CEROP

Cross-Border Enterprise Risk & *Operations Predictor*

Two models, two scores, each with a reason you can read.

Graduate machine-learning project · AIT 506, Westcliff University · 2026
Arthunya Kanoklertwongse (Aty) with Ankit Kumar, Divya Gopal and Aswin Shriram Thiagarajan
arthunya.github.io/cerop-console · live demo

THE PROBLEM

Cross-border buyers carry *two risks*

A supplier failing

Owned by credit and procurement. They read balance sheets and decide whether to extend terms or renew a contract.

Goods arriving late

Owned by logistics. They watch orders and decide where to add buffer stock or chase a carrier.

Commercial risk scores exist, but they're paid for, refreshed every so often, and rarely say why a score is what it is.

WHAT CEROP DOES

Two independent models, *on purpose*

Supplier risk

XGBoost on 85 financial ratios. Output: the chance a firm goes bankrupt, plus the factors behind it.

Shipment delay

Random Forest on 27 order features. Output: the chance an order arrives late, plus the factors behind it.

They're deliberately not merged. A solvent supplier can still ship late, and a failing one can ship on time right up to the end. One blended number would hide what each team needs to see.

DATA

Three public datasets, *two very different balances*

Taiwan bankruptcy (UCI)

6,819

firms, 1999–2009; 3.2% went bankrupt

DataCo supply chain (Mendeley)

171,962

orders after cleaning; 55% arrived late

NY Fed GSCPI

monthly

global supply-chain pressure, joined to each order's month

Taiwan bankruptcy (6,819 firms)



DataCo orders (171,962 after cleaning)



Counted from the label files: `results/web/dataset_facts.json` (notebook 07).

PIPELINE

From raw data to a score *you can question*

01
Data
three public sources

02
Cleaning
sentinel values, IDs and free text out

03
Leakage checks
nothing known only after delivery

04
Model selection
three candidates per layer

05
Tuning & threshold
search, then calibrate

06
Export
saved once, loaded read-only

07
Scoring + SHAP
every explanation adds up

08
Console
try it in a browser

One rule throughout: split first, then fit. Scalers, encoders and clipping bounds learn from training data only.

CATCHING LEAKAGE

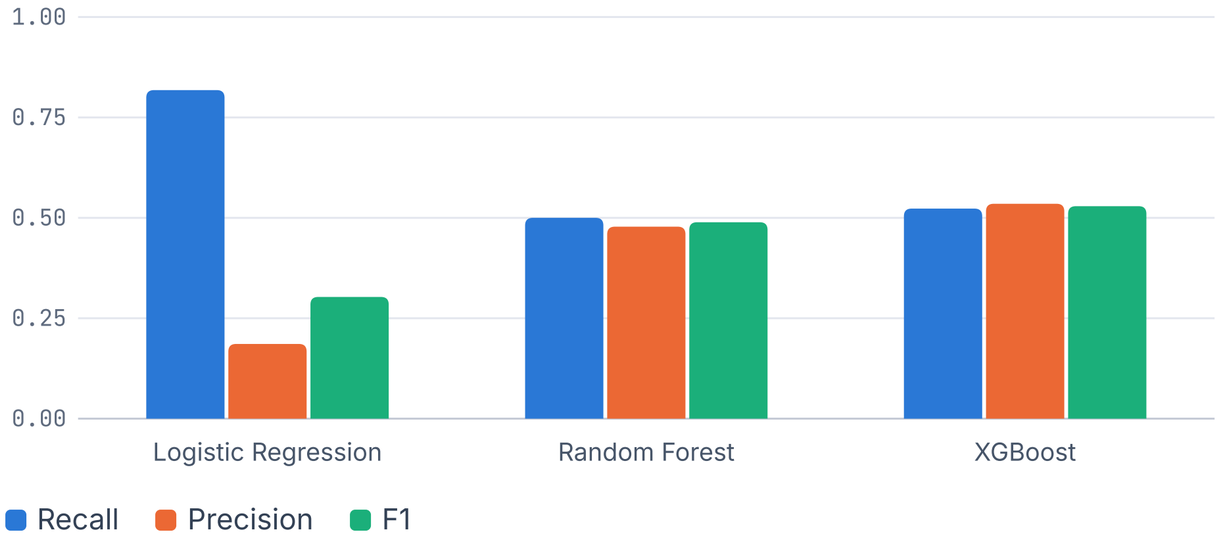
A model that knows the answer *looks great and fails*

- Actual shipping days and delivery status were the strongest predictors of lateness, and both are only known after delivery. Removed in Week 4.
- Order Status: kept in Week 4, then judged a possible leak in Week 6 and dropped, taking the model from 35 to 27 inputs.
- An unreliable tail: orders from October 2017 on behaved erratically, so 8,557 rows were removed before the split.
- Checks that must pass before any model runs: no missing values, every value finite, train and test with the same columns.

Notebook 03 (asserts in cells 10 and 32); row counts from Final Report Table 5.

CHOOSING THE FINANCIAL MODEL

Why *97% accuracy* is the wrong metric



Test set, threshold 0.50. Final Report Table 7.

A model that calls every firm safe would be 96.8% accurate on our test set and catch nothing.

So we compared recall, precision and F1. Logistic Regression catches the most failures (recall 0.818) but most of its alarms are wrong (precision 0.186). XGBoost has the best balance: F1 0.529, ROC-AUC 0.950.

FINANCIAL RESULT

XGBoost on 1,364 unseen firms

	Said solvent	Said bankrupt
Really solvent	1,300 cleared	20 false alarms
Really bankrupt	21 missed	23 caught

F1
0.529

ROC-AUC
0.950

For a credit analyst: it caught 23 of 44 bankruptcies, and 23 of its 43 flags were real. A strong pointer to which files to read first, not a decision on its own.

Official result: `results/week6_financial_tuning_results.csv`, row "XGBoost - Week 5 published".

WEEK 6

Tuning didn't help, *and that's a finding*

Layer	Configuration	F1 at 0.50
Financial	Week 5 settings (tuning split)	0.568
Financial	Tuned XGBoost (same split)	0.505
Operations	Baseline Random Forest	0.724
Operations	Tuned Random Forest	0.721

A bootstrap over 2,000 resamples put the financial change in PR-AUC between -0.135 and 0.025: indistinguishable from noise. We kept the Week 5 model.

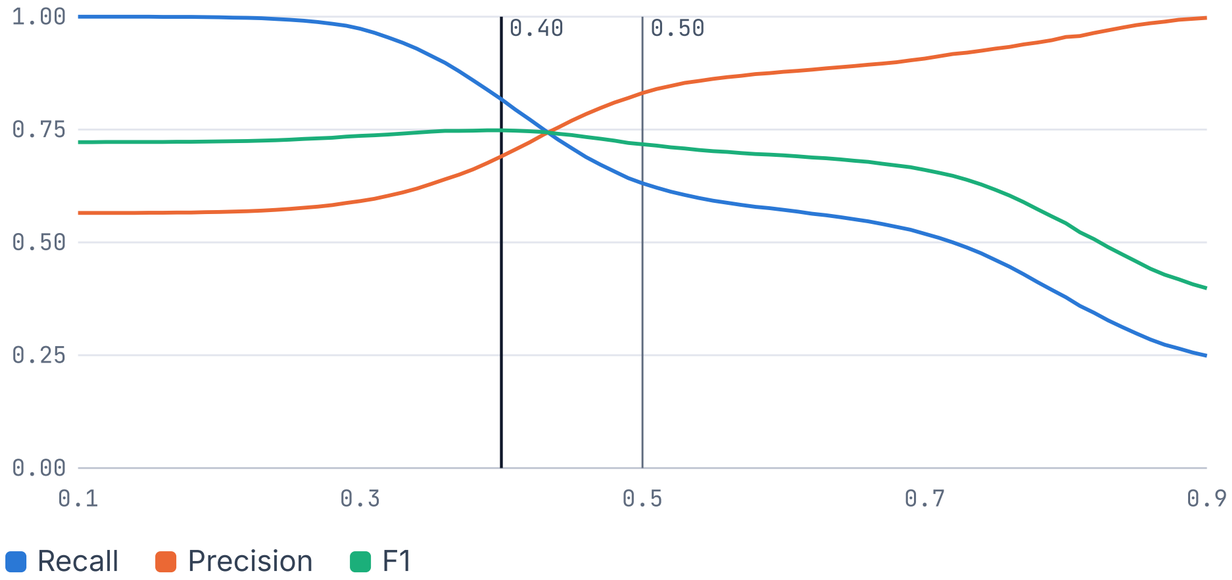
`results/week6_financial_tuning_results.csv`, `results/week6_operations_tuning_results.csv`, `results/week6_financial_metrics.json`.

THRESHOLD

The threshold is a *business decision*

Move the threshold

At `0.40` on the validation set: recall `0.817`, precision `0.690`, F1 `0.748`. This is the highest F1, so it became the threshold.



Validation curve: `results/week6_operations_threshold_results.csv`.

Moving the shipment model's line from 0.50 to 0.40:

Late orders caught
+3,705

False alarms
+4,200

Recall 0.825, F1 0.763 on 34,393 test orders. Whether that trade is worth it depends on what a missed delay costs.

EXPLAINING EACH CASE

Real SHAP, *checked to add up*

A supplier that failed

Bankruptcy risk · FIN-T0150 · went bankrupt **99.6%**

Starting point	**3.2**
Quick ratio	+11.1
EPS, last four seasons	+10.2
Total debt / net worth	+8.1
Retained earnings / assets	+7.3
Cash / total assets	+5.9
Allocation rate per person	+5.1
79 more factors combined	+48.7
Model score	**99.6**

An order that arrived late

Late risk · First Class · OPS-T02270 · arrived late **75.5%**

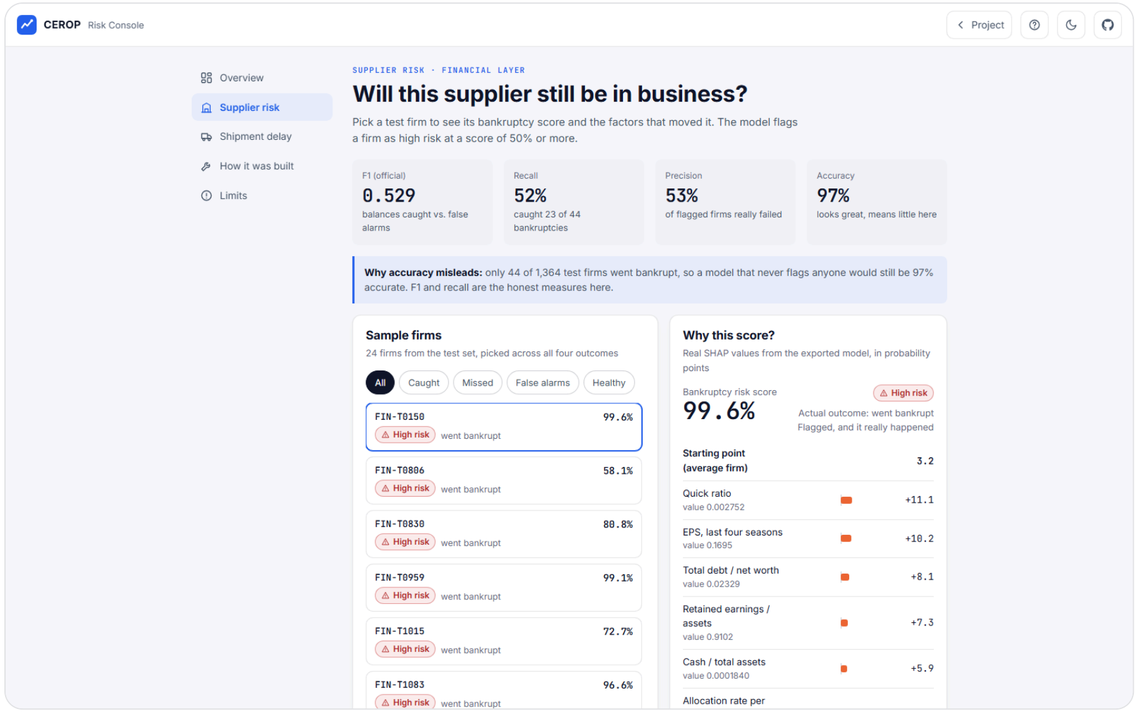
Starting point	**54.8**
Scheduled shipping window (1 day)	+18.2
Shipping Mode (First Class)	+16.4
Payment type (TRANSFER)	-10.7
Supply chain pressure (GSCPI) (-0.18)	-2.0
Order discount (12.00)	-1.3
Order Region (US Center)	+0.8
16 more factors combined	-0.7
Model score	**75.5**

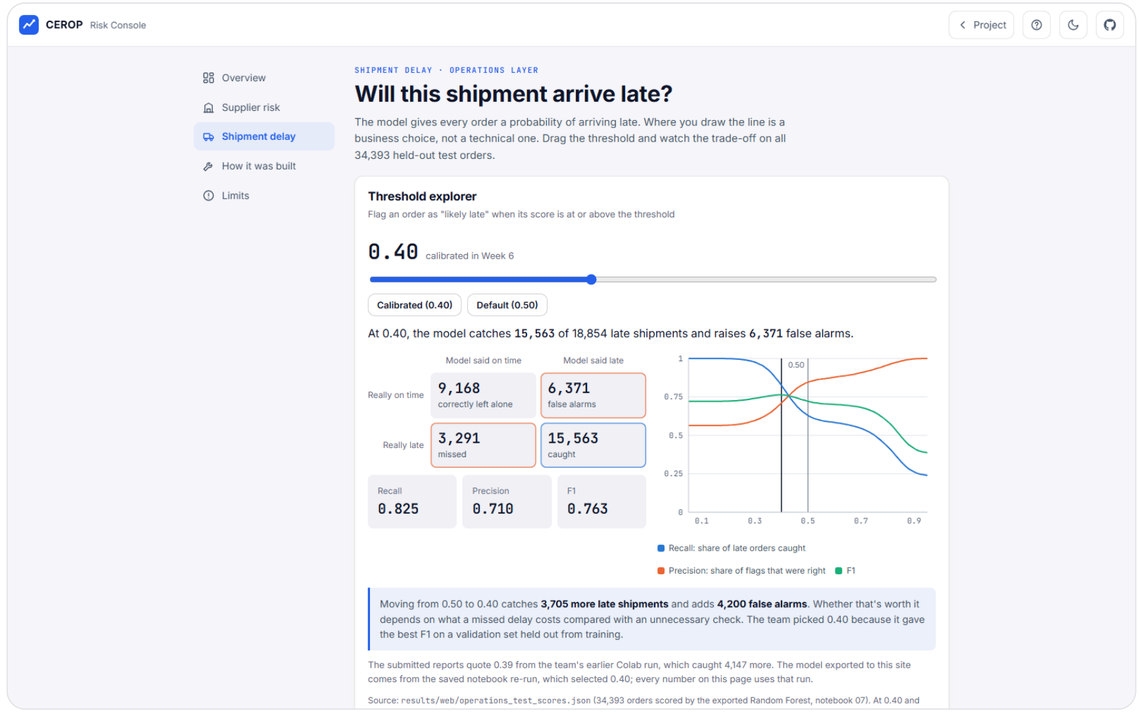
For every case shown, the starting point plus all feature contributions equals the score within 1e-6. Scheduled days and shipping mode carry the same information in this data, so read them together.

`results/web/financial_shap_cases.json`, `operations_shap_cases.json` (notebook 07). Numbers are percentage points.

THE LIVE CONSOLE

Try it on real *held-out cases*





Open the console

arthunya.github.io/cerop-console/console

Browse test cases, drag the shipment threshold, change a supplier's ratios.

LIMITS AND LESSONS

What this *can't* tell you

- Results moved between runs: F1 0.533 locally vs 0.529 reported, same code and seed. Library versions.
- Only 44 bankrupt test firms, so wide uncertainty.
- Taiwan, 1999–2009: not validated elsewhere.
- Collinear shipping features: scheduled days and shipping mode are one signal.
- Prototype scope: no monitoring, retraining or live feed.

What I'd do next

- Pin library versions from day one.
- Bootstrap intervals on every headline metric.
- Export one raw-to-probability pipeline per layer.
- Pick thresholds from real costs, not F1.
- Test on newer, non-Taiwanese data.

TEAM & LINKS

Who built it

Arthunya Kanoklertwongse	financial layer, leakage checks, demo
Ankit Kumar	financial model selection and tuning
Divya Gopal	pipeline integration, operations selection and tuning
Aswin Shriram Thiagarajan	operations EDA and cleaning

My role: the financial layer, verifying the operations leakage, the live demo build (export, scoring, SHAP, console), coordination and report writing.

Project site
Live demo
Final report (PDF)
Code on GitHub
Portfolio
This deck as PDF