# A Framework for Advanced Fraud Detection Using Causal-Informed Graph Neural Networks and MLOps

## Abstract

This paper presents a comprehensive framework for building a sophisticated, production-ready fraud detection system that moves beyond traditional predictive modeling. By integrating **Graph Neural Networks (GNNs)** for relational data analysis, **Causal Inference** for strategic decision-making, and **Explainable AI (XAI)** for model transparency, this system addresses the complex, interconnected nature of modern financial fraud. The entire project is structured around a robust **MLOps (Machine Learning Operations)** lifecycle, ensuring reproducibility, scalability, and maintainability from initial data exploration to a containerized API deployment. This paper details the theoretical underpinnings of each component, the practical workflow for implementation, and the strategic advantages of such a holistic approach.

## 1. Introduction

Financial fraud is a dynamic and adaptive challenge. Fraudsters rarely act in isolation, often forming sophisticated rings that exploit relationships between users, devices, and payment instruments. Traditional machine learning models, which treat transactions as independent events, struggle to capture these complex, networked behaviors. This limitation necessitates a paradigm shift toward models that can natively understand and reason over relational data.

This project addresses this gap by proposing a system built on three advanced ML pillars:

1. **Graph Neural Networks (GNNs):** To model the transaction ecosystem as a graph, enabling the system to learn from the rich web of connections and identify fraudulent patterns that are invisible to tabular models.
2. **Explainable AI (XAI):** To provide transparent, human-interpretable reasons for a model's predictions, a critical requirement for regulatory compliance and for providing actionable insights to fraud investigators.
3. **Causal Inference (Causal ML):** To elevate the system from a predictive tool to a strategic one. Instead of only asking "Is this transaction fraudulent?", we can answer "What would be the impact on fraud rates if we implemented a new security measure?".

This paper documents the complete workflow, from initial setup to the deployment of a production-grade API, providing a blueprint for developing advanced, high-impact machine learning systems.

## 2. Core Methodologies and Technologies

## 2.1. Graph Neural Networks (GNNs)

**Why GNNs?** Financial transactions are not independent. A single credit card can be used by multiple users, and a single user can have multiple cards. These relationships form a natural graph structure. GNNs are a class of deep learning models designed specifically to operate on graph data. They work by passing messages between connected nodes, allowing each node's representation (or "embedding") to be enriched by the features of its neighbors.

**How They Were Utilized:** In this project, we constructed a graph where each transaction was a **node**. We then created **edges** between transactions that shared key identifiers, such as the same `card_id` or `email_domain`. This explicitly links potentially collusive activities. We implemented a **GraphSAGE** model, a popular inductive GNN architecture, which learns to aggregate feature information from a node's local neighborhood. This allows the model to learn powerful representations that capture not just the transaction's own features but also the "context" of its surrounding transactions, making it highly effective at detecting fraud rings.

## 2.2. MLOps: The Production Backbone

MLOps is the practice of applying DevOps principles to machine learning workflows. It is essential for building reliable and scalable systems.

**Key Components Utilized:**

- **Feature Store (Feast):** A feature store acts as a central repository for feature data. By defining our features in Feast, we ensured consistency between the data used for training our baseline model and the data that would be used for live inference. This solves the critical problem of train-serve skew.
- **Experiment Tracking (MLflow):** Machine learning is an iterative process. We used MLflow to meticulously track every experiment, including model parameters, performance metrics (AUPRC, ROC AUC), and the model artifacts themselves. This provided a complete, auditable history of our model development, allowing us to compare the GNN's performance directly against the XGBoost baseline.
- **Containerization (Docker):** To ensure our application could run reliably in any environment, we packaged it into a Docker container. The `Dockerfile` specifies the operating system, dependencies, and code, creating a portable and self-contained image. This eliminates "it works on my machine" problems and is a standard practice for deploying microservices.
- **API Deployment (FastAPI):** A model is only useful if it can be accessed. We used FastAPI to create a high-performance, asynchronous REST API. This API exposes our trained GNN, allowing other services to send a transaction's node ID and receive a real-time fraud probability in return.

## 2.3. Explainable AI (XAI)

As models become more complex, their "black box" nature becomes a liability. XAI techniques aim to make model decisions understandable.

**How It Was Utilized:** We implemented **GNNExplainer**, a specific XAI technique for graph models. For a given prediction (e.g., flagging a transaction as fraudulent), GNNExplainer identifies the most influential components of the computation:

- **The critical subgraph:** The specific neighboring nodes (other transactions) that most contributed to the prediction.
- **The key node features:** The specific attributes of the transaction and its neighbors that were most important.

This provides invaluable, actionable intelligence. Instead of just knowing a transaction is risky, a fraud analyst can see the exact cluster of related transactions that triggered the alert, immediately pointing them to a potential fraud ring.

### 2.4. Causal Inference

Causal Inference moves beyond correlation to estimate the cause-and-effect relationships in data.

**Why It's Advanced:** While our GNN can predict risk, it cannot, by itself, tell us what would happen if we intervened. For example, would forcing multi-factor authentication (MFA) on a group of users actually reduce fraud, or would it just shift fraudulent activity elsewhere?

**How It Was Utilized:** We used the powerful node embeddings generated by our trained GNN as features for a causal model. Specifically, we used the **Double Machine Learning (DML)** technique from Microsoft's EconML library. We simulated a hypothetical intervention (a new security check) and used DML to estimate its **Average Treatment Effect (ATE)** on the fraud rate. The GNN embeddings served as rich "confounders" in the DML model, allowing it to control for the complex factors that influence both the treatment and the outcome, leading to a more reliable causal estimate. This provides a data-driven way to evaluate the potential ROI of business decisions before they are implemented.

## 3. Project Workflow and Structure

The project was executed in five distinct phases, organized within a professional file structure to ensure modularity and clarity.

1. **Phase 1: Data Exploration & Setup:** Initial analysis of the dataset, handling of missing values, and visualization of the critical class imbalance.
2. **Phase 2: Feature Store & Baseline:** An XGBoost model was trained on features served by a Feast feature store. This established a strong, reproducible performance baseline tracked with MLflow.
3. **Phase 3: Graph Construction & GNN Training:** The core of the project, where tabular data was transformed into a graph and a GraphSAGE model was trained to outperform the baseline.
4. **Phase 4: Causal Analysis & Explainability:** The trained GNN was used to generate explanations with GNNExplainer and to estimate causal effects with DML.
5. **Phase 5: API Deployment & Containerization:** The final model was packaged into a Docker container and exposed via a FastAPI REST API for real-time inference.

## 4. Conclusion and Future Work

This project successfully demonstrates a modern, end-to-end workflow for building an advanced fraud detection system. By combining the predictive power of GNNs with the strategic insights from Causal ML and the transparency of XAI, all within a robust MLOps framework, we have created a system that is not only powerful but also reliable, interpretable, and ready for production.

Future work could involve exploring more complex heterogeneous graphs (with different node types like users, merchants, and devices), experimenting with different GNN architectures like Graph Attention Networks (GAT), and automating the entire pipeline with CI/CD for continuous training and deployment.