네트워크 분석 기반

자금세탁·금융사기 의심거래 탐지 모델 개발

목차

네트워크 분석 기반 자금세탁·금융사기 의심거래 탐지 모델 개발 프로젝트

프로젝트 배경

- **기존 이상금융거래 탐지 방식: Rule 기반**
  - 사기 가능성이 높은 경우를 룰로 설정한 뒤 필요 시 업데이트
  - ex) 500만원 이상 현금 인출, 1일 이내 50회 이상 거래

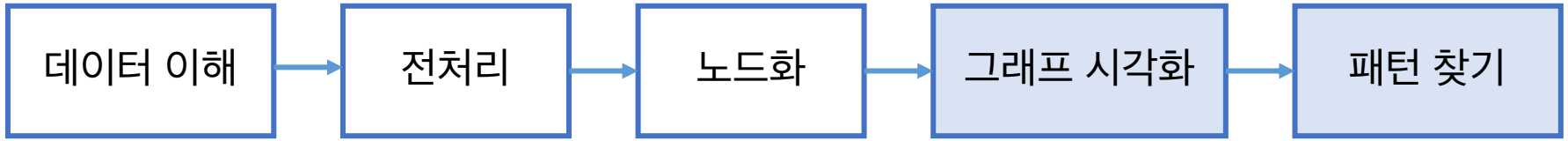
- **한계**
  - 빠르게 진화하는 자금세탁 및 사기 수법으로 효과적 이상금융거래 탐지에 한계
  - 이상금융거래가 조직화되고 있어 단일 거래 내역에 초점을 맞춘 룰 기반 방식으로는 관계 파악이 어려움

사기계좌와 정상계좌의 관계적 특성과 패턴을 파악할 수 있는 네트워크 분석 기반 모형 개발 필요

프로젝트 목표

- **목표 1: 네트워크 분석을 통한 시각화 및 패턴 발굴**

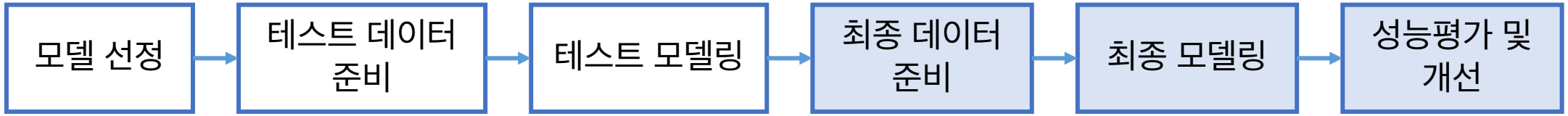
  NetworkX 패키지를 이용해 거래 네트워크 시각화 후 이상금융거래 패턴 발굴



NetworkX
Network Analysis in Python

- **목표 2: 이상금융거래 탐지 모델 개발**

  GNN(Graph Neural Network)을 기반으로 사기계좌와 정상계좌를 분류하는 모델 개발

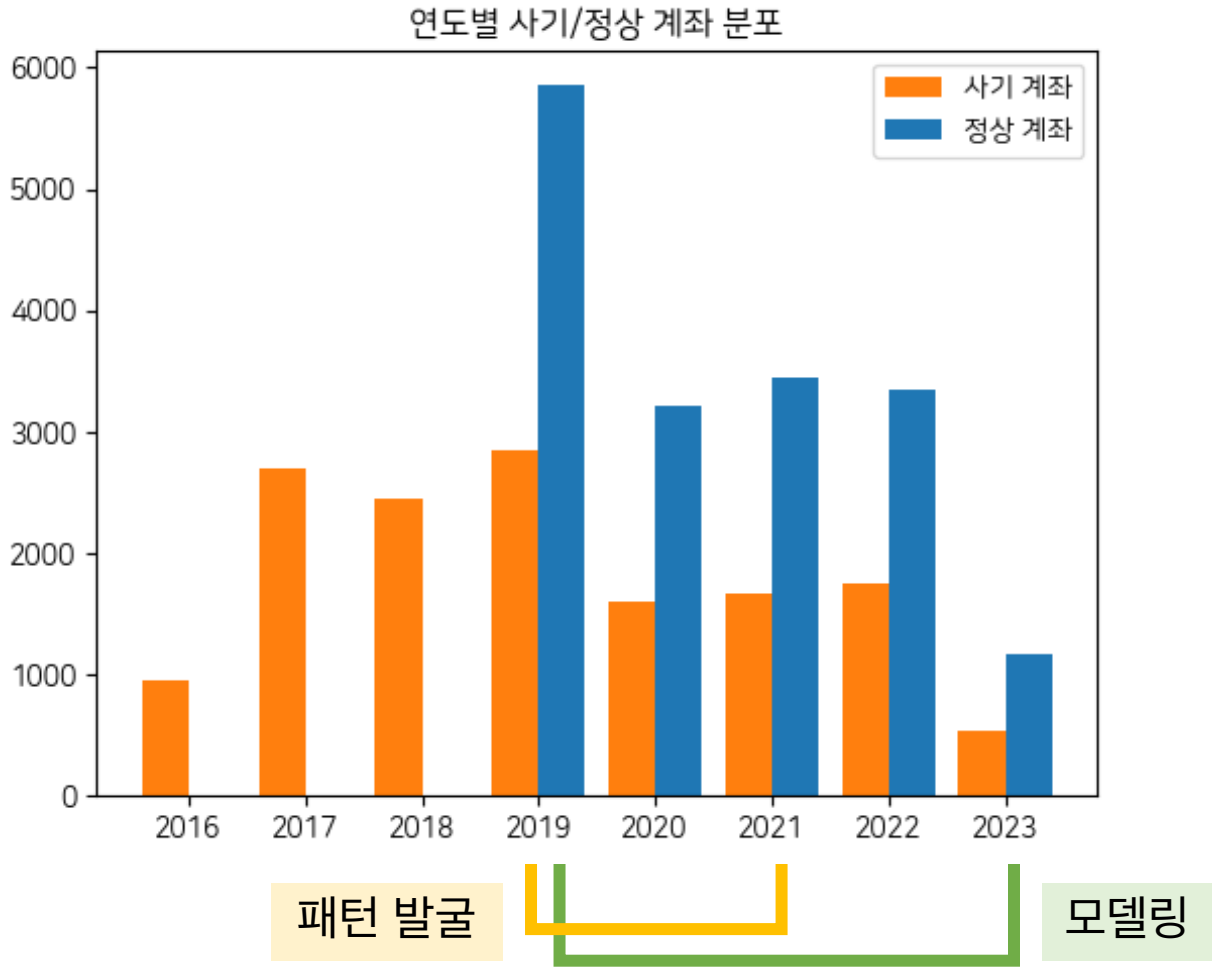


DGL DEEP GRAPH LIBRARY

PyTorch

데이터 이해

전체 데이터셋의 연도별 분포



	시드 사기 계좌	시드 정상 계좌
계좌 수	약 14000개	약 17000개
기간	16년~23년	19년~23년

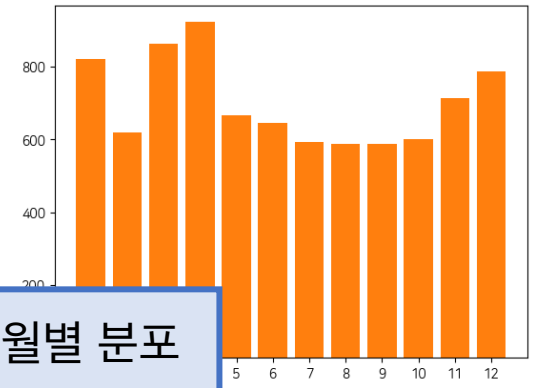
- 시드 계좌에서 퍼져 나가는 Depth 2까지의 거래내역 데이터 사용

데이터 이해

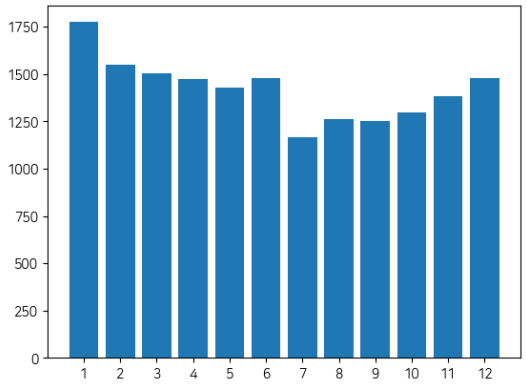
정상 계좌 데이터 요청 시, 아래와 같이 사기 계좌 데이터와 최대한 유사한 분포를 갖도록 추출 요청

월별 분포

사기 계좌

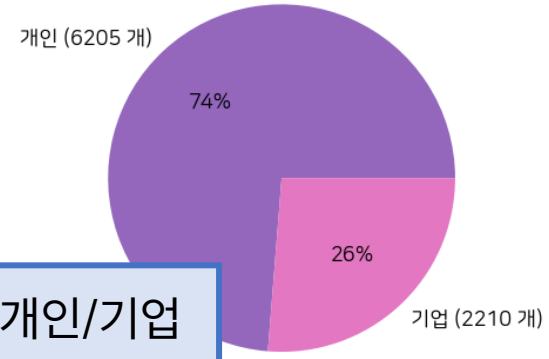


정상 계좌

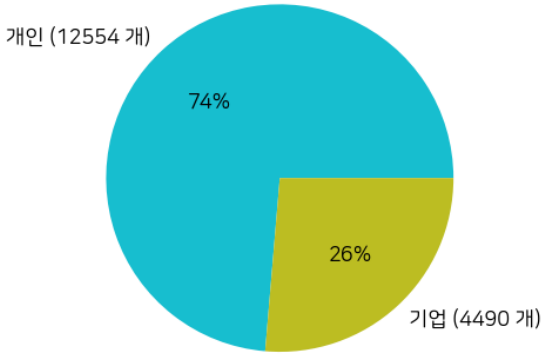


개인/기업

사기 계좌

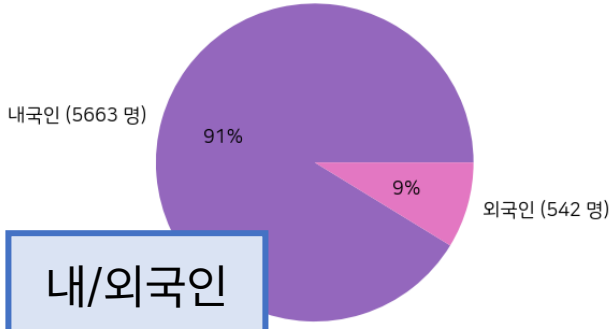


정상 계좌

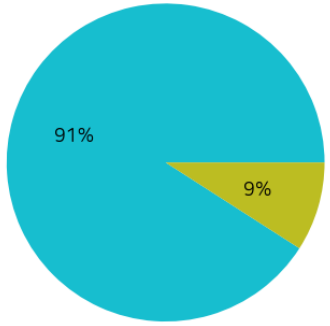


내/외국인

사기 계좌

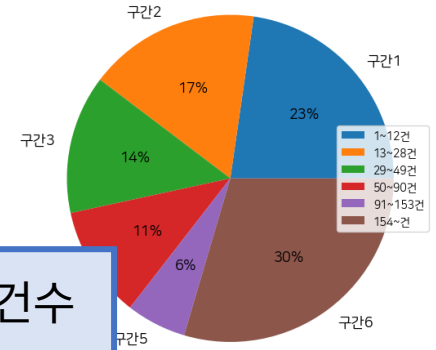


정상 계좌

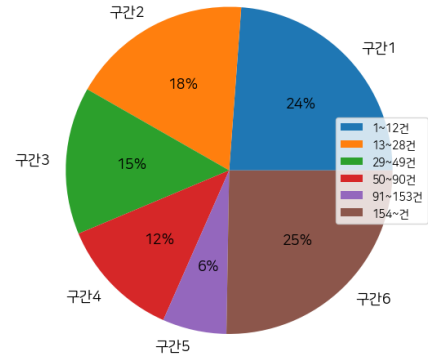


거래 건수

사기 계좌



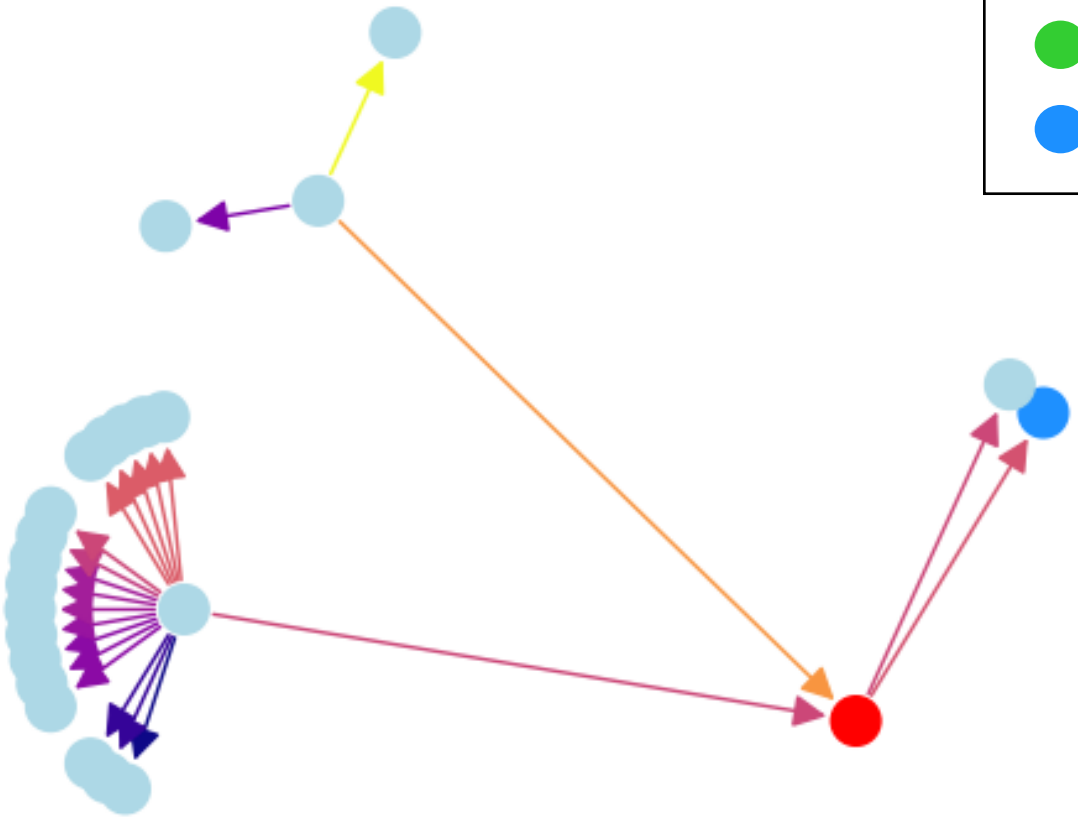
정상 계좌



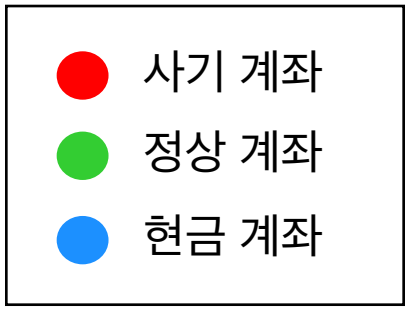
네트워크 그래프 시각화

사기 네트워크 그래프

CASE_ID : 103

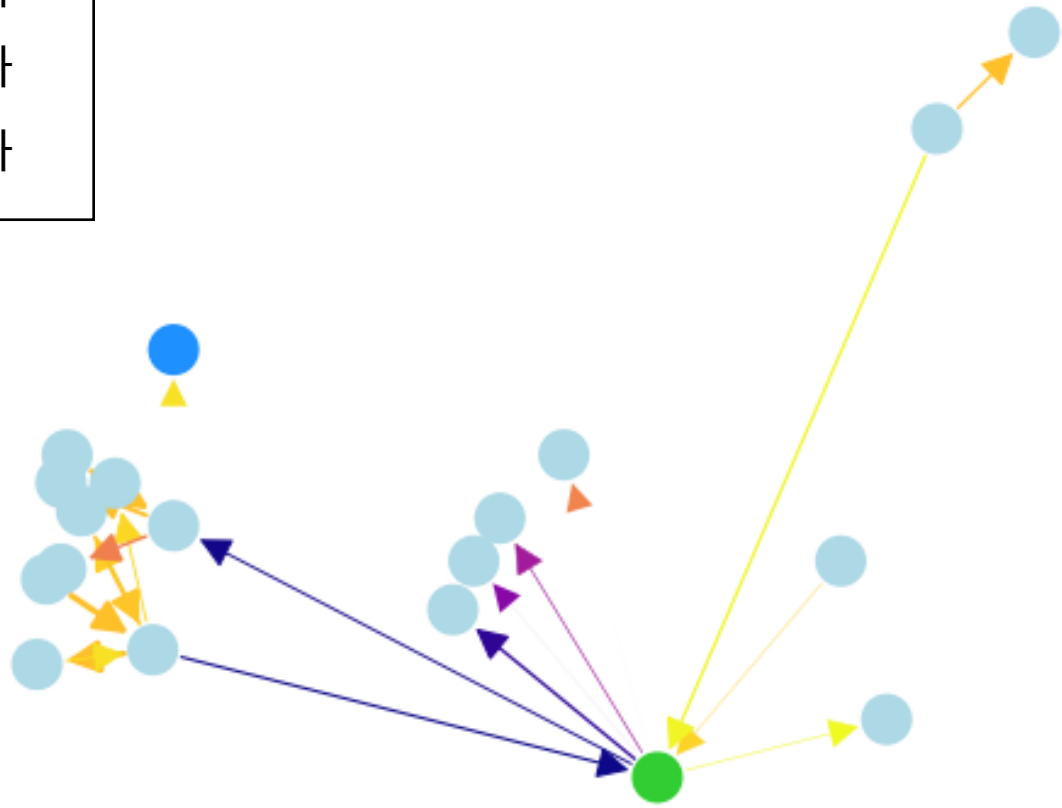


최초 거래 최종 거래



정상 네트워크 그래프

CASE_ID : 100710



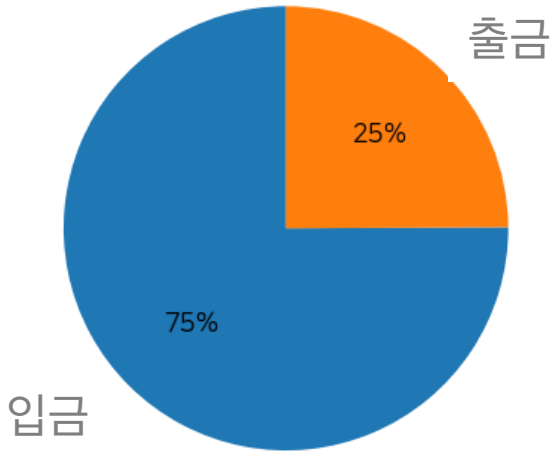
최초 거래 최종 거래

패턴 발굴 - 1. 가설 수립 및 검증

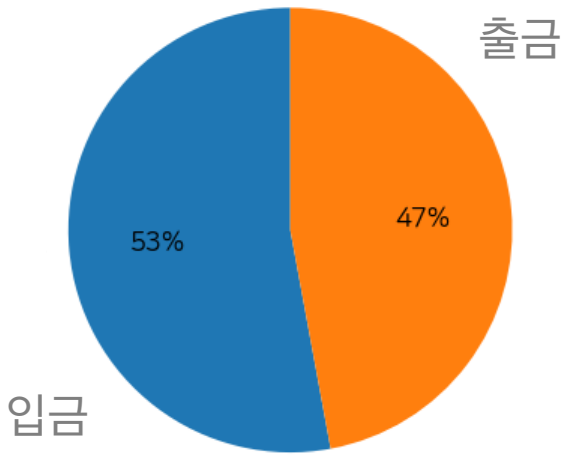
가설 1. 사기 계좌가 정상 계좌보다 입금 거래 비율이 높다.

- 사기 계좌는 정상 계좌보다 집금 계좌의 성격을 갖는 경우가 많기 때문인 것으로 나타남

사기 계좌



정상 계좌



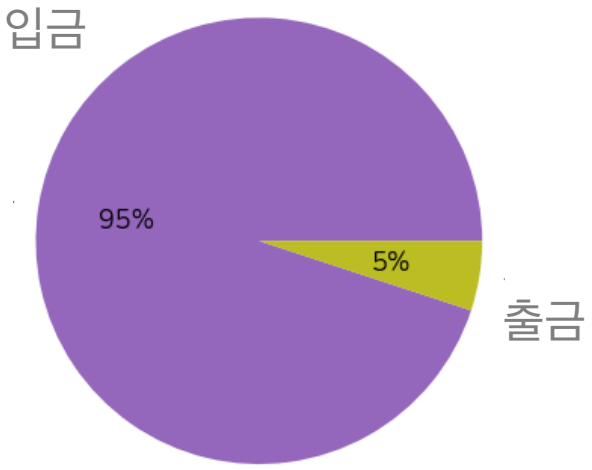
전체 거래 내역 중 입출금 거래 비율		
	입금	**출금**
사기 계좌	782,163 (75%)	259,633 (25%)
정상 계좌	1,452,784 (53%)	1,292,934 (47%)

패턴 발굴 - 1. 가설 수립 및 검증

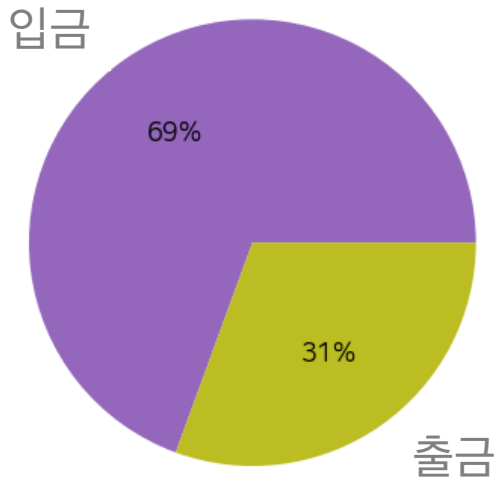
가설 1. 사기 계좌가 정상 계좌보다 입금 거래 비율이 높다.

- 특히 현금 거래 중에서는 사기 계좌 거래 내역의 대부분이 입금 거래

사기 계좌



정상 계좌

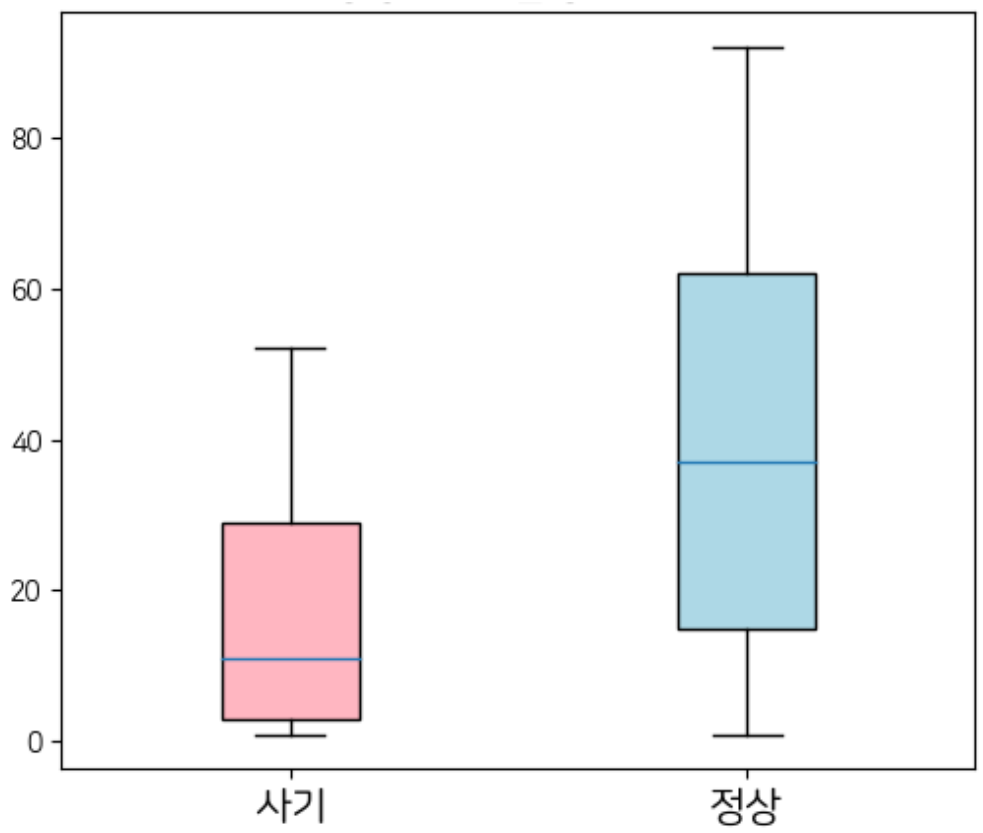


현금 거래 내역 중 입출금 거래 비율			
	현금 입금	**현금 출금**	**총 거래 건수**
사기 계좌	52,708 (95%)	2,800 (5%)	1,041,796
정상 계좌	8,690 (69%)	3,828 (31%)	2,745,718

패턴 발굴 - 1. 가설 수립 및 검증

가설 2. 사기 계좌의 거래 지속 기간이 정상 계좌보다 짧다.

- 사기 계좌는 단기간에 뚜렷한 사기 의도를 가지고 거래를 하는 경향을 보임

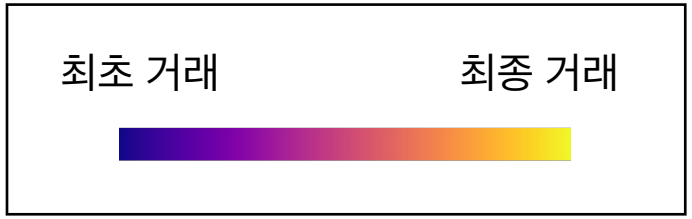


거래 내역별 지속 기간 비교		
	사기 계좌	**정상 계좌**
count	531,529	1,012,748
mean	19.02	40.50
std	20.33	27.42
min	1	1
25%	3	15
50%	11	37
75%	29	62
max	92	92

패턴 발굴 - 2. 의심 그룹 탐구

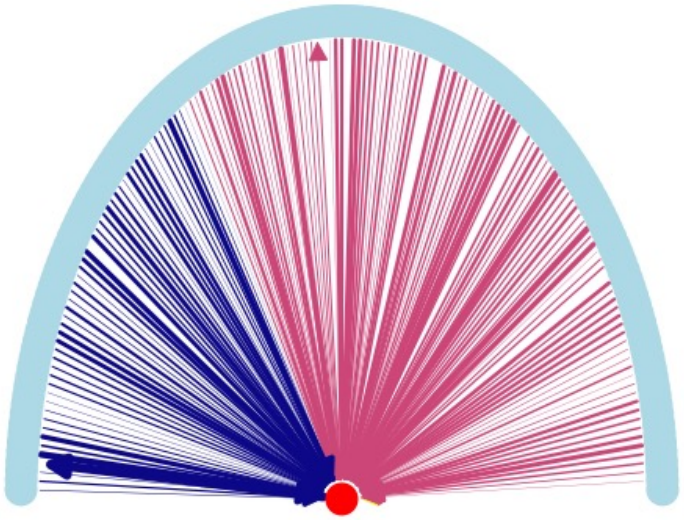
의심 그룹: 단기간에 다수의 외국인 거래가 이루어진 그룹

- 거래 건수에 비해 영어 적요가 많고, 모든 거래가 단기간에 이루어진 경우 사기일 가능성이 높다.
- 검증 기준: 영어 적요 거래 내역 수 / (거래 지속 기간 * 총 거래 건수) > 0.01

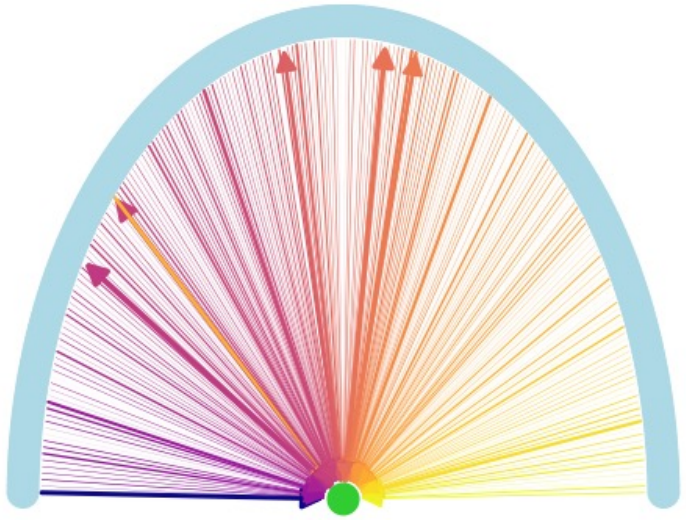


검증 기준을 만족하는 계좌 수	
사기 계좌	**정상 계좌**
128개	18개

사기 케이스

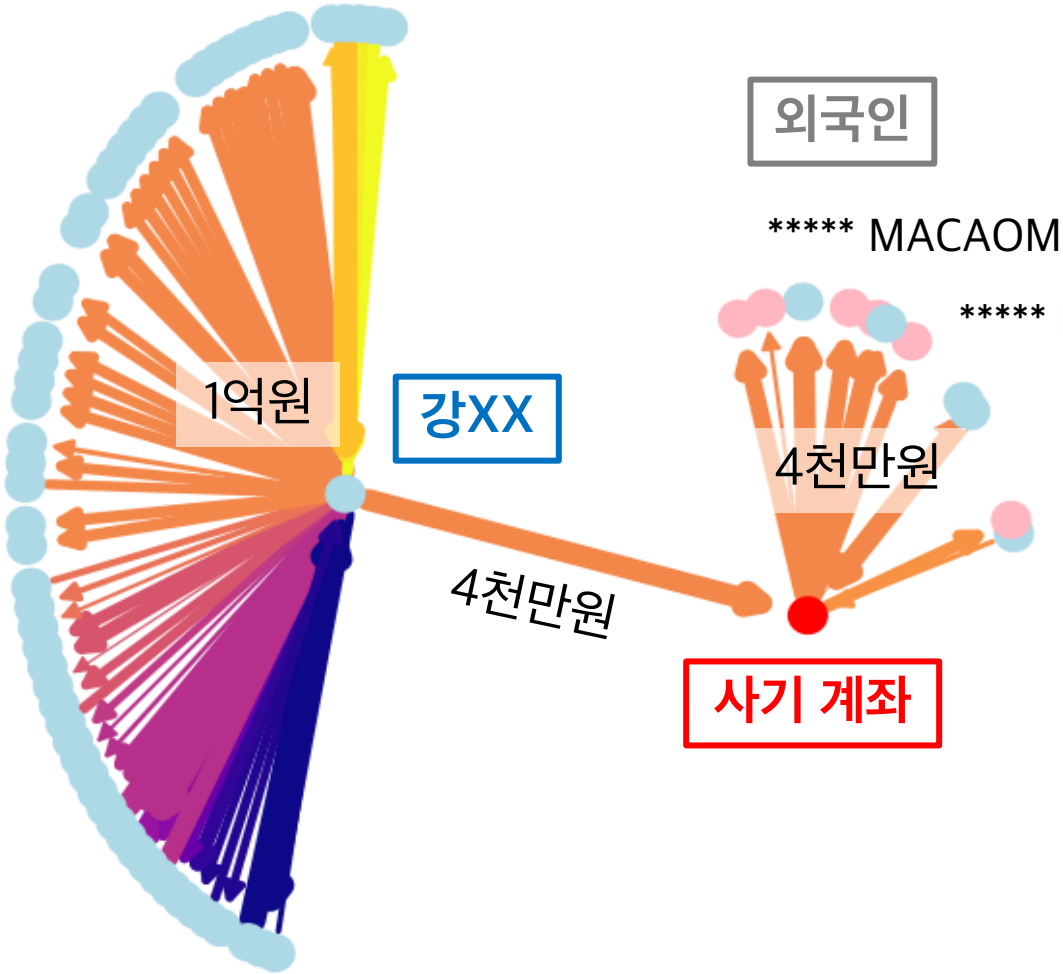


정상 케이스



패턴 발굴 - 2. 의심 그룹 탐구

의심 그룹: 단기간에 다수의 외국인 거래가 이루어진 그룹

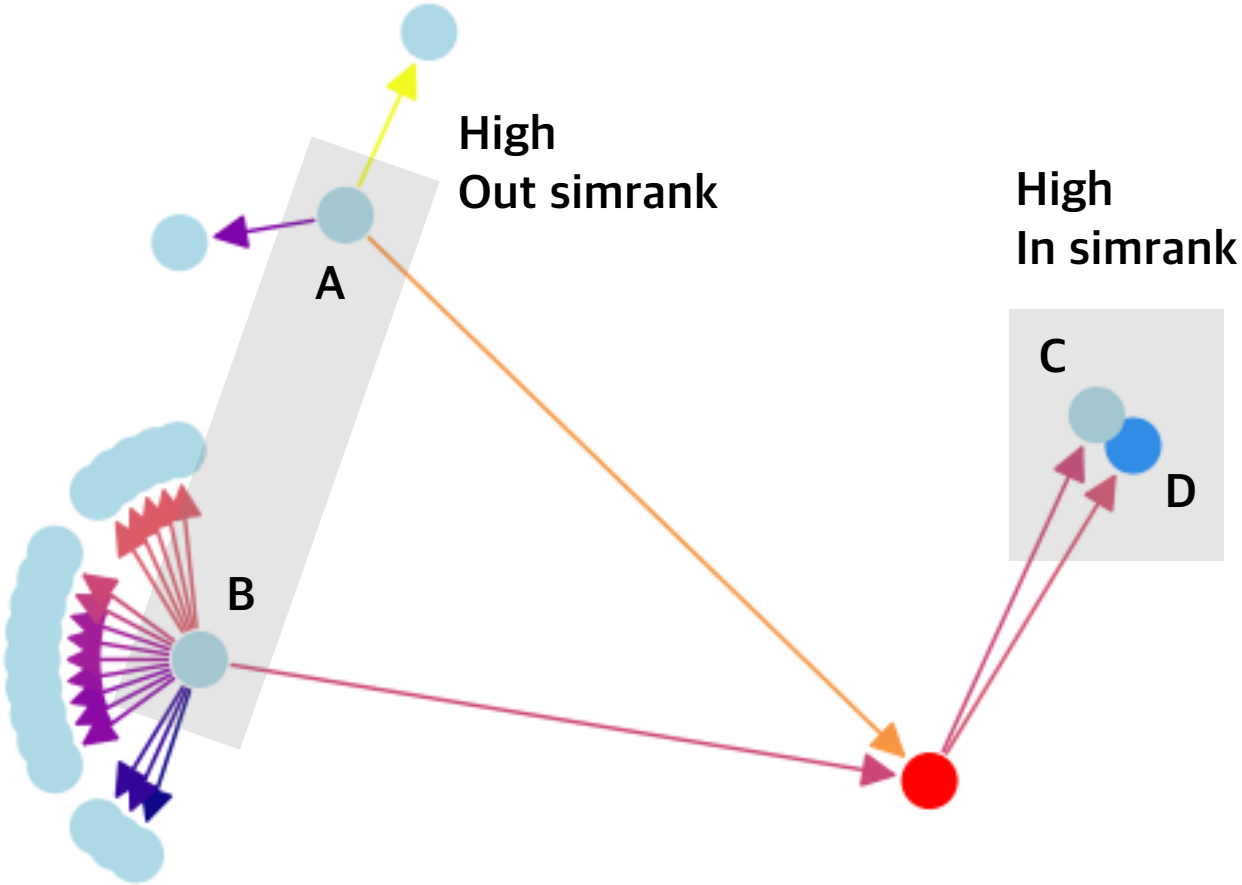


- **강XX 계좌에서 집금한 금액 일부를 사기 계좌로 송금 후 같은 날 여러 개의 외국인 계좌로 분산하여 그대로 송금**
- **해외의 유령 회사를 통해 자금 세탁을 시도한 것으로 보임**
- **강XX 계좌는 아직 적발되지 않은 사기 계좌로 예상됨**

패턴 발굴 - 3. 네트워크 분석 지표 활용

Simrank similarity 를 활용한 구조적 등위성 분석

- Simrank: Two objects are considered to be similar if they are **referenced by similar objects**



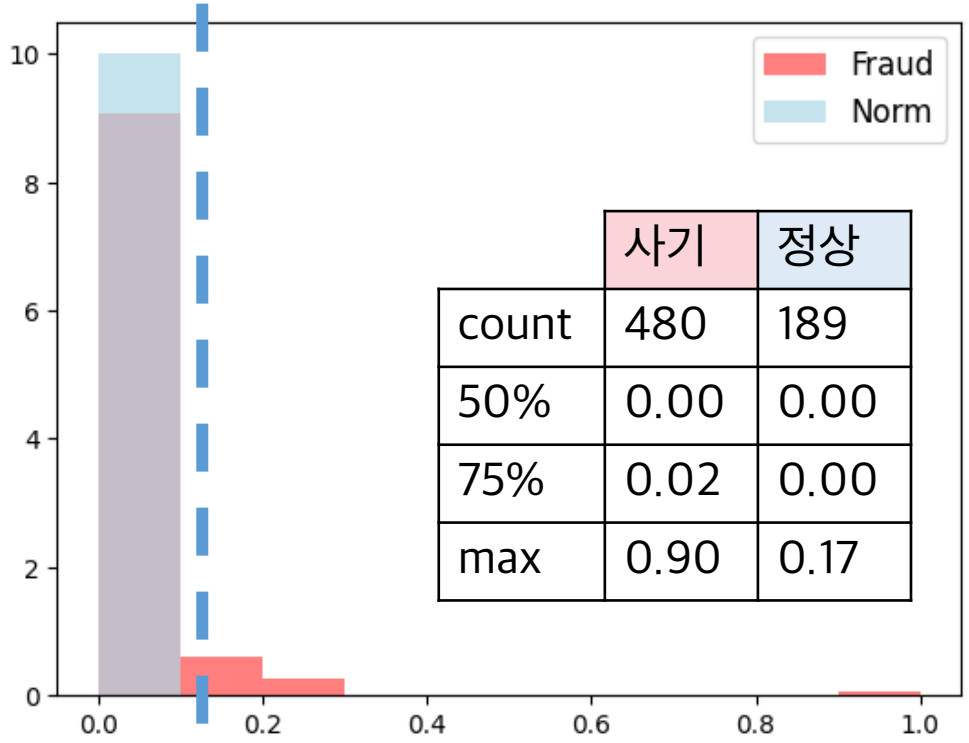
- 조직적으로 이루어지는 사기 네트워크 내에서 집금 등 동일한 역할을 수행하는 계좌 파악 가능

패턴 발굴 - 3. 네트워크 분석 지표 활용

Simrank similarity 를 활용한 구조적 등위성 분석

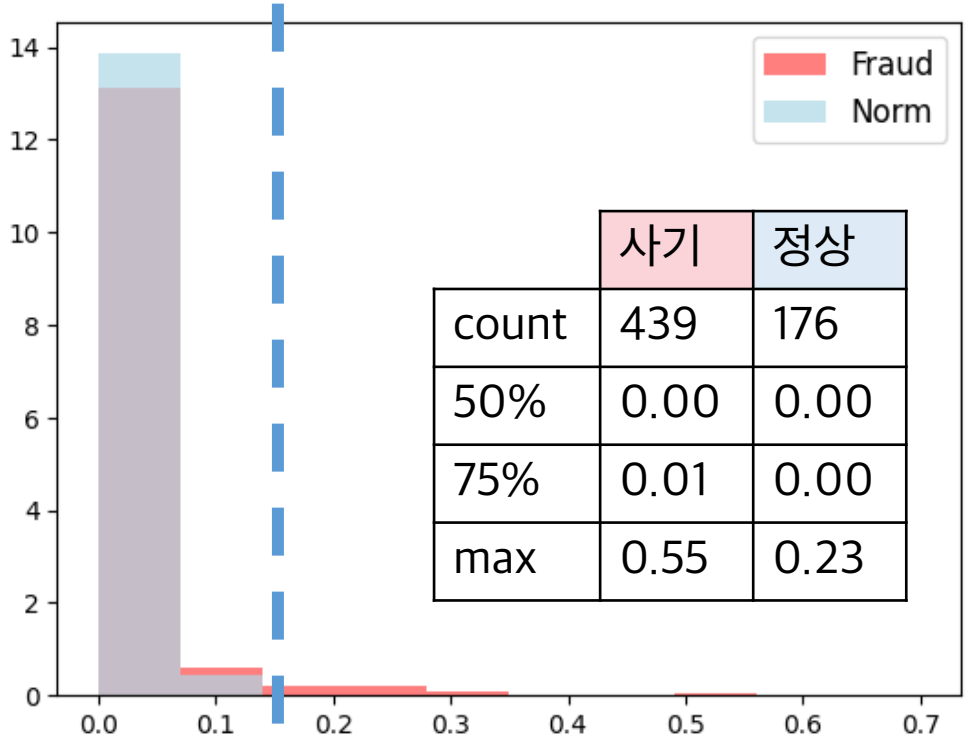
- 사기-정상 계좌 간 simrank 범위를 초과하면 사기일 가능성이 높은 계좌로 해석 가능

[In simrank]



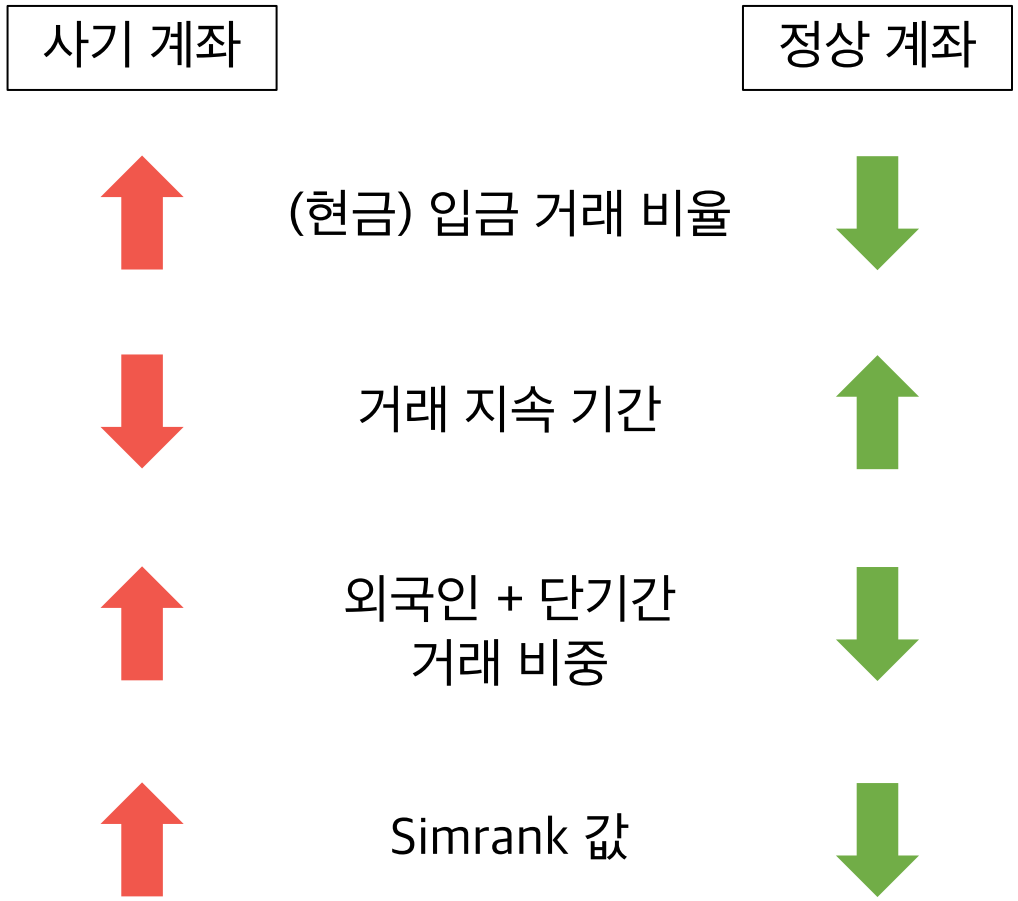
	사기	정상
count	480	189
50%	0.00	0.00
75%	0.02	0.00
max	0.90	0.17

[Out simrank]



	사기	정상
count	439	176
50%	0.00	0.00
75%	0.01	0.00
max	0.55	0.23

패턴 발굴 - 4. 인사이트

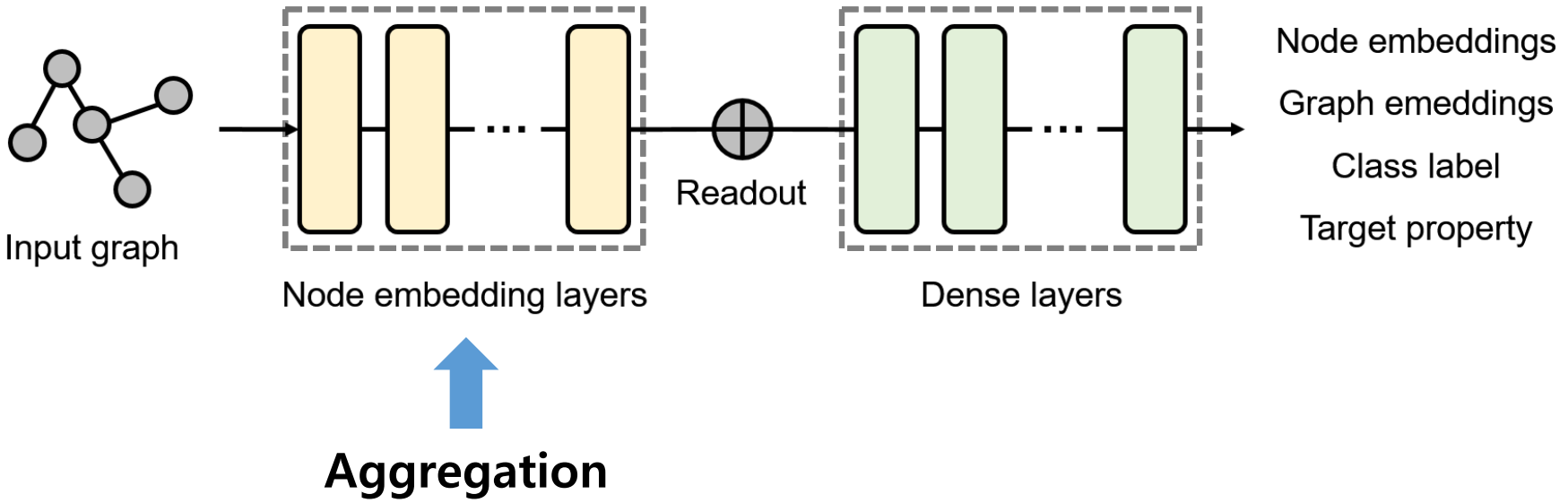


- **패턴 발굴 인사이트를 바탕으로 모델링 과정에서 거래 기간, 현금 거래 관련 변수를 Feature로 추가함**

모델 개요 - 1. GNN

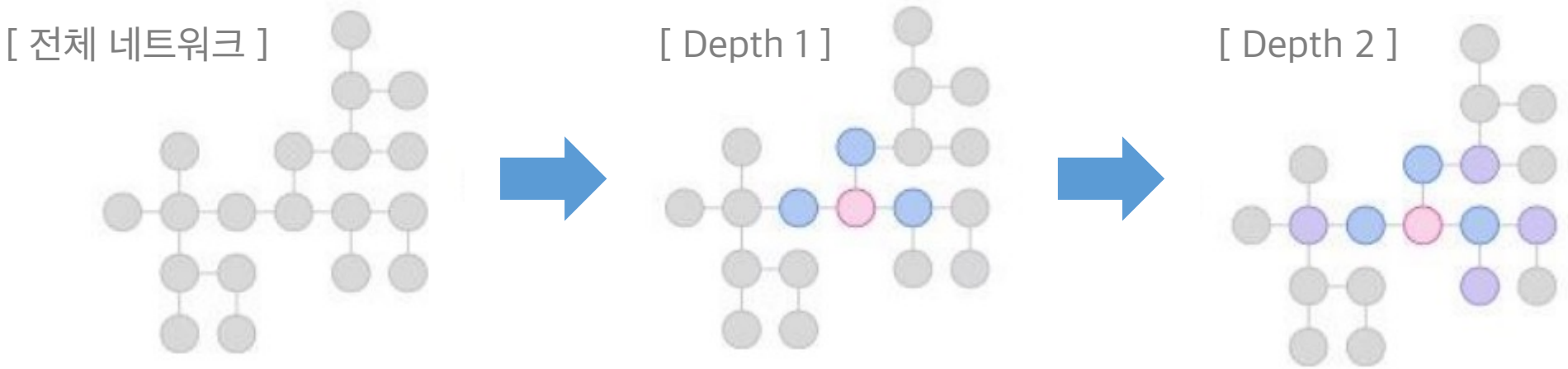
- **GNN**
  - 그래프 데이터를 neural network 로 접근하는 방법
  - 노드의 특성을 학습한 뒤 이를 임베딩하여 다양한 작업을 수행 (노드 분류, 엣지 예측, 그래프 분류)
  - 그래프는 인접행렬이 달라도 동일한 그래프일 수 있으므로 정보들을 취합하는 Aggregation 과정이 필요

[GNN 기본 구조]



모델 개요 - 2. EdgeGATConv

- **EdgeGATConv 기반 그래프 분류**
  - 계좌 정보인 Node Feature와 거래 내역 정보인 Edge Feature를 모두 사용해야 함 → EdgeGATConv
  - Convolution Layer를 하나씩 쌓을 때마다 각 노드별로 이웃 노드들의 정보를 취합하는 방식
  - 사기계좌의 Depth 2 거래 내역 데이터를 활용하기 때문에 EdgeGATConv Layer를 2번 쌓은 후,
    최종적으로 선형 레이어를 통과시켜 그래프에 대한 값 추출



최종 모델 - 1. 데이터셋 준비

모델링에 사용할 노드, 엣지, 그래프 데이터를 각각 별도의 파일로 저장

[node data]

	graph_id	node_id	feat
0	276	0	1.0,0.0,0.0,0.0
1	504	1	1.0,0.0,0.0,0.0
2	1066	2	0.0,1.0,1.0,0.0
3	391	3	1.0,0.0,0.0,0.0
4	327	4	1.0,0.0,0.0,0.0

In_degree, Out_degree,
현금성 여부, 활성화 기간

[edge data]

	dst_id	src_id	graph_id	feat
0	221425	3953389	0	13.815510557964275,1.0,13.815510557964275,0.0
1	5241618	4901865	1	16.472267464678932,4.0,16.472267464678932,74.0
2	1132852	3893086	2	15.761420707019589,3.0,15.761420707019589,57.0
3	3471020	634827	3	11.512925464970229,1.0,11.512925464970229,0.0
4	4420367	6206783	4	14.478198531039512,3.0,14.478198531039512,41.0

총 거래 금액, 총 거래 건수,
현금 거래 금액, 거래 지속 기간

[graph data]

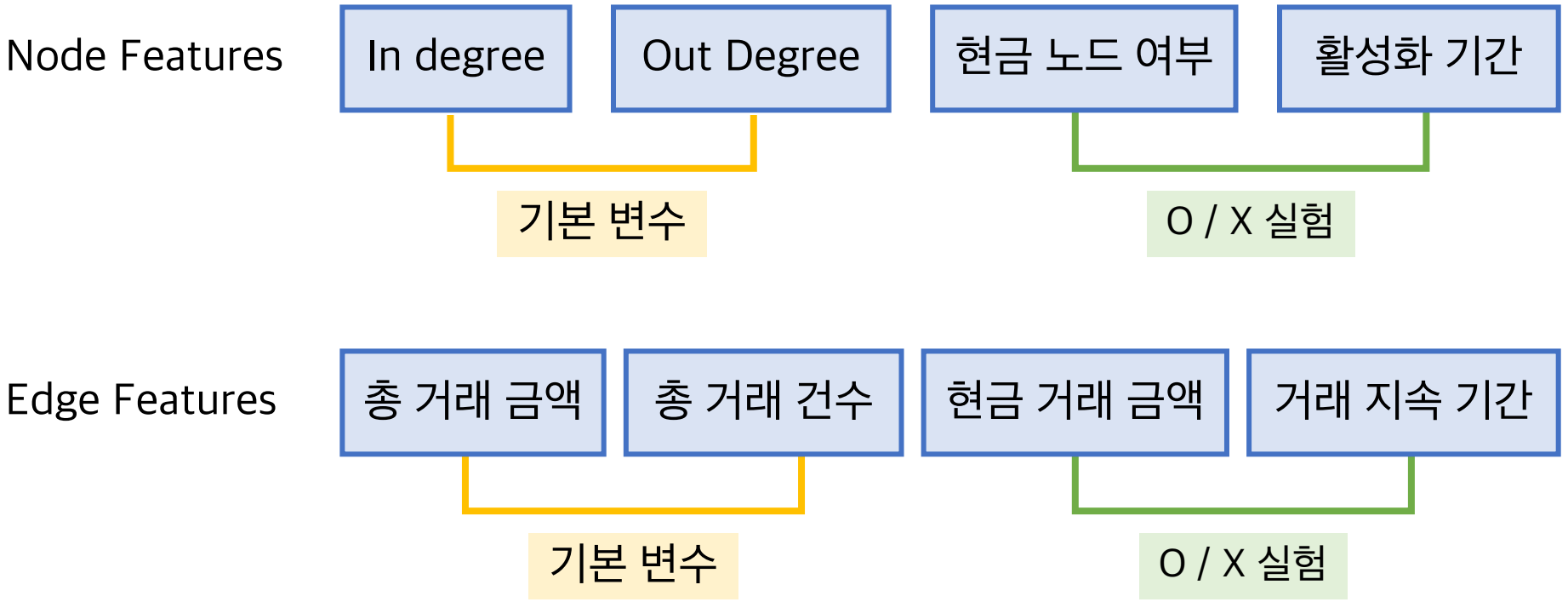
	graph_id	label
0	0	1
1	1	0
2	2	0
3	3	0
4	4	0

라벨
(사기: 1)

최종 모델 - 2. Feature 실험

중간 발표에서 사용한 기본 변수 외에, 검증한 가설을 바탕으로 변수를 추가하여 각각 모델 성능 확인

- 노드와 엣지 피쳐에서 현금 변수와 기간 변수를 모두 추가했을 때 가장 좋은 성능을 보임



최종 모델 - 3. 모델링

모델 프로세스 및 코드는 다음와 같음

기존의 Node, Edge Features

↓ EdgeGAT Layer1

새로운 Node Features, 기존의 Edge Features

↓ EdgeGAT Layer2

새로운 Node Features, 기존의 Edge Features

↓ Readout Layer
Linear Layer

최종 예측값

```python
class EdgeGATModel(nn.Module):
    def __init__(self, in_feats, edge_feats, hidden_feats, num_heads):
        super(EdgeGATModel, self).__init__()
        self.edge_gat1 = EdgeGATConv(in_feats = in_feats,
                                     edge_feats = edge_feats,
                                     out_feats = hidden_feats,
                                     num_heads = num_heads)
        self.edge_gat2 = EdgeGATConv(in_feats = hidden_feats,
                                     edge_feats = edge_feats,
                                     out_feats = hidden_feats,
                                     num_heads = num_heads)
        self.Linear = nn.Linear(hidden_feats, 1)

    def forward(self, graph, node_feats, edge_feats, num_heads):
        hidden1 = self.edge_gat1(graph, node_feats, edge_feats)
        hidden1 = torch.mean(hidden1, dim=1, keepdim=True)
        hidden1 = hidden1.view(hidden1.shape[0], -1)
        hidden1 = F.leaky_relu(hidden1)

        hidden2 = self.edge_gat2(graph, hidden1, edge_feats)
        hidden2 = torch.mean(hidden2, dim=1, keepdim=True)
        hidden2 = hidden2.view(hidden2.shape[0], -1)
        hidden2 = F.leaky_relu(hidden2)
        graph.ndata['h'] = hidden2

        hg = dgl.mean_nodes(graph, 'h')
        hg = self.Linear(hg)

        return hg
```

최종 모델 - 4. 성능 평가 및 개선

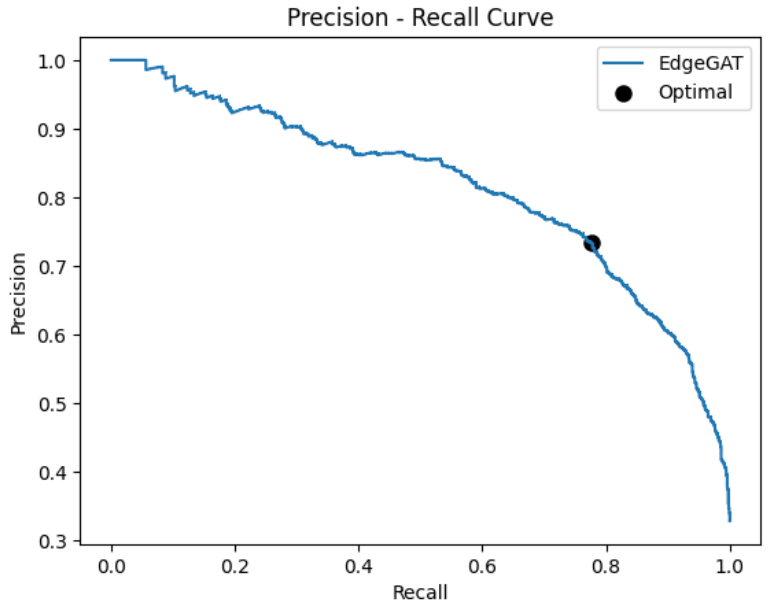
- Optuna를 이용해 하이퍼파라미터 튜닝 후, 최적 파라미터를 바탕으로 사기/정상을 판별하는 threshold 설정
- **성능 평가 지표로 미루어 보았을 때 효과적인 의심거래 탐지 모델 개발에 성공함**

[하이퍼파라미터 튜닝]

```
sampler = TPESampler(seed = 12)
study = optuna.create_study(
    study_name = 'EdgeGAT_opt',
    direction = 'minimize',
    sampler = sampler)
study.optimize(objective, n_trials = 50)
print('Best Score : ', study.best_value)
print('Best Trial : ', study.best_trial.params)
```

hidden_dim: 6 learning_rate: 0.01
num_heads: 5 batch_size: 64
num_epochs: 70

[Optimal Threshold 설정]



[최종 모델 성능]

Accuracy : 84%

Recall : 86%

F1 Score : 78%

결론

프로젝트 의의

- 네트워크 분석을 도입함으로써 조직적인 이상금융거래를 포착하기 어려운 **룰 기반 방식의 한계 극복**
- 발굴한 이상 거래 패턴을 바탕으로 **기존 룰 추가 또는 정교화 등 강화 가능성 확인**

기대 효과

- **탐지 모델을 룰 기반 방식과 결합**할 시 의심 계좌를 보다 효과적으로 파악할 수 있을 것으로 기대됨
- 1차로 탐지 모델을 통해 의심 계좌 그룹을 파악한 후, 2차로 발굴한 패턴과 기존 룰로 의심 계좌를 세부적으로 살펴보면 기존의 룰 기반 방식보다 단축된 시간과 높은 정확도로 작업 수행 가능할 것

감사합니다