CEROP: Cross-Border Enterprise Risk & Operations Predictor

Final Report

Group 4

Ankit Kumar

Arthunya Kanoklertwongse

Aswin Shriram Thiagarajan

Divya Gopal

Westcliff University

AIT 506: Machine Learning

Professor Oueichek Ibba

August 19, 2026

CEROP: Cross-Border Enterprise Risk & Operations Predictor

Introduction

Cross-border trade decisions usually run on two separate checks: does the counterparty look financially sound, and has their delivery track record been reliable. CEROP (Cross-Border Enterprise Risk & Operations Predictor) treats those as one connected question instead of two. The project combines a financial-distress classifier built on the Taiwanese Bankruptcy Prediction dataset with a delivery-risk classifier built on the DataCo Smart Supply Chain dataset, and enriches the operations side with the Federal Reserve Bank of New York's Global Supply Chain Pressure Index (GSCPI) so a macro-level pressure signal sits alongside firm-level and shipment-level data.

Commercial tools already score parts of this problem Dun & Bradstreet's Supplier Evaluation Risk Rating and Moody's supplier risk scorecards both exist for a reason, and their existence is itself evidence that the underlying business need is real (Dun & Bradstreet, 2026; Moody's, n.d.). What these platforms tend not to do is put a company's financial health and its operational delivery performance in front of a decision-maker at the same time, on a shared macroeconomic backdrop. That gap is where CEROP sits: not a replacement for analyst judgment, but a decision-support layer that surfaces two independent risk scores side by side instead of asking a risk or procurement team to consult two disconnected systems.

This report follows the project from problem framing through model finalization across eight weeks of coursework, condensed here into six sections: framing the problem and proposing the approach (Week 2), getting and exploring the two source datasets (Week 3), preparing and integrating the data into model-ready form (Week 4), selecting and shortlisting candidate models (Week 5), and finalizing both models under a shared tuning-and-calibration protocol (Week 6). Each section keeps the two layers Financial and Operations visibly separate where their data,

targets, and class balance genuinely differ, and brings them back together wherever a shared pattern or a shared decision actually holds across both. The full Cross-Layer Synthesis from Week 6, along with the project's overall conclusion, closes out the report.

Frame the Problem and Proposal

Frame the Problem

The objective is to minimize the business and monetary losses enterprises face when they extend credit or enter contracts with cross-border partners without visibility into two risks that often compound each other: financial distress and failure to deliver products through the supply chain. Third-party risk platforms already put a price on part of this problem Dun & Bradstreet's Supplier Evaluation Risk Rating, for instance, scores how likely a supplier is to cease operations within 12 months which confirms there is a real business case here. CEROP's contribution is integrating the financial-risk view and the operations-risk view (late delivery) into one system, with macroeconomic context factored in, rather than treating the two as separate assessments.

The system is meant to work as a decision-support tool for risk, procurement, and finance teams before they extend credit, renew a contract, or evaluate a supplier relationship. Outputs from the financial-distress model and the delivery-risk model are kept distinct, presented the way other risk-scoring platforms present information to decision-makers (Censinet, Inc., 2025), so the results support rather than replace human judgment.

Several commercial platforms already address parts of this problem. Dun & Bradstreet's Supplier Evaluation Risk (SER) Rating assigns suppliers a score of 1 to 9 based on payment behavior, net worth, employee count, and legal filings, and its PAYDEX Score separately tracks payment timeliness (Dun & Bradstreet, 2026). Moody's (formerly Bureau van Dijk/Orbis) offers supplier risk scorecards built on geography, ownership, financial stability, and regulatory exposure

(Moody's, n.d.). Industry guidance also documents standard practices such as security questionnaires, continuous monitoring, and Likelihood × Impact scoring models (Censinet, Inc., 2025). What all these tools share is that they are proprietary, subscription-based, updated periodically rather than continuously, and rarely fold macroeconomic indicators directly into a company-level score the gap CEROP targets.

This is a supervised, batch/offline binary classification problem. Both target datasets carry a historical label bankruptcy status for the financial layer and late-delivery status for the operations layer so training happens on fixed, versioned offline data rather than a streaming setup.

Accuracy alone is not appropriate for the financial-risk layer because the Taiwanese Bankruptcy dataset is heavily imbalanced (about 3.23% positive class); precision, recall, F1-score, and AUC-ROC are used instead. This mirrors the evaluation approach of the dataset's original authors, who assessed models on Type I and Type II error rates and ROC curves rather than accuracy alone (Liang et al., 2016). For the operations layer, comparable published work on the same DataCo dataset reports accuracy, F1-score, and AUC-ROC as standard metrics (Ahmed et al., 2025).

Recall is weighted more heavily than precision for both layers, and the reasoning carries a specific cost asymmetry. In the original bankruptcy study, a Type I error classifying a company that will go bankrupt as safe was treated as more costly than a Type II error flagging a safe company as risky (Liang et al., 2016). That maps directly onto CEROP's business objective: missing a genuinely high-risk cross-border partner costs more than an unnecessary manual review.

For minimum performance targets, the original benchmark study reported Random Forest reaching 87% accuracy on the financial-distress task, against 69% for Logistic Regression and 50% for Linear Discriminant Analysis (Liang et al., 2016); more recent work using additional

imbalance-handling techniques on the same dataset has reached 0.93–0.94 across accuracy, F1, and AUC. The team's own goal was set at 85% or better accuracy and 80% or better recall. For the operations layer, published accuracy ranges from roughly 90% (standard ensembles) to over 96% (hybrid deep-learning approaches), with a comparable framework reporting an F1 of 0.91 and an AUC-ROC of 0.943 on the DataCo dataset; the team's target for standard, non-deep-learning ensemble methods covered in this course was 85% or better on both accuracy and F1.

Related research supports the modeling approach chosen here. Random Forest, XGBoost, and SMOTE-style resampling have all been applied to corporate bankruptcy prediction and to supply-chain delivery-risk prediction to handle class imbalance (Liang et al., 2016; Ahmed et al., 2025), and the DataCo dataset itself has been used across multiple published comparisons of different modeling approaches (Constante et al., 2019), giving the team a valid reference point for model selection and validation rather than starting from scratch.

Arthunya's background in international financial services and FinTech, including prior research on vendor and third-party risk detection frameworks, shapes how the team reads specific features which financial ratios genuinely signal risk, and where a model's output might be statistically sound but not practically true. A traditional financial or credit analyst would work through this manually: ratio analysis covering liquidity, solvency, and profitability; a red-flag checklist for late payments, litigation history, and ownership changes; and third-party credit reports such as Dun & Bradstreet's PAYDEX and SER ratings (Dun & Bradstreet, 2026). Supply-chain risk gets a similar manual treatment tracking past shipment delays and geographic risk by hand. That process is slow, error-prone, and does not scale across a large cross-border vendor portfolio, which is the specific gap CEROP is built to close.

A few working assumptions carry through the rest of the project. Even though the financial dataset covers only Taiwanese firms from 1999–2009, the underlying financial-ratio patterns are assumed to generalize to enterprises operating across other borders. The DataCo late-delivery patterns (2015–2019), while drawn from a single company's logistics network, are assumed to represent general operational-risk patterns. GSCPI, though measured at a global macro level, is assumed to correlate meaningfully with company-level operational risk an enhancement that applies only to the operations layer, since the Taiwanese Bankruptcy dataset has no date field to merge against a monthly macro index. None of these assumptions are treated as settled; Week 3's correlation analysis and descriptive comparisons across the two datasets test them directly, and any assumption that fails verification (for example, no measurable GSCPI correlation) is documented and the modeling approach adjusted rather than silently dropped. The course also does not require real-time deployment; a batch-trained, periodically retrained model is an acceptable stand-in for a real-time monitoring system.

Proposal

Selected Problem Statement

A firm dealing with foreign suppliers may make a decision without a full picture of the risk it is taking on. Financial health is typically assessed after the fact using accounting information, while delivery performance is tracked separately through shipping and logistics data. Evaluating the two in isolation makes it harder to notice when a partner's financial problems are linked to late deliveries or other operational trouble. Commercial resources such as Dun & Bradstreet's Supplier Evaluation Risk Rating and Moody's supplier risk scorecards confirm this is a legitimate business need, but they typically examine financial or operational data separately rather than producing one consolidated view.

Significance and Industrial Impact

Enterprises need something that fills this gap for the sake of sustainability and supply-chain resilience. A single node failing in a cross-border supply chain a supplier in real financial distress combined with volatile shipment patterns can produce disproportionate downstream consequences, a dynamic the Federal Reserve Bank of New York tracks through its Global Supply Chain Pressure Index (Global SCPI, n.d.). Academic work continues on both sides of this problem independently. CEROP's contribution is combining macroeconomic pressure signals with firm-specific finance and operations metrics, moving the assessment process from after-the-fact reporting toward earlier, evidence-based risk identification.

AI Smart Intervention Planning

CEROP is architected as a dual-layer supervised classification system, complemented by a macroeconomic signal layer, converging on a single decision-support output.

Layer 1 Financial Distress Classification. Built on the Taiwanese Bankruptcy Prediction dataset from the UCI Machine Learning Repository: 6,819 firms and 95 financial-ratio features. Ensemble classifiers such as Random Forest and gradient-boosted methods (XGBoost) are well suited to this dataset's severe class imbalance (about 3.23% positive class), based on their performance in prior research on the same data. The final model is determined during Week 5's model-comparison activity rather than fixed in advance.

Layer 2 Supply Chain Operations Risk Classification. Built on the DataCo Smart Supply Chain dataset (180,519 records, 53 features, with a built-in delivery/logistics-delay-risk label; Ahmed et al., 2025), predicting the probability of delivery or logistics delay. Comparable published work reports roughly 90% accuracy for standard ensemble methods and over 96% for

hybrid deep-learning methods (Liang et al., 2016; Ahmed et al., 2025), setting a reasonable performance benchmark.

Data Enrichment Layer Macroeconomic Pressure. Both classification layers are supplemented by the Global Supply Chain Pressure Index (n.d.), measured monthly by the Federal Reserve Bank of New York. Prior frameworks combining this index with operational data have shown the macro-feature carries meaningful signal for delivery-risk prediction.

The output of both layers is consolidated into a global risk report giving risk and procurement teams a combined financial and operational profile for a vendor before extending credit, renewing a contract, or approving a new relationship replacing the current manual, siloed review process.

Getting and Exploring the Data

Financial Distress Layer Taiwan Bankruptcy Dataset

The Taiwanese Bankruptcy Prediction Dataset was loaded via the UCI Machine Learning Repository, confirming the expected structure of 6,819 firms across 96 columns (95 financial-ratio features plus the binary target, Bankrupt?). An initial completeness check returned zero missing values across all columns.

The target variable confirmed a heavily imbalanced class distribution: 96.77% non-bankrupt (6,599 firms) versus 3.23% bankrupt (220 firms), consistent with the figure cited from Liang et al. (2016) in the Week 2 submission. This confirmed the need for the precision, recall, F1-score, and AUC-ROC evaluation approach already committed to in the Frame the Problem document, rather than relying on raw accuracy.

Correlation analysis identified Debt ratio % (r = 0.250), Current Liability to Assets (r = 0.194), and Borrowing dependency (r = 0.177) as the features most strongly associated with

bankruptcy, alongside Net Income to Total Assets (r = -0.315) and ROA(A) before interest and % after tax (r = -0.283) as the strongest negative predictors. Two features, Current Liabilities/Equity and Current Liability to Equity, returned identical correlation values (r = 0.154), flagging likely multicollinearity for Week 4.

Histograms of the six features most strongly correlated with bankruptcy were generated to visually assess separability between bankrupt and non-bankrupt firms, and a correlation heatmap of the eleven most relevant features visualized relationships among the strongest predictors and the target.

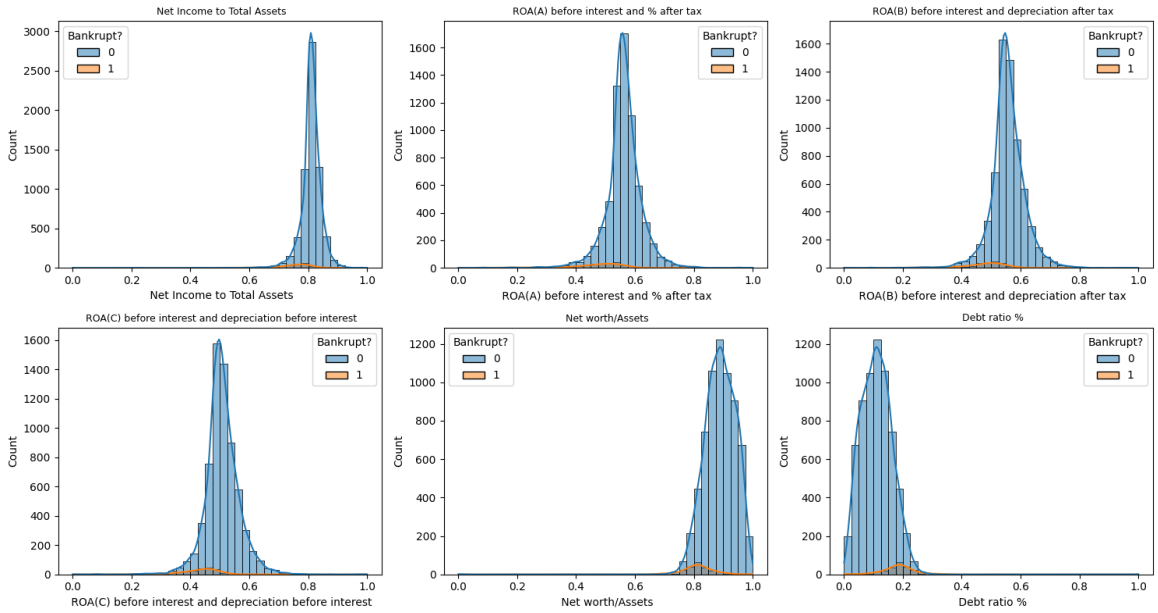


Figure 1. Distribution of top correlated financial ratios by bankruptcy status (0 = non-bankrupt, 1 = bankrupt).

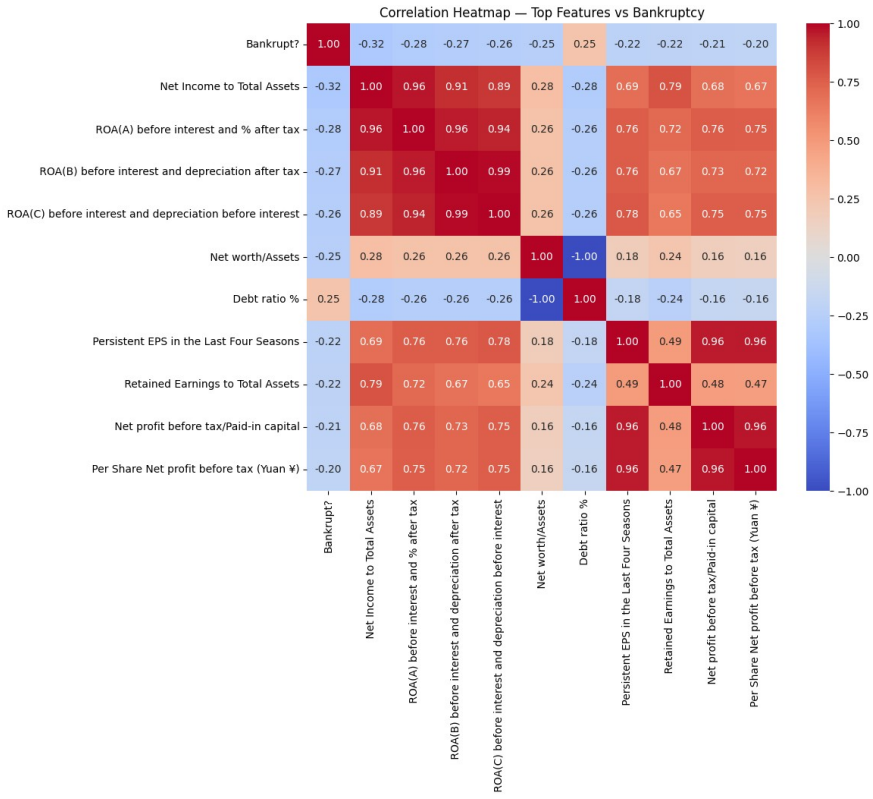


Figure 2. Correlation heatmap of the top 11 features most associated with bankruptcy.

A more detailed outlier check was run after the six top-correlated features turned out to be cleanly bounded between 0 and 1 a result that prompted verification across the full feature set rather than assuming the pattern held throughout. Extending the range check to all 95 numeric features found that 71 fell within a normalized [0, 1] range, while 24 did not. Those 24 non-normalized features showed maximum values in the billions despite a minimum of 0, a pattern inconsistent with realistic financial ratios and indicative of sentinel or placeholder values standing in for true missing data which also explains the apparent absence of missing values noted earlier.

Row-level analysis separated these 24 features into three tiers based on the percentage of records affected:

Tier	# Features	Description
Minor (< 2% affected)	15	Isolated data-entry errors; correctable by capping at

Tier	# Features	Description
		the 99th percentile (e.g., Current Ratio 0.01%, Quick Ratio 0.13%, Cash/Current Liability 0.67%)
Borderline (~3% affected)	1	Interest-bearing debt interest rate (3.24% of records affected); flagged for closer review before deciding to cap or exclude
Severe (18–88% affected)	8	Majority of records affected; too little reliable data remains to justify imputation

Note. Sentinel Value Treatment Tiers.

The eight severely affected features are Total Asset Growth Rate (88.2% of records), Cash Turnover Rate (62.2%), Research and Development Expense Rate (59.1%), Inventory Turnover Rate (43.2%), Quick Asset Turnover Rate (34.9%), Operating Expense Rate (33.5%), Current Asset Turnover Rate (18.1%), and Fixed Assets Turnover Frequency (18.0%). None of these eight features appear among the top correlated predictors identified above, so this data-quality issue does not compromise the features prioritized for modeling.

Based on this analysis, the financial-distress layer needed a tiered cleaning strategy rather than one uniform approach: the 71 already-normalized features and the 15 minor-outlier features would be capped at the 99th percentile where needed and scaled normally; the borderline feature (Interest-bearing debt interest rate) would be reviewed individually before a capping or exclusion decision; and the 8 severely affected features would be flagged as candidates for exclusion from the final modeling feature set, since imputing values where most of the data is unreliable would add more noise than signal.

Operations Layer DataCo Smart Supply Chain Dataset

The DataCo Smart Supply Chain dataset (Constante et al., 2019) was loaded via Kaggle, confirming 180,519 order-line records across 53 raw columns. Seven columns containing personally identifiable or non-analytical information (customer names, email, password, street address, product image, and description fields) were removed as an initial structural pass, bringing the working dataset to 37 columns ahead of the deeper cleaning planned for Week 4.

A completeness check across the retained columns found the dataset largely complete, with missing values concentrated in a small number of fields notably Order Zipcode and Product Description flagged for handling during Week 4 preprocessing rather than at the exploration stage.

Late_delivery_risk, the operations-layer target, showed a moderate class split: roughly 55% of orders flagged late versus about 45% delivered on time. This near-even balance stands in sharp contrast to the severe 3.23%/96.77% imbalance in the Financial layer, meaning the two layers ultimately need different evaluation strategies Operations can lean more on standard accuracy-adjacent metrics, while Financial cannot.

An IQR-based outlier scan across the numeric columns found the largest concentration of extreme values in Benefit per order (18,942 outliers, 10.49% of records) and Sales per customer (1,943 outliers, 1.08% of records). Unlike the Financial layer's sentinel-value problem placeholder values in the billions masquerading as clean data these outliers reflect genuine business variance, such as orders with unusually large losses or unusually large per-customer sales, and were flagged for IQR-based clipping rather than removal in Week 4.

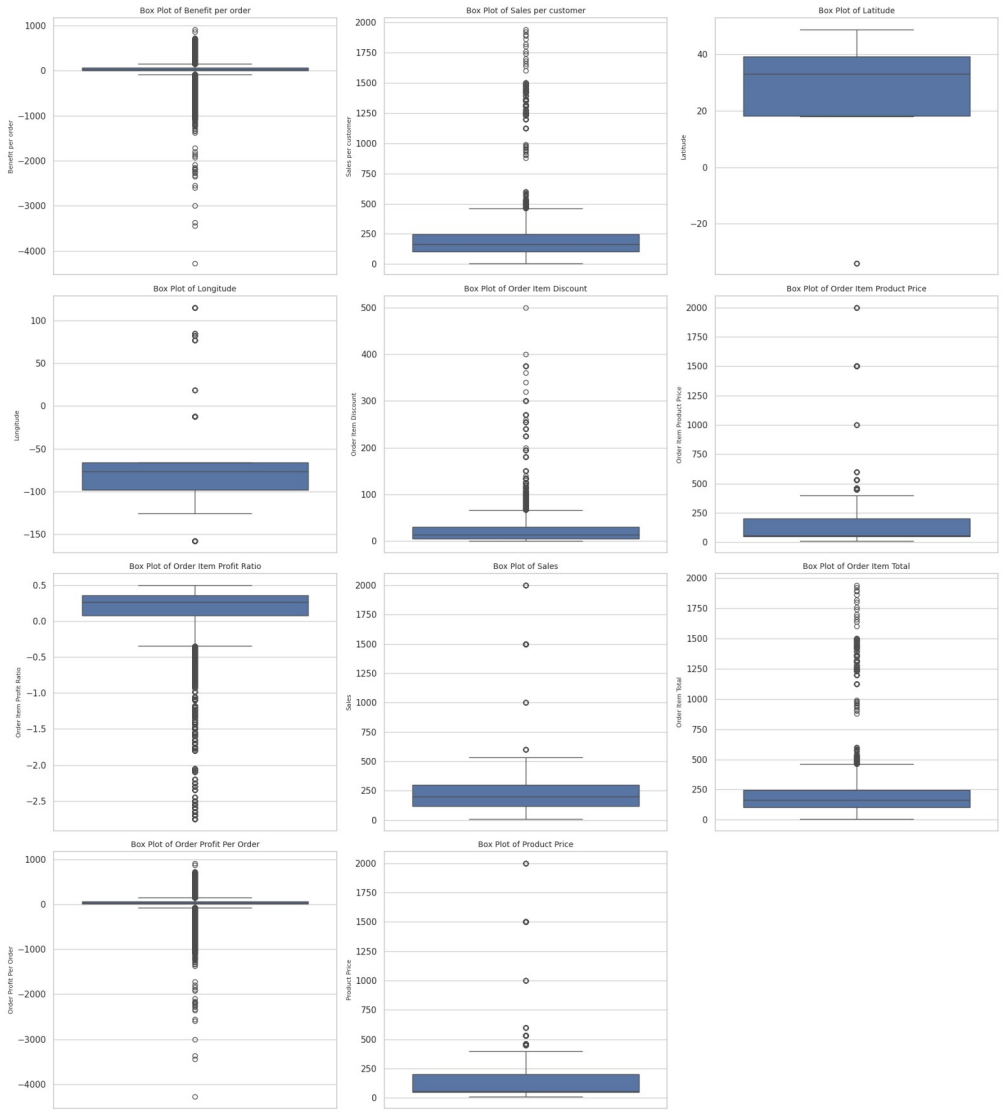


Figure 3. Box plots of key numeric operations features showing the spread of values before outlier treatment.

Histograms of key numeric features (Benefit per order, Sales per customer, Days for shipping (real), Order Item Discount, Order Item Product Price, and Order Item Quantity) showed right-skewed distributions typical of transactional order data, with most orders clustered at low values and a long tail of larger orders.

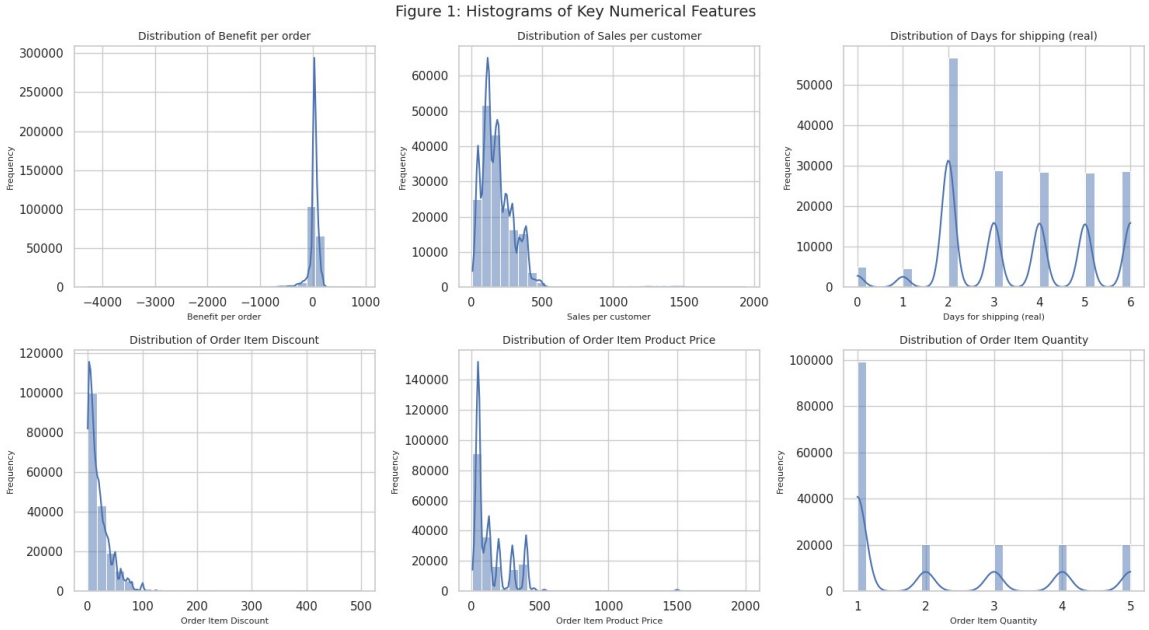


Figure 4. Distribution of six key numeric operations features.

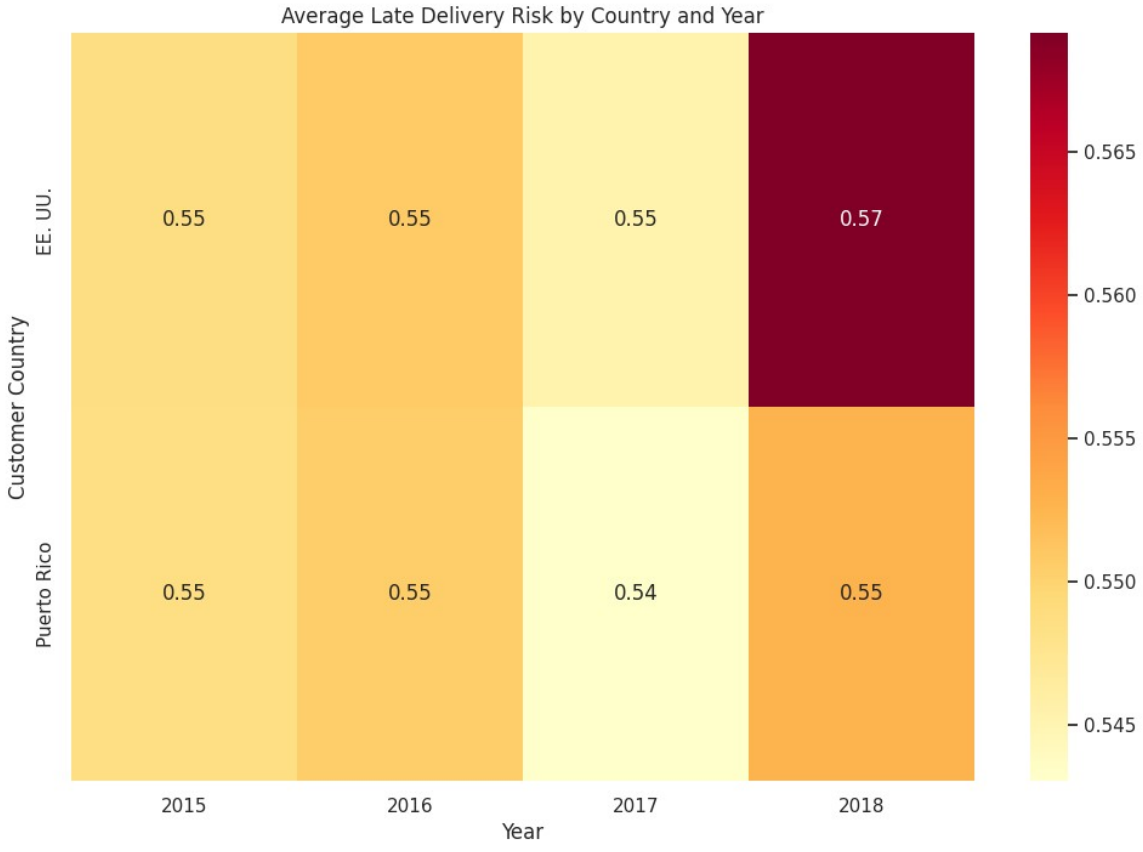
A correlation heatmap against Late_delivery_risk identified Days for shipping (real) (r = 0.40) and Days for shipment (scheduled) (r = -0.37) as by far the strongest linear associations with late delivery; every other numeric feature correlated at or near r = 0.00. This carried a direct warning into Week 4: Days for shipping (real) records the actual shipping duration, which is only known once a delivery has already happened. Using it as a predictor would leak the outcome directly into the model, which is why this column along with the related Delivery Status field was excluded during Week 4 preprocessing. Days for shipment (scheduled), known at the time an order is placed, was retained as a legitimate predictor.

Figure 5. Correlation heatmap of the top 15 features most associated with Late_delivery_risk.

Late-delivery risk was further broken down by country and year. The highest average late-delivery rates appeared for orders shipped to the United States and Puerto Rico (both roughly 54–57%), broadly in line with the dataset's overall late-delivery rate rather than an extreme geographic outlier.

Figure 6. Average late-delivery risk by customer country and year.

A daily time-series view of total Sales, total Profit, and the daily Late_delivery_risk rate checked whether the data behaved consistently across the multi-year window it covers. Sales and profit held a stable daily pattern from January 2015 through roughly September 2017, then showed sharp, erratic spikes and drops afterward a signal that data collection became unreliable toward



the end of the observed period. The daily late-delivery rate itself stayed noisy but centered around 50–60% throughout, with no strong long-run trend.

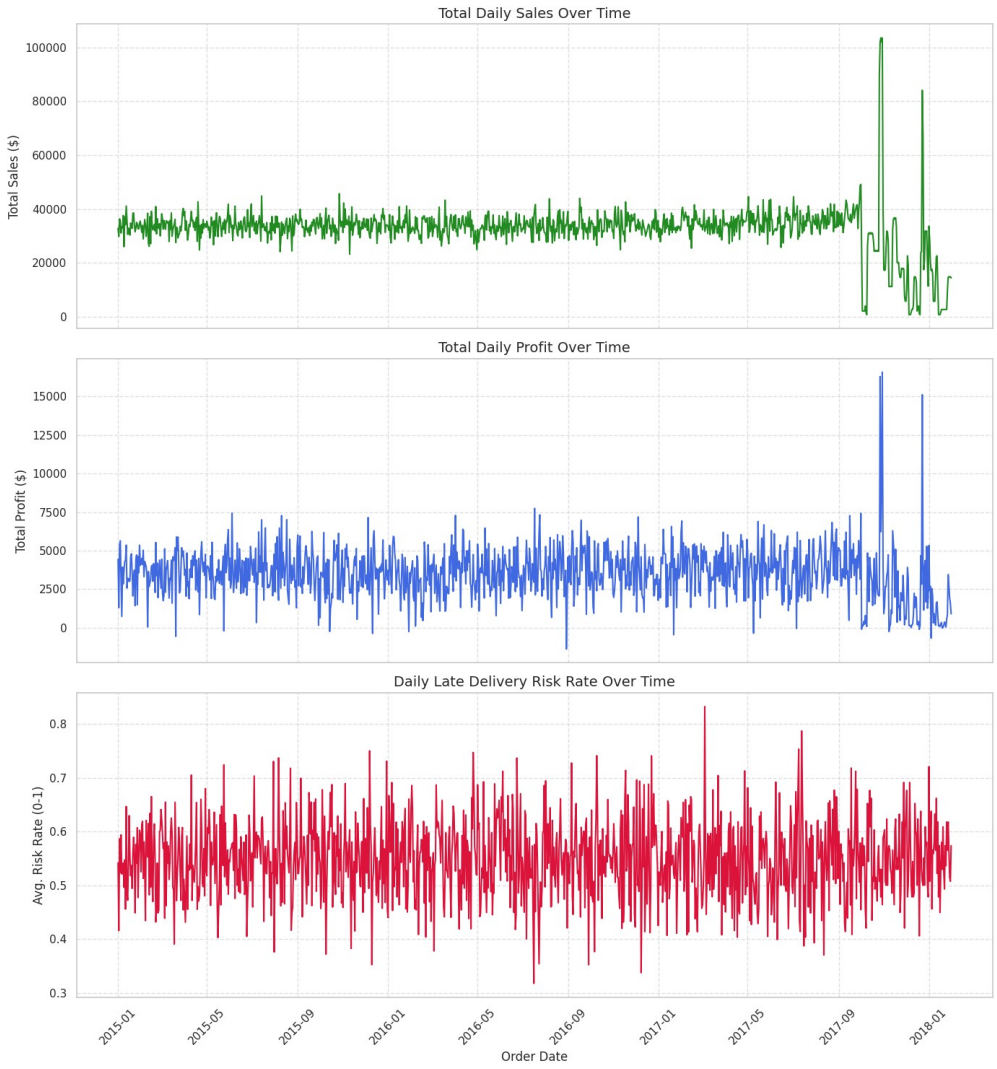


Figure 7. Daily total sales, total profit, and late-delivery risk rate over the full date range of the dataset.

The Operations layer entered data preparation with three concrete tasks flagged from EDA: exclude Days for shipping (real) and Delivery Status as target-leakage columns, since both are only known after a delivery outcome is determined; resolve the identified outliers in Benefit per order and related financial fields via IQR clipping fitted on training data only; and exclude the unreliable tail of the dataset identified in the time-series view later implemented in Week 4 as the removal of records after 2017-10-01.

Preparing Data

Financial Distress Layer Taiwan Bankruptcy Dataset

This section prepares the Taiwanese Bankruptcy dataset for modeling, building on the Week 3 data-quality findings that sorted the dataset's 95 financial-ratio features into fifteen minor-outlier features, one borderline feature, eight severely corrupted features, and one pair of multicollinear features. Each group received a tiered cleaning treatment, followed by feature engineering and a leakage-safe train/test split.

All cleaning was done on a copy of the original dataframe so the raw data stayed intact. Each transformation capping, dropping, feature engineering was written as a standalone, reusable function rather than inline code, so the same logic runs on both the training and test sets without duplication. This enforces a strict fit/apply separation: capping thresholds are learned only from the training data, then applied unchanged to the test data, so no test-set information touches the cleaning process.

Tier	# Features	Action	Rationale
Minor (< 2% affected)	15	Capped at 99th percentile	Isolated data-entry errors; capping removes corrupted

Tier	# Features	Action	Rationale
			values without losing the feature
Borderline (3.24% affected)	1 (Interest-bearing debt interest rate)	Capped, same treatment as minor tier	Behaves closer to the minor tier than the severe one; worth keeping as a predictor
Severe (18–88% affected)	8	Dropped entirely	Too little reliable data left to justify imputing it

Note. Sentinel Value Treatment Tiers and Feature Actions.

"Average Collection Days," one of the fifteen minor-outlier features, shows the capping effect clearly. Before treatment, the training set contained a sentinel value as extreme as 9.73 billion, consistent with the sentinel pattern flagged in Week 3. After the 99th-percentile cap (threshold = 0.021), the maximum dropped to 0.021, putting the feature back in a business-plausible range without discarding the record.

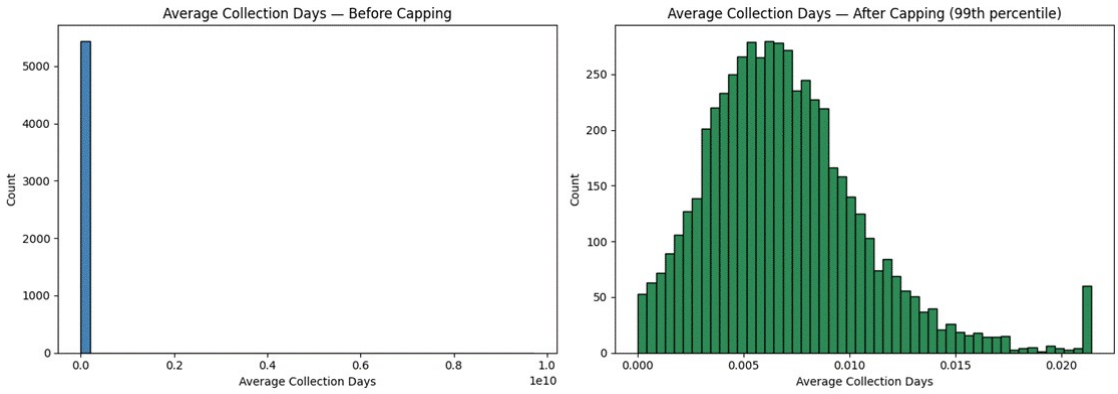


Figure 1. Distribution of "Average Collection Days" before and after 99th-percentile capping.

Week 3 found that Current Liabilities/Equity and Current Liability to Equity returned identical correlation with the target (r = 0.154). The corrected code verified both columns were identical across all 6,819 records, with a maximum absolute difference of 0.0. Current Liability to Equity was dropped; Current Liabilities/Equity stays as the representative feature.

Two new features were built from existing ratios to capture relationships no single ratio covers on its own:

Feature	Formula	Business Rationale
Profitability_Composite	Mean of ROA(A), ROA(B), ROA(C)	Averages three ROA variants into one signal, cutting noise from any single calculation method
Compounding_Leverage_Risk	Current Liability to Assets × Borrowing dependency	Flags firms carrying heavy short-term liabilities while also leaning on borrowing; neither factor alone looks alarming, but together they compound distress risk

Note. Summary of Composite Features Built From Financial Ratios.

The dataset was split 80/20 using stratified sampling on the target variable (Bankrupt?), preserving the class imbalance (96.77%/3.23%) in both sets, with capping thresholds learned from the training set only. The cleaned dataset holds 85 features, down from 95 (−8 severely corrupted, −1 exact duplicate, −3 ROA variables, +2 engineered). Training set: 5,455 records. Test set: 1,364 records consistent with an 80/20 split of the original 6,819 firms. A post-cleaning check found zero missing values in either set.

Operations Layer DataCo Smart Supply Chain Dataset

This section documents the preprocessing pipeline built for the supply-chain risk component of CEROP, using the DataCo Smart Supply Chain dataset to predict Late_delivery_risk. The report follows the standard data-preparation checklist: cleaning, feature selection, feature engineering, feature scaling, plus a section on unsupervised learning applied to

a small feature subset. From Week 3's EDA, the dataset was known to contain duplicate columns, high-cardinality features like order or customer ID, and other issues requiring attention.

All data transformations were built as reusable functions using scikit-learn's preprocessors, encoders, and data splits: the data was split into training and test first, followed by one-hot encoding of categorical variables, then standard scaling of numeric columns. The goal was to clean the data, ensure no missing values remained, remove or replace outliers, drop unneeded columns, and select features based on domain knowledge while avoiding multicollinearity.

A separate feature-scaling and preprocessing step supported unsupervised clustering (DBSCAN) on a subset of features, to check for underlying patterns or groupings that could feed downstream layers.

Data cleaning covered five actions: filling missing values; identifying and dropping duplicate rows; checking for and removing redundant columns; eliminating high-cardinality features such as PII and IDs that would not contribute meaningfully to modeling; and using the Winsor (IQR) method to eliminate outliers and replace them with boundary values. Missing values were checked directly via isna().sum() after IDs and redundant columns were removed. Columns like Order Country and Customer City, originally in Spanish, were translated to English for consistency and possible joins with the financial layer.

Several columns were dropped to prevent data leakage, remove redundancy, and manage cardinality. Leakage columns Delivery Status, Days for shipping (real), and shipping date (DateOrders) were removed since they describe the delivery outcome and would leak the target. PII and free-text columns (Customer Email, Customer First/Last Name, Customer Password, Customer Street, Product Image, Product Description) were dropped. Redundant ID columns (Order Customer Id, Order Item Cardprod Id, Product Category Id) duplicated Customer Id,

Product Card Id, and Category Id and were removed, along with Latitude and Longitude as redundant geographic information. High-cardinality columns (Order Id, Order Item Id, Customer Zipcode, Order Zipcode, Customer City, Order City, Order State, Product Name, Customer Id) were also dropped.

Order Item Total and Order Item Product Price were removed for their high correlation with Sales and Product Price. Order Profit Per Order was dropped as perfectly correlated ($r = 1.0$) with Benefit per order, confirmed by identical outlier bounds and clipped-value counts during outlier treatment; Sales per customer was dropped as a near-duplicate of Sales ($r \approx 0.99$). Records after 2017-10-01 were removed (about 8,557 rows) because the data showed sudden spikes in order counts and sales after that date, consistent with the unreliable tail identified in Week 3's time-series view.

The main outlier handling applied the IQR (Winsor) method with a factor of 1.5 to Benefit per order and Order Item Profit Ratio, clipping values to calculated upper and lower bounds fit on the training set.

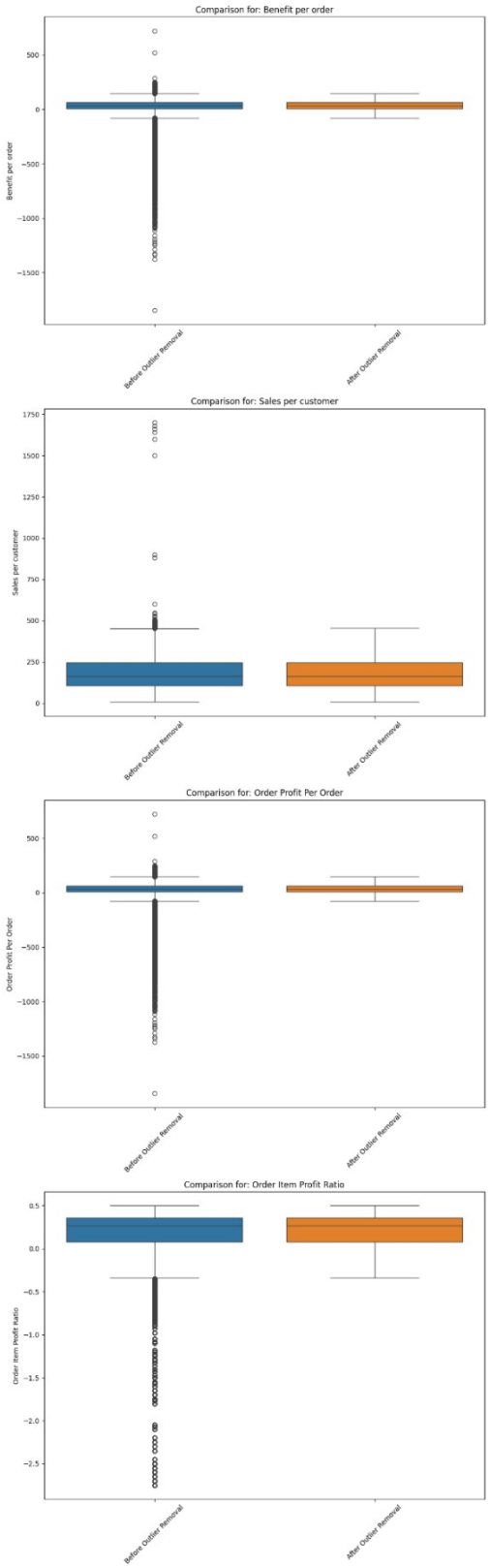


Figure 2. Box plots showing before and after outlier removal.

Feature selection combined domain knowledge with correlation-based analysis to keep only features contributing to prediction of the target. Target-leakage columns (Delivery Status, Days for shipping (real), shipping date), PII/non-tabular columns, and duplicate ID columns were excluded as discussed above. Zipcode columns were dropped since other geo identifiers (City/State) were retained. Order Status was kept as it describes order and payment status known before delivery. The final selected columns before feature engineering were: Type, Days for shipment (scheduled), Benefit per order, Category Name, Customer Country, Customer Segment, Customer State, Department Name, Market, Order Country, Order Item Discount, Order Item Discount Rate, Order Item Profit Ratio, Order Item Quantity, Sales, Order Region, Order Status, Product Price, and Shipping Mode.

Feature engineering decomposed the order date into year, month, day of week, and hour, then dropped the raw datetime column; Year was treated as a weaker candidate given the dataset spans only three years. Scikit-learn's TargetEncoder was applied to high-cardinality categorical columns (City, State, Country, Category Name, Department Name, Product Name), with an 8-or-fewer-unique-values threshold separating one-hot encoding from target encoding, converting categorical variables to numeric values without an explosion of new columns. After feature engineering, the working feature list was: Type, Days for shipment (scheduled), Benefit per order, Category Name, Customer Country, Customer Segment, Customer State, Department Name, Market, Order Country, Order Item Discount, Order Item Discount Rate, Order Item Profit Ratio, Order Item Quantity, Sales, Order Region, Order Status, Product Price, Shipping Mode, Order_Year, Order_Month, Order_DayOfWeek, and Order_Hour.

Since a time-series model was not being used, all numeric columns from the steps above, along with the date components, were standardized using scikit-learn's StandardScaler within the

same ColumnTransformer as the other encoders, applying each feature-processing step sequentially by column type.

A DBSCAN-based component was added to flag orders with anomalous profit and discount patterns identified during EDA. A feature subset (Benefit per order, Order Item Profit Ratio, Sales, Order Item Discount Rate) was scaled independently of the main pipeline, fit on the training split, and applied to test, since DBSCAN's distance-based clustering needs its own comparable scale separate from the full one-hot and target-encoded feature space. Orders that did not resemble any dense neighborhood were labeled as noise, an effective unsupervised signal for detecting anomalies such as potential fraud. The analysis found 3 clusters and 18 noisy transactions.

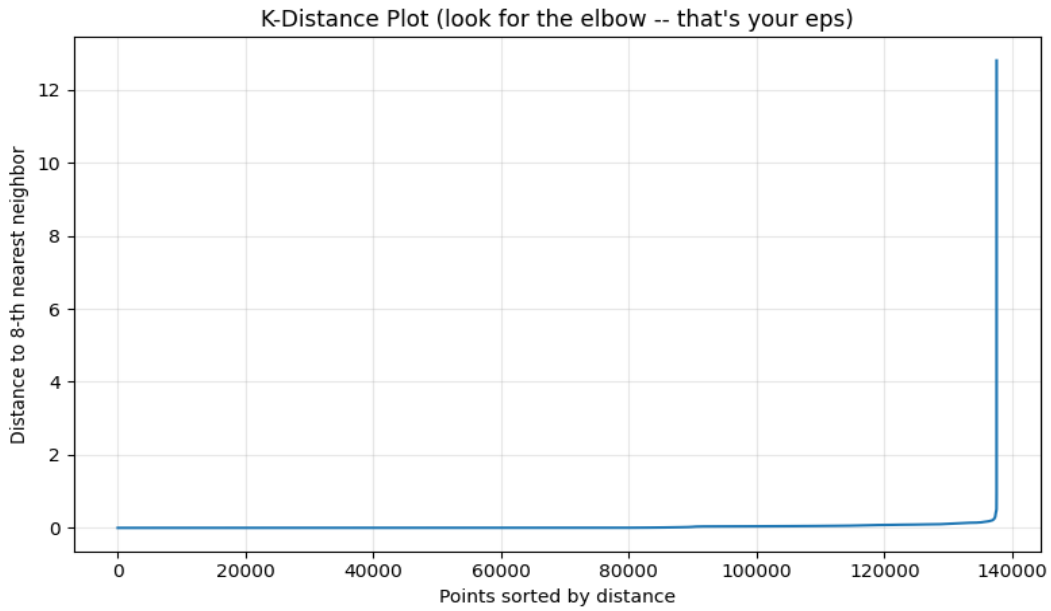


Figure 3. K-Distance plot for the DBSCAN model.

The resulting pipeline removes unreliable spikes in the dataset, excludes target-leakage and PII columns, resolves redundant identifiers, encodes categorical features by a cardinality rule, handles outliers without discarding data, and adds an unsupervised anomaly-detection component.

This connects directly to the Financial layer and feeds the overall ML modeling supporting CEROP's business use case.

Pipeline Integration

This section covers the Week 4 integration step: taking the two cleaned layers produced independently this week and running them through a single preprocessing pipeline that produces model-ready matrices for Week 5. The financial layer comes from the Taiwanese Bankruptcy dataset, with Bankrupt? as the target; the operations layer comes from the DataCo Smart Supply Chain dataset, with Late_delivery_risk as the target and the GSCPI series merged in only on the operations side. Integration work centered on three things: making sure both layers split their data before anything gets fitted, making sure the pipeline's numbers matched the numbers in the two layer notebooks, and saving the results so Week 5 could start from files instead of re-running everything.

The financial cleaning steps were taken directly from the financial-layer notebook. For the operations layer, the same column removals, feature engineering, outlier treatment, and encoding decisions from the operations notebook were followed in the combined pipeline. Since the two notebooks were built separately, integration was checked by comparing row counts, train/test split sizes, and final feature counts against the original results the numbers matched, confirming both layers were brought together correctly.

Both datasets are split 80/20 with stratified sampling on their target before any data-dependent step runs. Everything that learns a parameter from data the 99th-percentile cap thresholds on the financial side, the IQR bounds on the operations side, StandardScaler on both, the OneHotEncoder category vocabulary, and the TargetEncoder mappings is fitted on the training split only and applied to the test split without re-fitting. One operation deliberately runs before the

split: the November–December 2017 tail of the DataCo data was removed from the full dataset first, because it is a data-quality filter rather than something learned from the data. TargetEncoder deserves a specific note since it is the one transformer that touches the labels; it is fitted with cv=5 cross-fitting and automatic smoothing, so no row contributes to its own encoded value, and categories that appear only a handful of times get pulled toward the global mean instead of memorizing a rate from a few orders.

The financial layer ran exactly as specified in the layer notebook: fifteen minor-outlier features plus the borderline Interest-bearing debt interest rate were capped at the 99th percentile using training-derived thresholds, eight severe-outlier features were dropped, Current Liability to Equity was dropped after verifying it as an exact duplicate, the three ROA variables were replaced by Profitability_Composite, and Compounding_Leverage_Risk was added. The 6,819 firms split into 5,455 training and 1,364 test records, preserving the 96.77%/3.23% class balance in both. After cleaning, the financial dataset had 85 features.

The operations layer left out the November–December 2017 records identified as unreliable, taking the dataset from 180,519 rows to 171,962. From there, 137,569 records were used for training and 34,393 for testing, with stratified sampling holding the class balance at 54.82% late against 45.18% on time. Structural cleaning dropped the three leakage columns, the PII/free-text columns, the confirmed duplicate ID pairs, the high-cardinality identifiers and geographic name columns, and the redundant price/profit columns Order Item Total, Order Item Product Price, Order Profit Per Order ($r = 1.0$ duplicate of Benefit per order), and Sales per customer ($r \approx 0.99$ near-duplicate of Sales). The order date was decomposed into year, month, day of week, and hour, plus a month-start key for the GSCPI join, and the raw timestamp was dropped.

Outliers in Benefit per order and Order Item Profit Ratio were clipped at 1.5× IQR, with bounds computed on the training split alone and then applied to both splits a deliberate difference from the layer notebook (which clipped the training split only), made to keep both matrices on a consistent scale without leaking test-set information.

Categorical columns were handled by what they represent rather than only their cardinality. Transaction-related columns Type, Customer Country, Customer Segment, Order Status, Shipping Mode were one-hot encoded with drop="first"; Shipping Mode mattered most here, since Week 3 found it was the strongest driver of late delivery. Columns describing geography or product taxonomy (Category Name, Department Name, Market, Order Country, Order Region) were target-encoded, since the useful summary is a category's historical late-delivery rate rather than dozens of indicator columns. This produced 35 features: 13 numeric (including GSCPI), 17 one-hot columns, and 5 target-encoded columns differing from the layer notebook's reported 37 due to the Market and Customer Country encoder reassignments described below. Customer State was left out of the final model since the same location information was already available through country and region fields.

Encoder	Columns	Output columns
StandardScaler	Days for shipment (scheduled), Benefit per order, Order Item Discount, Order Item Discount Rate, Order Item Profit Ratio, Order Item Quantity, Sales, Product Price, Order_Year, Order_Month, Order_DayOfWeek, Order_Hour, GSCPI	13
OneHotEncoder(drop="first"	Type, Customer Country, Customer Segment, Order	17

Encoder	Columns	Output columns
)	Status, Shipping Mode	
TargetEncoder	Category Name, Department Name, Market, Order Country, Order Region	5
Dropped	Customer State	0

Note. Column groups and encoder assignment.

The integration deliberately does not include DBSCAN_Cluster or Is_Anomalous_Profit. These were created through a separate unsupervised anomaly-analysis step in the operations notebook, but the original fitted DBSCAN helper was not available as a reusable component in the integration workflow; rather than recreating clusters that might not match the source notebook, these features were left out and documented for possible reintegration before model comparison. The 3 clusters and 18 noise points reported in the operations notebook remain valid as a standalone unsupervised analysis.

The GSCPI file from the Federal Reserve Bank of New York contains 342 monthly records from January 1998 through June 2026. The date field was converted into a monthly key and matched with Order_YearMonth in the DataCo dataset. The merge succeeded with no missing GSCPI values in either split, and was completed after the train-test split still safe, since GSCPI is an external published series rather than a value calculated from the project data. GSCPI was added only to the operations layer, since the Taiwanese Bankruptcy dataset has no reporting date to match against. Week 3's analysis had already shown GSCPI's relationship with late-delivery risk was weak (monthly correlation of about -0.06 across 37 months, with late-delivery rate staying within a fairly narrow 51.9%–56.8% range); GSCPI was kept as a candidate feature regardless, so its

value could be tested during Week 5 modeling rather than removed on the initial correlation result alone.

The financial dataset had no missing values before or after cleaning. In the operations dataset, the main missing-data issues came from Product Description (completely empty) and Order Zipcode (missing in about 86% of records); both were removed during structural cleaning. After preprocessing, all four training and test matrices were checked and confirmed to contain no missing or infinite values.

The Week 4 task list included a feature comparing scheduled versus actual shipping time. This was not added: actual shipping time is only known after delivery, while CEROP is meant to predict late-delivery risk at the time an order is placed. Since Late_delivery_risk itself is based on whether actual shipping time exceeds scheduled shipping time, a delivery-gap feature would closely reproduce the target and create data leakage. Days for shipping (real) was removed for this reason, while Days for shipment (scheduled) was retained since it is known at order time.

Each layer runs the same assertions before anything is written: train and test have identical column counts, neither contains nulls, and all values are finite. The pipeline then saves ten files processed X and y files for the training and test portions of both layers, plus the fitted financial scaler and operations preprocessor so Week 5 can transform any new data with the same encoder objects rather than risk silently producing a different feature space by refitting.

Layer	Rows in	Rows after cleaning	Train	Test	Features
Financial (Taiwan)	6,819	6,819	5,455	1,364	85
Operations (DataCo)	180,519	171,962	137,569	34,393	35

Note. Row and feature counts for both layers after preprocessing.

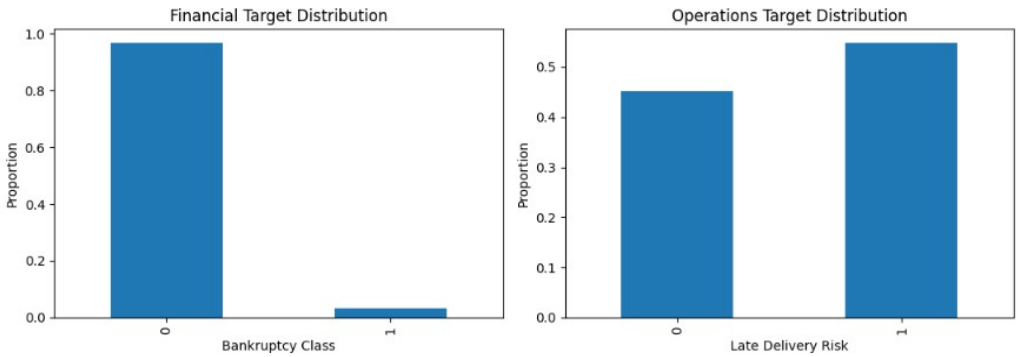


Figure 4. Target class distribution for the financial and operations training datasets.

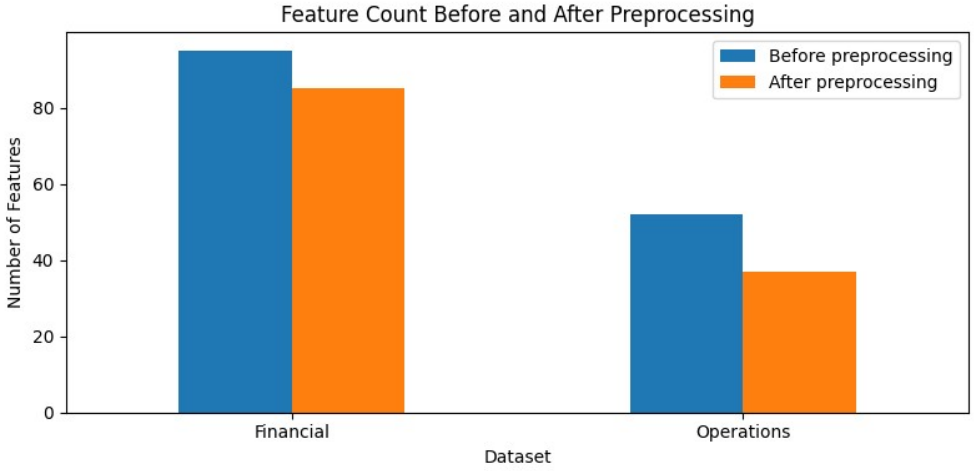


Figure 5. Feature count before and after preprocessing for both layers.

Three integration decisions differ from the individual layer notebooks, documented here for clarity:

Decision	Layer notebook	Integration	Reason
IQR clipping scope	Train only	Train and test, using train bounds	Keeps both matrices on one scale; bounds carry no test information
Market encoding	One-hot	Target encoded	Treated as a geographic rollup with the other geography columns
Customer Country encoding	Target encoded	One-hot	Only two levels, so one indicator column is simpler and readable

Note. Integration decisions that differ from the layer notebooks.

Both cleaned layers were saved and ready for modeling in Week 5. The financial layer needed extra attention given only 3.23% of records belong to the bankrupt class, so accuracy alone could be misleading and models would need evaluation on precision, recall, and F1-score alongside class weighting or resampling. GSCPI would be tested in Week 5 to see whether it adds useful information to the operations model, given its weak Week 3 correlation. Target-encoded features would also need review during feature-importance analysis so they do not overshadow more practical operational variables such as Shipping Mode.

During the Week 4 class, the instructor recommended incorporating an unsupervised-learning component into the project, such as clustering or anomaly detection. In response, the team extended the operations-risk layer with a DBSCAN-based anomaly-detection component applied to order-level financial features (Benefit per order, Order Item Profit Ratio), building on Week 3's observation that these features showed wide variance including losses on some orders that could correlate with cancellations or fraud-flagged orders. This updates the Week 2 Frame the Problem framing (item 4, originally described as purely supervised) to reflect the instructor's suggestion: the two primary classification layers remain supervised, while the new component addresses a gap those layers cannot detect anomalous transactions that do not fit either labeled risk category. Model exploration and tuning for this component continued in Weeks 5 and 6.

Model Selection and Shortlisting

Week 4 delivered two cleaned, leakage-checked datasets the financial-distress layer (Taiwan Bankruptcy Prediction) and the operations layer (DataCo Smart Supply Chain) each with its own target and its own risk profile. This week picks up where that left off: three candidate models were trained per layer, evaluated on the held-out Week 4 test split, and shortlisted using the same evaluation framework across both accuracy, precision, recall, F1-score, and ROC-AUC with the shortlist decision resting on whichever metric actually reflects the cost of a wrong prediction rather than on accuracy alone.

The two layers are kept separate below. They do not share a target, a feature set, or even a comparable class balance, so folding them into one narrative would blur two distinct decisions into one. What they do share where tree-based models beat linear ones, and where recall took priority over precision is picked back up in the Cross-Layer Synthesis at the end of Week 6.

Model Selection Financial Risk Layer

The financial layer picks up directly from the Week 4 preprocessing pipeline, using the saved train/test arrays (5,455 training rows, 1,364 test rows, 85 features) with Bankrupt? as the target. Three models were trained and compared: Logistic Regression, Random Forest, and XGBoost.

Everything else here follows from the imbalance. Only 3.23% of firms in the training set are actually bankrupt; a model predicting “not bankrupt” for every firm would score 96.8% accuracy without learning anything, which makes accuracy close to useless. All three models were trained with explicit class weighting class_weight="balanced" for Logistic Regression and Random Forest, scale_pos_weight for XGBoost with evaluation leaning on precision, recall, F1, and ROC-AUC instead.

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Logistic Regression	0.878	0.186	0.818	0.303	0.902
Random Forest	0.966	0.478	0.500	0.489	0.943
XGBoost	0.970	0.535	0.523	0.529	0.950

Note. Financial layer model comparison (test set, n = 1,364, 44 bankrupt firms). Class weighting applied to all three models.

At first glance, Logistic Regression looks like the strongest choice for catching bankruptcies, with 36 of 44 true cases (81.8% recall) against XGBoost's 23 (52.3%). That comparison is misleading, though it compares the three models at their own default 0.5 decision threshold, and a threshold is not a fixed property of a model; it can be moved. Held to the same 81.8% recall, the gap reverses: XGBoost catches the same 36 bankruptcies while raising only 82 false alarms, versus 140 for Logistic Regression 41% fewer unnecessary reviews for identical detection. Logistic Regression's apparent edge is a threshold artifact, not better separation between classes; its 18.6% precision at the default threshold means roughly five out of every six firms it flags are not actually going bankrupt.

Random Forest sits close behind XGBoost on every metric (F1 0.489 vs. 0.529, ROC-AUC 0.943 vs. 0.950), making it a solid fallback if XGBoost proves harder to tune or explain to a non-technical audience.

XGBoost is the shortlisted model for the financial layer, with Random Forest retained as a backup. The call rests on ROC-AUC and the threshold-adjusted comparison above, not on accuracy, since accuracy cannot separate a useful model from one that just guesses the majority class here. Logistic Regression is dropped from the shortlist: with 85 correlated financial ratios, a linear decision boundary cannot separate the classes cleanly, and its 18.6% precision is exactly what that failure looks like. One caveat worth stating plainly with only 44 bankrupt firms in the

test set, a single misclassified firm shifts recall by about 2.3 percentage points, so the differences between models are directionally reliable but not precise to the decimal point.

Note. Full training and evaluation code: Week5_Financial_Model_Selection.ipynb

Model Selection Operations Risk Layer

The operations layer builds on the same Week 4 outputs (137,569 training rows, 34,393 test rows, 35 features) with Late_delivery_risk as the target. Three models went into the comparison: Logistic Regression, Random Forest, and a linear-kernel SVM.

The classes here sit close to balanced 54.8% of orders in the training set are late, 45.2% are on time which makes accuracy a meaningful number on its own, and no class weighting was needed. Precision, recall, and F1 are still reported alongside it, mainly to keep the comparison consistent with the financial layer.

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Logistic Regression	0.726	0.887	0.573	0.696	0.774
Random Forest	0.752	0.873	0.640	0.739	0.855
SVM (linear)	0.725	0.887	0.570	0.694	0.774

Note. Operations layer model comparison (test set, n = 34,393, 18,854 late deliveries). Class balance close to even, so no class weighting was applied.

Random Forest wins here cleanly, with no threshold artifact to untangle it leads on every metric that matters at the same default decision point. It correctly identifies 12,068 of the 18,854 late deliveries in the test set (64.0% recall) while holding 87.3% precision, and its F1 (0.739) sits well above both linear models (0.696 and 0.694). Logistic Regression and the linear SVM land almost identically to each other, which makes sense given both fit a linear boundary over the same

35 features; neither captures the non-linear interactions between shipping mode, order timing, and regional patterns that Random Forest's tree splits pick up naturally.

A 5-fold stratified cross-validation confirmed the ordering was not an artifact of one split. Random Forest holds the best CV F1 (0.728 ± 0.002), close to the test-set F1 of 0.739, and neither linear model closes the gap under repeated resampling.

Random Forest is the shortlisted model for the operations layer. Unlike the financial layer, there is no need to weigh F1 against recall separately Random Forest wins both at once, so the statistical result and the business priority point to the same choice. Recall matters more than precision here for a specific operational reason: a missed late delivery means losing the chance to warn the customer or step in before the order shows up late, while a false positive just triggers an unnecessary review, a minor cost by comparison. One limitation carried into Week 6: Random Forest still misses about 36.0% of late deliveries at its default decision threshold, making threshold tuning the obvious next step rather than a settled question.

Note. Full training and evaluation code: AIT506_Week5_Operational_Model_Selection.ipynb

Cross-Layer Synthesis Week 5

Two patterns hold across both layers despite the datasets having almost nothing else in common. Tree-based models beat linear ones in both cases, by similar margins in direction if not in size: XGBoost outperforms Logistic Regression by 22.6 points of F1 on the financial layer (0.529 vs. 0.303), while Random Forest outperforms Logistic Regression by 4.3 points on the operations layer (0.739 vs. 0.696). The smaller gap on the operations side likely shares the same underlying cause a linear decision boundary cannot capture interactions between correlated financial ratios any better than it can capture interactions between shipping mode, order timing, and region.

Recall was prioritized over precision in both layers, but for different reasons. On the financial layer, a missed bankruptcy means an undetected failing counterparty. On the operations layer, a missed late delivery means losing the chance to warn the customer before the order shows up late. Different businesses, the same shape of asymmetry: a false negative costs more than a false positive.

Where the two layers diverge is how much work it took to reach that recall priority. The financial layer's 3.23% bankruptcy rate meant the shortlist decision required actively correcting for a threshold artifact. The operations layer's near-balanced 54.8%/45.2% split needed no such correction. Severe imbalance does not just lower the achievable recall it distorts which model looks best until the comparison is done correctly. Both shortlisted models, XGBoost for financial and Random Forest for operations, carried the same open item into Week 6: neither is deployed at its optimal decision threshold yet.

Finalizing Models

Week 5 settled a model-selection question for each half of CEROP independently: which architecture, at default settings, best separates risk from non-risk in each domain. XGBoost was shortlisted for the Financial layer (bankruptcy prediction on 85 Taiwan-firm ratios) and Random Forest for the Operations layer (late-delivery prediction on DataCo supply-chain records). Week 6 narrows to the next question for each: given the chosen architecture, what is its best configuration, and does finding it actually improve anything?

Both layers were tuned under the same protocol a validation split carved out of training data, a hyperparameter search scored by a metric matched to each layer's imbalance, threshold calibration on validation only, and one sealed evaluation on the test set, so that “before” and “after”

are comparable within each layer. The two layers gave different answers, which is the subject of the Cross-Layer Synthesis closing this section.

Note. Full Finalizing Models code Week6_Financial_Model_Finalization.ipynb AIT506_Week6_Operations_Model_Tuning.ipynb

Financial Risk Layer

A before/after comparison is only meaningful if both sides are measured identically. Two design decisions enforced this. First, a stratified 20% validation split was carved out of the training rows: 4,364 rows for fitting (141 bankrupt) and 1,091 for selection (35 bankrupt). Every subsequent decision the hyperparameter search, the Random Forest re-tune, the threshold calibration, and the ensemble comparison used only these two partitions; the test set was not read until final evaluation. Second, the Week 5 configuration was refit on the identical 4,364-row split to serve as the baseline, since comparing the published Week 5 model (trained on all 5,455 rows) directly against a tuned model trained on 4,364 rows would confound the effect of tuning with the effect of training-set size.

Split	Rows	Bankrupt	Rate	Used for
Tuning fit	4,364	141	0.0323	Cross-validated hyperparameter search
Validation	1,091	35	0.0321	Model choice and threshold calibration
Test (sealed)	1,364	44	0.0323	One final evaluation

Note. Data partition and the role of each split.

XGBoost was tuned with RandomizedSearchCV over 40 sampled configurations, scored by 5-fold stratified cross-validation (200 model fits total), using average_precision rather than accuracy as the scoring function given the imbalance. The search space held 15,360 combinations, so the 40 draws covered 0.26% of the grid randomized search was chosen over exhaustive grid search because GridSearchCV over the same space at 5-fold would have required 76,800 fits.

Random Forest was re-tuned under an identical budget (40 draws, 5-fold, same scorer, same split) so a tuned model outperforming an untuned rival would demonstrate something real about the underlying ranking, not just about tuning effort.

Two operating points were selected on the validation split for threshold calibration: the F1-optimal threshold, balancing missed bankruptcies against false alarms without importing an external cost assumption, and the most precise threshold achieving at least 80% recall, matching the recall band Week 5 used as a reference. Both were fixed before the test set was opened.

The search converged on a configuration scoring 0.4257 cross-validated PR-AUC, against 0.3895 for the Week 5 baseline scored under the identical cross-validation object an improvement of +0.0362 by the conventional reading. That reading turned out to be too generous: every configuration's cross-validation error bar overlapped the best score, the visual signature of a search whose ranking is not statistically identified.

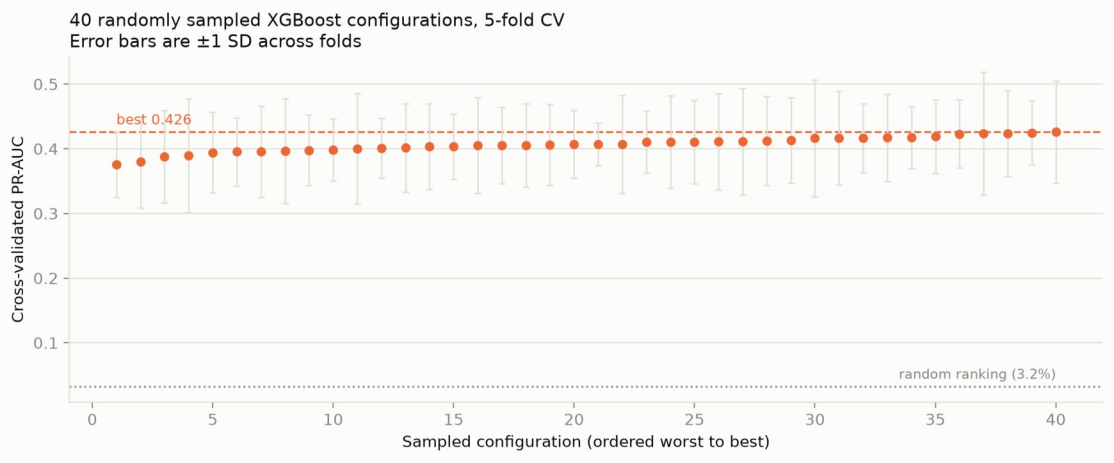


Figure 1. Cross-validated PR-AUC across 40 sampled XGBoost configurations, ordered worst to best with ±1 SD error bars across the 5 folds.

Model	Selected configuration	CV PR-AUC
XGBoost (tuned)	colsample_bytree=0.8, learning_rate=0.05, max_depth=8, min_child_weight=1, n_estimators=100,	0.4257 (SD 0.0791)

Model	Selected configuration	CV PR-AUC
Random Forest (tuned)	scale_pos_weight=10, subsample=0.9 class_weight=balanced, max_depth=None, max_features=sqrt, min_samples_leaf=8, min_samples_split=10, n_estimators=800	0.3994
XGBoost (Week 5 baseline)	colsample_bytree=0.8, learning_rate=0.1, max_depth=4, min_child_weight=1, n_estimators=300, scale_pos_weight=29.95, subsample=0.8	0.3895 (SD 0.0590)

Note. Selected hyperparameters and cross-validated scores.

The F1-optimal threshold on the tuned model came out to 0.335, well below the 0.5 default the expected direction under class imbalance, since the default is calibrated for balanced data and is systematically too conservative when positives are rare.

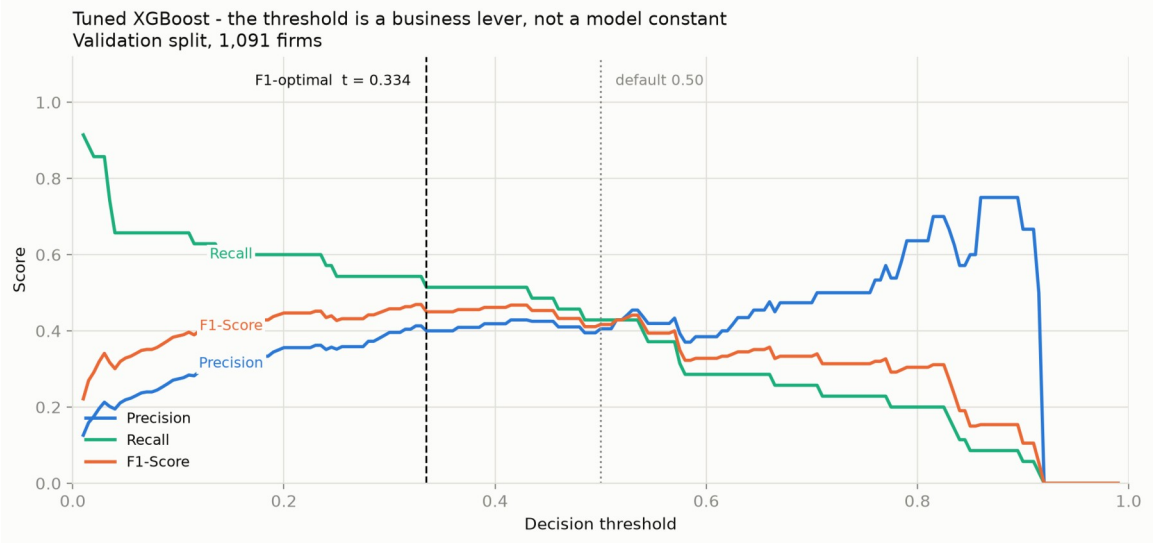


Figure 2. Precision, recall, and F1 across decision thresholds

(tuned XGBoost, 1,091-firm validation split).

With all hyperparameters and thresholds locked, the test set was opened once. Tuning reduced F1 from 0.5682 to 0.5055 (-0.0627) and PR-AUC from 0.4902 to 0.4250 (-0.0652).

Threshold calibration recovered part of the recall (0.523 to 0.591, catching 26 of 44 bankruptcies rather than 23) but at the cost of false alarms rising from 24 to 42, so F1 fell further to 0.4643. The aggressive recall-targeted threshold caught 35 of 44 at the cost of 124 false alarms.

Model	Thr.	Prec.	Recall	F1	ROC-AUC	PR-AUC	Caught
XGB Week 5 published	0.500	0.535	0.523	0.529	0.950	0.476	23/44
XGB Week 5 config	0.500	0.568	0.568	0.568	0.947	0.490	25/44
XGB tuned	0.500	0.489	0.523	0.505	0.946	0.425	23/44
XGB tuned @ calibrated	0.335	0.382	0.591	0.464	0.946	0.425	26/44
XGB tuned @ recall≥80%	0.030	0.220	0.795	0.345	0.946	0.425	35/44
RF tuned	0.500	0.306	0.750	0.434	0.944	0.461	33/44
RF tuned @ calibrated	0.791	0.429	0.477	0.452	0.944	0.461	21/44
RF tuned @ recall≥80%	0.331	0.233	0.795	0.361	0.944	0.461	35/44
Stacking (XGB+RF)	0.500	0.229	0.795	0.355	0.943	0.454	35/44
Stacking (XGB+RF) @ calibrated	0.949	0.415	0.500	0.454	0.943	0.454	22/44
Stacking (XGB+RF) @ recall≥80%	0.518	0.235	0.795	0.363	0.943	0.454	35/44

Note. Test-set performance before and after tuning. Accuracy is omitted since the majority-class baseline already scores 96.8%.

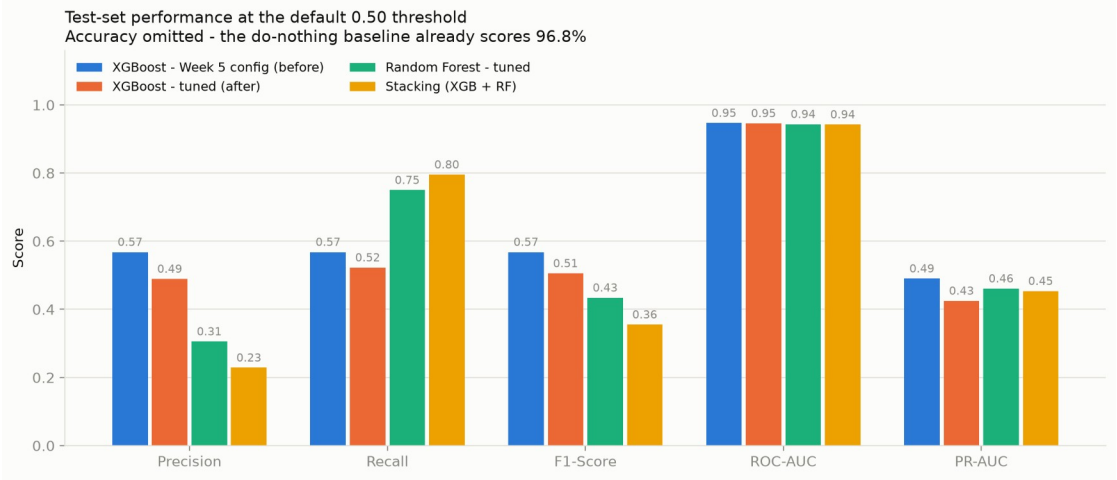


Figure 3. Test-set metrics by model at the default threshold.

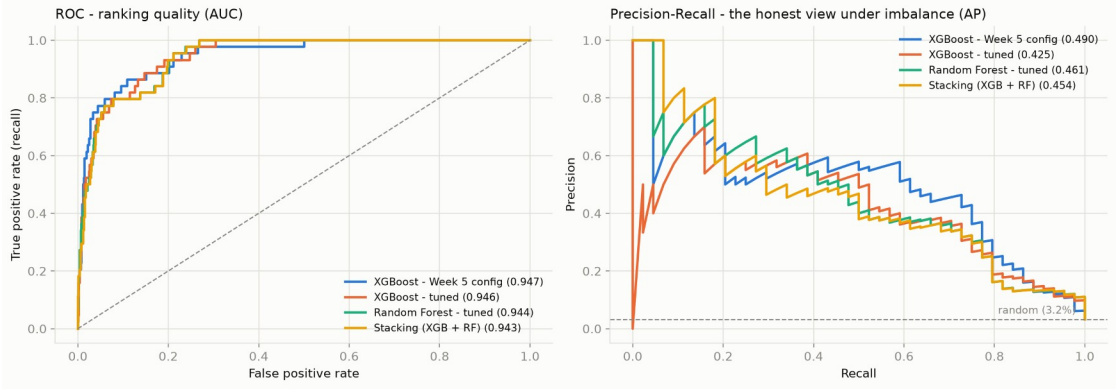


Figure 4. ROC and precision-recall curves on the test set.

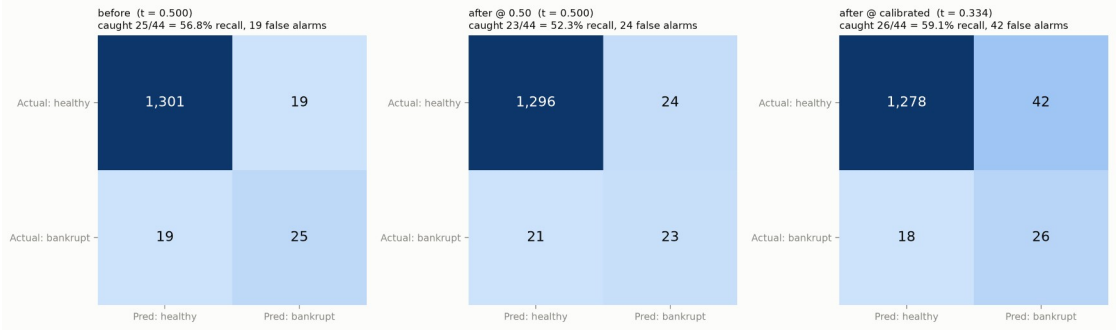


Figure 5. Confusion matrices before tuning, after tuning, and after calibration.

Random Forest did not overtake XGBoost on either surface available for selection: cross-validated PR-AUC of 0.3994 against 0.4257, and validation PR-AUC of 0.3794 for the baseline against 0.3780 for the tuned model. On the sealed test set, tuned Random Forest edges tuned

XGBoost on PR-AUC (0.4610 against 0.4250) and catches 33 of 44 bankruptcies against 23, though at 75 false alarms against 24 a reversal reported because it happened, not because it changes the recommendation, since it appears only on the surface with the smallest positive count and widest error bars, not on either surfFace the protocol permits for selection. An optional stacking ensemble (tuned XGBoost + tuned Random Forest with a Logistic Regression meta-learner) did not beat the tuned single model on validation PR-AUC and was not adopted; on the test set it scored 0.4536, between the two base learners a valid negative finding, reported as one.

A negative result is only useful if its cause is identified, since the cause determines the remedy. The hypothesis that the search simply failed to find anything better is rejected: it located a configuration scoring +0.0362 PR-AUC above baseline in cross-validation, so whatever went wrong happened between cross-validation and deployment, not during the search. The hypothesis that the search's ranking signal is smaller than its own measurement noise holds decisively the spread between the best and worst of the 40 configurations is 0.0498 PR-AUC, while the mean fold-to-fold standard deviation within a single configuration is 0.0649, a signal-to-noise ratio of 0.77, with all 40 configurations lying within one standard deviation of the winner. The mechanism is arithmetic: 141 bankrupt firms divided across 5 folds leaves roughly 28 positives per fold, and average precision estimated on 28 positives is inherently high-variance, so taking the maximum over 40 such estimates largely selects whichever configuration drew the most favorable fold assignment the winner's curse, and exactly the pattern observed.

A third hypothesis, that the specified grid could not reach the baseline, also holds: the grid caps scale_pos_weight at 20 while the empirical class ratio on the tuning split is 30.0, so no number of draws could have recovered the Week 5 configuration since it sits outside the space being searched. A diagnostic rerun with that axis widened confirms the mechanism the search then

selects scale_pos_weight = 30 and reaches validation PR-AUC of 0.3985, ahead of both the baseline and the specified-grid winner. That diagnostic was scored on validation only and deliberately excluded from the exported deliverables, since the test set had already been spent and re-tuning against it would be the exact failure this protocol exists to prevent.

A paired bootstrap over the test set with 2,000 resamples places the before/after difference in PR-AUC at -0.0652, with a 95% confidence interval of [-0.1351, +0.0249] an interval that straddles zero. The tuned model outperforms the baseline in only 8.8% of resamples, a minority but not a decisive one. In units that matter to a credit committee, the entire before/after difference amounts to 2 firms out of 44.

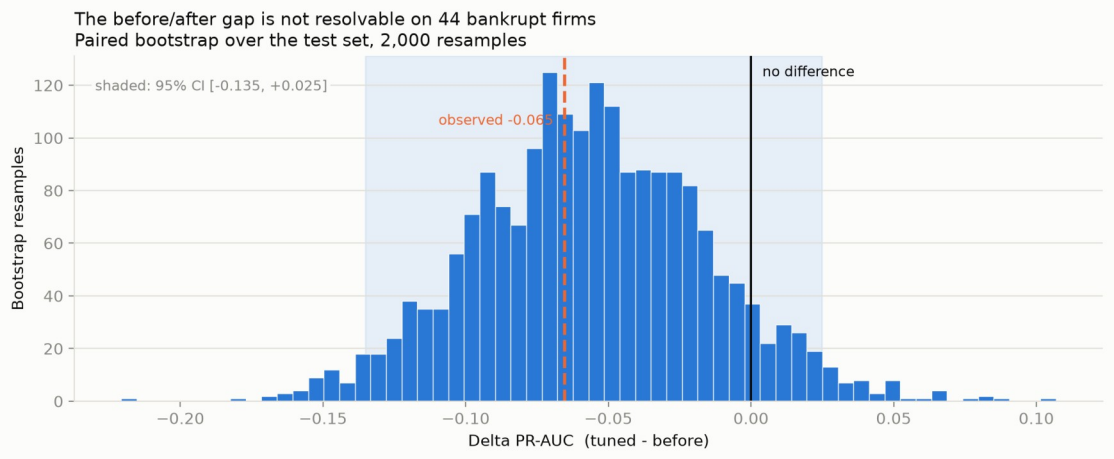


Figure 6. Bootstrap distribution of the before/after difference in PR-AUC.

The correct claim, then, is not that tuning degraded the model, but that tuning produced no measurable improvement, and this test set cannot resolve differences of the size under discussion. The 44-firm positive count flagged as a caveat in Week 5 turns out to be the binding constraint on the whole exercise it limits what the search can detect, what the validation split can select, and what the test set can confirm.

Limitations: the validation split contains only 35 bankrupt firms, so thresholds calibrated on it carry substantial uncertainty (one reason the calibrated model does not outperform the default

on test data); model selection used a single stratified split and a single 5-fold cross-validation, so repeated cross-validation with several seeds would give a more stable ranking at proportionally higher compute cost; the bootstrap resamples test rows but treats the fitted models as fixed, quantifying evaluation uncertainty rather than the additional variance from refitting on resampled training data; and results are reported for a single random seed (random_state = 42), so seed-to-seed variation in the selected configuration should be expected given the noise levels documented above.

Operations Risk Layer

The updated dataset contains 137,569 training rows and 34,393 test rows, with a late-delivery target rate of roughly 54.82% in both. A stratified 80/20 split of the training data created 27,514 validation rows and 110,055 training rows; validation was used only to adjust the prediction threshold, not for hyperparameter search. For the hyperparameter search, 50,000 records were sampled from the 110,055 to control computational cost, since RandomizedSearchCV can be expensive on datasets this large.

RandomizedSearchCV tested 25 random combinations drawn from n_estimators (200 to 800), max_depth ([10, 15, 20, unlimited]), min_samples_leaf (1 to 8), min_samples_split (2 to 10), and max_features ([sqrt, log2, 0.5]), using F1 as the target metric with 3-fold cross-validation. Prediction thresholds from 0.1 to 0.9 were then applied to the resulting models on the validation set, selecting the threshold that maximized F1. Both the untuned baseline and the fine-tuned model were run on the training data to generate final models for comparison on the test set, with the baseline compared at threshold 0.5 and the tuned model at its calibrated threshold.

Hyperparameter	Selected Value
n_estimators	500

Hyperparameter	Selected Value
max_depth	unlimited
min_samples_split	5
min_samples_leaf	1
max_features	0.5
Best CV F1 score	0.7051

Note. Selected Random Forest hyperparameters (Operations layer).

Precision and recall moved in opposite directions across the threshold range: a lower threshold flags more cases as late, catching more true late deliveries (higher recall) but also producing more false alarms (lower precision). The best validation F1 of 0.748 occurred at a threshold of 0.39, with precision of 0.676 and recall of 0.838 better than the standard 0.5 threshold.

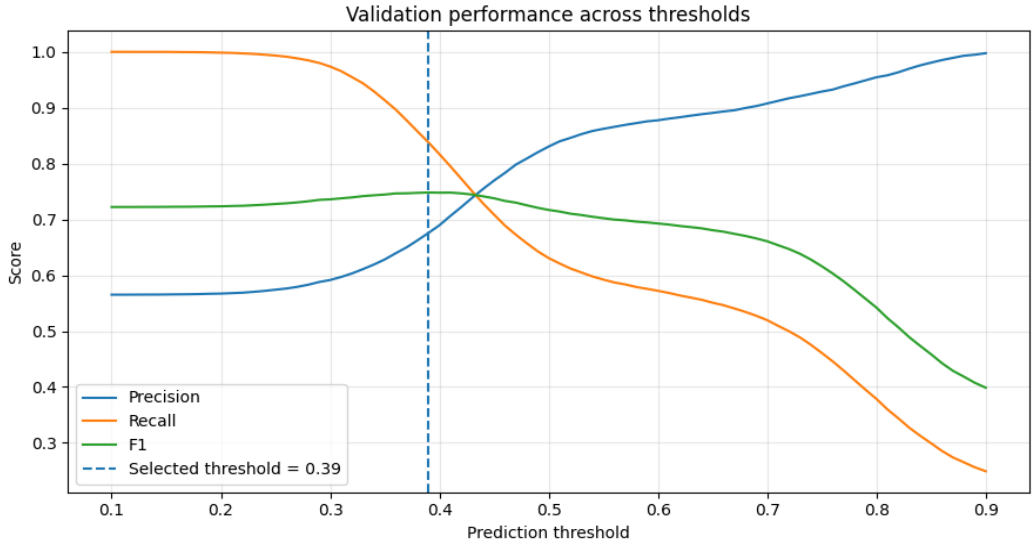


Figure 1 (Operations). Validation precision, recall, and F1 score across different decision thresholds.

Model	Threshold	Accuracy	Precision	Recall	F1	ROC-AUC	PR-AUC
Corrected Baseline RF	0.50	0.7315	0.8261	0.6461	0.7251	0.8138	0.8620

Model	Threshold	Accuracy	Precision	Recall	F1	ROC-AUC	PR-AUC
Tuned RF at 0.50 (default)	0.50	0.7337	0.8456	0.6290	0.7214	0.8258	0.8730
Tuned RF at 0.39 (calibrated)	0.39	0.7124	0.6944	0.8490	0.7639	0.8258	0.8730

Note. Final test performance by model configuration (Operations layer).

At the default threshold, tuning produced a mixed result F1 dipped slightly from 0.7251 to 0.7214, but ROC-AUC increased from 0.8138 to 0.8258 and PR-AUC improved from 0.8620 to 0.8730, meaning the tuned model ranks positive cases better overall even though F1 was marginally lower. The real improvement came from threshold calibration: moving the tuned model's decision threshold from 0.50 to 0.39 improved recall from 0.6290 to 0.8490 and F1 from 0.7214 to 0.7639, at the cost of precision dropping from 0.8456 to 0.6944.

Even though the F1 gain over baseline is not dramatic, the calibrated tuned model is the better fit for this use case. In an operations-risk context, missing a late delivery is generally more costly than flagging an on-time delivery as "at risk" a false negative is more expensive than a false positive, so it is better to double-check an order that turns out fine than to let a risky order go unnoticed. The calibrated model cut false negatives from 6,994 to 2,847, though false positives increased from 2,166 to 7,045.

Model	Threshold	TN	FP	FN	TP
Corrected Baseline RF	0.50	12,975	2,564	6,672	12,182
Tuned RF at 0.50 (default)	0.50	13,373	2,166	6,994	11,860
Tuned RF at 0.39 (calibrated)	0.39	8,494	7,045	2,847	16,007

Note. Confusion matrices of all evaluated models (Operations layer).

The practical effect of this trade-off depends on CEROP's stakeholder tolerance: if reviewing a flagged order is cheap, recall-oriented calibration is the right approach; if each flag triggers a costly manual process, this may not be the best practical model. RandomizedSearchCV was used instead of exhaustive GridSearchCV because the full parameter space ($4 \times 4 \times 4 \times 3 \times 3 = 576$ combinations) would have made a full grid search more expensive than the gain obtained justified.

Cross-Layer Synthesis

CEROP produces two independent risk signals this week, and they are not equally ready to hand to a business stakeholder. The Operations side improved in a way a supply-chain manager can act on right now: tuning plus threshold calibration pushed the model's F1 from 0.725 to 0.764, mostly by catching more late deliveries. False negatives dropped from 6,994 to 2,847 out of 34,393 orders about 4,100 additional at-risk shipments now getting flagged for review instead of just showing up late with no warning. The cost is more false alarms (2,166 jumped to 7,045), but in a shipping context a false alarm just means a review that turns out fine, while a missed flag means an unhappy customer and zero chance to fix anything beforehand the whole reason to recommend the calibrated, lower-threshold model over the raw tuned one at its default cutoff.

The Financial layer tells a different story, and it is just as useful for being a “no.” Tuning did not improve bankruptcy detection; the Week 5 configuration still catches more failing firms with fewer false alarms than anything RandomizedSearchCV turned up. For a lender or procurement team deciding which vendors need closer scrutiny, the current model is already close to the best this data and budget can produce. What would help is sourcing more bankruptcy examples the whole search was constrained by having only 141 failing firms to learn from.

Combined, CEROP's two halves sit at different points of business readiness. Operations is ready to deploy with a concrete, quantified improvement: fewer missed late deliveries, with a known and acceptable increase in manual reviews. Financial is ready to deploy as-is, with a clear recommendation that the next investment should go toward more labeled bankruptcies or additional financial signals rather than more search time on the same 5,455 firms. Neither layer can be merged into a single score a firm's balance sheet and a shipment's delivery risk answer different business questions for different stakeholders but a combined CEROP report can present both signals side by side: financial risk as a stable model to monitor as-is, next to delivery risk as a newly calibrated model that will surface more flags with review capacity planned for it.

Week 6 fine-tuned both shortlisted models under a shared protocol, and the two landed in different places. Hyperparameter search improved the Operations layer's ranking ability and, combined with threshold calibration, its practical F1 as well. Run on the Financial layer, the same search strategy optimized noise instead of signal a result traced to the 30:1 class imbalance and a positive-class count too small for the search to resolve reliably. Both outcomes are reported as found rather than adjusted to look better, consistent with the accuracy-trap discipline established since Week 2. For Week 7, the path forward is finalizing deployment specifications for both layers the Financial layer at its Week 5 configuration, the Operations layer at its Week 6 tuned and calibrated configuration and defining how the two independently scored risk signals are surfaced together in a single CEROP report, since their units of analysis are too different to merge into one model.

Conclusion

Across six weeks, CEROP went from a stated business problem to two finalized, evaluated classifiers running on a shared preprocessing and evaluation protocol. The Financial layer's story is one of a well-shortlisted model (XGBoost) that tuning could not meaningfully improve not because the search failed, but because only 141 bankrupt firms in the tuning split meant the search was optimizing noise nearly as often as signal. A paired bootstrap confirmed the before/after difference in PR-AUC was not distinguishable from zero. The honest recommendation for this layer is to ship the Week 5 configuration as-is and direct future effort toward sourcing more labeled bankruptcy examples, not toward further hyperparameter search on the same 5,455 firms.

The Operations layer tells a different story. Its near-balanced 54.8%/45.2% class split gave RandomizedSearchCV a stable signal to work with, and tuning plus threshold calibration (moving the decision cutoff from 0.50 to 0.39) pushed F1 from 0.725 to 0.764 by catching roughly 4,100 additional at-risk shipments that the untuned model would have missed. That gain came at the cost of more false alarms, but in an operations context a false alarm is a cheap manual review while a missed late delivery is an unhappy customer with no warning an asymmetry the team weighted consistently from Week 2's performance-measure discussion through to the final calibration decision in Week 6.

Neither layer produces a single merged risk score, and that is a deliberate design choice rather than a limitation the team ran out of time to fix: a firm's balance sheet and a shipment's delivery timeline answer different business questions for different stakeholders, and forcing them into one number would hide more than it revealed. What CEROP delivers instead is two independently validated, independently calibrated risk signals that a procurement or credit team can read side by side one stable and ready to monitor as-is, the other newly tuned with a known,

quantified trade-off between catching more risk and reviewing more orders. That two-signal structure, evaluated honestly at every step rather than adjusted to look better than the data supports, is the project's core contribution.

References

Ahmed, K. R., Ansari, M. E., Ahsan, M. N., Rohan, A., Uddin, M. B., & Rivin, M. A. H. (2025). Deep learning framework for interpretable supply chain forecasting using SOM ANN and SHAP. Scientific Reports, 15(1), Article 26355.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13, 281–305.

Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324

Brownlee, J. (2021). Tour of evaluation metrics for imbalanced classification. Machine Learning Mastery. https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/

Censinet, Inc. (2025, December 24). Ultimate guide to vendor risk scoring 2025. https://censinet.com/perspectives/ultimate-guide-to-vendor-risk-scoring-2025

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785–794). https://doi.org/10.1145/2939672.2939785

Constante, F., Silva, F., & Pereira, A. (2019). DataCo smart supply chain for big data analysis [Data set]. Mendeley Data. https://doi.org/10.17632/8gx2fvg2k6.5

Davis, J., & Goadrich, M. (2006). The relationship between precision-recall and ROC curves. In Proceedings of the 23rd International Conference on Machine Learning (pp. 233–240). https://doi.org/10.1145/1143844.1143874

Dun & Bradstreet. (2026, March 20). Supplier Evaluation Risk (SER) rating. https://www.dnb.com/en-us/smb/resources/credit-scores/supplier-evaluation-rating-declined-ser-info.html

Efron, B., & Tibshirani, R. J. (1993). An introduction to the bootstrap. Chapman & Hall.

Federal Reserve Bank of New York. (n.d.). Global Supply Chain Pressure Index (GSCPI). Retrieved July 8, 2026, from https://www.newyorkfed.org/research/policy/gscpi

Géron, A. (2022). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems (3rd ed.). O'Reilly Media.

Liang, D., Lu, C.-C., Tsai, C.-F., & Shih, G.-A. (2016). Financial ratios and corporate governance indicators in bankruptcy prediction: A comprehensive study. European Journal of Operational Research, 252(2), 561–572. https://doi.org/10.1016/j.ejor.2016.01.012

Liao, J. G., & Chin, K.-V. (2007). Logistic regression for disease classification using microarray data: Model selection in a large p and small n case. Bioinformatics, 23(15), 1945–1951. https://doi.org/10.1093/bioinformatics/btm270

Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. R News, 2(3), 18–22.

Moody's. (n.d.). Supplier risk management solutions. Retrieved July 8, 2026, from https://www.moodys.com/web/en/us/solutions/compliance-third-party-risk/supplier-risk.html

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in

Python. Journal of Machine Learning Research, 12, 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html

Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLOS ONE, 10(3), Article e0118432. https://doi.org/10.1371/journal.pone.0118432

Taiwanese Bankruptcy Prediction [Data set]. (2020). UCI Machine Learning Repository. https://doi.org/10.24432/C5004D

Varma, S., & Simon, R. (2006). Bias in error estimation when using cross-validation for model selection. BMC Bioinformatics, 7, Article 91. https://doi.org/10.1186/1471-2105-7-91

Wolpert, D. H. (1992). Stacked generalization. Neural Networks, 5(2), 241–259. https://doi.org/10.1016/S0893-6080(05)80023-1