WHITE PAPER

Building ML That Ships:

A Decision-Maker's Playbook for Fintech Teams

A practical guide to evaluating, planning, and deploying machine learning in financial products, without the failed pilots, blown timelines, and compliance surprises.

Published by Code-B | 2026

code-b.dev

Executive Summary

Machine learning in fintech is widely discussed and more accessible than ever. Yet most initiatives in financial services still fail to reach production. The issue is rarely the technology itself; it is how projects are scoped, executed, and aligned with business goals.

This playbook is designed for decision-makers, not data scientists. It focuses on the practical questions that matter: whether to start, how to structure an initiative, what it will cost, how long it will take, and how to align teams around it.

By the end of this guide, you will be able to assess your organisation's readiness for a specific ML initiative; choose the right approach, whether in-house, SaaS, or with a development partner; and build a clear, credible business case with realistic timelines and expectations.

This playbook is developed by the team at Code B, based on experience working on ML-driven fintech systems and observing where most projects succeed and where they fail for avoidable reasons.

Section 1: Why Most Fintech ML Projects Stall Before Production

1.1 The Gap Between Pilot and Production

Most machine learning initiatives in financial services succeed at the pilot stage but fail to reach production. This is rarely due to poor model performance. In controlled environments, models are built on clean, structured data with no constraints around latency, compliance, or system integration. Production environments, however, introduce all of these complexities at once.

This gap is where most projects stall. A typical pattern emerges: the team demonstrates a promising prototype, leadership aligns on the opportunity, and a timeline is defined. Progress appears steady initially, but over time, momentum slows. After several months, the initiative is deprioritised or quietly shelved, often without a clear explanation of what went wrong. In many cases, the issue is not technical feasibility but a lack of readiness to move from experimentation to production.

1.2 Common Failure Patterns in Fintech ML Projects

Across fintech organisations, a consistent set of failure patterns explains why many ML initiatives do not progress beyond the pilot stage. These are not isolated issues but recurring structural gaps in how projects are planned and executed.

Unclear Problem Definition
Many projects begin without a clearly defined business objective. When goals are broad or ambiguous, it becomes difficult to determine scope, allocate resources, or measure success. A

viable ML initiative requires a specific, measurable outcome tied to a defined dataset. Without this clarity, the project lacks a concrete direction and cannot reach a meaningful conclusion.

Weak Data Foundation
Data challenges are often underestimated at the outset. Teams may assume that relevant data is readily available, only to discover later that it is incomplete, unstructured, or inconsistently labeled. In practice, data preparation frequently takes significantly longer than expected. Without a thorough data audit early in the process, timelines slip and model performance is compromised.

Late-Stage Compliance Integration
Regulatory requirements are sometimes treated as a final step rather than a foundational consideration. As a result, compliance questions arise late in the development cycle—around explainability, bias, and auditability—when the model is already built. Addressing these requirements at that stage often requires reworking model design or retraining systems, adding significant delays and cost.

Lack of MLOps and Lifecycle Management
Even when models are successfully deployed, many projects fail to establish ongoing monitoring and maintenance. Without systems to track performance, detect model drift, and schedule retraining, model accuracy degrades over time. This decline may go unnoticed until it begins to impact business outcomes, undermining the value of the entire initiative.

In most cases, these challenges are not isolated technical issues but symptoms of a broader gap between experimentation and production readiness. Addressing them early is essential for any fintech organisation aiming to build and scale machine learning systems successfully.

Section 2: Defining the Right Problem Before Building Any Model

2.1 Why Vague Goals Lead to Failed ML Projects

A critical distinction in any machine learning initiative is the difference between a business goal and a well-defined ML objective. A statement such as “we want to reduce fraud” reflects intent, but it is not actionable. In contrast, a goal like *reducing the false positive rate on card-not-present transactions from 4.2% to below 2% within twelve months, without increasing the fraud miss rate beyond 0.8%,* provides clarity on performance, trade-offs, and success criteria.

This level of precision is essential because machine learning systems operate within defined constraints. A clear objective determines what data is required, how performance will be measured, and what level of accuracy is acceptable. Without it, teams cannot align on priorities or make informed decisions during development.

Vague goals also create alignment issues across stakeholders. Product teams, data scientists, and leadership may each interpret success differently. These misalignments often surface late in the process, when the model is already built and the team is unable to answer a fundamental question: *is this ready for production?*

2.2 A Framework for Structuring ML Problems

To move from a business idea to a buildable ML solution, the problem must be structured clearly. The following four components provide a practical framework for defining machine learning initiatives in fintech.

Outcome Definition
Start by identifying a measurable outcome. Define success in terms of a clear before-and-after metric, such as reducing fraud loss rate, lowering default rates, or improving processing time. This creates a concrete target that guides development and evaluation.

Data Source Mapping
Determine what data is available to support the objective. This includes identifying relevant datasets, assessing whether they are labeled, and understanding their volume and historical coverage. Without sufficient and reliable data, even well-defined problems cannot be solved effectively.

Decision Mapping
Clarify the decision that the machine learning system will influence or automate. Identify how this decision is currently made, what inputs it relies on, and the cost of incorrect decisions. This ensures that the model is aligned with real operational workflows.

Constraint Definition
Define the boundaries within which the system must operate. These may include latency requirements, explainability standards, fairness considerations, and integration with existing systems. These constraints shape both the model design and its deployment strategy.

2.3 Practical Examples of Problem Framing

Fraud Detection in Payments
A vague objective, such as improving fraud detection lacks direction. A well-framed objective specifies measurable targets, such as reducing fraud losses on card-not-present transactions from 0.9% to below 0.5% within nine months, while maintaining a false positive rate under 1.8% and ensuring response times remain under 80 milliseconds. This framing clearly defines performance expectations, acceptable trade-offs, and system requirements.

Credit Scoring for Digital Lending
Instead of broadly aiming for better credit decisions, a structured goal focuses on measurable

impact. For example, increasing approval rates for thin-file borrowers by 20% while maintaining a 90-day default rate below 3.2% provides a clear balance between growth and risk. This also highlights the need for alternative data sources, such as transaction-level insights.

KYC Automation in Digital Banking
A general goal to improve onboarding speed becomes actionable when defined as achieving a 70% straight-through processing rate for identity verification within six months while reducing average completion time from 11 minutes to under 3 minutes. This establishes clear operational targets and aligns the system with user experience improvements.

2.4 Indicators That a Problem Is Not Ready

Certain signals indicate that an ML initiative is not yet ready for development:

- Success is defined in qualitative terms without measurable metrics
- Data sources have not been identified or validated
- The stakeholder responsible for the outcome is not involved in the project
- Timelines are driven by external pressures rather than business needs
- Compliance and regulatory considerations have not been addressed early

Defining the problem correctly is the most important step in any machine learning initiative. When done well, it aligns stakeholders, clarifies scope, and significantly increases the likelihood of moving from concept to production.

Section 3: What Your Data Can Actually Support, and What It Cannot

3.1 Why Data Preparation Takes Longer Than Anyone Thinks

In most production ML projects, 60 to 70% of total project time is spent on data work: collection, cleaning, labelling, feature engineering, and pipeline construction. Most teams estimate this at 20 to 30 %. The gap between estimate and reality is the most predictable source of timeline and budget overruns in fintech ML.

Financial data is harder to work with than it looks. Transaction data is high-volume but often inconsistently labelled across different channels and time periods. Fraud labels are noisy: a transaction marked as legitimate at the time may be disputed weeks later. Credit data has survivorship bias built in; you only have repayment data for loans that were approved, not for the applicants who were declined. These are solvable problems, but they require time and expertise.

3.2 The Data Audit Framework

Run this five-dimensional audit before committing to model development.

Volume. How many labelled examples exist for the outcome you want to predict? Supervised fraud models typically require tens of thousands of confirmed fraud cases to train meaningfully. Credit models need at least twelve months of repayment outcomes for a substantial proportion of historical applicants. If volume is below the threshold, the options are to collect more data before starting, use transfer learning from a pre-trained model, or use synthetic data generation to augment the training set.

Quality. What is the completeness rate of key features? What is the null rate? Are there systematic gaps by time period, geography, or customer segment? Is there evidence of data drift between the period the training data covers and the current period?

Labelling. For supervised learning problems, are the labels reliable? For fraud specifically, understand the difference between confirmed fraud labels, disputed transaction labels, and inferred fraud labels. The difference matters significantly for model quality.

Recency. How old is the data? Financial behaviour, fraud patterns, and credit risk characteristics change over time. A model trained predominantly on pre-2023 data may have learned patterns that no longer hold in the current environment.

Accessibility. Is the data in a system that the ML team can actually access? Is it governed in a way that permits use for model training under GDPR, RBI data localisation requirements, and internal data governance policies? Data that exists but cannot be legally accessed is not available.

3.3 The Alternative Data Question

Alternative data sources, such as rent payment history, utility bills, mobile phone usage patterns, e-commerce transaction data, and open banking feeds, can significantly improve credit model accuracy for thin-file borrowers. The practical considerations are data supplier agreements, integration cost, regulatory scrutiny of alternative data in credit decisions, and bias risk if the data proxies for protected characteristics.

Alternative data is worth pursuing when the primary data is genuinely thin, the integration cost is proportionate to the expected accuracy improvement, and the data has been reviewed for potential fairness implications before incorporation.

3.4 Feature Engineering: The Work Between Raw Data and Model Input

Feature engineering is the process of transforming raw transaction records and account histories into the numerical inputs a model can learn from. Three fintech-specific examples: converting raw transaction timestamps into behavioural frequency features, deriving velocity signals from transaction sequences, and creating account-age and relationship-depth features from core banking records.

Feature engineering is where domain knowledge matters most. A data scientist who understands financial behaviour will build better features than one who does not, regardless of

model sophistication. This is one of the strongest arguments for choosing a development partner with genuine fintech experience.

Section 4: Model Selection Without the Jargon

4.1 The Three Questions That Determine Your Model Family

Rather than starting with model names, start with three practical questions.

What is the structure of your data?

Tabular data, rows and columns of transaction records and customer attributes, are best handled by gradient boosting approaches like XGBoost or LightGBM. Sequential data, transaction sequences over time or market time series call for recurrent architectures like LSTM networks. Text data: customer messages and regulatory documents call for transformer-based models. Image and document data, identity verification, bank statement processing, calls for convolutional approaches.

What is the nature of your outcome?

Predicting a known category from historical examples (fraud vs not fraud, default vs repayment) is supervised learning. Looking for unusual patterns without labelled examples is unsupervised. Optimising a decision sequence over time (trading strategy and dynamic credit line management) is reinforcement learning.

What are the latency and explainability requirements?

A fraud scoring model that must respond in under 100 milliseconds at transaction volume has different architectural constraints from a monthly portfolio risk model that runs in batch. An adverse action model that must explain each credit decision has different requirements from an internal analytics model.

4.2 The Explainability Trade-Off Explained Plainly

The most accurate models for many fintech problems are ensemble and deep learning approaches. But these are also the hardest to explain. Logistic regression is fully transparent but less accurate. The practical resolution is to build the most accurate model you can and apply an explainability layer, SHAP or LIME, on top of it. This gives you the accuracy of a complex model with the auditability a regulator expects.

This approach has limits. Post-hoc explanations are approximations, not complete descriptions of model behaviour, and some regulators are becoming more sophisticated about this distinction. For the highest-stakes decisions, those that trigger adverse action notices, some institutions are moving back toward inherently interpretable models to eliminate the approximation risk.

4.3 Why Starting Simple Is Almost Always Right

Before building a sophisticated ensemble model, build the simplest possible model that could work, logistic regression for credit scoring or a rule-based threshold as a fraud baseline, and measure it against the current system. This gives you a performance benchmark, surfaces data quality issues early, and provides a reference point for evaluating whether additional model complexity delivers proportionate improvement.

Teams that skip the baseline step and go straight to a complex model have no way of knowing whether the complexity is earning its keep. Start simple. Move to complexity only when the baseline falls short of the target.

Section 5: Build, Buy, or Partner, The Decision That Determines Your Timeline

5.1 The Three Paths and What They Actually Cost

Building in-house

It gives you full control over the model, the data, the infrastructure, and the team. The hidden cost is time. Hiring a team with genuine fintech ML experience takes six to twelve months. Getting that team productive on your specific problem takes another three to six months.

First production output is typically twelve to eighteen months from the hiring decision. The loaded annual cost for a meaningful in-house ML team in the US or UK market is £800,000 to £1.5 million plus infrastructure and tooling.

Buying ML SaaS

It is the fastest path to deployment. Mature platforms exist for fraud detection, credit scoring, and KYC automation and can be integrated in weeks. The trade-off is customisation. These platforms are built for broad applicability, not for the specific characteristics of your transaction data or customer base.

Their models are trained on their network's data, not yours, which means their performance on your population may differ significantly from their published benchmarks.

Partnering with a specialist development firm

Partnering with a specialist development firm sits between the two. A good partner builds to your specifications, on your infrastructure, using your data. The models belong to you. The team transfers knowledge to your internal people rather than creating a dependency.

The timeline to first production output is typically three to six months. Cost is project-based rather than ongoing headcount, which makes it easier to budget incrementally.

5.2 When Each Path Makes Sense

In-house makes sense when ML is genuinely core to your competitive differentiation, when the model itself is the product, when you need to iterate continuously on proprietary data, and when you have the runway to absorb an eighteen-month build-up period. This describes a small number of fintech companies.

SaaS makes sense when your use case is standard, your volume is within the platform's designed range, and you do not need the model to reflect the specific characteristics of your customer base. It is the right choice for early-stage companies that need fraud detection to run quickly without distraction from their core product build.

A development partner makes sense when you need a customised solution, want to own the result, and cannot absorb the timeline and cost of building in-house from scratch. It is also the right choice when you are making your first significant ML investment and want experienced guidance through the decisions that have the most downstream consequences.

5.3 The Five Questions That Separate Capable Partners from Generalists

1. **Fintech-specific experience.**

   Can they show you production systems they have built in financial services, specifically in use cases adjacent to yours? Generic ML expertise does not transfer cleanly to a regulated financial environment.

2. **The MLOps handoff.**

   What happens after the model is deployed? Do they build the monitoring infrastructure, the drift detection, and the retraining pipeline? Do they document the system in a way your internal team can operate and evolve without ongoing dependency?

3. **Compliance and explainability.**

   Do they build SHAP or LIME explanation layers as standard? Have they produced model documentation for regulatory review before? Do they understand the requirements of the jurisdictions you operate in?

4. **Data architecture, not just model architecture.**

   Can they design a feature store, a data pipeline, and a training infrastructure that your team can maintain? Or do they deliver a model file and call it done?

5. **Transparency on timeline and scope.**

Do they give you a realistic view of data preparation time, compliance review time, and integration complexity? Or do they present an optimistic timeline that assumes clean data and no regulatory friction?

5.4 Risk in Partner Proposals

- Proposals that skip or minimise the data audit phase
- Timelines that assume clean and fully labelled data without having first examined it
- Scopes that describe model development without MLOps infrastructure
- Engagements structured so that the partner retains all model knowledge, and the client cannot operate independently after delivery

Section 6: The Compliance-First Approach to ML in Finance

6.1 Why Compliance at the End Is the Most Expensive Mistake

Here is a scenario most fintech ML teams will recognise. The model is built, validated, and passes performance benchmarks. The team presents it to legal and compliance for sign-off before launch. Legal asks three questions: how does the model explain its decisions to an affected individual, how has it been tested for discriminatory impact, and what is the audit trail for the training data and model versions? None of this was built in.

The launch is delayed by two to four months while the team retrofits explainability, conducts bias testing, and reconstructs documentation that was never created. In some cases, the explainability retrofitting requires changes to the model architecture, which means going back to training.

This is the default outcome when compliance is treated as a review gate rather than a design requirement.

6.2 The Regulatory Landscape in Plain Language

EU AI Act (2026).

ML systems that make or assist in consequential decisions about individuals, such as credit scoring, insurance pricing, and AML monitoring, are designated high-risk. High-risk systems require documented risk management processes, technical documentation of model design and performance, human oversight mechanisms, transparency to affected individuals, and bias testing. The high-risk provisions became enforceable in August 2026.

SR 11-7 (United States).

The Federal Reserve's model risk management guidance applies to banks and bank holding companies and covers model development, validation, ongoing monitoring, and documentation requirements. It is the most operationally detailed of the major frameworks and represents the baseline standard for model governance at any institution with serious regulatory exposure in the US market.

FCA Model Risk Principles (United Kingdom).

Apply to UK-regulated firms alongside the Consumer Duty obligations. The FCA has been explicit that AI and ML models are in scope and that firms must be able to explain model decisions in plain language to affected customers.

RBI Guidelines (India).

Evolving and placing particular emphasis on explainability in credit decisions and data localisation. For fintech companies operating in India, a rapidly growing market, compliance with RBI expectations is not optional.

6.3 Building Compliance In: The Four Practical Steps

Step 1: Compliance scoping at the problem definition stage.

Before writing a line of code, identify which regulatory frameworks apply to this specific model, what documentation will be required for sign-off, and what explainability and fairness standards must be met. This takes one or two days and prevents months of rework later.

Step 2: explainability architecture.

Decide at model design time, not after, whether the model will use post-hoc explanations (SHAP or LIME) or an inherently interpretable architecture. If post-hoc, build the explanation layer into the serving infrastructure from the start. Make sure adverse action notice generation is a designed feature, not an afterthought.

Step 3: bias testing as part of model validation.

Before deployment, run the model's output distribution across protected demographic groups. Document the results, the methodology, and any mitigations applied. This is not just a regulatory requirement; it is a quality test that often reveals feature engineering problems affecting overall model performance.

Step 4: documentation from day one.

Every model should have a model card, a structured document describing the model's purpose, training data, performance characteristics, known limitations, and intended use context. Model versioning, data lineage, and inference logs should be part of the production architecture from the start, not added later when a regulator asks for them.

Section 7: From PoC to Production: The Deployment Roadmap

7.1 What a Valid PoC Actually Proves

A well-scoped PoC proves that a specific ML approach can produce a meaningful improvement over the current baseline on representative historical data. It does not prove that the system will perform at production latency, that the feature pipeline will run reliably at scale, that the model will maintain performance as real-world patterns evolve, or that the system will satisfy regulatory requirements.

The right response to a successful PoC is not "We are done" — it is "We have validated the approach, and we can now scope the production build properly."

7.2 The Five Gates Before Production Deployment

Gate 1: performance validation.

The model must meet the target metric threshold defined in Section 2 on held-out data not used in training or development. The held-out set must be representative of the actual population the model will encounter in production.

Gate 2: latency and scalability testing.

The model must meet its response time requirement under a realistic load. For transaction scoring models, this typically means a sub-100-millisecond response at peak transaction volume. Testing on a laptop with a small dataset does not constitute this validation.

Gate 3: compliance sign-off.

Model documentation, the explainability layer, bias testing results, and audit trail infrastructure must be reviewed and approved by legal and compliance before deployment.

Gate 4: integration testing.

The model must function correctly when connected to the live data pipeline, the serving infrastructure, and any downstream systems that consume its output. Integration failures are common and time-consuming to diagnose.

Gate 5: rollout readiness.

Monitoring dashboards are in place, drift detection thresholds are configured, retraining triggers are defined, and the incident response process for model failures is established.

7.3 The Three-Stage Rollout Sequence

- **Shadow mode.**

The model runs in parallel with the existing system, scoring every decision without the scores being acted on. This reveals how the model behaves on live data before it has any customer impact. Run shadow mode for at least two to four weeks, longer if fraud or credit outcomes take time to resolve.

- **Champion/challenger.**

  Route a defined percentage of live decisions through the ML model and the rest through the existing system. Compare outcomes systematically. This stage produces the evidence that the model is genuinely performing better in production, not just in the test set.

- **Full deployment.**

  Reached only after champion/challenger data confirms the model is meeting its target performance metrics in the live environment.

7.4 MLOps Essentials, What Must Be Running on Day One

- Performance monitoring dashboard tracking the model's key metrics in real time
- Data drift detection that alerts when incoming data distribution deviates significantly from the training distribution
- A defined retraining cadence or trigger-based retraining system
- Model versioning so every deployed version is identifiable, and rollback is possible
- An incident response process defines who is notified and what steps are taken when the model produces anomalous outputs

Section 8: Measuring ROI, What to Track and When to Expect It

8.1 The Timeline Problem

Financial ROI from ML in fintech, measured in reduced fraud losses, lower default rates, or reduced operational headcount, typically takes twelve to twenty-four months to fully materialise from the start of the project. This is not because ML is slow to work. It is because the project has to be built, validated, deployed, ramped through shadow mode and champion/challenger phases, and then run for long enough to accumulate statistically meaningful outcome data.

Teams that present ML investment to a board based on six-month ROI projections create a credibility problem when the six-month mark arrives. The honest framing: leading indicators will be visible within three to six months of launch, and financial ROI will be fully measurable within twelve to twenty-four months.

8.2 Leading Indicators to Track Before Financial ROI Is Measurable

Fraud detection: false positive rate improvement, reduction in manual review volume, and time to detection.

Credit scoring: approval rate on the target thin-file population, model AUC improvement over baseline, and reduction in manual underwriting interventions.

KYC automation: average processing time per application, straight-through processing rate, document rejection rate.

8.3 ROI Benchmarks by Use Case

Typically delivers the fastest financial ROI. Institutions report forty to sixty per cent reductions in false positives, which translates directly into reduced manual review cost and customer friction.

Payback on a well-scoped fraud ML implementation for a mid-sized payment processor is typically nine to fourteen months. Mastercard's 2025 research with FT Longitude found that 42% of card issuers saved more than five million dollars in fraud attempts over two years after deploying AI.

ROI is driven by two levers: reduced defaults from better risk discrimination, and higher approval volume from better identification of creditworthy thin-file borrowers. Full ROI measurement takes eighteen to twenty-four months, given loan repayment cycle lengths. Research from Neontri and Codiste shows that financial institutions report a 15 to 25% improvement in default prediction accuracy versus traditional scorecard-only approaches.

Delivers the most predictable ROI because the cost reduction is immediate and measurable: processing time per application, cost per reviewed transaction, and headcount avoided. Payback is typically twelve to eighteen months.

Section 9: Getting Internal Buy-In, How to Present the ML Business Case

9.1 Who You Are Trying to Convince and What They Care About

The board or investors care about three things: what is the financial impact, what is the risk (operational, regulatory, and reputational), and is this the right time given other capital priorities? They do not need a technical explanation. They need a financial model and a risk assessment.

The compliance and legal team cares about regulatory exposure, explainability, and audit trail. Their default response to any new AI initiative is scepticism, and that is healthy. Bring them into the problem definition stage, not as a review gate at the end.

The engineering or product team cares about integration complexity, maintenance burden, and the impact on existing roadmap priorities. They need a realistic assessment of what the ML

system will require from them in terms of data pipeline support, API integration, and ongoing infrastructure.

9.2 The Three Numbers That Matter Most

A defensible ML business case comes down to three numbers expressed in currency, not percentages.

Expected loss reduction: what the ML system saves annually in fraud losses, reduced defaults, or avoided compliance penalties. Use conservative benchmarks from the ROI section, applied to your own volume figures.

Expected efficiency gain: expressed in cost per transaction, FTE equivalent, or cost per processed application. Apply the same conservatism.

Cost of delay: what every quarter of delay costs in continued losses, continued manual processing costs, or competitive disadvantage from competitors who have already shipped. This is the most powerful number in the business case and the one most often missing.

9.3 Setting Honest Expectations

The decision-maker who says "the model will be live in three months and will save us two million in year one" is creating a credibility problem. The decision-maker who says "we will have a validated PoC in eight weeks, a production system in five months, measurable leading indicators at month six, and full financial ROI data at month eighteen" is presenting a credible plan that will hold up to scrutiny.

The credibility you build by setting honest expectations in the first conversation is worth more than any optimistic projection. You will need that credibility when a regulator asks a hard question, when the data turns out to be less clean than expected, or when the champion/challenger phase takes an extra month to produce conclusive results.

About Code-B

Code B is a fintech-focused software development company that builds production-grade machine learning systems across fraud detection, credit scoring, KYC automation, robo-advisory platforms, and trading infrastructure.

Our approach covers the full lifecycle, from problem definition and proof of concept to deployment with MLOps. Solutions are built on your infrastructure using your data, with your team enabled to manage and evolve the system independently after delivery.

code-b.dev | manager@code-b.dev