AI in Healthcare: From Pilot to Production

Why most pilots stall, and what it actually takes to operationalize

Faris Fyzee | M.Sc. Computer Science, Georgia Tech (AI Specialization)

Executive summary. Healthcare organizations are not short on AI pilots. They are short on AI that survives contact with production systems, clinical workflows, and regulatory scrutiny. Industry surveys consistently report that the majority of healthcare AI initiatives never reach production, and of those that do, a significant share are quietly decommissioned within a year. The bottleneck is rarely the model. It is data readiness, governance, integration, and ownership. This paper lays out where pilots stall, what operational readiness actually requires, how to sequence the work, and how to structure governance so that the investment continues to pay back over time.

Why AI pilots stall

Pilots typically succeed in controlled conditions and fail the moment they meet the real environment. The failure is rarely dramatic. More often, the pilot produces a respectable accuracy number, a slide deck circulates, and then the project slowly loses altitude as the organization realizes the work required to turn the prototype into something operable. The recurring failure modes are consistent across payers, providers, and health systems:

- **No production-grade data.** Pilots run on hand-curated extracts that were cleaned, joined, and labeled by a data scientist over several weeks. Production requires continuous, reconciled feeds from claims, RCM, and EHR systems with known lineage, refresh cadence, and error handling. The distance between a curated CSV and a production pipeline is measured in months of engineering work, not days.
- **No clinical or operational owner.** A data science team cannot deploy into a clinical workflow alone. Without a named operational owner — a VP of revenue cycle, a CMIO, a service-line leader — the work has nowhere to land. The pilot produces a model; no one owns the decision to put it in front of a user.
- **Governance treated as a final gate.** Compliance, privacy, and model risk reviews introduced at the end of the project force rework or kill the deployment outright. A governance body seeing a model for the first time two weeks before go-live is effectively being asked to rubber-stamp it, and when they refuse, the program loses months.
- **Integration underestimated.** Getting a prediction into an EHR at the right point in the workflow, with the right user, at the right time, is harder than building the model. It requires vendor coordination, interface work, security review, and user experience testing that is routinely left out of the original budget.
- **No measurable target.** Pilots often optimize accuracy metrics rather than an operational KPI (denial rate, length of stay, time-to-authorization, avoidable readmissions). Accuracy without a

business outcome cannot be defended to a CFO, and when the next budget cycle comes, the program is the first one cut.

- **No plan for the day after go-live.** Organizations treat deployment as the finish line. In reality it is the starting line for monitoring, retraining, drift management, and workflow refinement. Without a plan and a team for that work, the model degrades in silence until someone notices it is doing more harm than good.

Data readiness: claims, RCM, and EHR

Operational AI in healthcare lives or dies on the data layer. The model is the visible part; the data pipeline is where the real investment goes. Three domains matter most, and each has characteristic failure patterns that need to be understood before any model work begins.

Claims data is structured but lagging and often incomplete on the provider side. Adjudication status, remit codes, and payer-specific edits must be normalized before any model sees them. Organizations frequently discover that their "claims data" is actually a blend of submitted claims, adjudicated claims, and pending claims, each with different fields populated. Two to three months of reconciliation work is a realistic estimate before claims data is trustworthy for training or inference, and that work has to be re-done whenever a new payer contract, a new EDI format, or a new remit code is introduced.

RCM data sits across multiple systems — charge capture, coding, billing, collections — and is frequently the most neglected. Denial reason codes, work queue states, A/R aging buckets, and payer follow-up activity are the operational signals that drive ROI for any revenue cycle AI use case. If this data is not captured at the transaction level with timestamps and user attribution, AI cannot move the needle on revenue cycle. The most common symptom: an organization that wants to predict denials but has no reliable record of when a denial was worked, by whom, and with what outcome.

EHR data is rich but messy. Structured fields capture a fraction of the clinical picture; the rest lives in notes, flowsheets, and free-text fields. Any production use case must account for documentation variability across providers, version differences in the EHR, and the reality that "mapped to a code" does not mean "clinically accurate." Problem lists drift out of date, diagnoses are carried forward without review, and medication reconciliation is inconsistent. A model trained on EHR data without understanding these patterns will learn the documentation habits of the organization, not the underlying clinical truth.

Before any model work begins, the organization should be able to answer, for each data source: Where does this come from? How fresh is it? Who owns it? What is its known error rate? What happens when the source system changes? If those questions do not have clear answers, the project is not ready to start — and starting anyway guarantees rework later.

Governance, compliance, and integration

These three are frequently treated as separate workstreams. In practice they are the same problem: can this model be deployed, trusted, and maintained in a regulated environment? Pulling them apart is convenient for org charts but dangerous for delivery.

- **Governance** must be embedded from day one, not bolted on. That means a model inventory maintained in one place, documented intended use and known limitations, bias and performance monitoring built into the pipeline, and a clear escalation path when drift is detected. The governance body should meet the model team early — during scoping, not during sign-off — and should have the authority to shape the design, not just approve it.
- **Compliance** — HIPAA, state privacy laws, CMS rules, and increasingly state-level AI disclosure requirements — is not a checkbox. PHI handling, minimum necessary standards, Business Associate Agreements with any vendor whose model touches patient data, and comprehensive audit logging all need to be designed into the architecture. State AI regulations are moving quickly, and organizations that have not centralized their model inventory will not be able to respond to disclosure or impact-assessment requirements when they arrive.
- **Integration** is where most projects die quietly. A model that requires a clinician to open a separate tab, log in again, and interpret a probability score will not be used. Inference must be delivered inside the workflow the user is already in — EHR inbox, work queue, prior-auth screen, discharge planning view — with latency measured in seconds, not minutes, and with an action the user can take in one or two clicks. The integration cost is almost always larger than the modeling cost, and it is where EHR vendor dependencies, interface engine capacity, and security review cycles compound.

A useful discipline: for any AI use case, before approving the build, the team should produce a one-page workflow sketch showing exactly where the prediction appears, who sees it, what they do with it, and what happens if they ignore it. If the team cannot produce that sketch, they do not yet understand the use case well enough to build it.

What it takes to operationalize

Moving from pilot to production is a program, not a project. It requires a different organizational posture, a different budget model, and a different set of success metrics. The organizations that do this successfully share a consistent set of patterns:

- **Named accountable owner on the operational side** — not IT, not data science — typically a VP of revenue cycle, CMIO, COO delegate, or service-line leader whose performance is directly affected by the outcome. This person owns the decision to deploy, the decision to roll back, and the relationship with the users.
- **Production data pipelines built before the model,** not after. If the pipeline cannot run reliably at the required cadence with real data, the model cannot run. Teams that invert this sequence — build the model on extracts, then try to productionize the pipeline — consistently run six to twelve months late.

- **Tight feedback loops.** Model outputs are logged, outcomes are tracked, and performance is reviewed on a set cadence, monthly at minimum. The review should include both technical metrics (accuracy, calibration, drift) and operational metrics (usage rate, override rate, downstream KPI movement). Without the operational view, the team is flying blind on whether the model is actually helping.
- **Clear decommission criteria.** A model that underperforms should have a defined threshold for rollback and a named decision-maker. Organizations without this end up with zombie models in production — technically still running, increasingly out of calibration, and politically inconvenient to shut down because no one wants to be the person who pulled the plug.
- **Change management treated as first-class work.** Training, workflow redesign, super-user programs, and incentive alignment consume more effort than the technical build. Budget accordingly. A model that is technically correct but procedurally rejected by the users is a failure, and rescuing it after the fact is far more expensive than doing the change work up front.
- **A realistic cost model for the full lifecycle.** Pilots are often funded as capital projects; production AI is an operating cost. Monitoring, retraining, vendor fees, governance overhead, and incident response do not end at go-live. Organizations that do not plan for ongoing cost end up either under-investing in maintenance or quietly shutting models down when the next budget cycle tightens.

Sequencing: what to do first

A common mistake is to pick the most exciting use case first. The right first use case is the one that is boring enough to succeed: stable data, a clear operational owner, a measurable KPI, and a workflow integration that does not require heroic EHR work. A successful boring deployment builds the data pipelines, governance habits, and deployment muscle that more ambitious use cases will need later. A failed exciting deployment sets the program back by a year and damages the credibility of the team.

In practice, the sequence that works most often is: revenue cycle automation first (where the ROI is measurable and the data is relatively clean), operational analytics second (staffing, scheduling, throughput), and clinical decision support last — because clinical use cases carry the highest regulatory, safety, and change-management burden and should be attempted only after the organization has established the habits of production AI on lower-stakes work.

Bottom line

The gap between a working pilot and a production deployment is not a technical gap. It is an operational, governance, and integration gap. Organizations that treat AI as a clinical or financial operations program — with owners, data contracts, measurable KPIs, and a realistic view of the full lifecycle cost — move past the pilot phase. Those that treat it as a data science initiative stay stuck, regardless of how good their models are.

The question leadership should ask is not "can we build this model?" It is "can we run this model every day, in our workflow, under our regulators, for the next three years?" If the answer is no, the pilot is not ready to scale, and no amount of additional modeling work will change that.