Project Proposal

Graph Neural Networks (GNNs) Learning Path & Fraud Detection System

Mentors : Kavin, Soham Panchal and Ayush Inamdar

Objective:

In this project we would be teaching through the complete evolution of machine learning models. Starting with classical ML, it progresses to deep learning, culminating in Graph Neural Networks (GNNs).

Final Project - Brief:

The final project focuses on detecting fraudulent accounts and transactions within a financial network. By modeling accounts as nodes and money transfers as edges, a graph structure is created that reveals hidden fraud rings and multi-hop laundering patterns. A GNN-based model is then developed to uncover relational anomalies that classical models cannot detect.

Timeline:

1. **Foundations of Machine Learning**
    - Understand supervised learning and evaluation metrics.
    - Implement Logistic Regression, Decision Trees, Random Forests, and XGBoost.
    - Build a baseline fraud classifier on tabular data.
2. **Introduction to Deep Learning & CNNs**
    - Learn convolution, feature maps, and hierarchical feature extraction.
    - Train a CNN on MNIST/CIFAR to understand feature learning.
    - Identify limitations of CNNs on non-grid relational data.
3. **Understanding Graph Structures**
    - Learn nodes, edges, adjacency matrices, and graph connectivity.
    - Study directed, undirected, and heterogeneous graphs.
    - Construct and visualize sample graphs using NetworkX.
4. **Message Passing & GNN Fundamentals**
    - Study the Message Passing Neural Network (MPNN) framework.
    - Understand neighborhood aggregation and multi-hop context.
    - Learn why GNNs are ideal for relational tasks like fraud detection.
5. **Implementing GNN Architectures**
    - Implement GCN, GraphSAGE, and GAT using PyTorch Geometric.
    - Train and evaluate node classification models on citation datasets.
    - Compare inductive vs transductive learning approaches.
6. **Constructing the Fraud Detection Graph**

- Convert transactional data into nodes (accounts) and edges (transfers).
- Engineer node/edge features such as amount, frequency, and recency.
- Apply temporal train/validation splits to avoid leakage.

7. **Final Fraud Detection System**
   - Train GraphSAGE/GAT models for fraud classification.
   - Evaluate using ROC-AUC, PR-AUC, and Recall@K.
   - Identify suspicious patterns such as fraud rings and money flows.