

BEYOND DEPLOYMENT

The Model Nobody Owned

Why putting machine learning into production turned out to be a problem of accountability, not engineering

Giovanni Leonardi

August 2019



The Model Nobody Owned

Transformation · No. 283

Originally written August 2019 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

"The right question was never whether the model was in production, but who could stop it, on what evidence, and by when."

1. The question that emptied the room

A model had been making credit decisions for eleven months when someone from the second line finally asked the question that mattered. Not how accurate is it — that number was on the wall, and it was reassuring. The question was quieter and harder: show me the model that declined this particular applicant back in March, and prove to me it would make the same decision today. The room went quiet, because the honest answer was that no one could. The model had been retrained four times since March. The training data had been refreshed twice. The version that made the March decision no longer existed anywhere anyone could point to, and the version now running had never been asked to reconstruct it.

Everyone in that room had prepared for the wrong hard part. We had spent two years learning how to ship models — how to get them out of a notebook and into production reliably, repeatably, at scale. The literature, the conference talks, and the tooling that had begun to gather under the banner people were starting to call MLOps all addressed the same problem, and addressed it well. But that is an engineering problem, and the thing that broke in that meeting was not an engineering problem. We had placed a new kind of decision-maker inside an organisation built to hold people accountable, and nobody had noticed the category error until the auditor did.

That is the argument of this piece, stated plainly: the hard part of putting machine learning into production was never the deployment. It was governance — reconciling a probabilistic, drifting, opaque artefact with an organisation's existing machinery for accountability, sign-off, and control. The textbooks left that out, and they left it out for an honest reason. It is not a technical chapter.

2. What the textbooks were answering

The engineering story had matured impressively, and quickly. We had reproducible pipelines, models packaged into containers, feature stores to keep training and serving inputs consistent, registries to record versions, monitoring dashboards, and the discipline of running a new model quietly in shadow before letting it decide anything. Each of these was a real advance, and I would not give any of them back.

But look at the assumption underneath them all. The engineering discipline treated the model as an asset — a thing you build, deploy, version, and maintain — and assumed that once you could deploy reliably and monitor continuously, governance was a downstream formality you would bolt on later. The celebrated moment was the launch: the model goes live, the dashboard turns green, the uplift is demonstrated in a percentage point of accuracy, and the team moves on to the next one.

The trouble is that an asset sits there. A model decides. Treating the second as though it were the first is the original sin from which most of the pain that followed descended.

3. The decision no one owned

Governance functions — risk, audit, compliance, the whole second line — were built on two assumptions that machine learning quietly violates.

- Systems are deterministic. You can test them exhaustively, and tomorrow they will do what they did today. A model is statistical: it behaves differently on data it has not seen, and the world it was trained on keeps moving underneath it. You cannot sign a model off the way you sign off a rules engine, because what you are approving is not a fixed behaviour but a distribution of future behaviours, most of which you have not observed.
- Decisions are made by accountable humans. An underwriter, a claims handler, a credit officer — a named person who could be asked to justify a call and, if necessary, be held to it. Automate that decision and the accountability structure still points at a person, but that person no longer decides. The model decides; the human is left carrying responsibility for a choice they did not make and frequently cannot explain.

So the model falls into a gap. It is too technical for the risk committee, who cannot interrogate it. It is too consequential to be left with the data scientists, who never signed up to be accountable for lending policy or claims strategy. In the end it is owned by no one who combines the authority to stop it with the understanding to know when it should be stopped. This is not a failure of any individual. It is a structural vacancy, and the model sits in it.

A model in production is not a deployed asset. It is an automated decision — and every decision an organisation makes still needs a single person who can be held to it.

4. When the model met the regulator

If the accountability gap was the disease, the regulator was the symptom that finally made it visible. GDPR had been in force for a little over a year, and its provisions on automated decision-making — the expectation that a data subject can be given meaningful information about the logic involved — had moved from the legal team's watch-list into live operational demands. The collision was almost comic in its predictability. Compliance asks for an explanation an ordinary person could understand.

The model is a gradient-boosted ensemble of several hundred trees. The local-explanation techniques the team reaches for produce a plausible story for any single decision, but it is a story even the modellers would not fully stand behind, and everyone in the room knows it.

Reproducibility went through the same transformation. Almost overnight it stopped being a scientific nicety and became a legal and audit requirement. In regulated industries the instinct was not new — model risk management had been a serious discipline since the years after the financial crisis, when institutions learned expensively what an unexamined model could cost. What was new was applying that discipline to models that retrained themselves weekly on data nobody had frozen.

The mechanism of failure was mundane, which is why it was so common. A model validated at an AUC of 0.82 in development; six months later, nobody has re-measured it. The first sign of decay is not a monitoring alert but a business one: approval rates drift upward, then a cluster of complaints arrives, then a manual review discovers the model has been quietly generous to a segment of applicants whose behaviour had shifted. The monitoring dashboard existed the whole time. It simply belonged to no one whose job it was to read it, and act.

5. "The tooling will close the gap"

The strongest objection to everything I have said is that I am describing a transitional problem and mistaking it for a permanent one. The case runs like this. Registries now record every version. Feature stores make inputs reproducible. Monitoring catches drift earlier every quarter. Continuous delivery for models is arriving. Give it eighteen months, the optimist says, and the reproduce-and-monitor problem is solved; governance is then just a matter of wiring a checklist to the pipeline. My complaint is a snapshot of an immature field, not a condition of the thing itself.

I have real sympathy for this view, and I want all of the tooling it promises. But it answers the wrong question, and it answers it well enough to disguise that it has done so.

- A registry records who deployed a model. That is not the same as who is accountable for it. Provenance is not authority.
- Monitoring detects drift. It does not decide who acts on the drift, under what mandate, or by when. An alert with no owner and no stopping authority is only a louder version of the dashboard nobody read.
- The questions that actually govern a model are not technical, and no tool answers them. What error rate is acceptable, and to whom? Who bears the cost of a wrong automated decision? Who is permitted to veto a model, or pull it out of production, over the objections of the team that built it?

*"No feature store confers the authority to stop a model.
That authority has to be given to a person, on purpose."*

Better tooling makes governance possible. It has never, on its own, made governance happen. The two are separated by an act of organisational will that no amount of infrastructure supplies.

6. What "in production" should have meant

The reframe is simple to state and hard to live: the right question was never whether the model was in production, but who could stop it, on what evidence, and by when. A model with no answer to that question is not in production. It is merely running, unsupervised.

Four commitments follow, and they belong to the organisation chart far more than to the platform.

1. Every model in production has one accountable owner — and it is the business owner of the decision the model makes, not the data scientist who built it and not the platform team that deployed it.
2. It carries a stated tolerance, agreed before launch: what it is allowed to get wrong, and by how much, before someone is obliged to act.
3. Its monitoring is tied to authority. Whoever reads the drift has both the mandate and the means to pull the model, without first assembling a committee.
4. It has a retirement trigger written down at birth. A model is the one asset that decays silently, and it can do the most damage while still appearing to work.

None of this is exotic. It is the ordinary discipline of accountable decision-making, applied to a decision-maker that happens to be made of statistics rather than of a job description. The textbooks left it out not because their authors were careless but because governance is not a chapter in an engineering book. It lives in the org chart — in who is allowed to say no.

The organisations that came through these two years well were not the ones with the most elegant pipelines. They were the ones who, before the model ever went live, could already answer the auditor's question, and knew exactly whose job it was to answer it.