FREE SAMPLE CHAPTER

Validating AI in GxP

A Practitioner's Guide

Chapter 3
Why AI validation breaks traditional CSV

Sachin Bhandari

TrustBridge Compliance (formerly CSV Interactions)

CHAPTER 3

Why AI validation breaks traditional CSV (and what still carries over)

A 25-year CSV veteran opens a Tier 3 AI validation file and finds the model has been retrained twice since PQ sign-off, neither time through change control. No error has appeared anywhere in the system log; the signature page on the VMP is current, and the model the signatures sit behind is two versions ahead. CSV taught the same engineer to validate once, manage change, and revalidate when something changes, and AI ignores all three rules. The system does not stay put. The change is not always a human decision. And the validated state degrades without a single error message or downtime event. CSV got more right than the current discourse admits; the question is what carries over, and what breaks.

3.1 Four assumptions that CSV makes

CSV grew up around software that was deterministic and stable. The four assumptions underneath the discipline are usually inherited rather than stated; AI breaks each one.

1. **Same input, same output.** The test script's expected-result column collapses for probabilistic outputs.
2. **The system stays put.** Retraining and vendor model updates change behaviour without a change-control event.
3. **Behaviour depends on configuration.** Training data is now part of the validated configuration; the CSV baseline never listed it.
4. **Failure is detectable.** A drifting model produces no error, no downtime, no user complaint. Failure is silent.

> **Important to remember.**
>
> *CSV assumed Cell A: static parameters and deterministic output. The four assumptions map cleanly onto the cell. The next four sections take each assumption in turn.*

3.2 Why expected results break

The first fracture is on the output axis: the move from Cell A's deterministic output to Cells B and D's probabilistic output. Traditional CSV test scripts have an expected result column. The actual result either matches or it does not.

For probabilistic models there is no single expected result. The output is a distribution: a confidence score, a probability, a class label that may itself be the most likely class out of several. Two correctly-functioning probabilistic models given identical inputs may legitimately produce different outputs. The script's expected-result column collapses.

Three responses fail. Ignoring the issue and treating the most likely output as the expected. Giving up on scripted testing and falling back on aggregate accuracy alone. Writing expected results post hoc, after observing outputs on the test set. All three produce test packages that conceal variance or mask the framework choice the validation should have made explicit.

Three responses work. Scripted testing where the input is a labelled dataset and the expected result is a metric (sensitivity, specificity, F1, calibration error) against a pre-defined acceptance criterion. Structured exploratory testing for boundary cases and underrepresented inputs, with a charter, session notes, and defect reports. Hybrid testing for Tier 2 and above: a scripted phase for performance metrics and an exploratory phase for boundary behaviour. Chapter 10 develops the hybrid approach.

3.3 Why the system does not stay put

The second fracture is on the parameter axis: the move from Cell A's static parameters to Cells C and D's adaptive parameters.

Most AI systems break the validated-state assumption in two ways. The system can change without a human decision through retraining on production data. A vendor update to a hosted ML system can change inference behaviour without an operational change at the customer. Both modes update the model parameters or pipeline behaviour. Both can pass through the audit trail invisibly if the configuration baseline does not capture the model artefact hash, the framework version, the dependency versions, and the inference pipeline.

The CSV instinct is to forbid the change. That instinct is correct for critical GMP under Annex 22, where dynamic models are not permitted. It is incomplete for non-critical use, where a retraining cadence can be a feature rather than a bug.

The CSV principle that survives is that no change to the validated state is permitted without a documented governance step. The AI translation is that retraining is the governance step, with its own trigger log, data quality gate, delta validation, and approval workflow. The principle holds. The mechanism changes. Chapter 13 develops retraining governance in detail.

3.4 Why behaviour depends on training data

The third fracture is in the configuration baseline. CSV baselines list the system version, the patch level, the parameter set, the user roles, the integration endpoints. They do not list the data on which the system was trained because, traditionally, the system was not trained.

For AI systems, training data is part of the validated configuration. A model trained on a different dataset is a different model. A model trained on a contaminated dataset is a contaminated model. ALCOA+ applies to training data with the same force it applies to operational records.

This is the layer at which most quality systems are not yet structured: unvalidated extractions from validated sources, retrospective labels without reliability evidence, tolerated data quality issues, all outside the original Annex 11 text and all within scope of Annex 22. Chapter 7 develops this layer in full.

The CSV principle that survives is configuration management. The AI translation is that the configuration baseline now includes training data lineage, label quality, pre-processing specification, exclusion justification, and the version of the dataset used to train each deployed model.

3.5 Why failure is silent

The fourth and most operationally consequential fracture is in failure detection. A traditional system that fails produces an error, a downtime, or a user complaint. The audit trail captures it. The deviation is opened. The CAPA loop closes the issue.

> **Caution.**
>
> *A drifting AI model does not produce errors. It does not go down. The system log shows a normally functioning service. Outputs continue to flow at the expected rate, with the expected confidence, into the expected workflows. The HITL reviewer signs the same form they signed yesterday. Failure is silent until the monitoring data exposes it.*

What has happened is that the relationship between input and correct output has shifted. Concept drift, when the world has changed and the right answer is now different. Data drift, when the input distribution has moved. Label drift, when the prevalence of each output class has changed. None of these announces itself.

The only signal is the monitoring data. Performance metrics calculated against a labelled sample of production outputs. PSI and KS statistics on input features. Output distribution and confidence trends. Override rate and direction. Periodic and event-driven review records that turn signal into action. Chapter 11 develops continuous monitoring and drift detection.

The CSV principle that survives is that the validated state must be maintained. The AI translation is that monitoring is the mechanism by which the state is maintained. PQ is the moment the monitoring plan starts running, not the moment the validation ends.

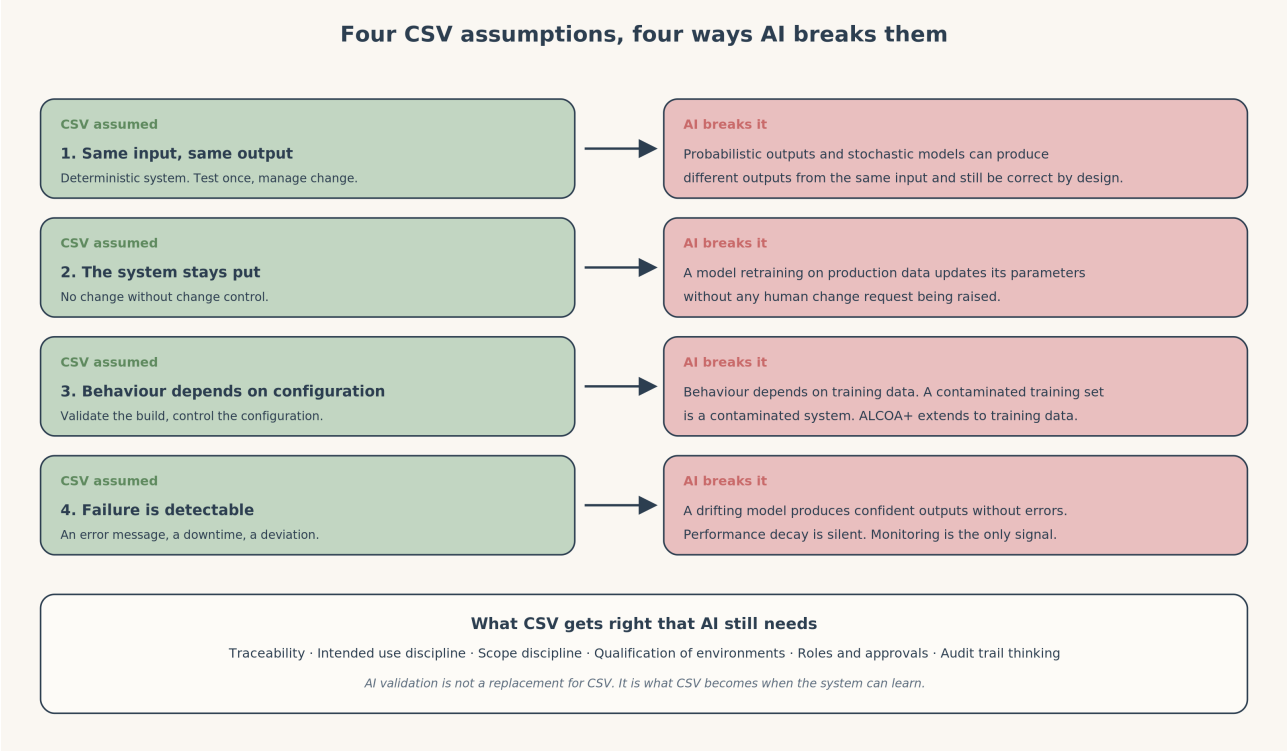

Figure 3.1. CSV assumptions versus AI realities.

The figure pairs four CSV assumptions with how AI breaks each one.

3.6 What CSV gets right that AI still needs

The discourse sometimes implies that AI validation is a fresh discipline. It is not. Most of the principles that built CSV continue to apply. The mechanisms change; the principles hold.

CSV principle	AI extension
Traceability. Any decision must be reconstructable from the records.	Add training data lineage, model version tags on every GxP record, and feature attribution sample logs.
Intended use discipline. What the system will and will not do is written down before deployment.	The Intended Use Statement gains an input sample space, output format, exclusions, and SME approval.
Scope discipline. The validated scope is what the validation actually covered.	Add the sample space, the subgroups, and the explicit exclusions.
Qualification of environments. The system runs in a qualified environment with controlled changes.	Extend to model artefact integrity, framework versions, dependency versions, and inference pipeline.
Roles and approvals. No record reaches the validated state without a named, qualified approver.	Add SME approval of intended use, QA approval of acceptance criteria, named owners of monitoring and retraining.
Audit trail thinking. The audit trail is the inspector's primary evidence.	Extend to feature attribution logs, model version tags, alert logs, and review records.

Table 3.1. The six CSV principles that carry over, with their AI extensions.

> **Important to remember.**
>
> *The discipline that AI requires is CSV plus six extensions, not CSV abandoned and rewritten. Organisations with a mature CSV programme are not starting from zero; they are extending. The inspector is more likely to be a CSV inspector who has read Annex 22 than a data scientist who has read CSV. Speak the language of CSV with the AI extensions on top.*

3.7 The CSA mindset shift

CSA is the second important continuity. The shift from documentation-heavy CSV to risk-based, critical-thinking-led CSA was already underway before the FDA finalised the guidance in September 2025. AI accelerates the shift.

The CSA mindset has three components. **Risk-based scope:** do more where the consequence of error is greater, less where it is smaller. **Reused evidence:** rely on validated platform behaviour and vendor test evidence where it is appropriate to rely on it. **Critical thinking:** design the test scope to answer the validation question, not to fill a template.

For AI, this translates as a depth schedule (the schedule Chapter 2 introduced). Tier 1 systems may rely on vendor evidence and lightweight checks. Tier 2 systems require scripted OQ against pre-defined acceptance criteria. Tier 3 systems require full IQ, OQ, and PQ with structured exploratory testing for boundary behaviour. Tier 4 requires maximum depth and pre-deployment regulatory engagement is recommended.

CSA's risk-based scope applies on top of the acceptability matrix cell. A Cell A system at Tier 2 receives the scripted OQ depth. A Cell B system at Tier 2 (an LLM-backed deviation drafting tool, for example) receives a different OQ design: the test scope addresses the probabilistic output through behavioural and grounding tests rather than a single-expected-result matrix. A Cell C system requires a change-control overlay that the CSV framework does not produce by default. A Cell D system on a critical workflow is out of scope under Annex 22; the CSA conversation does not start.

3.8 Worked example: a deviation classification model under CSA

A deviation classification model is being introduced into a Veeva-based eQMS. The model takes the deviation text, the affected product, the site, and the process step as inputs and returns a Critical, Major, or Minor classification recommendation with a confidence score. The QA reviewer must explicitly accept or override the recommendation before the deviation proceeds. Tier 2.

A CSV-only approach would produce a long IQ, OQ, and PQ stack written in the language of expected results. Hundreds of test scripts each with a single input and a single expected output. A change control framework that struggles to express retraining. A configuration baseline that does not list the training dataset. A periodic review that asks whether the system is still installed correctly.

A CSA-with-AI approach produces the following.

IQ. Verify the model artefact hash matches the validated artefact. Verify the framework version, the inference library version, and the dependency versions. Verify the access controls on the model artefact (who can overwrite, retrain, redeploy). Verify the data pipeline configuration (input pre-processing, feature engineering, output post-processing).

OQ. Execute scripted OQ against pre-defined acceptance criteria on a held-out test set: sensitivity at least 0.90 for Critical class, specificity at least 0.85, F1 at least 0.88, calibration error within 0.05. Test boundary behaviour: deviations with very short text, deviations from a site underrepresented in training data, deviations with unusual product-step combinations. Test the HITL override workflow.

PQ. Execute PQ in production with real users for four weeks. Monitor sensitivity, specificity, F1, calibration, override rate, and override direction throughout. Document any anomalies and their resolution before PQ sign-off.

Configuration baseline. Model artefact hash, framework versions, dependency versions, training dataset version, pre-processing specification, post-processing specification.

Monitoring plan. PSI on input features. KS on continuous features. Output distribution and average confidence. Sensitivity and specificity calculated weekly on a labelled sample. Override rate target 5 to 15%; alert if outside.

Change control. Retraining trigger log. Data quality gate. Delta validation against the OQ acceptance criteria. Deployment approval by QA and BPO. VMP update within 10 business days.

This is CSA with AI extensions running through every stage. It is shorter than a comparable traditional CSV package on the low-risk parts and longer on the high-risk parts. The package is also legible to a CSV-trained inspector because the structure is recognisable.

3.9 Mini case: the validation that did not survive PQ

A second-tier pharma site, 2024. A team validates a complaint triage model with a traditional CSV approach. The OQ test scripts have expected results matching the most likely class for each test case. The PQ runs for two weeks against pre-rehearsed scenarios. The model passes. The system goes live.

Three months later, a quality engineer compares the model's classifications to the manual triage decisions over the period. There is a 14% disagreement rate. About a quarter of the disagreements are clearly the reviewer's call. The remainder fall into two patterns: the model under-classifies a particular product family (underrepresented in training data) and over-classifies complaints from one site (which has noisy, short-text complaint records).

The CSV validation passed. A CSA-with-AI validation would have caught both patterns at OQ (subgroup acceptance criteria stratified by product family and site) and confirmed them at PQ (override rate analysis with direction and reason codes). The remediation costs the site three months of rework, a CAPA, and a delayed sponsor audit. CSV did exactly what it was designed to do for a deterministic system. The system was not deterministic, and the validation framework had to be the AI framework.

3.10 Do and Don't

Do	Don't
Treat training data as a configuration item. The dataset version that trained each deployed model is part of the validated state.	Don't try to validate an AI system with a long traditional OQ script with one expected result per row. The pass record will mask variance.
Replace the expected-result test script with a metric-based scripted OQ for Tier 2 and above, supplemented by structured exploratory testing for boundary behaviour.	Don't assume that a traditional change control framework can govern retraining. The change is not human-initiated; the framework has to be redesigned for it.
Plan monitoring as part of the validated state. The monitoring plan should be operational on the day of go-live.	Don't write off CSV as obsolete. It is the foundation that the AI extensions sit on.
Keep the CSV traceability discipline. Every decision must be reconstructable from the records. AI does not change this.	Don't extract training data from a GxP source system without validating the extraction. The source is validated; the extraction is a separate question.

3.11 Inspection question

How does this AI system's validation framework address the four CSV assumptions that AI breaks (deterministic outputs, static system, configuration-only behaviour, detectable failure), and which CSV principles are explicitly retained?

3.12 Key takeaways

1. CSV makes four assumptions: same input produces same output, the system stays put, behaviour depends on configuration, failure is detectable. AI breaks all four.

2. CSV gets six things right that AI still needs: traceability, intended use discipline, scope discipline, qualification of environments, roles and approvals, audit trail thinking.
3. The expected-result test script collapses for probabilistic models. Replace with metric-based scripted OQ plus structured exploratory testing.
4. For non-critical AI, the validated state is maintained through retraining governance rather than by preventing change.
5. Training data is a configuration item. ALCOA+ applies.
6. Monitoring is part of the validated state. PQ is the start of the monitoring plan, not the end of the validation.
7. CSV plus six extensions is the right mental model. CSV abandoned and rewritten is not.

Read the rest

That was 1 chapter of 19, alongside 15 appendices of templates, SOPs, quick reference guides and an inspection question bank.

The full handbook takes you from classifying an AI system and tiering its risk, through building the evidence, to keeping it valid after go-live and standing in front of an inspector.

It's a digital handbook you can open at the chapter you need on a Monday morning.

Get the book → trustbridge-compliance.com/book