

Case Study: Re-Engineering a Global Price-Tier Forecasting Engine

How Niche Intelligence Forward Deployed Engineering delivered a more powerful model, constraint conformity by design, a fully automated self-checking deliverable and AI-orchestrated execution

Niche Intelligence | Forward Deployed Engineering

October 2026

At a Glance

Client	A prominent global market intelligence firm
Program	Quarterly price-tier forecast for a global consumer-device market, delivered to the client's end customers
Engagement	Forward Deployed Engineering (FDE): embedded modeling, data engineering, QA automation and AI orchestration
Scope	2 sales channels, 3 price tiers, 28 geographies, 9 leading OEMs, 45 quarterly and annual periods (2022–2030)
Headline outcome	A production-grade forecasting system that improves every part of the program: • More powerful model: a two-segment lognormal mixture with price spread measured from data, delivering the non-normal distribution the client named as a primary goal • Constraint conformity by design: every client rule codified in a 40-rule data contract and enforced by construction and constrained optimization; boundary-pinned prices, annual-versus-quarter breaks and source-total breaks driven to zero • More credible forecasts: smoothing and a vendor stabilizer that reduce implausible swings while preserving every total • One consistent story: country and vendor views reconciled to each other at every tier, region and period • Automated, self-checking deliverable: the full Excel workbook, with live QC, pivots and exception reports, generated by code • Evidence-backed assurance: regression tests, a ~49-check audit, and fingerprinted, reversible changes • New visibility: upstream source-data issues surfaced for the first time • AI-orchestrated execution: a month of manual preparation replaced by a single command that runs in about two minutes

Niche Intelligence replaced a fragile, manually assembled spreadsheet-and-notebook process with a statistically grounded, fully automated engine. Every client constraint is written down in a governing data contract, enforced in code and verified by tests and audits.

Key Results

Outcome	Before	After
Price-distribution model	Symmetric normal curve	Two-component lognormal mixture with data-derived dispersion
Pinned tier-edge ASPs (modeled, vendor engine)	137	0
Pinned tier-edge ASPs (modeled, country engine)	9	0
Annual ≠ sum-of-quarters tier breaks	99 per year per metric	0 across all years
Vendor tier totals ≠ source basis	12 material breaks	0 material breaks
Written constraint specification	Informal bullet list	40-rule versioned data contract
Automated verification	Manual spot checks	24 regression tests, ~49-check evidence audit, live in-workbook QC
Execution steps	Multiple notebooks, manual parameter edits, hand-built Excel tabs and pivots	One command: <code>python run.py --report-period YYYYQ#</code>
Quarterly preparation time	~1 month of manual effort	~2 minutes end-to-end runtime

The Client and the Challenge

The client publishes a quarterly forecast that splits each market's unit and revenue totals into three retail price tiers by dollar amount. The forecast covers every country, every major OEM and both the certified pre-owned (CPO) and non-CPO channels, through 2030. End customers use it to size premium, mainstream and value segments, so the numbers must be internally consistent and defensible.

Before the engagement, the program had problems on four fronts: the modeling assumptions, the engineering underneath them, the deliverable, and the process used to produce it.

Modeling Assumptions That Did Not Fit the Market

- **A symmetric curve applied to skewed prices.** The first-generation engine assumed a normal (bell-curve) price distribution. Device prices are positive, right-skewed and clustered into value and premium segments. A normal curve ignores all three: it can place volume at negative prices and it misallocates premium volume. Its spread was also assumed rather than measured from the price differences between OEMs, so tier shares reflected a modeling choice more than the market's actual price mix.
- **A distribution assumption that broke under aggregation.** The vendor engine fitted a separate normal curve to each OEM in each country. A sum of normal curves with different means and spreads is not itself normal. When vendor tiers rolled up, the country-level shape therefore contradicted the normal distribution the country engine assumed for the same market.

- **No parity between the country and vendor engines.** Each engine modeled the same markets on its own, with its own distribution and its own reconciliation logic. The client was effectively publishing two competing versions of every market, and vendor tiers did not reliably add up to country tiers.
- **Coherence and business rules forced after the fact.** Tier units, revenue and average selling price (ASP) were not derived from one consistent curve. Reconciliation passes were needed to make them agree, which pushed ASPs onto the \$200 and \$500 tier boundaries and weakened the rule that ASP equals revenue divided by units. Likewise, nothing in the model's structure kept CPO volume out of the sub-\$200 tier; compliance depended on later corrections.
- **Abrupt transitions and an undefined extension method.** Nothing smoothed the change from frozen history to modeled quarters, or from detailed forecasts to the 2028–2030 extension years. This produced large tier growth swings that were hard to defend. Vendor detail also ends with the source data, and there was no consistent, constrained way to split later country totals across OEMs.

Engineering and Data-Integrity Gaps

- **Constraints that lived in people's heads.** The client's rules existed only as an informal list, with no consistent enforcement. Examples: frozen history must never change; tiers must sum to the source totals; ASP must equal revenue divided by units; ASPs must sit inside their tier but not at its edge; CPO must have no sub-\$200 volume; regions must equal the sum of their countries; growth should be smooth.
- **Totals that did not tie out, and history at risk.** Annual tiers could drift from the sum of their quarters, and vendor tier totals left residuals against the source data. No single, enforced policy guaranteed that delivered history matched the client's last submission across both engines.
- **Problems that went unseen.** Constraint violations passed through without any record. Analysts had no exception report showing what broke, where, or what to do about it.
- **Fragile, unauditable workbooks.** The legacy engines were large, interlinked Excel workbooks with error-prone formulas, no version control, no automated tests and no audit trail. Any change risked breaking downstream calculations without warning. Region and vendor names were spelled differently in different places, which risked dropped rows or duplicate entries when users filtered the data.

A Deliverable End Users Could Not Fully Trust

- **Hand-built, inconsistent workbooks.** The workbook contained hard-coded tables where live pivots were required, and positional cell references instead of keyed lookups. Precision was inconsistent: some places rounded to 4 decimals, others to 5, and some truncated instead of rounding. Conditional formatting was applied where it wasn't needed, there were no named ranges, and inherited sheets held thousands of cached formula errors. The result was hard to trust and hard to maintain.

A Slow, Error-Prone Process

- **Month-long preparation.** Analysts spent roughly a month on each quarterly build. They staged inputs, worked through linked workbooks in sequence, edited parameters by hand, rebuilt Excel tabs and pivots, and reconciled numbers across files.
- **Execution confusion.** Report periods and years were hard-coded separately in each workbook, and the country and vendor workbooks had to be updated in the right order with matching settings. A missed edit or a stale period reference could silently pull the wrong data.

The client asked for a more powerful model that kept every constraint intact. They specifically wanted a non-normal price distribution, together with a process their team could run reliably every quarter.

Our Approach: Forward Deployed Engineering

Niche Intelligence embedded engineers directly in the client's modeling workflow, with no hand-off of requirements to a remote team. Our FDE model combines four disciplines in one team:

1. **Quantitative modeling.** We redesigned the statistical core around how prices actually behave.
2. **Data and software engineering.** We turned exploratory notebooks into a modular, tested, reproducible codebase.
3. **Governance.** We translated informal business rules into a versioned, auditable data contract.
4. **AI orchestration.** We used agentic AI tooling to accelerate development, validation and documentation, and designed the pipeline so it can be driven programmatically end to end.

Every change followed the same disciplined loop: snapshot, isolated candidate run, evidence-backed comparison against a baseline, then promotion with cryptographic fingerprints of the promoted outputs.

The Solution

Architecture

The solution has two coordinated engines and an automated assembly layer:

- **Country Engine.** Splits each country's CPO and non-CPO totals into tiers, rolls countries up to regions and the global total, and hands off a compact intermediate file.

- **Vendor Engine.** Splits each OEM's totals into tiers and reconciles them to the Country Engine, so both views tell one consistent story. It also allocates the 2028–2030 extension years and assembles the final deliverable.
- **Shared library.** Eleven focused modules (about 5,500 lines of production Python). They cover tier vocabulary and bounds, period windows, constrained repair, smoothing, stabilization, the history policy, workbook assembly, pivot generation and governed business-rule overrides.
- **Orchestration.** A single command runs both engines in the correct order with the same period settings. It applies parameter overrides without editing source notebooks and reports pass/fail gates and elapsed time.

All time windows (frozen history, modeled detail, extension, anchor and share-base years) come from one input, the report period. This removes the risk of hard-coded years drifting out of date.

A More Powerful Model: The Lognormal Mixture

We replaced the bell curve with a **two-component lognormal mixture**. Prices are positive and right-skewed, and real demand clusters into a large value segment and a smaller premium segment. The mixture models both:

- Each group's mean price comes directly from its revenue divided by units.
- Price spread (coefficient of variation) is **measured from the cross-vendor price mix**, with guardrails for very narrow or very wide groups, rather than assumed.
- A value component and a premium component are blended around the group mean. Their weights are calibrated so the mixture mean always equals the observed ASP.
- Tier unit shares are the area under the curve inside each tier. Tier ASPs are the curve's average within each band. Both are computed in **closed form** from the lognormal distribution function, with no simulation.

Units, revenue and ASP all come from the same curve. As a result, **tier ASPs fall inside their bands and revenue equals units times price by construction**, without the reconciliation passes the old approach needed. For the CPO channel, the model shifts the distribution so all mass sits at or above \$200, which **structurally guarantees** that CPO has no sub-\$200 volume.

A backtest against held-out history placed **89.2% of vendor-level units in the correct tier**, against a theoretical ceiling of 92.9%.

Constraint Engineering and the Data Contract

We codified the client's rules into a **versioned data contract of 40 numbered rules** across ten categories: source authority, dimensions, numeric integrity, tier eligibility, conservation and aggregation, time and frozen history, growth behavior, exceptions and QC, tolerances, and deliverables. Each rule maps to the code that enforces it and the check that verifies it.

Key mechanisms:

- **Constrained repair with no pinning.** When an upstream step leaves a tier ASP on or near a boundary, a constrained quadratic optimizer finds the nearest feasible split. It preserves both unit and revenue totals exactly and keeps every ASP at least one cent inside its band. Compliant groups are left untouched. Infeasible groups are surfaced for review instead of being forced.
- **Frozen history, verbatim.** Delivered history is written back exactly as the client supplied it and re-applied at the end of the pipeline. Cleanup steps are explicitly prevented from touching frozen cells.
- **Tie-out at every level.** Tiers sum to the source totals. Annual figures equal the sum of quarters, with annual ASP computed as total revenue over total units, never an average of ASPs. Regions equal the sum of their countries at tier level. Vendor detail equals country totals.
- **Transparent exceptions.** Every exception records its rule, scope, observed values, allowed bounds and a concrete corrective action. Modeled issues are separated from frozen-source exceptions, so nothing passes silently.

Smoother, More Credible Forecasts

- **Distribution-shift smoothing.** Tier mixes glide from the last full year of history into the forecast using an exponentially weighted blend. A biproportional (RAS) fit then spreads the glided mix back to quarters, preserving every quarterly total exactly.
- **Vendor tier stabilizer.** A constrained optimizer reduces extreme same-quarter year-over-year swings in vendor tier volumes. It holds every vendor-quarter total fixed and never relaxes tier bands. Its denominator and materiality floors stop tiny tiers from generating misleading growth rates.
- **Governed business overrides.** Client-requested judgment calls, such as a premium-only floor for a specific OEM's CPO line or an upward tier-shape adjustment for selected OEMs, are isolated as period-gated, audited rule modules. They are not edits buried in the model.

Automated, Self-Checking Deliverable

The final workbook is generated in full by code, with no manual Excel steps and no Excel installation required. It contains:

- Country and vendor engines, plus long-format flatfiles whose values are **live lookups** into the engines.
- Source inputs, frozen history, and live QC tabs that reconcile outputs to inputs at country and vendor level.
- An interactive ASP chart with a country selector, growth-rate views with targeted highlighting, and pivot reports.
- Exception, reconciliation and growth-violation sheets.
- Named ranges throughout, so every formula reads clearly.

Pivot and workbook structures are generated directly at the file-format level. This let us meet the client's formatting requirements precisely and reproducibly every quarter.

Quality Assurance and Auditability

- **Regression tests.** 24 automated tests cover boundary cases, CPO rules, history policy, exception routing and business-rule invariants. All pass.
- **Evidence-backed audits.** A read-only audit runs about 49 checks across tens of thousands of output rows. It produces an HTML report, charts, a source-file manifest and CSV evidence for every finding.
- **Safe change management.** Every material change is preceded by a timestamped snapshot. Candidate and baseline runs execute in isolation, and promoted outputs are recorded with SHA-256 fingerprints and rollback paths.
- **Sensitivity testing.** A/B runs compared the stabilizer on and off and swept its penalty weight across four settings, so the tuning decisions rest on evidence.

Human-in-the-Loop AI Orchestration

Niche Intelligence uses agentic AI tooling as a force multiplier across the engagement:

- **Accelerated engineering.** AI coding agents, supervised by our engineers, analyzed legacy notebooks, refactored them into modules, drafted tests and diagnostics, and produced documentation. This compressed weeks of work into days.
- **Machine-readable governance.** The data contract, regression tests and evidence audits give both people and AI agents an unambiguous definition of "correct". Automated changes can be validated against it before promotion.
- **Programmatic control.** The one-command orchestrator exposes the report period, engine selection and stabilizer settings as parameters. Either a user or an AI agent can run, compare and validate a full quarterly cycle without opening a notebook.
- **Non-technical explainability.** An interactive explainer page walks business stakeholders through the method, formulas and checks in plain language.
- **Human-in-the-Loop Forward Deployed Engineering.** Human expertise remains essential to the system's architecture, strategic direction, continuous improvement, troubleshooting, testing, validation, and execution. AI accelerates these processes, while human oversight, technical judgment, and engineering expertise ensure their effectiveness, accuracy, and reliability.

Impact

Dimension	Impact for the client
Time to deliver	Quarterly execution reduced from about a month to the roughly two minutes the model takes to run (Country engine about 13 seconds, Vendor engine and workbook about 110 seconds).
Modeling power	A two-segment lognormal mixture replaces the symmetric normal curve, giving the client

Dimension	Impact for the client
	the non-normal distribution asked for and a better fit to real prices. Price dispersion is now calculated automatically from the market's price mix, and the tier split, ASPs, units and revenue all follow from that one curve, with no manual inference. Every constraint is enforced by the model's design and by optimization instead of fragile spreadsheet formulas. Because tier shares respond directly to each market's ASP and dispersion, results are more precise and traceable to observed data. The process runs fully automated, and analysts can still apply judgment through governed, audited controls.
Constraint conformity	All the client's constraints are now enforced by the model itself and checked automatically every run. Tiers always sum to the source totals, every ASP equals revenue divided by units and sits inside its tier, and there are no negative values or revenue without units. Pinned ASPs were cut to zero in both engines, and the CPO sub-\$200 rule is guaranteed by the model's design. Annual figures always equal the sum of their quarters, regions equal their countries at tier level, and vendor detail matches the country engine exactly. Frozen history is kept exactly as the client delivered it. Large growth swings are controlled by smoothing and a constrained stabilizer that preserve every total, and any rate above 50% is flagged for review.
Governance	Informal rules replaced by a 40-rule versioned data contract that maps each rule to its enforcement and verification.
Manual effort and confusion	Manual parameter edits, run-order mistakes and hand-built Excel tabs eliminated. One command with consistent settings replaces them.
Transparency	Every exception carries scope, values and a corrective action. Live QC tabs let end users verify the deliverable themselves.
Risk reduction	Snapshots, isolated candidate runs, fingerprinted promotions and regression tests make every change reversible and provable.
Upstream insight	Audits surfaced source-data inconsistencies, such as historical restatements and source totals that do not reconcile, that were previously invisible. The client can now resolve them at the source.
Scalability	Adding a market, adjusting a tier rule or rolling to a new quarter is a configuration change, not a rebuild.
AI orchestration	The pipeline can be driven, validated and extended by supervised AI agents, which compounds productivity in every future cycle.

By the Numbers

Metric	Value
Geographies modeled	28 (23 markets, 4 regional roll-ups, global total)
OEMs modeled	9 plus vendor total
Periods per run	45 (36 quarters and 9 annual totals, 2022–2030)
Vendor engine cells per metric family	1,680 rows × 45 periods
Frozen-history cells restored per run	2,916, with 100% unit coverage
Vendor tier cells stabilized per run	17,289

Metric	Value
Data contract rules	40
Automated regression tests	24 (all passing)
Automated audit checks	~49, each with CSV evidence
Production Python	~5,500 lines across 11 modules, plus a one-command orchestrator
End-to-end runtime	~2 minutes

What Comes Next

The data contract also records the remaining integration opportunities. That openness is part of the value: the client knows exactly what is certified and what is next. Planned enhancements include:

- Independent source-reconciliation proofs at country and vendor grain, including extension years.
- A single, explicit cutoff for delivered history across all components.
- Configuration-driven export horizons and harmonized tolerances.
- Stable rule identifiers and source lineage on every exception record.
- Seam smoothing into the extension years, informed by external market-intelligence distributions.

Why Niche Intelligence

Niche Intelligence's Forward Deployed Engineers work inside the client's problem: their data, their constraints, their timeframes. In one engagement we delivered stronger statistical models, governed constraint frameworks, fully automated deliverables and an AI-ready operating models. The client now gets a defensible, audit-ready forecast in minutes rather than weeks, and its analysts can spend their time on insight instead of assembly.

Niche Intelligence. Embedded experts. Production-grade systems. Measurable impact.