Lesson 27: Classification Metrics

Data Science with Python – BSc Course

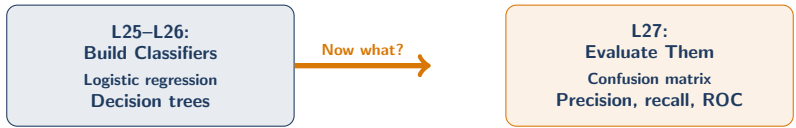
Data Science Program

BSc Course

45 Minutes

Previously on L25–L26...

We built classifiers. But how do we know they're any good?



- L25: Logistic regression outputs **probabilities** → need a threshold
- L26: Decision trees create **non-linear boundaries** → but how accurate?
- Today: The metrics toolkit to answer "Is this model actually useful?"

Building a model is only half the job – evaluating it properly is the other half

Learning Objectives

The Problem: Our classifier has 95% accuracy – is that good? What if 95% of samples are one class? How do we properly evaluate classification models?

After this lesson, you will be able to:

- Build and interpret **confusion matrices** (the foundation)
- Calculate **precision, recall, and F1 score** (and know when each matters)
- Plot and interpret **ROC curves and AUC** (threshold-independent evaluation)
- **Choose metrics** appropriate for your specific problem and cost structure

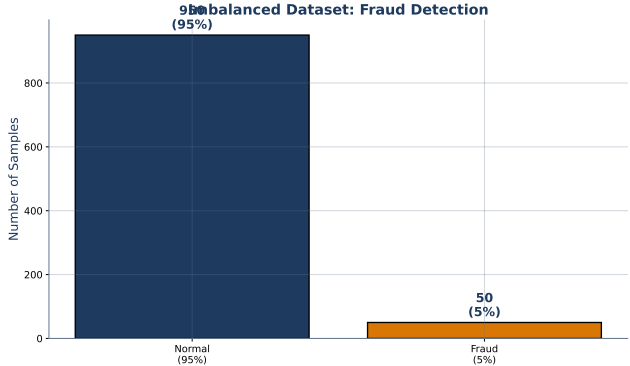
---

Finance Application: Evaluating fraud detection and default prediction models

The Accuracy Trap

Your spam filter has 99% accuracy. Impressive?

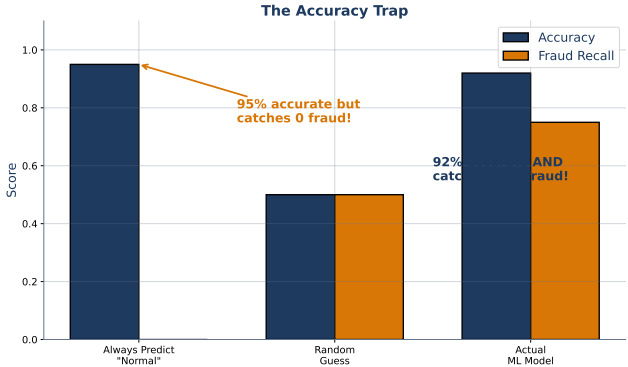
- What if only 1% of emails are spam?
- A model that **always** says "not spam" gets 99% accuracy but catches **zero** spam



1% fraud rate: predicting "no fraud" gives 99% accuracy but catches zero frauds

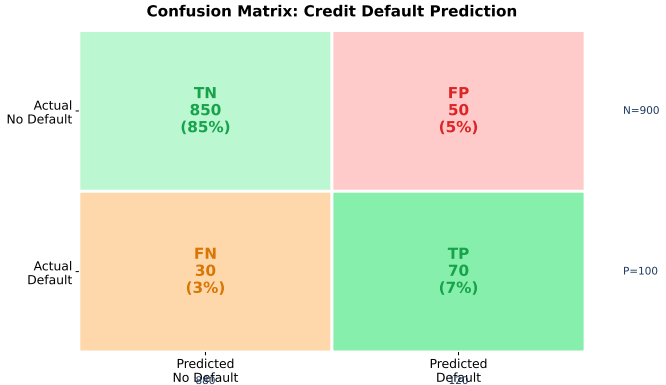
The Accuracy Trap: Visualized

High accuracy, useless model – the trap in action



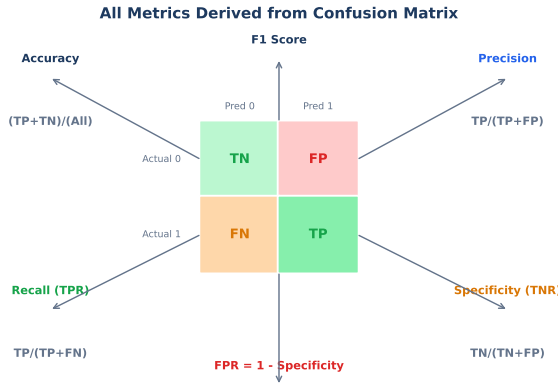
Rule: Never use accuracy alone when classes are imbalanced

Confusion Matrix: Visual Overview



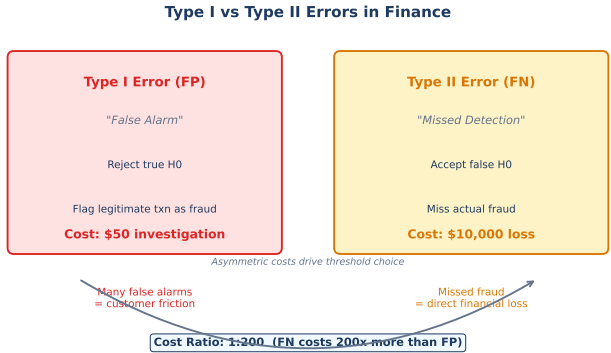
Self-study: Visual representation of the 2x2 confusion matrix

Metrics Overview



Self-study: How all classification metrics relate to each other

Type I vs Type II Errors: Visual



Self-study: Visual comparison of Type I (FP) and Type II (FN) errors

Beyond Accuracy: The Confusion Matrix

The 2×2 Scorecard – Where Exactly Does Our Model Fail?

	Predicted: No	**Predicted: Yes**
Actual: No	TN (correct rejection)	FP (false alarm)
Actual: Yes	FN (missed detection)	TP (correct detection)

Spam Filter Analogy:

- **TP:** Correctly flagged spam → good!
- **TN:** Correctly let real email through → good!
- **FP:** Flagged real email as spam → you miss an important email
- **FN:** Let spam through → annoying but less harmful

Accuracy hides the **types** of errors. The confusion matrix reveals them.

The confusion matrix is the foundation – all other classification metrics derive from it

Type I vs Type II Errors

Connection to Hypothesis Testing (L15)

Error	Statistics (L15)	Classification
Type I (α)	Reject H_0 when true	FP (false positive)
Type II (β)	Fail to reject H_0 when false	FN (false negative)

Finance: Fraud Detection

- H_0: Transaction is legitimate (class 0)
- Type I (FP): Flag legitimate transaction $\rightarrow$ customer annoyance, wasted investigation
- Type II (FN): Miss actual fraud $\rightarrow$ **financial loss** (usually much worse!)

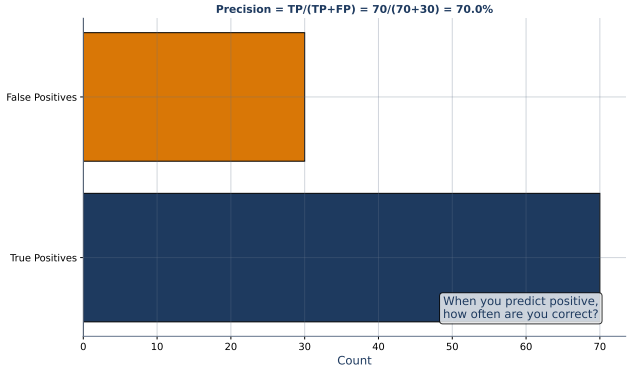
Key insight: FP and FN have different **costs**. Choosing a metric means deciding which error matters more.

The α/β trade-off in statistics = precision/recall trade-off in ML

Precision: Quality of Positives

"Of things I flagged, how many were correct?"

- Precision = $\frac{TP}{TP+FP}$ (high = few false alarms)
- Prioritize when **FP is costly** (unnecessary surgery, blocked cards)

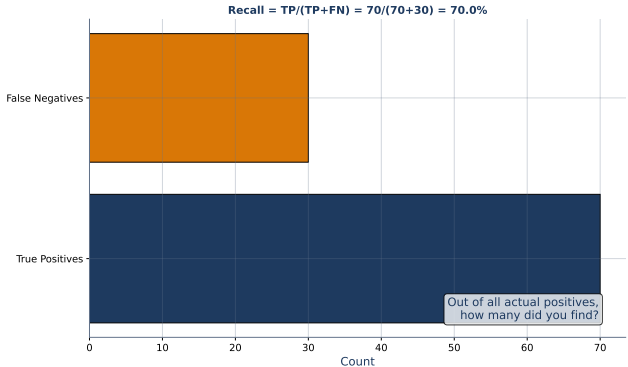


High precision = few false alarms. Important when FP is costly.

Recall: Coverage of Positives

"Of actual positives, how many did I find?"

- Recall = $\frac{TP}{TP+FN}$ (high = few missed positives)
- Prioritize when **FN is costly** (missed cancer, undetected fraud)

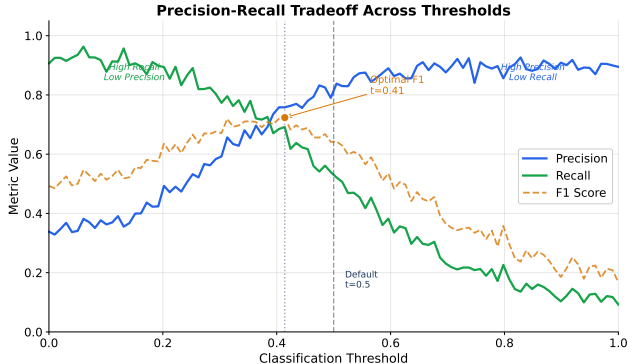


Memory: Precision = Predictions correct. Recall = Real positives Recovered.

The Precision-Recall Tradeoff

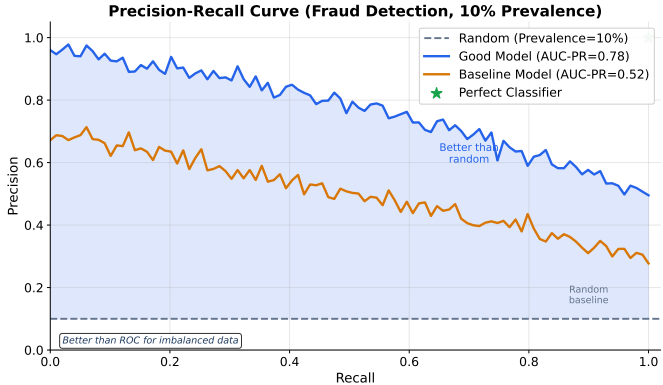
You can't maximize both simultaneously

- Raising threshold: ↑precision ↓recall (fewer but confident predictions)
- Lowering threshold: ↓precision ↑recall (catch more, more false alarms)



Trade-off: Every threshold choice sacrifices one metric for the other

Precision-Recall Curve

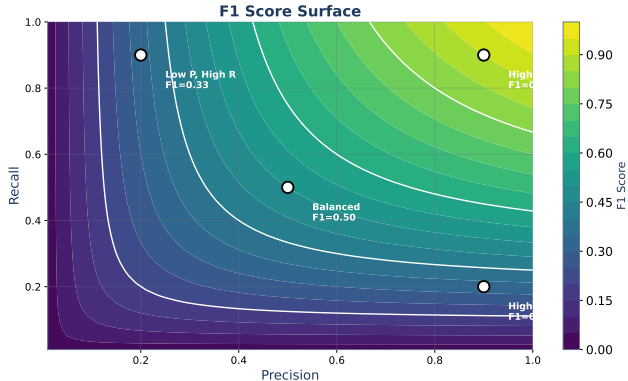


Self-study: The PR curve shows precision vs recall at every threshold

F1 Score: The Balanced View

Harmonic mean of precision and recall

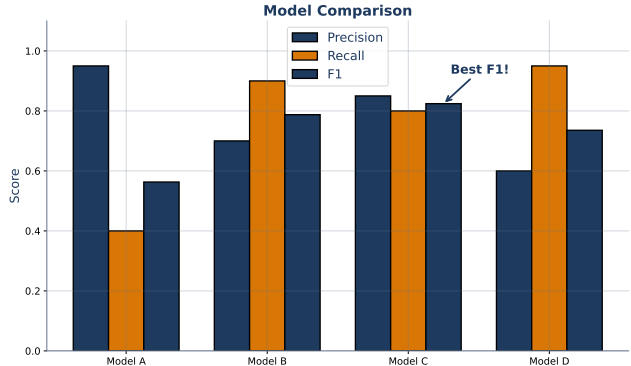
- Arithmetic: $(1.0 + 0.0)/2 = 0.5$ looks OK, but finds **zero** positives!
- Harmonic: $F_1(1.0, 0.0) = 0$ – correctly reflects failure



$F_1 = 2 \cdot \frac{P \cdot R}{P + R}$. F1 high only when BOTH P and R are high.

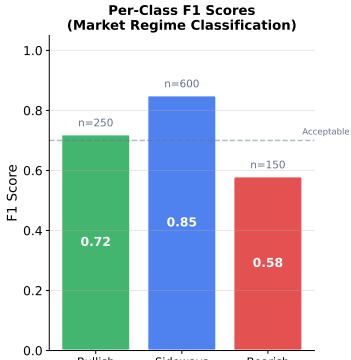
F1 Score: Model Comparison

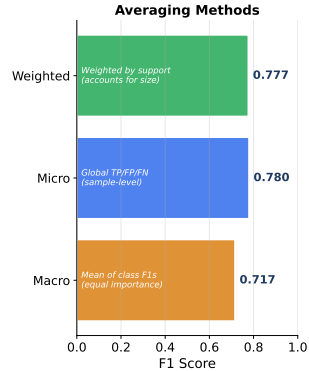
Compare multiple models on the same dataset



F1 = 0 if either precision or recall is 0. Max F1 = 1 (perfect).

Multiclass Classification Metrics





Self-study: Macro, micro, and weighted averaging for multiclass problems

ROC Curve: The Complete Picture

"What if we could evaluate ALL thresholds at once?"

Analogy – Metal Detector at Airport Security:

- Low sensitivity: catches only large weapons (few false alarms)
- High sensitivity: catches everything including belt buckles (many false alarms)
- ROC curve traces this trade-off across **all** sensitivity settings

Reading the Curve:

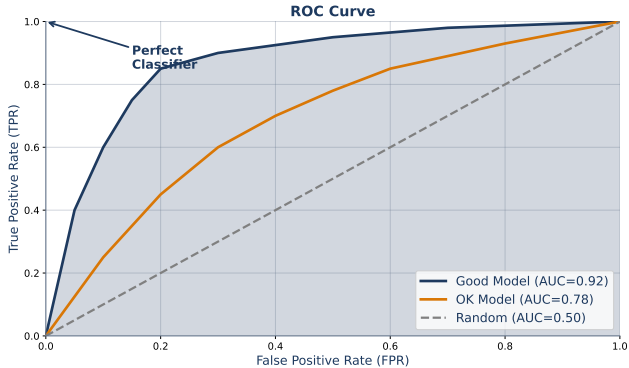
- **X-axis:** False Positive Rate (FPR) = FP / (FP + TN)
- **Y-axis:** True Positive Rate (TPR) = TP / (TP + FN) = Recall!
- **Top-left corner** = perfect (catches all positives, zero false alarms)
- **Diagonal line** = random guessing (coin flip)

ROC = Receiver Operating Characteristic – originally from WWII radar signal detection

How ROC Curves Are Built

Each point = one threshold setting

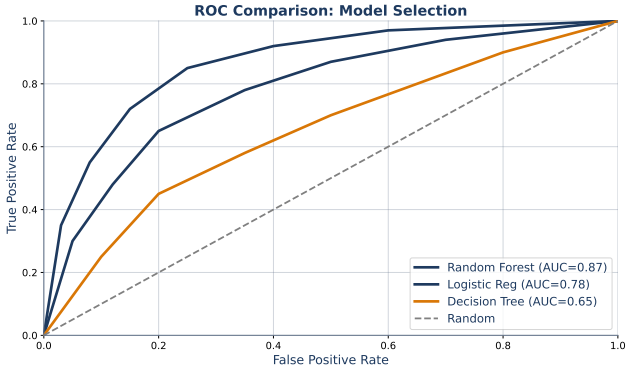
- Threshold = 1.0: predict all negative → (0, 0). Threshold = 0.0: all positive → (1, 1)
- Each threshold gives one point; trace the curve



Good models bow toward the top-left corner. Diagonal = random guessing.

ROC: Model Comparison

Curve closer to top-left = better classifier

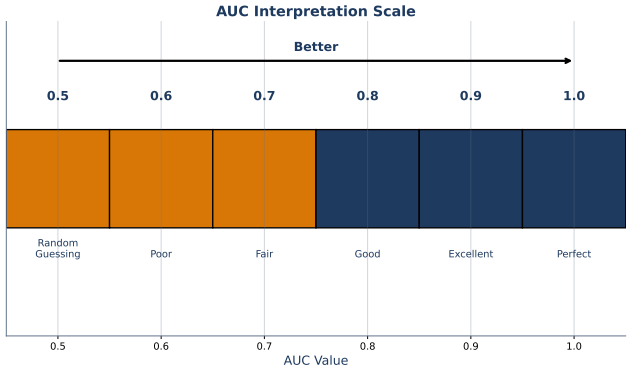


Perfect classifier: curve goes straight to top-left corner (TPR=1, FPR=0)

AUC: One Number to Rule Them All

AUC = probability that model ranks a random positive higher than a random negative

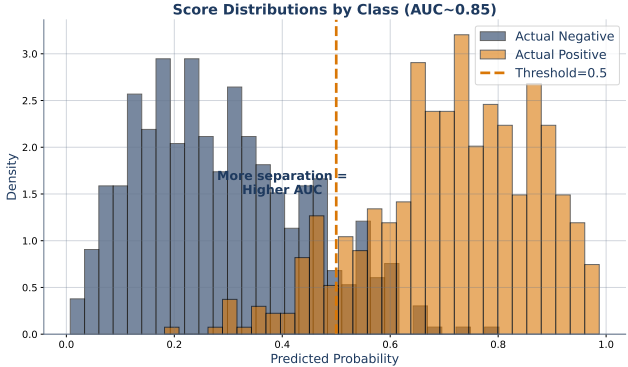
- 0.5 = random, 0.7 = acceptable, 0.8 = good, 0.9+ = excellent
- **Threshold-independent** – summarizes all thresholds in one number



AUC = 0.5: random. AUC = 1.0: perfect separation. Widely used for model comparison.

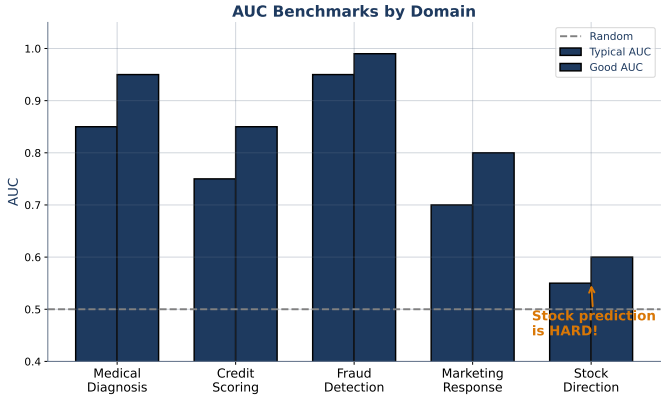
AUC: Score Distributions

Good model: class score distributions are well separated



AUC = probability that model ranks a random positive higher than a random negative

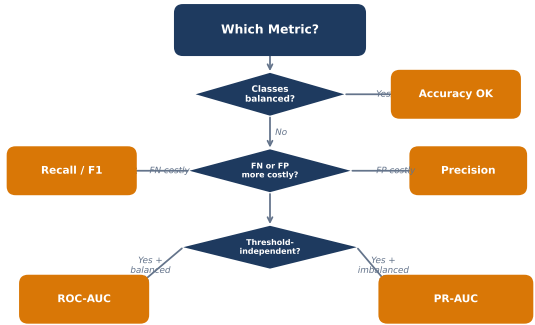
AUC: Industry Benchmarks



Self-study: AUC benchmarks across different industries and applications

Which Metric Should I Use?

The decision flowchart for metric selection

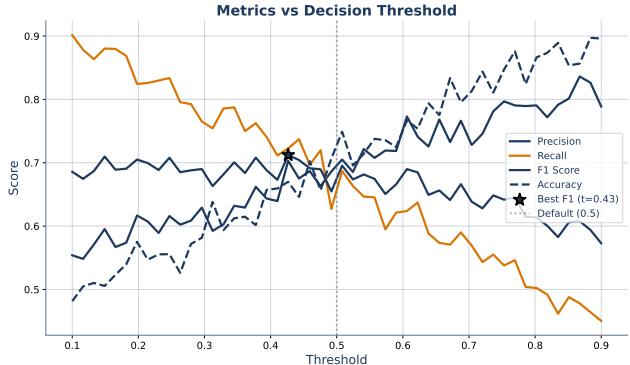


Match your metric to your problem's cost structure – there is no universal best metric

Threshold Tuning: Why 0.5 Is Wrong

Default 0.5 assumes equal costs. Reality: FP $\neq$ FN costs.

- **Lower threshold:** FN costly (cancer screening – don't miss cases)
- **Raise threshold:** FP costly (trading – false signals cost money)



Optimal threshold: $t^* = C_{FP}/(C_{FP} + C_{FN})$. Always tune for business costs.

Threshold: Cost-Based Selection

Self-study: Visualizing how cost ratios determine optimal threshold

Common Mistakes in Classification Evaluation

Evaluation Pitfalls Checklist

- ✗ Using accuracy on imbalanced data
- ✗ Comparing models on different test sets
- ✗ Not tuning threshold for business costs
- ✗ Reporting ROC-AUC for highly imbalanced data (use PR-AUC)

Comprehension Check

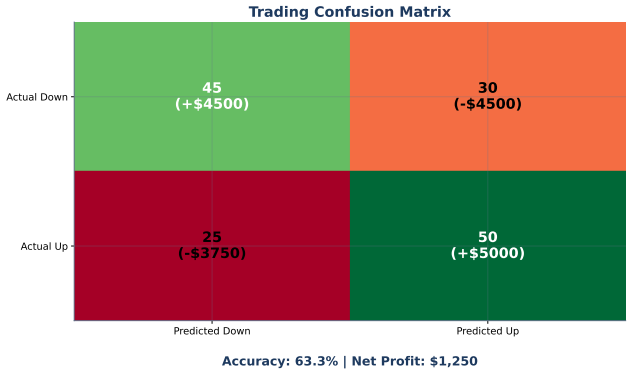
Your fraud model has AUC=0.95 but recall=0.10. Good?

No – ranks well but catches almost no fraud at current threshold.

A good AUC with poor recall means the threshold needs adjustment

Finance: Trading Confusion Matrix

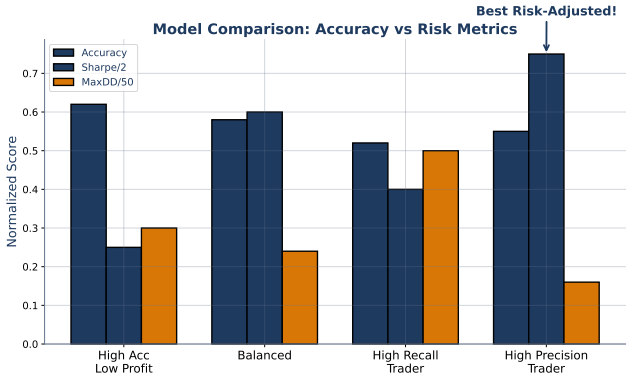
Trading: FP = false signal (losses) vs FN = missed opportunity (forgone profit)



In trading, false signals cost real money – precision typically matters more than recall

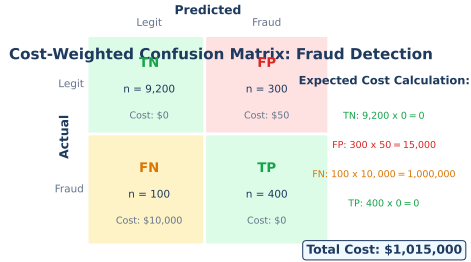
Finance: Strategy Performance

Match metric to cost structure



Different strategies need different metrics: fraud → recall, trading → precision

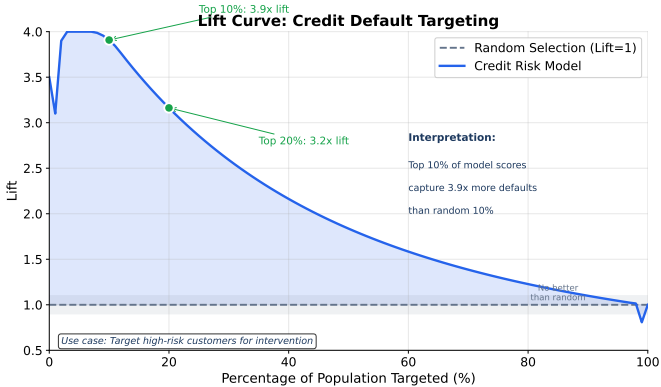
Finance: Fraud Cost Matrix



FN dominates total cost despite being only 100 cases (100 x 10k = 1M)

Self-study: Quantifying the dollar cost of each cell in the confusion matrix

Finance: Lift Curve



Self-study: Lift curves show how much better a model is than random selection

Hands-On Exercise (25 min)

Task: Evaluate a Default Prediction Model

1. Fit logistic regression on credit data (target: default yes/no)
2. Generate confusion matrix with `confusion_matrix()`
3. Calculate precision, recall, F1 using `classification_report()`
4. Plot ROC curve and calculate AUC
5. Try thresholds 0.3 and 0.7 – how do metrics change?

Deliverable: Confusion matrix heatmap + ROC curve with AUC annotation.

Extension: Calculate expected cost at each threshold using a custom cost matrix

Lesson Summary + Preview

Beyond the Grade – What We Learned:

- **Confusion matrix:** TP, TN, FP, FN – the foundation for everything
- **Precision** = quality of positives, **Recall** = coverage of positives
- **F1 score:** balanced view (harmonic mean penalizes imbalance)
- **ROC/AUC:** threshold-independent performance measure
- **Choose metrics** based on business costs, not convenience

Next Lesson – Finding Needles in Haystacks:

What if only 0.1% of your data is the class you care about? L28 tackles **class imbalance**: SMOTE, cost-sensitive learning, and evaluation under extreme skew.

Memory: Precision = Positive predictions correct. Recall = Real positives Recovered.