask it a few things, it does well on some and badly on others, and then you sit down to write "the test". You will write questions like the ones you have just seen it handle, worded the way it seemed to like. You will not mean to. The result is a test that the assistant passes and that tells you little about next month's real questions.

So the questions come first, with their reference answers, and the date they were written is recorded. In the case data, the last question was written on 20 February 2026 and the first answer was recorded on 9 March 2026. The notebook checks this with one line. It is worth having that line in your own project, because the order is the first thing a careful reader of your results will ask about.

What one test question needs

A question on its own cannot be graded. Each row of `questions.csv` carries what a grader, a script and a later reader need.

Field	Example	Why it is there
Question	how long do i get a dvd for	Asked the way a member asked it
Reference answer	Films are lent for 7 days.	What graders mark against
Source passage	B0R-1	Where the answer is, so a citation can be checked
Type	number	So results can be split by kind of question
Origin	desk_log	Who wrote it: the team, or a real user
Fact id	film_loan	Marks questions that ask about the same fact
Written on	11 February 2026	Proof that it came before the answers

Two origins

The case has 84 questions written by the project team and 66 taken from a log of what members asked at the front desk, spelling and all. You need both. The team knows the documents and will cover them evenly. The team also writes in the documents' own words, which makes its questions easier than real ones. Lesson 5 measures how much easier.

Four types

Type	Questions	What it is
number	60	The answer is a figure: an amount, a count, a time, a date or a length of time
lookup	42	A fact in words
condition	21	A rule with an exception or a second part. "The first replacement card is free. Each later one costs \$4."
unanswerable	27	The documents do not say

The last type is the one most often left out, and it matters most. Members will ask about parking, and the documents say nothing about parking. An assistant that answers "the documents do not cover that" and an assistant that invents a parking lot are very different products. Answerable questions cannot tell them apart. In this case 27 of the 150 questions have no answer in the documents, and for those the correct behavior is to say so.

Coverage

Count the answerable questions per passage. A passage with none is a part of the documents you will know nothing about. Here every one of the 40 passages has at least 2.

Hold part of it out

You will change the assistant many times before launch. Each time you will look at what it got wrong and fix that. After a few rounds it is good at *these* questions, and you no longer know whether it is good at questions in general. The scikit-learn user guide calls testing on the data you learned from "a methodological mistake", and warns that knowledge of the test set can leak into a system through repeated tuning. The same holds when the thing being tuned is a prompt or a search step.

The defense is to split the questions once, at the start.

- The **development set** is yours to read, tune against and argue about.
- The **held-out set** is used only to report a result. Nobody tunes against it.

In the case, 50 questions are for development and 100 are held out. Every result in lessons 2 to 6 that is reported as a finding is a held-out result. Where a lesson needs to choose something, such as a threshold, it chooses on the development set.

The trap in a simple split

Some questions ask about the same fact in different words. "How long can I keep a book?" and "book loans are how many weeks" are one fact. The case has 7 facts with two questions each. If one twin lands in development and the other in the held-out set, you will tune on a question whose answer is also being held out. The held-out score is then partly a score on things you have seen.

A plain random split of rows does this readily. With the course's seed it separates 3 of the 7 twinned facts. Exercise 3 asks you to measure how often it happens over a thousand seeds. Real question sets have many more near-duplicates than this one, because people word the same worry in many ways.

Split by fact, and keep the mix

The function below moves whole facts, so twins stay together. It also splits each question type on its own, so the held-out set has the same mix of types as the whole set. Both properties come from how the function is built. They do not depend on the seed.

```
def split_by_group(qtype, fact_id, heldout_share=2 / 3, seed=SEED):
    """Return 'heldout' or 'dev' for each question.

    Questions with the same fact_id always get the same side.
    Each question type is split on its own.
    """
    qtype = np.asarray(qtype)
    fact_id = np.asarray(fact_id)
    rng = np.random.default_rng(seed)
    side = {}
    for t in sorted(set(qtype)):
        facts = fact_id[qtype == t]
        names = sorted(set(facts))
        sizes = {name: int((facts == name).sum()) for name in names}
        quota = round(len(facts) * heldout_share)  # how many questions of this type to hold out
        taken = 0
        for i in rng.permutation(len(names)):  # facts in random order
            name = names[i]
            if taken + sizes[name] <= quota:
                side[(t, name)] = "heldout"
                taken += sizes[name]
            else:
                side[(t, name)] = "dev"
    return [side[(t, f)] for t, f in zip(qtype, fact_id)]
```

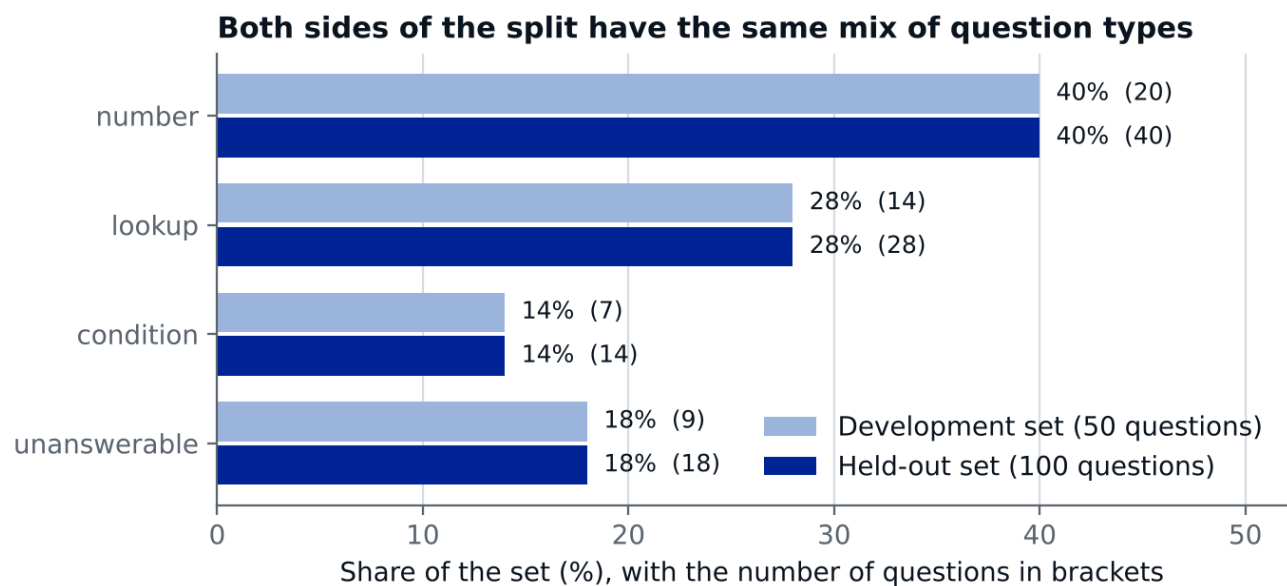


Figure 1.1. The split holds out 100 of 150 questions and keeps the four types in the same proportions on both sides. Made by the lesson 1 notebook.

Lock the held-out set

A held-out set is worth something only while it stays the same and stays unseen. Three habits protect it.

1. **Give it a fingerprint.** The notebook works out a short code from the held-out questions and their reference answers. For the case it is `5d39ee13967b5721`. If anyone adds, drops or rewords a held-out question, the code changes. Lesson 6 prints the code in the launch memo.
2. **Keep it away from the people tuning.** A separate file, a separate folder, and a rule that results on it are produced by a script, not by reading.
3. **Log each use.** Every time the held-out set informs a decision it becomes a little less unseen. Lesson 6 returns to this.

The NIST AI Risk Management Framework asks for the same discipline in one line of its MEASURE function: test sets, metrics and the tools used in testing are to be documented (MEASURE 2.1). A fingerprint and a dated question file are a cheap way to start.

How many questions is enough?

Lesson 3 answers this with care. For planning, this is roughly how far the true figure can sit from a measured 80%, 19 times in 20.

Held-out questions	25	50	100	200	400
Give or take, in points	16	11	8	6	4

With 100 held-out questions you can tell 80% from 65%. You cannot tell 80% from 76%. That is why every score in this course is written with its count beside it.

COMMON MISTAKES

- Writing the questions after trying the assistant, or rewording them when it stumbles.
- Leaving out questions the documents cannot answer.
- Using only the team's own questions.
- Splitting rows at random when several questions share a fact.
- Reading held-out failures, fixing them, and then reporting the held-out score as if nothing had happened.

EXERCISES

1. **Another seed.** Run `split_by_group` with `seed=1`. Is any fact split across both sides? Is the mix of question types still the same on both sides? Why does the seed not matter for those two properties?
2. **Thin spots.** List the passages with the fewest answerable questions. Pick one, read it, and write two new questions for it on paper: one the passage answers, and one that sounds as if it should be answered there but is not.
3. **How often does a plain split leak?** Repeat the plain random split of rows 1,000 times with different seeds. In what share of the splits is at least one fact split across both sides? What is the average number of split facts?

The other five lessons

Lesson 1 ends with a locked set of 100 held-out questions. The rest of the course uses them.

Lesson	The question it answers
2. Grading answers, and why two graders must agree	Who decides whether an answer is right, and do they agree?
3. Is the difference real?	Is one version really better, or was it luck?
4. Grounded or not	Does each answer stay inside its source?
5. Failure hunting	Where are the failures hiding, and can our checks fail at all?
6. The launch memo	What do we tell the person who decides?

The full course has the course book, six notebooks, the complete teaching dataset (recorded answers from two made-up assistant versions, two graders' marks, claim labels and planted bad answers), and a notebook of worked answers to all eighteen exercises.

Seven Sky Consulting. sevenskyconsulting.com