

“How do I find true fraud when I only have labels for defaults, and fraud is just a hidden minority inside that default population?”

Let's unpack everything very clearly, step by step, so there's zero ambiguity.

## 1. Problem Setup — Why This Is Hard

### 1.1 What labels you actually have

The dataset gives you labels for defaulters (e.g., DEFAULT\_IND = 1 or similar).

In reality, only some of those defaulters are fraudsters.

Many defaulters are just:

genuinely over-leveraged,

hit by financial hardship,

or making poor but non-fraudulent decisions.

So if you treat all defaulters as fraud, you are:

Over-labeling → you are calling many “honest but unlucky” customers fraud.

Mixing behavioral patterns of fraudsters with normal defaulters.

### 1.2 The true target: Fraud, not Default

Business goal: detect fraud (intentional deception), not just high risk / non-payment.

Technical reality: fraud label is missing → no direct supervised signal for fraud.

So:

You have labels for default (proxy label).

Fraud is a “minority within a minority”:

Overall population → many good customers.

A small subset → defaulters.

Inside that subset → an even smaller, hidden subset = fraudsters.

This becomes:

A weakly supervised + anomaly detection problem where:

“Default” provides a weak proxy for “risk/fraud potential”.

“Fraud” must be discovered as anomalies inside that default/high-risk region.

Supervised learning alone cannot solve this because:

You don't have clean fraud labels.

Training a classifier on DEFAULT\_IND gives you a default model, not a true fraud model.

Fraud behavior is not the same as generic default behavior.

Many defaulters behave in “normal but risky” ways.

Fraudsters often show weird, inconsistent, or extreme patterns.

Fraud is rarer than defaults.

A standard default model will be optimized to catch all defaulters, not to differentiate the special few who are fraud.

So you need something more nuanced.

## 2. Overall Solution — A Two-Stage Hybrid System

To handle this, you designed a two-layer pipeline:

Stage 1 (Supervised):

Use default labels as a weak supervision layer to detect high-risk / suspicious behavior, not to directly detect fraud.

Output: A filtered subset of customers who are “high risk”, where fraud is more likely to be hiding.

Think of this as narrowing down the haystack.

Stage 2 (Unsupervised):

On this risky subset, use anomaly detection to isolate true fraud-like outliers:

Global anomalies with Isolation Forest

Local/density-based anomalies with K-Means

Combine the two into a dual anomaly score.

So conceptually:

Full population → Supervised Risk Model → High-risk subset → Anomaly models → Fraud candidates

This pipeline respects reality:

You use the labels you have (defaults) to narrow the field.

You use unsupervised learning to search for fraud where it is most likely to exist: among the riskiest accounts.

### 3. Stage 1 in Detail — Supervised “Weak Labeling” Layer

#### 3.1 Input Features

You worked with multiple feature families:

Transactional features

e.g., transaction amount patterns, velocity, frequency, time-of-day activity, cross-border transactions.

Behavioral features

e.g., payment regularity, utilization rate, sudden changes in usage, delinquency history.

Financial / profile features

e.g., income proxies, credit limits, product type, tenure, geography.

These capture how a customer uses the product and how their risk evolves over time.

### 3.2 Variable Selection with Information Value (IV)

Before modeling, you did IV-based feature selection:

What is IV (Information Value)?

A measure widely used in credit risk to quantify how predictive a variable is for a binary outcome (here: default vs non-default).

Why use IV?

To rank variables by their predictive power.

To drop uninformative or noisy variables.

To stabilize the model and reduce overfitting.

You:

Calculated IV for each variable with respect to DEFAULT\_IND.

Kept variables above a certain threshold.

Removed low-IV or redundant variables.

This cleaned up the feature space and directly contributed to:

- 12% improvement in F1 score for the XGBoost model after IV-based filtering.

### 3.3 Training XGBoost — But With a Purpose

You used XGBoost as the supervised model, trained on default labels, with this mindset:

You are not predicting “fraud = 1”.

You are predicting:

“High-risk default behavior that is correlated with eventual loss”

This model does two things:

Learns nonlinear interactions and complex patterns that correlate with default.

Produces a risk score for each customer:

Higher score → more default-like, riskier behavior.

Lower score → more normal, low-risk behavior.

You then:

Sorted customers by risk score.

Focused Stage 2 anomaly detection on the high-risk tail (e.g., top X% by score or defaulters with high predicted probabilities).

So Stage 1 is not a final fraud classifier.

It is a filter to define the “search region” for fraud.

### 3.4 Model Explainability with SHAP

You used SHAP (SHapley Additive exPlanations) to:

Globally:

Understand which features most drive default risk.

Validate that the model aligns with business intuition (e.g., very high utilization, erratic repayments, etc., should increase risk).

Locally (per customer):

For a given account, see:

Which factors push risk up.

Which factors pull risk down.

Benefits of SHAP here:

Trust-building with risk/fraud stakeholders:

They can see why someone is tagged as high-risk.

Feature refinement:

Using SHAP insight, you can engineer better features, drop misleading ones, and improve IV + model together.

Operational transparency:

When a case is escalated, analysts can see which behaviors drove the high-risk signal.

This SHAP-driven iterative refinement contributed to the 12% F1 uplift and made the Stage 1 filter reliable enough to build Stage 2 on top of it.

### 3.5 Role of Stage 1 (Conceptually)

Stage 1 serves to:

Reduce the search space:

From 100% of customers → only the riskiest portion.

Increase fraud density:

Fraudsters are more likely to be in the top-risk group than in the whole population.

Make the data manageable for unsupervised analysis:

Anomaly detection on the full population would be noisy, unstable, and less meaningful.

So:

Stage 1 = “Define where to look”

Stage 2 = “Figure out what is truly suspicious within that region”

#### 4. Stage 2 in Detail — True Fraud Discovery via Anomalies

Stage 2 operates on the filtered subset from Stage 1:

Typically high-risk defaulters / top-risk scores where fraud is more likely.

Here, you no longer rely on labels.

Instead, you ask the question:

“Within this already-risky group, who behaves so differently that they look like fraud?”

You used two complementary anomaly detection techniques:

##### 4.1 Isolation Forest — Global Outlier Detection

What it does:

Isolation Forest works by randomly partitioning the feature space:

If a point is an outlier, it can be isolated with fewer splits.

The average path length in the trees informs an anomaly score.

What it captures in your context:

Global outliers:

People whose patterns (spend amount, frequency, geography, category mix, etc.) are extremely far from the majority of high-risk defaulters.

Examples:

Someone doing unusually high cross-border spends.

Sudden extreme cash withdrawals inconsistent with their profile.

Abnormal spike in high-risk merchant categories.

Output:

Each account gets a global anomaly score (higher score = more anomalous).

#### 4.2 K-Means — Local / Density-based Anomalies

What it does:

K-Means clusters similar data points into K groups.

Each cluster has a centroid (average behavior).

The distance from a point to its cluster centroid can be used as a local anomaly score:

Large distance = behaves differently from peers in that cluster.

What it captures in your case:

Local/density-based anomalies:

Small behavioral groups that are unusual relative to the defaulter segment they belong to.

Examples:

Within a cluster of high-utilization, low-income customers, one subgroup shows:

Unusual merchant types,

Or a strange temporal transaction pattern,

Or inconsistent repayment trends.

These might not look extreme globally but are weird within their local neighborhood.

Output:

Each account gets a local anomaly score (distance-to-centroid-based).

#### 4.3 Why Both Models Are Needed

Isolation Forest:

Good for big, obvious outliers.

Might miss subtle but suspicious micro-behaviors.

K-Means:

Good for finding oddballs within a cluster.

Might struggle with extremely global outliers or poor cluster assignment alone.

By combining them, you:

Capture:

Extremely strange patterns (via Isolation Forest).

Subtle but locally odd patterns (via K-Means).

Avoid relying on a single notion of “anomaly”, which could be biased.

#### 4.4 Dual-Ensemble Anomaly Score

You combined:

Global anomaly score from Isolation Forest.

Local anomaly score from K-Means (distance to centroid).

Typical way conceptually (even if you didn't code it exactly this way):

Normalize both scores to a comparable scale (e.g., 0–1).

Create a combined score like:

FraudScore

=

■

.

GlobalScore

+

(

1

-

■

)

.

LocalScore

$\text{FraudScore} = \alpha \cdot \text{GlobalScore} + (1 - \alpha) \cdot \text{LocalScore}$

where

■

$\alpha$  controls the importance of global vs local anomalies.

Rank customers by this combined score.

Top-ranked customers are fraud suspects:

High risk from Stage 1.

Anomalous both globally and/or locally in Stage 2.

## 5. Final Outcomes and Metrics — What Improved, Exactly?

Your system delivered these key improvements:

### 5.1 Detection Quality

Recall  $\approx 75\%$

You were able to capture  $\sim 75\%$  of the (hidden) fraud cases.

Compared to a baseline, this means you are finding more fraud that would otherwise be missed.

False Positive Rate  $\downarrow \sim 30\%$

Even though you increased recall, you reduced FPR by  $\sim 30\%$ .

This means:

Fewer genuine customers are being wrongly flagged.

Fraud analysts spend less time on non-fraud cases.

This precision–recall trade-off is better because:

Stage 1 focuses on the high-risk segment.

Stage 2 refines that segment using robust anomaly scoring.

### 5.2 Operational Efficiency

You didn't stop at just model metrics. You built tools around the model:

Streamlit App

For interactive:

Exploration of the model outputs.

SHAP-based explanations of why someone is deemed high risk.

Viewing anomaly scores for individual accounts.

Useful for:

Data scientists / risk teams to experiment with thresholds.

Sanity check on model behavior.

Power BI Dashboards

For fraud analysts & management:

View distributions of fraud scores.

Monitor volumes of flagged cases.

Track trends over time (e.g., fraud score by region, product, merchant type).

Provides operational transparency and continuous monitoring.

Business impact:

■ Fraud investigation efficiency increased by ~25%

Meaning:

Analysts need to spend less time sifting through irrelevant alerts.

They are guided towards cases with higher anomaly + risk scores.

More fraud discovered for the same or even lower effort.

## 6. Why This Methodology Is Justified (and Smart)

Let's tie the logic together:

Supervised alone doesn't work:

Default labels  $\neq$  fraud labels.

A default model will learn:

"Who tends to not pay back?"

not

"Who is actively committing fraud?"

It confuses financial struggle with deception.

Unsupervised alone doesn't work well either:

Running anomaly detection on all customers:

Lots of noise from naturally diverse behaviors.

Very high false positives.

Little business focus.

Your two-stage hybrid method fixes this:

Stage 1 (XGBoost + IV + SHAP) uses default as a weak but valuable signal:

It narrows the search space to high-risk behavior.

It is interpretable and aligns with credit risk best practices.

Stage 2 (Isolation Forest + K-Means) focuses only on that high-risk region:

Finds outliers among outliers.

Targets the “minority within the minority” structure of fraud within defaulters.

Dual anomaly scoring:

Merges global and local perspectives on abnormality.

Explainability built-in:

IV + SHAP = transparent feature selection and model behavior.

Analysts and risk managers can understand:

Why someone is in the high-risk filter (Stage 1).

Why they are flagged as anomalous (Stage 2 scores & patterns).

Operational readiness:

Streamlit → experimentation & what-if analysis.

Power BI → live monitoring & business reporting.

This is not just a “Kaggle model”; it’s a full fraud detection workflow.