Digital Finance 3: Technology in Finance

Lesson 28: Supervised Learning - Classification

FHGR

December 11, 2025

Learning Objectives

By the end of this lesson, you will be able to:

- Explain classification problems and contrast with regression
- Understand logistic regression and probability estimation
- Interpret confusion matrices and classification metrics
- Calculate and interpret accuracy, precision, recall, and F1-score
- Use ROC curves and AUC for model evaluation
- Apply classification to credit scoring and fraud detection

Classification vs. Regression

Regression (Lesson 27):

- Predict continuous values
- $Y \in \mathbb{R}$ (e.g., stock return, price)
- Example: Predict bond yield (3.2%)
- Metrics: R^2, RMSE, MAE

Classification (Today):

- Predict discrete categories
- $Y \in \{0, 1, 2, \ldots, K\}$
- Binary: Default (Yes/No)
- Multi-class: Credit rating (AAA, AA, A, ...)
- Metrics: Accuracy, precision, recall

Focus Today: Binary classification (most common in finance).

Finance Examples:

Binary Classification:

- Loan default (Yes/No)
- Fraud transaction (Fraud/Legitimate)
- Stock direction (Up/Down)
- M&A success (Completed/Failed)

Multi-class:

- Credit ratings (10 grades)
- Customer segments (5 types)
- Market regime (Bull/Bear/Sideways)

Comparative analysis helps identify the right tool for specific requirements.

Logistic Regression: The Model

Why Not Linear Regression?

- For binary $Y \in \{0, 1\}$, OLS can predict $\hat{Y} < 0$ or $\hat{Y} > 1$
- Violates probability interpretation
- Residuals not normal

Logistic Regression:

- Predict probability: $P(Y = 1|X)$
- Output constrained to $[0, 1]$
- Use logistic (sigmoid) function

Model:

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p)}}$$

Logit Form:

$$\log\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

- Left side: Log-odds (logit)
- Right side: Linear combination (like linear regression)

Interpretation:

- $\beta_j > 0$: Increase in X_j increases probability of $Y = 1$
- $\beta_j < 0$: Decrease probability
- Magnitude: Change in log-odds per unit X_j

Estimation: Maximum Likelihood (not OLS).

Regression models predict continuous outcomes based on input features.

Example: Loan Default Prediction

Problem:

- Predict default (1) vs. repayment (0)
- Features:
  - Credit score
  - Debt-to-income ratio
  - Loan amount
  - Employment status

Fitted Model:

$$\log\left(\frac{P(\text{Default})}{1-P(\text{Default})}\right) = 2.5 - 0.015 \times \text{CreditScore} + 1.2 \times \text{DebtToIncome}$$

Interpretation:

- **Credit Score** ($\beta = -0.015$):
  - 10-point increase → log-odds decrease by 0.15
  - Higher score → lower default probability
- **Debt-to-Income** ($\beta = 1.2$):
  - 0.1 increase (10 pp) → log-odds increase by 0.12
  - Higher debt burden → higher default risk

Example Prediction:
Credit Score = 650, D/I = 0.4:

$$\log(\text{odds}) = 2.5 - 0.015(650) + 1.2(0.4) = -7.28$$

$$P(\text{Default}) = \frac{1}{1+e^{7.28}} = 0.0007 \ (0.07\%)$$

Case studies provide concrete evidence of technology impact and adoption patterns.

Making Predictions: Decision Threshold

From Probabilities to Classes:

- Logistic regression outputs $P(Y = 1|X) \in [0, 1]$
- Need decision rule to classify
- Default threshold: 0.5
  - If $P \geq 0.5 \rightarrow$ predict 1
  - If $P < 0.5 \rightarrow$ predict 0

Custom Thresholds:

- Threshold is tunable hyperparameter
- Lower threshold (0.3): More sensitive (more positives)
- Higher threshold (0.7): More specific (fewer false positives)

Example (Fraud Detection):

- Transaction 1: $P(\text{Fraud}) = 0.8 \rightarrow$ Flag as fraud
- Transaction 2: $P(\text{Fraud}) = 0.3 \rightarrow$ Depends on threshold
  - Threshold $= 0.5 \rightarrow$ Legitimate
  - Threshold $= 0.2 \rightarrow$ Fraud (more cautious)

Threshold Selection:

- Depends on cost of errors
- Fraud: False negative costly (miss fraud) $\rightarrow$ low threshold
- Spam: False positive costly (block good email) $\rightarrow$ high threshold
- Use ROC curve for tuning (later)

Market prediction is inherently difficult due to efficiency and noise.

Confusion Matrix

2x2 Table of Predictions vs. Actuals:

	Actual 0	**Actual 1**
Predict 0	TN (True Neg)	FN (False Neg)
Predict 1	FP (False Pos)	TP (True Pos)

Definitions:

- **TP:** Correctly predicted positive (fraud caught)
- **TN:** Correctly predicted negative (legit confirmed)
- **FP:** False alarm (Type I error, legit flagged as fraud)
- **FN:** Missed positive (Type II error, fraud missed)

Example (Fraud Detection):
Out of 1000 transactions:

	Not Fraud	**Fraud**
Predict Not	920 (TN)	10 (FN)
Predict Fraud	30 (FP)	40 (TP)

Interpretation:

- 40 frauds caught (TP)
- 10 frauds missed (FN) - costly!
- 30 false alarms (FP) - customer inconvenience
- 920 correct legitimate (TN)

All metrics derived from this matrix.

Key concepts from this slide inform practical applications in finance.

Classification Metrics: Accuracy

Accuracy:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Proportion of correct predictions
- Range: $[0, 1]$ (higher is better)
- Intuitive, widely reported

Example (previous slide):

$$\text{Accuracy} = \frac{40 + 920}{1000} = 0.96 \ (96\%)$$

Looks great! But...

Problem: Imbalanced Classes

Suppose only 50 frauds (5%) in 1000 transactions.

Naive Model: Predict all as "Not Fraud"

$$\text{Accuracy} = \frac{950}{1000} = 95\%$$

High accuracy, but useless model (misses all fraud)!

Lesson:

- Accuracy misleading for imbalanced data
- Common in finance (fraud 1-2%, default 2-5%)
- Need better metrics: Precision, Recall

Network metrics provide objective measures of adoption and ecosystem health.

Precision and Recall

Precision (Positive Predictive Value):

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Of predicted positives, how many are correct?
- Answers: "When model says fraud, is it really fraud?"
- High precision: Few false alarms

Recall (Sensitivity, True Positive Rate):

$$\text{Recall} = \frac{TP}{TP + FN}$$

- Of actual positives, how many did we catch?
- Answers: "What fraction of fraud did we detect?"
- High recall: Few missed positives

Example (Fraud):
TP=40, FP=30, FN=10, TN=920

$$\text{Precision} = \frac{40}{40 + 30} = 0.571 \ (57\%)$$

$$\text{Recall} = \frac{40}{40 + 10} = 0.80 \ (80\%)$$

Interpretation:

- 57% of flagged transactions are actual fraud (43% false alarms)
- 80% of frauds are caught (20% missed)

Trade-off:

- Lower threshold $\rightarrow$ Higher recall, lower precision
- Higher threshold $\rightarrow$ Higher precision, lower recall

Key concepts from this slide inform practical applications in finance.

F1-Score: Balancing Precision and Recall

F1-Score:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- Harmonic mean of precision and recall
- Range: $[0, 1]$ (higher is better)
- Balances both concerns
- Single metric for model comparison

Example:
Precision $= 0.571$, Recall $= 0.80$

$$F1 = 2 \times \frac{0.571 \times 0.80}{0.571 + 0.80} = 0.667$$

When to Use F1:

- Imbalanced classes (fraud, default)
- Need balance between precision and recall
- No strong preference for one over the other

Weighted F1:
If costs unequal:

$$F_\beta = (1 + \beta^2) \times \frac{\text{Prec} \times \text{Rec}}{\beta^2 \times \text{Prec} + \text{Rec}}$$

- $\beta > 1$: Favor recall (e.g., $\beta = 2$)
- $\beta < 1$: Favor precision (e.g., $\beta = 0.5$)

Alternative: Directly optimize business metric (expected cost/profit).

Key concepts from this slide inform practical applications in finance.

ROC Curve and AUC

ROC (Receiver Operating Characteristic):

- Plot TPR (Recall) vs. FPR across all thresholds
- **TPR** (True Positive Rate) = Recall = $\frac{TP}{TP+FN}$
- **FPR** (False Positive Rate) = $\frac{FP}{FP+TN}$
- X-axis: FPR (0 to 1)
- Y-axis: TPR (0 to 1)

Interpretation:

- Each point: Different threshold
- Top-left corner: Perfect classifier (TPR=1, FPR=0)
- Diagonal: Random guessing
- Further from diagonal: Better model

AUC (Area Under Curve):

- Aggregate metric: Area under ROC curve
- Range: $[0, 1]$
- AUC = 0.5: Random (no skill)
- AUC = 1.0: Perfect classifier
- AUC = 0.7-0.8: Good
- AUC = 0.8-0.9: Very good
- AUC ¿ 0.9: Excellent (or overfitting?)

Advantages:

- Threshold-independent
- Single number for model comparison
- Handles imbalanced classes

Finance: AUC 0.7-0.8 typical for credit/fraud models.

Key concepts from this slide inform practical applications in finance.

Application: Credit Scoring

Problem:

- Predict loan default (1) vs. repayment (0)
- Features: Credit score, income, debt, employment, history
- Imbalanced: 2-5% default rate

Model Comparison:

- **Logistic Regression:** AUC = 0.72
- **Random Forest:** AUC = 0.76
- **Gradient Boosting:** AUC = 0.78

Deployment:

- Set threshold based on business objective
- Example: Approve top 60% (by score)
- Monitor performance over time

Cost-Benefit Analysis:

- **True Negative:** Correctly reject bad loan → \$0 loss avoided
- **False Positive:** Reject good borrower → Lost profit (\$500)
- **True Positive:** Approve good loan → Profit (\$500)
- **False Negative:** Approve bad loan → Loss (\$5000)

Optimal Threshold:

- Not 0.5! Depends on costs
- FN (miss bad loan) 10x worse than FP (reject good)
- Use higher threshold (more conservative)
- Maximize expected profit, not accuracy

Regulatory: Must ensure fairness (no discrimination).

Real-world applications demonstrate the practical value of blockchain technology.

Application: Fraud Detection

Problem:

- Real-time detection of fraudulent transactions
- Highly imbalanced: 0.1-0.5% fraud rate
- Features:
  - Transaction amount
  - Time of day, day of week
  - Merchant category
  - Location (IP, GPS)
  - User behavior patterns (deviations)

Challenges:

- Extreme imbalance (99.5% legitimate)
- Real-time requirement (¡ 100ms)
- Adversarial (fraudsters adapt)
- False positives costly (block legit transactions)

Approach:

- Class reweighting or SMOTE (oversample minority)
- Ensemble methods (Random Forest, XGBoost)
- Anomaly detection (isolation forest, autoencoders)
- Multi-stage: Rules + ML

Typical Performance:

- Precision: 10-20% (many false positives)
- Recall: 60-80% (catch most fraud)
- F1: 0.20-0.30
- AUC: 0.85-0.95

Evolution:

- Fraudsters constantly evolve
- Models decay over time (concept drift)
- Continuous retraining required

Real-world applications demonstrate the practical value of blockchain technology.

Multi-Class Classification

Extension to K Classes:

- Binary: $Y \in \{0, 1\}$
- Multi-class: $Y \in \{1, 2, \ldots, K\}$
- Example: Credit ratings (AAA, AA, A, BBB, ...)

Approaches:

1. **One-vs-Rest (OvR):**
   - Train K binary classifiers
   - Class 1 vs. All, Class 2 vs. All, etc.
   - Predict class with highest probability
2. **One-vs-One (OvO):**
   - Train $\binom{K}{2}$ binary classifiers
   - All pairwise comparisons
   - Majority vote
3. **Softmax Regression (Multinomial Logistic):**
   - Direct K-class model
   - Outputs probabilities for all K classes

Softmax Model:

$$P(Y = k|X) = \frac{e^{\beta_k^T X}}{\sum_{j=1}^{K} e^{\beta_j^T X}}$$

- K sets of coefficients: $\beta_1, \ldots, \beta_K$
- Probabilities sum to 1

Evaluation:

- Confusion matrix now $K \times K$
- Multi-class accuracy, precision, recall (per class)
- Macro vs. Micro averaging
- AUC: One-vs-rest approach

Finance Example:
Predict credit rating transition:

- Features: Financial ratios, macro indicators
- Classes: Upgrade, Stable, Downgrade

Classification models assign discrete labels to observations.

Handling Imbalanced Data

Problem:

- Minority class underrepresented
- Model biased toward majority
- Poor recall for rare events

Resampling Techniques:

- **Undersampling:** Randomly remove majority class
  - Pro: Balanced dataset
  - Con: Lose information
- **Oversampling:** Duplicate minority class
  - Pro: Keep all data
  - Con: Overfitting risk
- **SMOTE:** Synthetic Minority Oversampling
  - Generate synthetic examples (interpolation)
  - Better than simple duplication

Algorithmic Approaches:

- **Class weights:** Penalize misclassifying minority more
  - In logistic regression: $w_1 = n_0/n_1$
  - Automatically adjusts loss function
- **Ensemble methods:** Bagging with balanced samples
- **Anomaly detection:** Treat minority as anomalies

Best Practice:

- Try multiple approaches
- Use stratified cross-validation
- Optimize F1 or AUC, not accuracy
- Domain expertise for feature engineering

Finance: Imbalance is the norm (default, fraud, rare events).

Quality data is the foundation for effective machine learning models.

Probability Calibration

Problem:

- Model outputs: Not always true probabilities
- Example: Predicts $P = 0.7$, but only 50% of such cases are positive
- Miscalibration common in ML

Why It Matters:

- Cost-benefit analysis requires true probabilities
- Regulatory reporting (PD, LGD in Basel)
- Decision-making (set reserves)

Calibration Methods:

- **Platt scaling:** Fit logistic regression on validation set
- **Isotonic regression:** Non-parametric calibration
- **Beta calibration:** Generalization of Platt

Evaluation:

- **Calibration plot:** Predicted prob vs. observed frequency
- **Brier score:**

$$BS = \frac{1}{n}\sum_{i=1}^{n}(p_i - y_i)^2$$

  - Lower is better
  - Measures both calibration and discrimination

Finance Application:

- Credit models: PD (probability of default) must be well-calibrated
- Regulatory requirement (IFRS 9, CECL)
- Miscalibration $\rightarrow$ Wrong provisions, capital requirements

Note: Logistic regression often well-calibrated out-of-the-box; tree models often need calibration.

Key concepts from this slide inform practical applications in finance.

Summary and Key Takeaways

Core Concepts:

- Classification: Predict discrete categories
- Logistic regression: Model probabilities
- Confusion matrix: TP, TN, FP, FN
- Threshold selection: Tune for business objective

Metrics:

- Accuracy: Misleading for imbalanced data
- Precision: Fraction of predicted positives correct
- Recall: Fraction of actual positives caught
- F1: Harmonic mean of precision and recall
- AUC: Threshold-independent performance

Finance Applications:

- Credit scoring: Default prediction
- Fraud detection: Real-time anomaly detection
- Both face severe class imbalance
- Cost-benefit analysis crucial

Practical Considerations:

- Handle imbalanced data (resampling, class weights)
- Calibrate probabilities for decision-making
- Monitor model decay over time
- Ensure fairness and explainability

Next Lesson Preview

Lesson 29: Algorithmic Trading Concepts

Topics to be covered:

- Types of algorithmic trading strategies
- Backtesting framework and pitfalls
- Overfitting in trading models
- Transaction costs and market impact
- Risk management in algo trading
- Realistic performance expectations

Preparation:

- Review time series concepts (stationarity, autocorrelation)
- Think: How would you test a trading strategy?