IBM Transactions for Anti Money Laundering (AML) Implementation by XGBoost

จัดทำโดย

1. นายจิรพัฒน์ ศรีทวี รหัสนิสิต 6610502005
2. นางสาวนภัสนันท์ ดามะพร รหัสนิสิต 6610502102
3. นายจารุกิตติ์ ผลวัฒนานุวงศ์ รหัสนิสิต 6610505306

เสนอ

Assoc. Prof. Kitsana Waiyamai, Ph.D.

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา

การทำเหมืองข้อมูล (DataMining) รหัสวิชา 01204465

ภาคเรียนที่ 1 ปีการศึกษา 2568

มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตบางเขน

สารบัญ

Complex Data Exploration

1. Overview Data

ข้อมูลที่ใช้ในโครงงานนี้มาจากสองแหล่งหลัก ได้แก่

- **HI-Small_Trans.csv** — ข้อมูลธุรกรรมทางการเงินระหว่างบัญชีของหลายธนาคาร ครอบคลุมธุรกรรมหลากหลายรูปแบบ
- **HI-Small_accounts.csv** — ข้อมูลเพิ่มเติมของแต่ละบัญชี เช่นชื่อธนาคาร หมายเลขบัญชี ประเภทนิติบุคคล และรหัสเอนทิตี

```
..                                SMALL
..                                HI     LI
.. Date Range HI + LI (2022)       Sep 1-10
.. # of Days Spanned               10     10
.. # of Bank Accounts              515K   705K
.. # of Transactions                5M     7M
.. # of Laundering Transactions     5.1K   4.0K
.. Laundering Rate (1 per N Trans)  981    1942
```

ภาพ : รายละเอียดของชุดข้อมูล

โครงสร้างของข้อมูลหลัก

คอลัมน์	รายละเอียด	ประเภท
Timestamp	วันเวลาเกิดธุรกรรม (ปี/เดือน/วัน ชั่วโมง/นาที)	Timestamp
From Bank / To Bank	รหัสตัวเลขของธนาคารต้นทาง / รหัสตัวเลขของธนาคารปลายทาง	int64
Account / Account.1	รหัสบัญชี (ฐาน 16) ของบัญชีต้นทาง / รหัสบัญชี (ฐาน 16) ของบัญชีปลายทาง	int64
Amount Paid / Amount Received	จำนวนเงินที่จ่าย / จำนวนเงินที่ได้รับ ในฝั่งบัญชีต้นทาง	float64
Payment Currency / Receiving Currency	สกุลเงินที่ใช้ในบัญชีต้นทาง / สกุลเงินที่ใช้ในบัญชีปลายทาง เช่น USD, EUR, JPY	String
Payment Format	รูปแบบการทำธุรกรรม เช่น เช็ค, โอนผ่านระบบ ACH, โอนผ่านสาย (wire), บัตรเครดิต เป็นต้น	String
Is Laundering	ป้ายกำกับระบุการฟอกเงิน — 0 หมายถึง ธรรมดา / 1 หมายถึง ฟอกเงิน	int64

ในส่วนของข้อมูลบัญชี (HI-Small_accounts.csv) ประกอบด้วย

- **Bank Name** – ชื่อธนาคาร
- **Bank ID** – รหัสธนาคาร
- **Account Number** – หมายเลขบัญชี
- **Entity Name** – ประเภทนิติบุคคล เช่น *Corporation*, *Sole Proprietorship*

2. Data Complexity

ข้อมูลนี้จัดอยู่ในกลุ่ม Complex Financial Transaction Data เนื่องจากมีความซับซ้อนหลายมิติ ได้แก่

มิติความซับซ้อน	รายละเอียด
Multi-Entity Structure	ธุรกรรมหนึ่งรายการเชื่อมโยงผู้โอน–ผู้รับ และธนาคารต้นทาง–ปลายทาง ซึ่งอาจอยู่ในเครือข่ายเดียวกันหรือข้ามประเทศ
Multi-Currency	มีทั้งสกุลเงินทั่วไปและสกุลเงินดิจิทัล เช่น USD, Euro, Shekel, Yen, Bitcoin
Temporal Dimension	มีตัวแปรเวลา (Timestamp) ที่สามารถแตกเป็น weekday, hour เพื่อวิเคราะห์พฤติกรรมในแต่ละช่วงเวลา
Imbalanced Class Problem	ป้ายกำกับไม่สมดุลสูง โดยธุรกรรมฟอกเงินมีเพียง 0.1019 % ของทั้งหมด (99.8981 % เป็นปกติ)
Relational Graph Structure	บัญชีจำนวนมากโอนเงินระหว่างกันซ้ำ ๆ จึงสามารถแปลงข้อมูลให้เป็นกราฟ เพื่อให้โมเดล GNN วิเคราะห์ความสัมพันธ์ได้

3. Data Exploration

- **แหล่งข้อมูล:** ข้อมูล Secondary จาก Kaggle หัวข้อ IBM Transactions for Anti Money Laundering (AML)
- **ขนาดข้อมูล:** 5,078 ,345 แถว × 11 คอลัมน์

```
1  data.shape
```

```
(5078345, 11)
```

ภาพ: ขนาดของข้อมูล

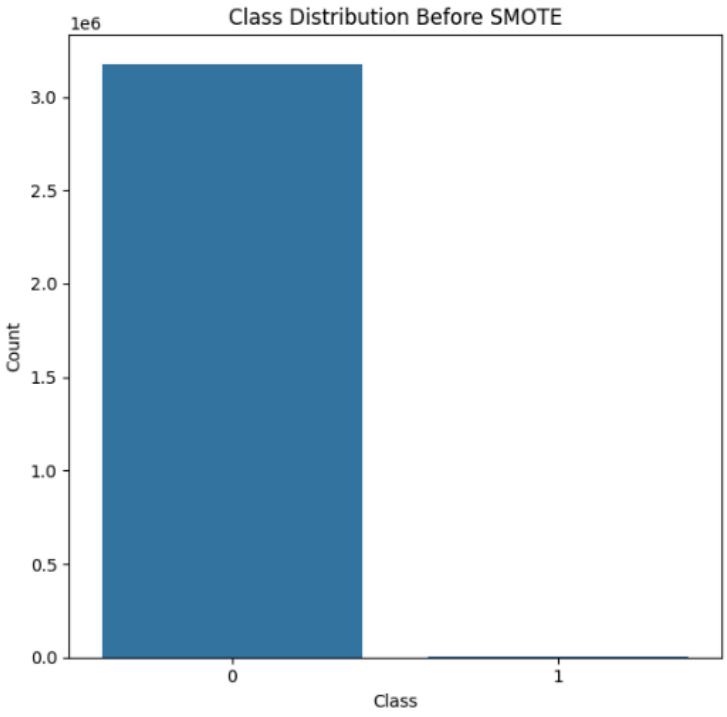
- **ค่าหาย (Null):** ไม่มีค่า null ในทุกคอลัมน์

```
Null Values in Each Column:
 Timestamp              0
From Bank               0
Account                 0
To Bank                 0
Account.1               0
Amount Received         0
Receiving Currency      0
Amount Paid             0
Payment Currency        0
Payment Format          0
Is Laundering           0
dtype: int64

Number of Duplicate Rows: 9
```

ภาพ : Null Values Counts

- **ข้อมูลซ้ำ:** พบแถวซ้ำ 9 รายการและถูกลบออกแล้ว
- **คัดกรองข้อมูลธนาคารที่ไม่เคยมีการฟอกเงินเลยออกจาก dataset:** เพราะจะได้ pattern ที่สัมพันธ์กับ laundering
- **การกระจายของคลาส:**
    - Is Laundering = 0 (99.898 %)
    - Is Laundering = 1 (0.102 %)



ภาพ : Bar Chart Class Distribution Before SMOTE

4. การวิเคราะห์เชิงเวลา (Time Exploration)

หลังจากแปลง Timestamp เป็น datetime แล้ว ได้สร้างฟีเจอร์ใหม่ดังนี้

- weekday (วันในสัปดาห์ที่เกิดธุรกรรม)
- hour (ชั่วโมงของวันที่ทำรายการ)

ข้อมูลนี้ช่วยให้ตรวจจับพฤติกรรมธุรกรรมในเวลาผิดปกติ เช่น การโอนเงินช่วงดึกหรือวันหยุด ซึ่งอาจสัมพันธ์กับพฤติกรรมการฟอกเงิน

5. การสำรวจโครงสร้างเครือข่าย (Graph Exploration)

หลังจากรวมข้อมูลผู้โอน–ผู้รับแล้ว แปลงข้อมูลเป็นกราฟดังนี้

- Node = บัญชีธนาคาร
- Edge = ธุรกรรมการโอนเงิน
- Edge Attributes

มิติความซับซ้อน	รายละเอียด
Timestamp	เวลาที่เกิดธุรกรรม ช่วยบอกลำดับเวลา / ความถี่
Amount Paid	จำนวนเงินที่โอนออกจากบัญชีต้นทาง
Amount Received	จำนวนเงินที่บัญชีปลายทางได้รับ
Payment Currency / Receiving Currency	สกุลเงิน ใช้ตรวจจับกรณี cross-currency
Payment Format	วิธีการชำระ เช่น wire, ACH, cheque ชี้พฤติกรรมผิดปกติ

- Node Features = ค่าเฉลี่ยยอดเงินที่โอนในแต่ละสกุล + รหัสธนาคาร (Encoding)
- Node Label = 1 หมายถึงหากบัญชีเคยเกี่ยวข้องกับธุรกรรมที่ถูกป้ายว่า Is Laundering = 1

ขั้นตอนเหล่านี้ช่วยให้ Graph Neural Network (GNN) สามารถเรียนรู้ความสัมพันธ์ระหว่างบัญชี และระบุ "กลุ่มเสี่ยง" ที่อาจเกี่ยวข้องกับการฟอกเงินได้

Complex Data Pre-Processing

การเตรียมข้อมูลสำหรับโมเดล XGBoost

1. การทำความสะอาดข้อมูล (Data Cleaning)

- โหลดข้อมูลจากไฟล์ HI-Small_Trans.csv และ HI-Small_accounts.csv
- ลบแถวที่ซ้ำกัน (พบ 9 แถวซ้ำ และถูกลบออก)
- ตรวจสอบค่าที่หาย (null) — ไม่พบค่า null ในทุกคอลัมน์หลัก
- แปลงคอลัมน์ Timestamp ให้เป็นชนิดข้อมูล datetime เพื่อใช้ในการแยกวันและเวลา
- ลบข้อมูลของธนาคารที่ไม่มีธุรกรรมฟอกเงินออก เพื่อให้ชุดข้อมูลมีความหลากหลายของคลาสมากขึ้น

2. การสร้างคุณลักษณะใหม่ (Feature Engineering)

สร้างฟีเจอร์ใหม่จากข้อมูลดิบเพื่อช่วยให้โมเดลเข้าใจพฤติกรรมธุรกรรมได้ดียิ่งขึ้น เช่น

- **weekday**: วันในสัปดาห์ที่เกิดธุรกรรม
- **hour**: ชั่วโมงของวันที่ทำธุรกรรม
- **type**: ประเภทการทำธุรกรรม (เช่น โอนให้ตัวเอง หรือ โอนออก)
- **sender_txn_count / receiver_txn_count**: จำนวนธุรกรรมของบัญชีแต่ละประเภท
- **sender_daily_count / receiver_daily_count**: จำนวนธุรกรรมต่อวันของผู้ส่งและผู้รับ

นอกจากนี้ยังมีการรวมข้อมูลจากไฟล์บัญชี (HI-Small_accounts.csv) โดยใช้ Account Number เป็นคีย์ เพื่อเพิ่มฟีเจอร์ด้านธนาคาร เช่น Bank ID, Bank Name, และ Entity Name

3. การเข้ารหัสและการปรับขนาดข้อมูล

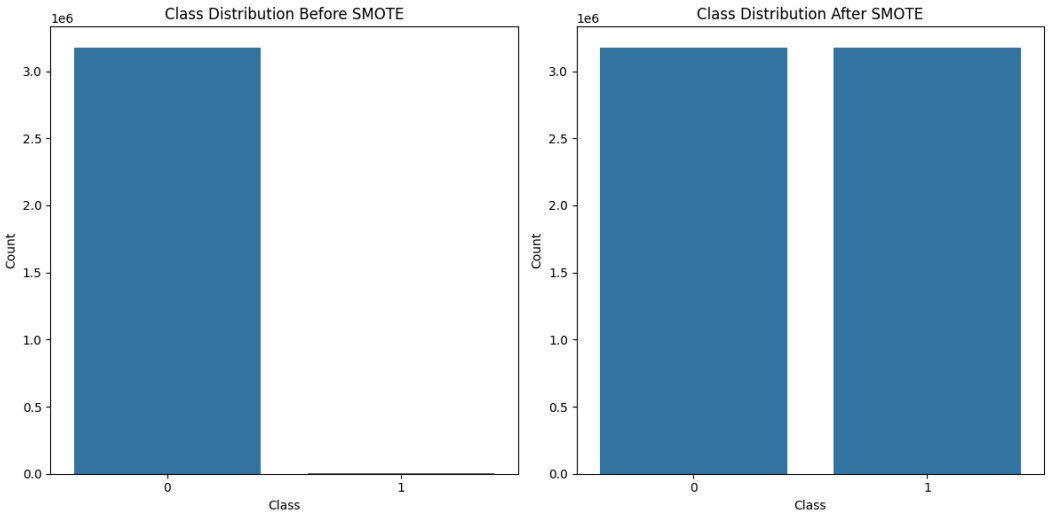
ประเภทของฟีเจอร์	วิธีแปลงข้อมูล	รายละเอียด
เชิงตัวเลข (Numeric)	SimpleImputer(strategy='mean') + StandardScaler()	เติมค่าที่หายด้วยค่าเฉลี่ย และปรับขนาดให้อยู่ในสเกลเดียวกัน
เชิงหมวดหมู่ (Categorical)	OneHotEncoder() และ BinaryEncoder()	แปลงค่าหมวดหมู่ เช่น สกุลเงิน และรูปแบบการชำระเงิน ให้เป็นเวกเตอร์ตัวเลข
คอลัมน์วันเวลา (Datetime)	แยกออกเป็น weekday, hour	ใช้แทนข้อมูลเชิงเวลาในรูปแบบที่โมเดลเข้าใจได้

4. การจัดการปัญหาคลาสไม่สมดุล (Imbalanced Data Handling)

จากการสำรวจพบว่า ข้อมูลธุรกรรมฟอกเงิน (Is Laundering = 1) มีเพียงประมาณ 0.1019% ของข้อมูลทั้งหมด

เพื่อป้องกันไม่ให้โมเดลเรียนรู้ไปยังคลาสส่วนใหญ่ จึงใช้เทคนิค SMOTE (Synthetic Minority Over-sampling Technique)

สร้างข้อมูลเทียมของคลาสส่วนน้อย (ฟอกเงิน) ให้มีจำนวนใกล้เคียงกับคลาสส่วนใหญ่



การเตรียมข้อมูลสำหรับโมเดล GNN (Graph Neural Network)

การทดลองใช้ GNN เพราะธุรกรรมฟอกเงินจริง "ไม่ได้เกิดจากธุรกรรมเดียว" แต่เกิดจากเครือข่ายของบัญชีและความเชื่อมโยงหลายชั้น ซึ่ง GNN สามารถ "เรียนรู้จากโครงสร้างความสัมพันธ์ (Graph Structure)" ได้โดยตรง

ข้อจำกัด (GPU): เนื่องจากข้อมูลมีขนาดใหญ่ ทางกลุ่มจึงทดลองโดย ไม่ได้ Imbalance Data เพิ่มครับ

1. การแปลงข้อมูลเป็นโครงสร้างกราฟ

ข้อมูลธุรกรรมถูกเปลี่ยนจากรูปแบบตาราง (tabular) ให้เป็น โครงสร้างกราฟ (graph structure) โดย

- **Node (โหนด)**: หมายถึง "บัญชีธนาคาร"
- **Edge (ขอบ)**: หมายถึง "ธุรกรรมระหว่างบัญชีผู้ส่งและผู้รับ"

เพื่อให้โหนดแต่ละตัวมีข้อมูลเชิงพฤติกรรมและธนาคารถูกต้องครบถ้วน มีการสร้างรหัสบัญชีเฉพาะ (unique ID) จากการรวม Bank ID และ Account Number

2. การเข้ารหัสและปรับค่าฟีเจอร์

- ใช้ LabelEncoder จาก scikit-learn เพื่อแปลงคอลัมน์หมวดหมู่ ได้แก่
    - Payment Format
    - Payment Currency
    - Receiving Currency
- ตรวจสอบให้แน่ใจว่าค่าหมวดหมู่ใน Payment Currency และ Receiving Currency สอดคล้องกันก่อนเข้ารหัส
- ค่าตัวเลข เช่น ยอดเงิน (Amount Paid, Amount Received) ผ่านการปรับสเกลด้วย Min–Max Normalization

3. การสร้างฟีเจอร์ของ Node และ Edge

- **Node Attributes:** ค่าเฉลี่ยยอดเงินที่จ่ายและรับในแต่ละสกุลเงิน + รหัสธนาคาร
- **Edge Attributes:** จำนวนเงินและสกุลเงินที่ใช้ในธุรกรรมแต่ละครั้ง
- **Node Labels:** บัญชีที่เกี่ยวข้องกับธุรกรรมที่มีป้ายกำกับ Is Laundering = 1 จะถูกตั้งค่าเป็น 1 ทั้งฝั่งผู้ส่งและผู้รับ

4. การรวมข้อมูลเข้าสู่ PyTorch Geometric

ข้อมูลทั้งหมดถูกรวมในวัตถุ Data() ของ PyTorch Geometric ซึ่งประกอบด้วย

- x คือ Node features
- edge_index คือ ความเชื่อมโยงของโหนด
- edge_attribute คือ คุณลักษณะของขอบ (Edge)
- y คือ Label ของแต่ละโหนด

สรุปการเตรียมข้อมูล

ประเด็น	XGBoost	GNN
ลักษณะข้อมูล	ตารางธุรกรรม (Tabular Data)	เครือข่ายธุรกรรม (Graph Data)
การเข้ารหัส	OneHotEncoder, BinaryEncoder	LabelEncoder
การปรับขนาด	StandardScaler	Min–Max Normalization
จัดการคลาสไม่สมดุล	ใช้ SMOTE / UnderSampling	ไม่ใช้ตรง ๆ (เรียนรู้จากโครงสร้างกราฟแทน)
หน่วยวิเคราะห์หลัก	ธุรกรรม	บัญชี
ผลลัพธ์การใช้ข้อมูล	โมเดล XGBoost สามารถจำแนกธุรกรรมได้อย่างแม่นยำ (AUC ≈ 0.95)	โมเดล GNN สามารถตรวจจับกลุ่มบัญชีที่มีความสัมพันธ์ผิดปกติในเครือข่ายได้ดี

Analytic technique

ภาพรวมของเทคนิคการวิเคราะห์

การวิเคราะห์ในโครงงานนี้ใช้เทคนิคการเรียนรู้ของเครื่องสองแนวทางหลักในการตรวจจับธุรกรรมฟอกเงิน (Anti-Money Laundering: AML) ได้แก่

1. **XGBoost (Extreme Gradient Boosting)** โมเดลแบบต้นไม้ (tree-based model) สำหรับข้อมูลเชิงตาราง
2. **Graph Neural Network (GNN)** โมเดลโครงข่ายประสาทเทียมที่ออกแบบมาเฉพาะสำหรับข้อมูลเชิงกราฟ

การใช้ทั้งสองเทคนิคนี้มีวัตถุประสงค์เพื่อเปรียบเทียบประสิทธิภาพของ Feature-based learning (XGBoost) กับ Relation-based learning (GNN) ในการตรวจจับธุรกรรมฟอกเงินจากข้อมูลธุรกรรมที่ซับซ้อนหลายมิติ

เทคนิคที่ 1: XGBoost (Extreme Gradient Boosting)

1. หลักการของ XGBoost

XGBoost เป็นอัลกอริทึมที่พัฒนามาจากแนวคิดของ **Gradient Boosted Decision Trees (GBDT)** ซึ่งทำงานโดยการสร้างต้นไม้หลายต้นแบบลำดับกัน (Sequential Boosting) โดยแต่ละต้นจะพยายามแก้ไขข้อผิดพลาดของต้นก่อนหน้า

สมการพื้นฐานของการปรับปรุงน้ำหนักในแต่ละรอบ คือ การใช้ค่า **Gradient ของฟังก์ชัน Loss** เพื่อหาค่าพารามิเตอร์ที่ทำให้ข้อผิดพลาดรวมลดลงมากที่สุด ซึ่งช่วยเพิ่มความสามารถในการจำแนกข้อมูลที่ซับซ้อนได้ดี

จุดเด่นของ XGBoost คือ

- รองรับข้อมูลที่มี missing values ได้โดยอัตโนมัติ
- มีระบบ regularization ป้องกัน overfitting
- ใช้งานได้ดีในข้อมูลขนาดใหญ่และไม่เชิงเส้น

2. การตั้งค่าพารามิเตอร์ (Hyperparameter Tuning)

Parameter	ค่า	รายละเอียด
objective	"binary:logistic"	ใช้สำหรับงานจำแนกแบบสองคลาส (ฟอกเงิน / ปกติ)
eval_metric	Logloss, error	ใช้ค่า logloss ในการประเมิน overfit
eta	0.1	ค่าการเรียนรู้ (learning rate) สำหรับการปรับน้ำหนัก
max_depth	16	ความลึกสูงสุดของแต่ละต้นไม้
subsample	0.8	สัดส่วนการสุ่มแถวข้อมูลที่ใช้ในแต่ละรอบ
colsample_bytree	0.8	สัดส่วนการสุ่มคอลัมน์ที่ใช้ในการสร้างต้นไม้
Alpha	0.1	regularization
lamda	1	regularization

3. ผลลัพธ์การฝึกโมเดล

ผลการเทรนแสดงให้เห็นว่าโมเดลมีความสามารถในการจำแนกธุรกรรมฟอกเงินได้ในระดับหนึ่ง โดยมี

- **อัตราการตรวจจับ (TPR / Recall) = 55.6%**
  หมายถึง โมเดลสามารถตรวจจับธุรกรรมฟอกเงินจริงได้มากกว่าครึ่งของทั้งหมด
- **อัตราการแจ้งเตือนผิดพลาด (FPR) = 0.57%**
  แสดงว่าโมเดลแทบไม่แจ้งเตือนธุรกรรมปกติผิด (False Positive ต่ำมาก)
- **Precision = 10.0%**
  หมายความว่า ในทุก 10 เคสที่โมเดลแจ้งเตือนว่า "ฟอกเงิน" จะมีเพียง **1 เคสที่เป็นจริง**
  ส่วนอีก **9 เคสเป็น False Positive**

สาเหตุหลักมาจากลักษณะของข้อมูลที่ **มีความไม่สมดุลสูงมาก (Extremely Imbalanced Data)** ธุรกรรมฟอกเงินจริงมีจำนวนน้อยมากเมื่อเทียบกับธุรกรรมปกติ ทำให้โมเดลเน้นทำนาย "ปกติ" เพื่อคง accuracy รวมสูง แม้ Precision ต่ำ (10%) แต่ใน AML การ "จับให้ได้" สำคัญกว่าความแม่นยำเฉพาะตัวเพราะหลังจากแจ้งเตือนแล้ว **มนุษย์หรือระบบขั้นถัดไปจะเป็นผู้ตรวจซ้ำอีกชั้นหนึ่ง**

เทคนิคที่ 2: Graph Neural Network (GNN)

1. หลักการของ GNN

Graph Neural Network (GNN) เป็นโมเดล Deep Learning ที่ออกแบบมาเพื่อเรียนรู้จากข้อมูลที่มี โครงสร้างเชิงกราฟ (Graph Structure) โดยที่แต่ละโหนด (Node) จะอัปเดตค่าคุณลักษณะของตนเองโดยการรวมข้อมูลจากเพื่อนบ้าน (Neighbor Aggregation) ซึ่งช่วยให้โมเดลเข้าใจ ความสัมพันธ์ระหว่างเอนทิตี (Entity Relationships) ได้

ในโครงงานนี้ใช้สถาปัตยกรรม Graph Attention Network (GAT) ซึ่งเป็นส่วนขยายของ GNN โดยเพิ่มกลไก Attention Mechanism เพื่อให้น้ำหนักความสำคัญกับเพื่อนบ้านแต่ละโหนดไม่เท่ากัน

2. การตั้งค่าโมเดล GNN

Parameter	ค่า	รายละเอียด
Model Type	Graph Attention Network (GAT)	ใช้ attention ในการรวมข้อมูลของเพื่อนบ้าน
Hidden Channels	16	จำนวนหน่วยในชั้นซ่อน
Output Channels	1	การจำแนกแบบ Binary
Activation Function	Sigmoid	ใช้สำหรับการทำนายแบบสองคลาส
Loss Function	BCELoss()	Binary Cross Entropy Loss
Optimizer	SGD	Stochastic Gradient Descent
Learning Rate	0.0001	ความเร็วในการปรับพารามิเตอร์
Epoch	100	จำนวนรอบการฝึกโมเดล
Data Split	RandomNodeSplit(train/val/test)	แบ่งข้อมูลโหนดแบบสุ่มสำหรับการเทรนและทดสอบ

3. การทำงานของ GNN ในโครงงานนี้

1. สร้างกราฟจากข้อมูลธุรกรรมโดยใช้บัญชีธนาคารเป็น Node และธุรกรรมเป็น Edge
2. สำหรับแต่ละ Node จะมี feature เช่น ยอดเงินเฉลี่ยที่โอน สกุลเงิน และธนาคารที่สังกัด
3. ใช้ GAT เพื่อให้โหนดเรียนรู้บริบทของเพื่อนบ้าน (neighboring accounts) ผ่าน Attention Weights
4. ผลลัพธ์ของโมเดลคือ การทำนายระดับบัญชี (Node Classification) ว่าบัญชีนั้นอาจมีส่วนเกี่ยวข้องกับการฟอกเงินหรือไม่

การเปรียบเทียบแนวทางระหว่าง XGBoost และ GNN

ประเด็นเปรียบเทียบ	XGBoost	GNN
โครงสร้างข้อมูล	ตาราง (Tabular Data)	กราฟ (Graph Data)
ลักษณะการเรียนรู้	วิเคราะห์ฟีเจอร์ของแต่ละธุรกรรม	วิเคราะห์ความสัมพันธ์ของบัญชีในเครือข่าย
จุดเด่น	ความเร็วสูง ใช้งานง่าย ตีความได้	เข้าใจบริบทของเครือข่าย และการโอนเงินต่อเนื่อง
ข้อจำกัด	ไม่เข้าใจความสัมพันธ์ระหว่างบัญชี	ต้องใช้หน่วยประมวลผลสูง **(GPU)** และการเตรียมข้อมูลซับซ้อน
ประเภทผลลัพธ์	จำแนกธุรกรรมรายรายการ	จำแนกความเสี่ยงในระดับบัญชี

Evaluation

1. วัตถุประสงค์ของการประเมินผล

การประเมินผลในโครงงานนี้มีเป้าหมายเพื่อวัดประสิทธิภาพของโมเดลในการตรวจจับธุรกรรมฟอกเงิน (Anti-Money Laundering Detection)
โดยเน้นการเปรียบเทียบประสิทธิภาพของสองแนวทางหลัก ได้แก่

1. **XGBoost** การจำแนกธุรกรรมในรูปแบบข้อมูลเชิงตาราง
2. **Graph Neural Network (GNN)** การจำแนกบัญชีในรูปแบบเครือข่าย (Graph-based)

เนื่องจากข้อมูลมีลักษณะ Class Imbalance สูงมาก (ธุรกรรมฟอกเงินมีเพียง 0.1%)
จึงเลือกใช้ตัวชี้วัด (evaluation metrics) ที่สามารถสะท้อนความแม่นยำของโมเดลได้ดีกว่า Accuracy เพียงอย่างเดียว

2. การประเมินผลโมเดล XGBoost

2.1 วิธีการประเมิน

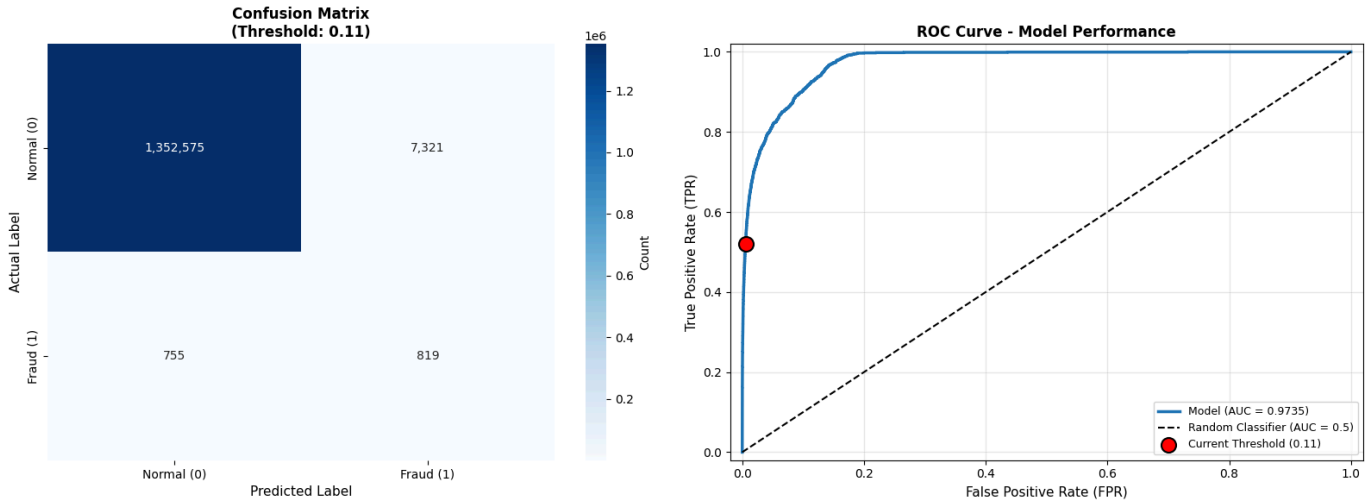
หลังจากทำการปรับสมดุลข้อมูลด้วย **SMOTE** แล้ว ข้อมูลถูกแบ่งเป็น

- **Training set:** สำหรับเรียนรู้พารามิเตอร์ของโมเดล
- **Test set:** สำหรับประเมินประสิทธิภาพสุดท้าย

ใช้ตัวชี้วัดหลักในการประเมินดังนี้

- **Accuracy:** สัดส่วนของผลทำนายที่ถูกต้องทั้งหมด
- **Precision:** ความถูกต้องของการทำนายว่าฟอกเงินจริง (ลด False Positive)
- **Recall:** ความสามารถในการจับธุรกรรมฟอกเงินจริง (ลด False Negative)
- **F1-Score:** ค่าเฉลี่ยเชิงกลมกลืนของ Precision และ Recall

2.2 ผลลัพธ์จากโมเดล XGBoost (Oversampling)

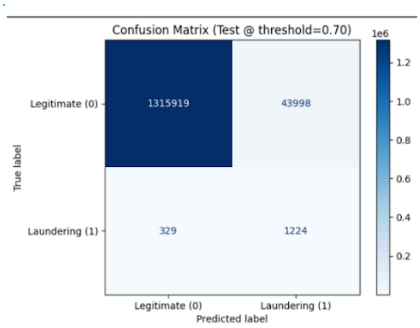


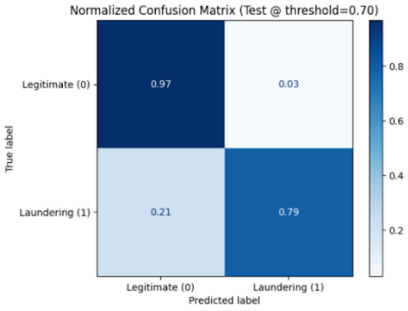
ภาพ : confusion matrix และ ROC Curve

จากผลการเทรน จะได้ค่าตัวชี้วัดดังนี้

Metric	ค่า (Evaluation Set)
Accuracy	0.99
Precision	0.10
Recall(TPR)	0.56
F1-Score	0.16

ผลลัพธ์จากโมเดล XGBoost (undersampling TH=0.7)





Metric	ค่า (Evaluation Set)
Accuracy	0.97
Precision	0.03
Recall(TPR)	0.79
F1-Score	0.05

1. การวิเคราะห์ ROC Curve

- **อัตราการตรวจจับ (TPR) ที่ 55.6%%** โดยมี อัตราการแจ้งเตือนที่ผิดพลาด (FPR ที่ต่ำมาก 0.54%)

2. การวิเคราะห์ Confusion Matrix

กราฟ Confusion Matrix แสดงผลลัพธ์การทำงานจริงของโมเดล เมื่อใช้เกณฑ์การตัดสินใจที่ 0.11 สรุปได้ดังนี้:

สถิติการตรวจจับ (คลาส 1: Fraud)

- True Positives (TP): 819 รายการ
    - ความหมาย: โมเดลสามารถตรวจจับเคสการฟอกเงินได้อย่างถูกต้อง 819 เคส
- False Negatives (FN): 755 รายการ
    - ความหมาย: โมเดล พลาดการตรวจจับ เคสฟอกเงิน 755 เคส (ประเมินว่าเป็น "ปกติ" ทั้งที่จริงคือ "โกง")

สถิติการแจ้งเตือน (False Alarms)

- False Positives (FP): 7,321 รายการ
    - ความหมาย: โมเดล แจ้งเตือนผิดพลาด 7,321 เคส (ประเมินว่าเป็น "โกง" ทั้งที่จริงคือ "ปกติ")
- True Negatives (TN): 1,352,575 รายการ
    - ความหมาย: โมเดลระบุธุรกรรมปกติได้อย่างถูกต้อง 1,352,575 เคส

สรุปตัวชี้วัดหลัก (ณ Threshold 0.11)

- **Accuracy:** 0.97 (โมเดลทำนายถูกเกือบทั้งหมด แต่ค่านี้สูงเพราะข้อมูลส่วนใหญ่เป็น “ธุรกรรมปกติ”)
- **Precision:** 0.03 (ในทุก 100 เคสที่โมเดลแจ้งว่า “ฟอกเงิน” มีเพียง 2–3 เคสเท่านั้นที่เป็นจริง)
- **Recall (TPR):** 0.79 (โมเดลสามารถตรวจจับธุรกรรมฟอกเงินจริงได้ประมาณ 79% ของทั้งหมด)
- **F1-Score:** 0.05 (ค่าเฉลี่ยระหว่าง Precision และ Recall ยังต่ำ เนื่องจากโมเดลแจ้งเตือนผิดบ่อย

สาเหตุหลัก

แม้โมเดลจะผ่านการปรับสมดุลด้วย Undersampling และ threshold สูง แต่ความไม่สมดุลเชิงโครงสร้างของข้อมูลยังคงอยู่ ฟีเจอร์ส่วนใหญ่ไม่ได้สะท้อนพฤติกรรมเชิงเครือข่ายของฟอกเงินจริง จึงทำให้โมเดล “จับได้เยอะแต่แม่นยำน้อย” (Recall สูง แต่ Precision ต่ำ)

ผลลัพธ์นี้เหมาะกับการใช้โมเดลเป็น ระบบแจ้งเตือนเบื้องต้น (Early Warning) ให้มนุษย์หรือตัวกรองขั้นถัดไปตรวจสอบต่อ

3. การประเมินผลโมเดล Graph Neural Network (GNN), No Imbalance Data

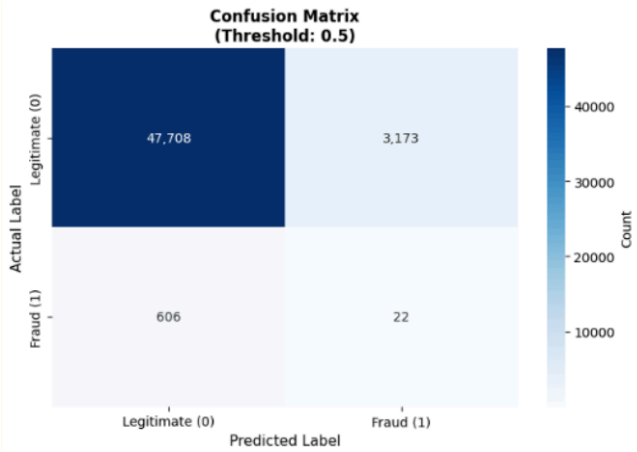
3.1 วิธีการประเมิน

ในฝั่งโมเดลกราฟ ใช้ข้อมูลในรูปแบบโครงสร้างกราฟที่สร้างจากบัญชีและธุรกรรม โดยแบ่งข้อมูลเป็นชุด Train / Validation / Test ด้วยฟังก์ชัน RandomNodeSplit() จาก PyTorch Geometric ประเมินความแม่นยำของการจำแนก (Node Classification Accuracy) ในแต่ละ Epoch

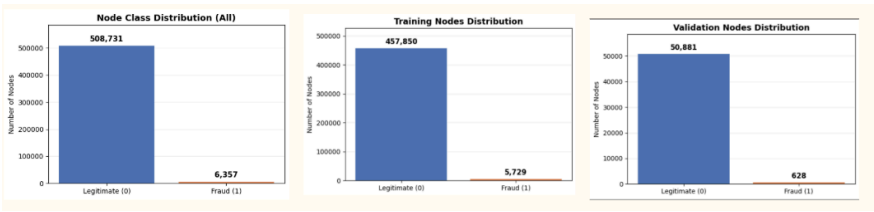
ตัวชี้วัดหลักที่ใช้ได้แก่

- **Training Loss / Validation Loss:** เพื่อดูการลู่เข้าสู่จุดสมดุล
- **Accuracy:** วัดสัดส่วนของโหนดที่ทำนายได้ถูกต้อง

3.2 ผลลัพธ์จากโมเดล GNN




```
Epoch: 010, Loss: 13.0348, Val Loss: 9.6795, Val Acc: 0.8988
Epoch: 020, Loss: 12.8660, Val Loss: 8.7422, Val Acc: 0.9094
Epoch: 030, Loss: 12.4076, Val Loss: 8.1823, Val Acc: 0.9158
Epoch: 040, Loss: 12.2839, Val Loss: 7.6819, Val Acc: 0.9213
Epoch: 050, Loss: 12.4332, Val Loss: 7.2408, Val Acc: 0.9263
Epoch: 060, Loss: 11.8950, Val Loss: 7.1120, Val Acc: 0.9275
Epoch: 070, Loss: 12.9962, Val Loss: 6.7850, Val Acc: 0.9308
Epoch: 080, Loss: 10.8557, Val Loss: 6.4532, Val Acc: 0.9344
Epoch: 090, Loss: 10.6530, Val Loss: 5.6826, Val Acc: 0.9421
Epoch: 100, Loss: 10.4606, Val Loss: 5.4734, Val Acc: 0.9443
```



จะสังเกตได้ว่าแม้จะใช้ GNN ซึ่งออกแบบมาสำหรับข้อมูลแบบกราฟ แต่เมื่อไม่ได้ปรับสมดุล class (no- sampling) ด้วยข้อจำกัดด้าน **GPU** โมเดลยังคง "กลบ signal ของธุรกรรมฟอกเงิน " ที่มีจำนวนน้อย ส่งผลให้ Recall ต่ำ (ตรวจจับน้อย) และ Precision ต่ำ (แจ้งผิดเยอะ) แม้ Accuracy จะดูดีแต่ไม่สะท้อนความสามารถจริงของระบบ

ในอนาคตสามารถต่อยอดได้โดยการเพิ่มเทคนิค **GraphSMOTE** จะช่วยให้โมเดลเรียนรู้จากบริบทของบัญชีฟอกเงินได้ดียิ่งขึ้นและเพิ่มความสามารถในการตรวจจับ (Recall)

4. สรุปผลการประเมิน

ผลการทดลองทั้งสองโมเดลแสดงให้เห็นว่า

- **XGBoost** สามารถจำแนกธุรกรรมฟอกเงินได้ เหมาะสำหรับใช้เป็น **Baseline Model** ในการคัดกรองธุรกรรมทั่วไป ใช้ทรัพยากรที่ต่ำกว่าแบบกราฟ
- **GNN (Graph Attention Network)** มีศักยภาพสูงในการตรวจจับบัญชีต้องสงสัยที่มีความสัมพันธ์ซับซ้อนโดยเฉพาะกรณีที่พฤติกรรมฟอกเงินเกิดเป็นกลุ่มเครือข่ายแต่แลกมากลับการใช้ทรัพยากรที่สูงในการเทรนและจำลองข้อมูล

ดังนั้น การผสานการทำงานของทั้งสองโมเดล XGBoost สำหรับการประเมินธุรกรรมรายกรณี และ GNN สำหรับการวิเคราะห์เชิงเครือข่าย จะช่วยสร้างระบบตรวจจับการฟอกเงินและครอบคลุมทั้งเชิงลักษณะและเชิงโครงสร้างของข้อมูลทางการเงินได้อย่างสมบูรณ์.

List of Possible Applications
การประยุกต์ใช้งานที่เป็นไปได้

1. ระบบตรวจจับการฉ้อโกงทางการเงิน (Fraud Detection System)

- ใช้หลักการเดียวกับ AML เพื่อวิเคราะห์ธุรกรรมบัตรเครดิตหรือ e-Payment
- ตรวจจับการใช้งานผิดปกติ เช่น การรูดบัตรในต่างประเทศซ้ำ ๆ หรือการโอนจำนวนเงินผิดปกติภายในช่วงเวลาสั้น
- ลดความเสียหายจากการโจรกรรมทางไซเบอร์และฟิชชิง

2. ระบบตรวจสอบธุรกรรมของลูกค้า (Customer Risk Scoring)

- ใช้ข้อมูลธุรกรรมและความสัมพันธ์ในเครือข่ายเพื่อคำนวณ “คะแนนความเสี่ยง” ของลูกค้า
- รองรับการทำ Credit Scoring ที่คำนึงถึงพฤติกรรมเครือข่าย ไม่ใช่เฉพาะข้อมูลส่วนบุคคล
- ช่วยให้ธนาคารประเมินความเสี่ยงของลูกค้าได้แม่นยำมากขึ้น

3. การตรวจสอบการระดมทุนหรือบริจาคที่ผิดปกติ (Suspicious Fund Flow Monitoring)

- ใช้วิเคราะห์เส้นทางการเงินของเงินบริจาคหรือกองทุนสาธารณะ
- ตรวจสอบการหมุนเวียนของเงินที่อาจเกี่ยวข้องกับการฟอกเงินหรือสนับสนุนกิจกรรมที่ผิดกฎหมาย
- ใช้ได้ทั้งในภาคเอกชนและองค์กรไม่แสวงหาผลกำไร