

Fighting Fraud with Machine Learning

Fraud detection that explains every flag, and caught its own data-leakage bug.

BUILT WITH

Fraud Detection

Random Forest

SHAP

Feature Engineering

Python

Scikit-learn

Imbalanced Learning

1M+

transactions at a 0.3% fraud rate

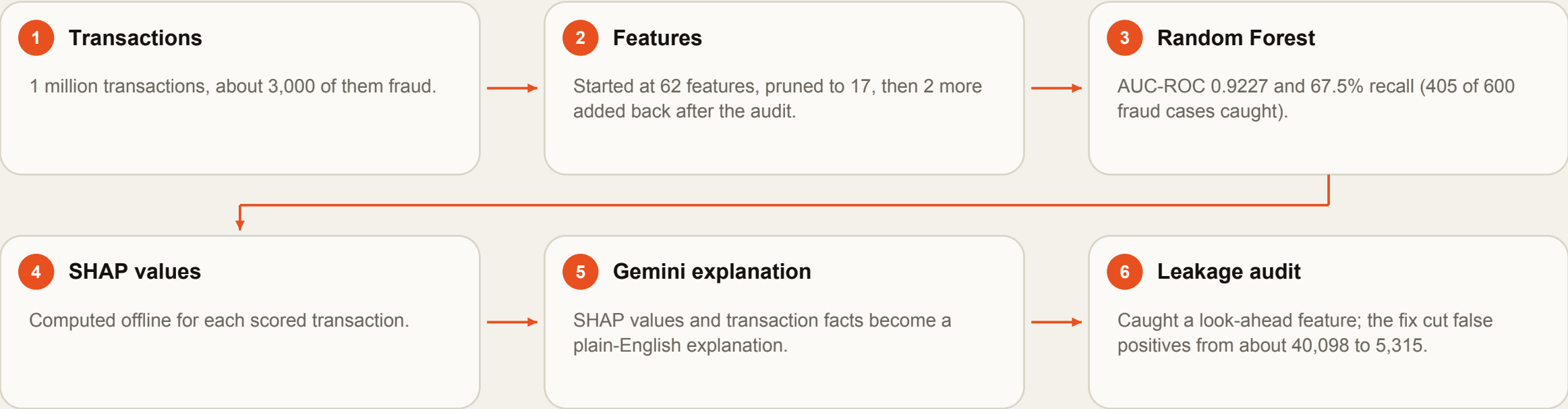
0.82 to 0.92

AUC-ROC after the leakage audit

40,098 to 5,315

approx. false positives after the fix

How it works



SHAP values and explanations are computed offline; only the follow-up chat on a case runs live.

WHAT EACH PART DOES

Feature engineering	62 features pruned to 17, then 2 added back after the audit.
Random Forest	Class-weighted, so fraud counts about 332 times more than normal in training.
SHAP and Gemini (offline)	Explain each flagged transaction in plain English, grounded in the SHAP numbers.
Dashboard (Next.js)	Static pages that read pre-computed results, so nothing heavy runs live.
Follow-up chat	The one live feature: ask about a single case, answered from that case's facts only.

Key decisions

01

Class weighting instead of synthetic fraud

Weighting keeps all the real training data and avoids made-up fraud patterns. SMOTE was tested and deliberately deferred.

02

Heavy work runs once, offline

SHAP and explanations are slow, so they are baked into static files. Only the cheap follow-up question is live.

03

The audit is the headline

An unexpectedly large AUC jump triggered a leakage check. It found a feature that used future data, in a model already deployed, and the fix was verified against the raw rows.

04

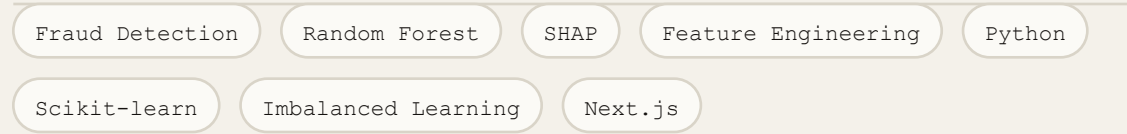
The mistakes are public

A findings file documents an earlier version that reported the wrong gain and an incomplete check, before it was corrected.



Stack, status and what is next

BUILT WITH



KNOWN LIMITS

- One of the original 17 features correlated with the leaky one, so the original 0.822 AUC already carried some diluted leakage.

SOURCE

<https://github.com/wailethewave/Fraud-risk-research-ML>