Clustering and Graph Representation Learning in Fraud Detection

Yisheng Jiang, Miao Wang

Under the Supervision of Professor Ali Hirsa, and Moffy Jiang from CraiditX

Industrial Engineering and Operations Research
COLUMBIA | ENGINEERING

氪信科技
CRAIDITX

Introduction

In the current financial world, the role of machine learning has become increasingly critical and irreplaceable in solving complex challenges such as fraud detection. Traditional methods, which rely heavily on human manual analysis that will incur errors and inconsistencies, are no longer feasible given the volume and complexity of data. This shift has led to the widespread adoption of advanced predictive modeling techniques that leverage Artificial Intelligence to improve accuracy and reduce human error.

Our project builds on previous efforts to apply **unsupervised learning** techniques, such as dimension reduction and clustering, to fraud detection. In addition to their efforts, we extend the analytical framework to include simple as well as complex classification models, from logistic regression to tree-based methods, to segment the feature space. Lastly, we combine the **Graph Representation** into the clustering.

We also systematized model comparisons including the impact of various classifiers, the treatment of imbalanced datasets, and a baseline comparison of the Graph Representation methods.

Evaluation Metrics

AUROC (Area Under the Receiver Operating Characteristic Curve): A metric used to the model's distinguishability between labels. The metric is calculated by computing the area under the ROC curve which plots the True Positive Rates and False Positive Rates at various thresholds.

KS Score (Kolmogorov-Smirnov): It quantifies the maximum difference between the True Positive Rate (TPR) and the False Positive Rate (FPR) across all possible thresholds. We define KS Score = max(TPR – FPR).

Both metrics require value to be closer to 1 so that the model is more capable of distinguishing between different labels.

Data

The provided raw dataset comes from CraiditX's internal client data and consists of 35,373 loan applications. For each application, its information may be related to another application, suggesting the potential existence of an underlying network from application to application. Our dataset is highly imbalanced, with only 3% of the labels being fraud cases, and 97% as non-fraudulent applications.

1. **[apply_info]** – Base Properties for an Individual Application
2. **[device_info]** – Base Properties for a Device Related to this Application
3. **[contacts_info]** – Contact Book Details on a Device Related to this Application
4. **[calls_info]** – Call Logs Details on a Device Related to this Application

Graph Representation

GraphSAGE (SAmple and AggreGatE)

GraphSAGE is a feature embedding framework for inductive representation learning on large, dynamic graphs. Unlike Graph Neural Networks, which are static, GraphSAGE is designed to generate feature representations inductively. It generates node embeddings by sampling from the node's neighborhood and aggregating the features through an aggregator function. This allows the GraphSAGE to efficiently handle new nodes in evolving graphs without retraining the model.

Algorithm 1: GraphSAGE embedding generation (i.e., forward propagation) algorithm

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$; non-linearity σ; differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, ..., K\}$; neighborhood function $\mathcal{N} : v \to 2^{\mathcal{V}}$

Output : Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{V}$

$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$;
for $k = 1...K$ **do**
  for $v \in \mathcal{V}$ **do**
    $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
    $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$
  end
  $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
end
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$

FI-GRL (Fast Inductive Graph Representation Learning)

FI-GRL is a framework designed to efficiently generate node embeddings by preserving essential graph topological information. It operates in two stages: (1) Use a random projection technique to reduce dimension while maintaining distances between nodes; (2) Extract features through a cost function that optimizes the fit between the original graph and a compressed representation.

Algorithm 2 FI-GRL: Fast Inductive Graph Representation Learning

Input: Graph $\mathcal{G} = (V, E, \mathbf{W})$ with totally n nodes; Unseen node set $\{v_i\}$;Dimension k, approximation ratio ϵ

Output: Low-dimensional vectors $(\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n, \cdots)$

Construct matrix $\mathcal{L} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ for $\mathcal{G}_1$
Construct a $d \times n$ matrix $\mathbf{R}$, whose entries are independently drawn from $N(0, 1)$, where d is $max\{4log(n)/\epsilon^2, k/\epsilon^2\}$
Compute each row of the matrix sketch $\mathbf{M}$, $\mathbf{M}_i = \frac{1}{\sqrt{d}}\mathbf{R}\mathcal{L}_i$, where $\mathcal{L}_i$ denotes ith row of $\mathcal{L}$
Compute k-singular value decomposition $\mathbf{M}_\mathbf{k} = \tilde{\mathbf{U}}_\mathbf{k}\tilde{\mathbf{\Sigma}}_\mathbf{k}\tilde{\mathbf{V}}_k^T$
Compute $\mathbf{Y} = \mathbf{D}^{-\frac{1}{2}}\tilde{\mathbf{U}}_\mathbf{k}$
for all unseen nodes v_j **do**
  Compute $\mathbf{b} = \frac{1}{\sqrt{d}}\mathbf{R}\mathcal{L}_j$
  Compute $\tilde{\mathbf{b}} = 1/\sqrt{\mathbf{D}_{jj}}\mathbf{b}\tilde{\mathbf{V}}_\mathbf{k}\tilde{\mathbf{\Sigma}}_\mathbf{k}^{-1}$
  Append $\tilde{\mathbf{b}}$ as a new row of $\mathbf{Y}$
end for
return $\mathbf{Y}$

Model Comparisons

Classifiers: Adaptive Boosting, Random Forest, and XGBoost Classifier

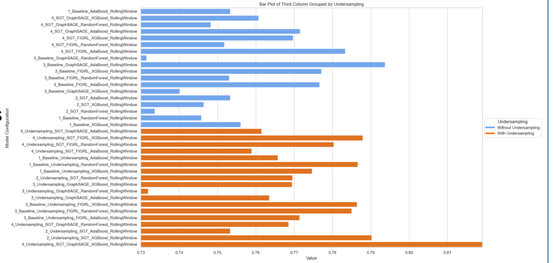
Undersampling: Undersampling the dataset based on a predefined ratio so that the resulting dataset has a more balanced number of labels for both classes.

Sequence Graph Transform (SGT) embedding: For the "calls_info" dataset, the dataset is compressed using SGT which captures temporal and relational patterns, highlighting the connections and timing of the calls.

Complexity: Each additional inclusion of a technique is considered one added complexity of the model, with finally the undersampling applied to all models

Rolling Window: A step size of 2 hours and a duration of 20 hours is applied as the rolling window parameter, to compare the time effect

Model	Raw Data	SGT	GraphSAGE	FI-GRL	UnderSample
Complexity 1	✓				
Complexity 2	✓	✓			
Complexity 3	✓		✓		
Complexity 3	✓			✓	
Complexity 4	✓	✓	✓		
Complexity 4	✓	✓		✓	
Complexity 1-4					✓



- Undersampling generally outperforms the one without
- GraphSAGE, SGT, Undersampling, XGB outperform the rest

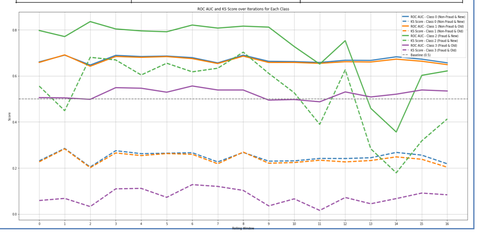
Feature Importance

Conditioning the Feature

The most importance feature is "is_new_client" which suggests whether the application is from a new client or not is crucial in our model. Conditioning the label, fraud and non-fraud, on the "is_new_client" indicator, we result in four labels:

1. For all Non-fraud cases, the model is stable, and performed well (AUROC around 0.7)
2. For Old Client & Fraud Cases, the model performed bad (AUROC around 0.5)
3. For New Client & Fraud Cases, the model fluctuates, suggesting the most prominent fraud is from New Customers

	New Client	Old Client
Fraud	New & Fraud	Old & Fraud
Non-Fraud	New & Non-Fraud	Old & Non-Fraud



Clustering and Dimension Reduction

K-Means Clustering – A clustering algorithm that partitions a dataset into K distinct clusters by minimizing the within-cluster sum of squares, iteratively refining the cluster centroids until convergence.

Spectral Clustering – Spectral Clustering uses the eigenvalues of a similarity matrix derived from the data to perform dimensionality reduction before clustering. The algorithm maps data points to a lower-dimensional space where it applies a standard clustering method like K-means, allowing it to identify clusters that may not be well-separated in the original space.

PCA – A linear dimensionality reduction technique that transforms the data into a new coordinate system where the greatest variation in the data expressed by the variances lie on the first few few coordinates, known as **Principal Components (PC)**. It determines the coordinates by iteratively solving for the next direction capturing the most variance

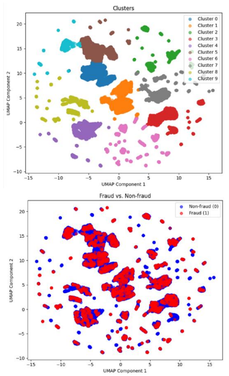
Kernel-PCA – kPCA applies the kernel trick to PCA, allowing it to capture non-linear relationships. The algorithm maps the original data into a higher-dimensional space using a kernel function and then performs PCA in this new space, enabling it to identify complex structures in the data.

UMAP – UMAP constructs a high-dimensional graph of the data and then optimizes a low-dimensional graph that preserves both local and global structures. It is designed to maintain the topological properties of the data while reducing dimensionality.

Laplacian Eigenmaps (LE) – LE creates a graph where the nodes represent data points and edges represent similarities. The algorithm then computes the eigenvectors of the Laplacian matrix of this graph, mapping the data to a lower-dimensional space that preserves the local neighborhood structure.

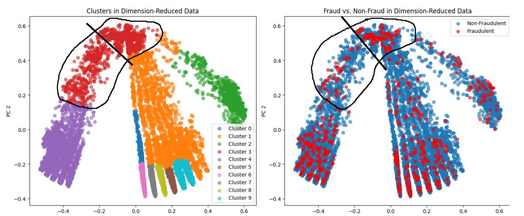
t-SNE – t-SNE reduces dimensionality by converting the similarities between data points into probabilities. It then attempts to minimize the difference between these probabilities in high-dimensional and low-dimensional spaces, effectively clustering similar data points together in the lower-dimensional space.

Model	AUROC	KS
K-Means++ only	0.544	0.081
K-Means++ with PCA (3 PC)	0.553	0.087
K-Means++ with kPCA (3 PC)	0.510	0.024
K-Means++ with t-SNE (3 PC)	0.550	0.086
K-Means++ with UMAP (2 PC)	0.538	0.066
K-Means++ with LE (7 PC)	0.610	0.1777



- Reduced dataset using UMAP is showing the labels are mixed significantly
- K-Means clustering is giving a slightly better than random guessing results

Spectral Clustering



Spectral Clustering Performance – Spectral clustering on dimension-reduced data (using kPCA) is performing better than K-Means in terms of classification results.

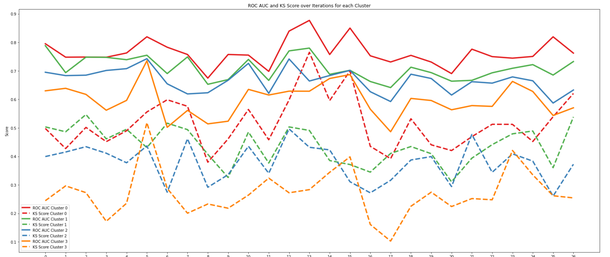
Cluster-Specific Performance – The table highlights the AUROC and KS scores for each cluster, indicating varying levels of model performance. Cluster 3 shows the highest AUROC (0.8245) and KS (0.5123), suggesting strong separability in this cluster.

Model	AUROC	KS	Model	AUROC	KS
Cluster 0	0.785	0.6	Cluster 5	0.5129	0.2059
Cluster 1	0.576	0.2045	Cluster 6	0.6185	0.2852
Cluster 2	0.6347	0.3124	Cluster 7	0.6276	0.3469
Cluster 3	0.8245	0.5123	Cluster 8	0.6007	0.3125
Cluster 4	0.6527	0.2713	Cluster 9	0.5336	0.2997

Further Investigation – Other clustering techniques may be more appropriate, and potentially projecting the data into higher dimension may help separate the data better.

Clustering with Graph Embeddings

Using k=4 clusters with K-Means++, corresponding to the conditioned 4 labels, a clustering is applied into the GraphSAGE embeddings with an XGBoost classifier.



The overall trend suggests that all clusters are moving in the same direction, suggest the underlying distribution remains similar.
The clustering segmented the data into 4 groups, and each group may suggest it is tailored to capture specific patterns and characteristics within the group, making it more precise.