Cost-Sensitive Payment Fraud Detection Using KNN and Random Forest

Payment fraud detection is a critical task in financial systems to prevent unauthorized transactions. Traditional machine learning methods often fail to address the **cost sensitivity** associated with fraud detection, where misclassifying fraud (false negatives) is far more costly than misclassifying legitimate transactions (false positives).

Here's a brief overview of the approach using **K-Nearest Neighbors (KNN)** and **Random Forest (RF)** in a cost-sensitive setting:

---

1. Cost Sensitivity in Fraud Detection

- **Imbalanced Data**: Fraudulent transactions usually represent a small fraction of all transactions, leading to imbalanced datasets.
- **High Cost of False Negatives**: Failing to detect a fraudulent transaction can result in significant financial losses.

- **Objective**: Design a system that prioritizes identifying fraud while minimizing false negatives, even at the expense of some false positives.

---

2. K-Nearest Neighbors (KNN)

- **How It Works**: KNN classifies a transaction by comparing it to the closest k neighbors in the feature space.
- **Cost Sensitivity**: Weight neighbors differently based on their label (fraud vs. non-fraud) or use cost-sensitive distance metrics.
- **Advantages**:
  - Simple and interpretable.
  - Effective for smaller datasets.
- **Challenges**:
  - Computationally expensive for large datasets.
  - Sensitive to feature scaling and irrelevant features.

3. Random Forest (RF)

- **How It Works**: RF is an ensemble of decision trees. Each tree is trained on a random subset of the data, and predictions are aggregated through majority voting or averaging.
- **Cost Sensitivity**:
  - Modify the decision threshold to favor fraud detection.
  - Use class weights to penalize misclassifications of fraudulent transactions more heavily.
- **Advantages**:
  - Handles high-dimensional data and feature importance ranking.
  - Robust to overfitting.
- **Challenges**:
  - Can be less interpretable compared to simpler models.

4. Combining KNN and RF for Cost-Sensitive Fraud Detection

- **KNN as a Preprocessor**:
  - Use KNN to identify a subset of suspicious transactions based on proximity to known fraud cases.
  - Pass these suspicious cases to Random Forest for a refined decision.
- **RF with Cost-Sensitive Learning**:
  - Adjust RF's class weights to focus on fraud detection or tune thresholds for cost-sensitive optimization.

---

5. Evaluation Metrics

Standard accuracy is insufficient due to class imbalance. Instead, use:

- **Precision**: How many predicted frauds are actually frauds.
- **Recall**: How many actual frauds are detected.
- **F1-Score**: Balances precision and recall.

- **Cost Metrics**: Quantify the financial impact of false positives and false negatives.

---

6. Practical Application

- **Feature Engineering**: Extract transaction-specific features (e.g., amount, time, location).
- **Scaling & Normalization**: Ensure feature values are comparable, especially for KNN.
- **Real-Time Capability**: Optimize RF and KNN for real-time fraud detection in production systems.

By leveraging the strengths of both KNN and Random Forest, and tailoring the system to the cost-sensitive nature of fraud detection, this approach can effectively reduce financial losses while maintaining high detection accuracy.

4o