Prediction of Suspected Money Laundering

CHI-LIN LI, WEI-RU CHEN, SHIH-CHIH LIN, and TSUNG-HAN LIU
National Tsing Hua University, Hsinchu, Taiwan R.O.C.

***Abstract*—The goal of this paper is to use anonymized customer account transactions and alert records to identify potential money laundering activity at E.SUN Bank. We first preprocess the data and perform feature engineering based on the features and patterns present. Then, we train a machine learning model – LightGBM on the data. Finally, we use the model to predict the likelihood of money laundering for each transaction and evaluate its performance.**

***Index Terms*—Money Laundering, Supervised learning, Time Series, LightGBM, Feature Engineering**

I. Introduction

Money laundering involves disguising the proceeds of illegal activities as legitimate funds which bring huge threats to finance industries. Therefore, detecting and preventing it helps to disrupt criminal networks, protect the financial system, and safeguard the public from the negative effects of illicit activities. This study aims to build a machine-learning model to detect the probability of money laundering. The data is provided by E.SUN Bank, which includes anonymized customer account transactions and alert records. Our goal is to develop a methodology that is capable of more accurately identifying suspicious transactions that should be reported and discovering hidden money laundering patterns. On the other hand, the false positive rate of the current bank system is so high that it's time-consuming and costly when manually evaluating the alert event. To address this issue, the model performance is measured by the precision of recall@N-1. In addition, we provide our insight into data and lessons learned, hopefully contributing to the development or implementation of an anti-money laundering system in the near future.

II. Related Work

In recent years, researchers have become increasingly interested in the Anti-Money Laundering (AML) problem [1]. The previous studies could be simply categorized into four aspects: Rule-based, Clustering, Classification, and Anomaly Detection. For classification and anomaly detection, various algorithms are employed in recent literature [2] as shown in Figure 1. Among the models, the paper [3] proposes a machine learning model for identifying suspicious financial transactions that may be related to money laundering. The authors found that their developed method using a supervised machine learning model – LightGBM [4] outperformed the bank's current approach to detecting money laundering transactions. On the other hand, the literature [5] analyzed the significant value of common features, indicating that occupation, country of origin, and the number of different account types across money laundering risk groups can effectively discriminate between suspicious and normal events.

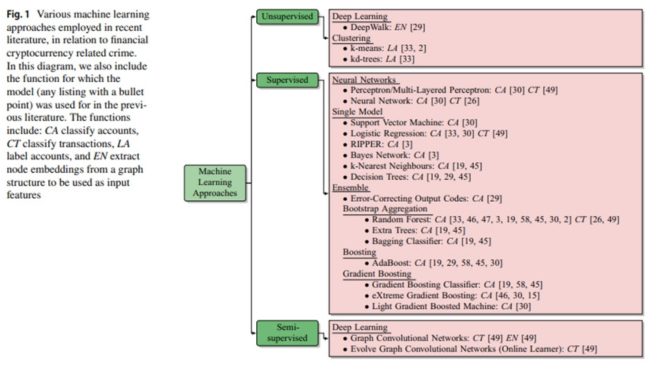

Fig. 1 Various machine learning approaches employed in recent literature, in relation to financial cryptocurrency related crime. In this diagram, we also include the function for which the model (any listing with a bullet point) was used for in the previous literature. The functions include: *CA* classify accounts, *CT* classify transactions, *LA* label accounts, and *EN* extract node embeddings from a graph structure to be used as input features

Machine Learning Approaches
Unsupervised
Deep Learning
• DeepWalk: EN [29]
Clustering
• k-means: LA [33, 2]
• kd-trees: LA [33]
Supervised
Neural Networks
• Perceptron/Multi-Layered Perceptron: CA [30] CT [49]
• Neural Network: CA [30] CT [26]
Single Model
• Support Vector Machine: CA [30]
• Logistic Regression: CA [33, 30] CT [49]
• RIPPER: CA [3]
• Bayes Network: CA [3]
• k-Nearest Neighbours: CA [19, 45]
• Decision Trees: CA [19, 29, 45]
Ensemble
• Error-Correcting Output Codes: CA [29]
Bootstrap Aggregation
• Random Forest: CA [33, 46, 47, 3, 19, 58, 45, 30, 2] CT [26, 49]
• Extra Trees: CA [19, 45]
• Bagging Classifier: CA [19, 45]
Boosting
• AdaBoost: CA [19, 29, 58, 45, 30]
Gradient Boosting
• Gradient Boosting Classifier: CA [19, 58, 45]
• eXtreme Gradient Boosting: CA [46, 30, 15]
• Light Gradient Boosted Machine: CA [30]
Semi-supervised
Deep Learning
• Graph Convolutional Networks: CT [49] EN [49]
• Evolve Graph Convolutional Networks (Online Learner): CT [49]

Fig. 1: Approaches on AML from Previous Literatures

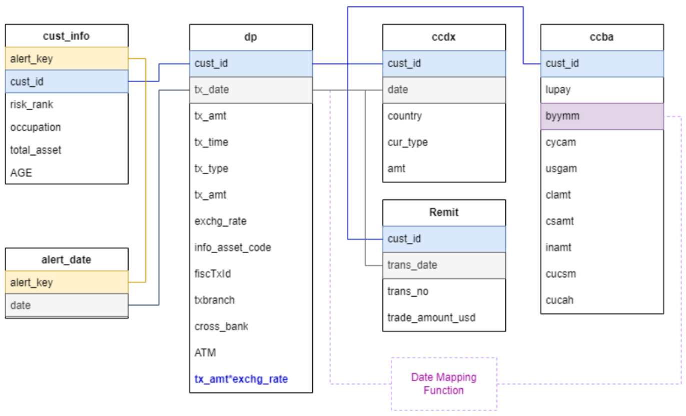

Fig. 2: Structure of E.Sun Data

III. Methodology

A. Data Overview

Before training the machine learning models, we must have a certain understanding of the data in order to further analyze the money laundering anomaly detection problem. The following table lists the data tables released in this competition. We studied, analyzed these tables, and listed the key attributes.

B. Database Structure

As shown in Figure 2, The universal key to join most tables is "cust_id". However, the "event_key" in alert records is the final prediction object, which lacks "cust_id". Therefore, the first step is to look up the corresponding "cust_id" in the cust_info table. For utilizing and transforming time series data, "date" (may be named differently in different tables) is a must.

Data	Description	Key Attributes
cust_info	Customer personal information	customer id, alert_key, risk_rank(0-3), occupation, total_asset, and ages
dp	Customer loan transaction information	customer id, transaction datetime, transaction type, transaction amounts, exchange rate, information asset code, transaction code, transaction branch, transaction by cross bank, and transaction by ATM
cdtx	Consumption transaction data	customer id, consumption date, consumption country, current type, and consumption amounts
ccba	Customer personal monthly bill	customer id, total payment last month,bill date, Quota used, amount of cash advance in installments this month, cash advance amount for this month, installment spending amount for this month, amount spent this month, amount of cash borrowed this month
remit	Foreign exchange transaction data	customer id, foreign exchange trading day (accounting day), transaction number, and transaction amount (equivalent to US dollars)
y_training	Alert key and date for training	alert_key and alert_date
y_test	Alert key and date for Testing	alert_key and alert_date

TABLE I: Description and Attributes of Tables

C. Data Exploration

1) Exploration on Custinfo Columns: We did some visualizations of the columns in the cust_info table, since the frequency of SAR(Suspicious Activity Reports) and Non-SAR data is heavily biased, we normalized the data to have a better view of the data distribution. In Figure 3, the occupation column, occupation 1, 3, 11, 13, 14, 15, 17 has a higher probability of containing SAR value. For the age column, the distribution of Non-SAR is more left skewed than SAR data, meaning that customers with a higher age are more likely to be SAR. For the SAR column, we observed the strange effect that higher risk rank does not necessarily mean the likeliness to be SAR, as observed from the box plot, for example in the risk rank = 1 bucket, SAR's frequency is higher than Non-SAR's, and in the risk rank = 3 category Non-SAR's frequency is larger than SAR's. In summary, the feature age and occupation are better features for models.

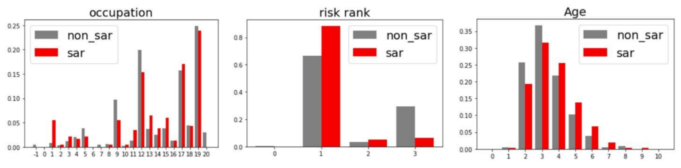

Fig. 3: Visualization of Categorical Attributes

2) Exploration on Time Series Data: We plot the transaction data of SAR and Non-SAR from the dp table. In Figure 4 and 5, there is no routine or specific pattern of normal and abnormal transaction data. It brings great difficulties in transforming or encoding them.

Data	Original # of records	Group by cust_id
custo_info	25,751	7708
dp	1,947,803	6105
cdtx	1,043,014	3945
ccba	59,075	4745
remit	17,167	1144
y_training	23,906 (alert_keys)	7264
y_test	3,850 (alert_keys)	736

TABLE II: Table Size and The Size after Group by "cust_id"

3) Data Quality: To better understand the effect of merging tables based on the "cust_id", we did some analysis on how much data is lost when merging data frames. Table II is the original table size and the size after grouping by "cust_id". Table III shows that there would be a great proportional data loss if we take more data into consideration. For example, if we only analyze the instances that have appeared in cust_info, dp (transaction record), and cdtx (consumption transaction data) tables, nearly half of the instances (train: 52%, testing: 44%) need to be dropped, which is not acceptable. Therefore, we take different strategies while merging different tables. First of all, basic transaction records should not be ignored, so we drop the instance that doesn't have any record in table dp. For the rest of the tables, we tend to fill the missing value by median and mode.

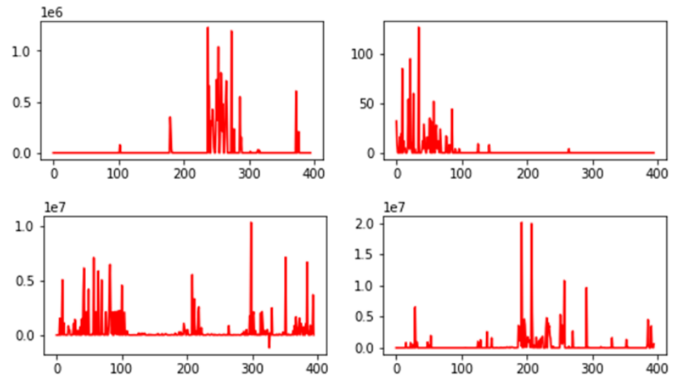

Fig. 4: Abnormal(SAR) Transaction

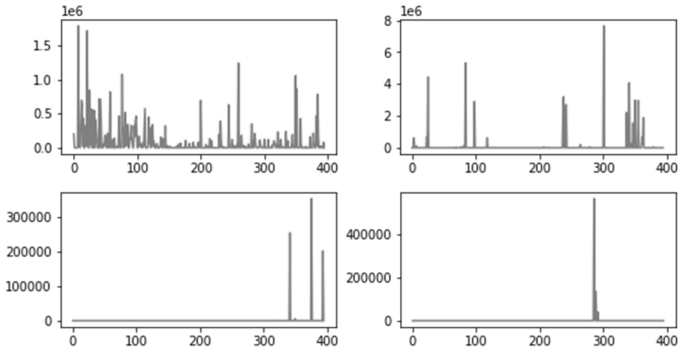

Fig. 5: Normal Transaction

Data	cust_info	Custo_info and dp	Custo_info, dp, and cdtx	Custo_info, dp, cdtx, and ccba	Custo_info, dp, cdtx, ccba, and remit
y_train	7264	5762 (-20%)	3466 (-52%)	3466 (-52%)	687 (-91%)
y_test	736 (alert_key #:1845)	706 (-4%)	409 (-44%)	409 (-44%)	104 (-86%)

TABLE III: Data Size after Inner Merge with Other Dataframes

4) Data Imbalanced: The data is highly imbalanced. Table IV shows the ratio of class is around 1:101 (234:23672). Intuitively, sampling methods and One-Class model need to be applied. However, the result doesn't work well. We will discuss the possible reasons in Section V.

Class	#
Non-SAR (Normal Transaction)	23672
y_test SAR	234

TABLE IV: Data Size after Inner Merging

D. Data Preprocessing

We use two aspects to do data preprocessing. And also uses the StandardScaler function to normalize some columns such as transaction amounts and total assets.for null data, we fill suitable values based on characteristics of different columns. For example, we fill in median values if columns are numerical.

1) Customer information Base Preprocessing: We merge train_x_alert_date and train_y_answer to a new dataframe: alert_train (alert_key / date / flag). Then, we merge cust info and alert_train for baseline model training.

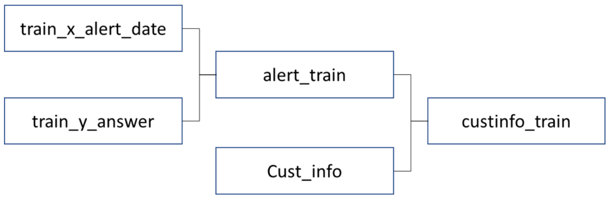

Fig. 6: Pre-processing workflow of customer-based

2) Transaction base preprocessing: We use another aspect to merge transaction data. The primary key for both dp and cdtx tables are "cust_id" and "transaction date" so we merge these two tables to obtain the dp_cdtx table. Then, we merge dp_cdtx and cust info table for model training.

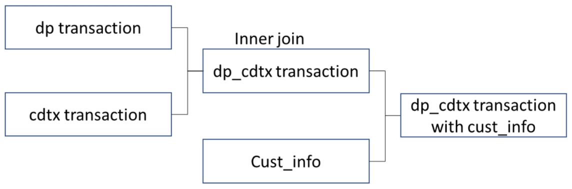

Fig. 7: Pre-processing Workflow of Transaction-based

E. Feature Engineering

1) Cust_info Data with Time Series Data: We extend the Section III-D1 and merge the time series data (mean, std, and counts of ccba/cdtx/dp/remit table) to a new dataframe (custinfo_features) for future training shown in Figure 8 below.

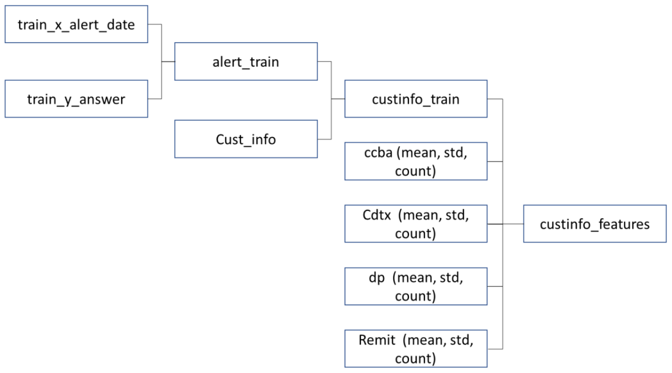

Fig. 8: Feature Engineering

2) Missing Value Handling: Before Feature Engineering, we process the data based on its attribute in Table V. If the data contains missing values, we would fill in it, transform the original data, and drop the outlier.

Types	Numerical	Categorical
Missing Value	Mean value	Mode value
Transformation	-	One-hot encode
Outlier	Drop outlier based on boxplot	-

TABLE V: Processing Strategy

F. Modeling

First, we implement our base model by Random Forest. It works by building an ensemble of decision trees and making predictions based on the majority vote of the individual trees. They can learn complex patterns in the data and make accurate predictions, making them a potentially useful tool for detecting money laundering. Secondly, we implement XGBoost and LightGBM. They all belong to gradient boosting algorithms, and can also be effective for detecting money laundering, a type of financial crime in which illicitly-gained proceeds are disguised as legitimate funds. However, it is important to note that Random Forest, XGBoost, and LightGBM, like any machine learning algorithm, are only as good as the data it is trained on. If the training data is imbalanced or does not accurately represent the characteristics of money laundering transactions, the model may not perform well. Therefore, when we develop the models, we already consider the problem of imbalanced data.

1) Hyperparameter: To control and reach the best result, the parameters of gradient boosting models should be fine-tuned. The most influential parameters include the objective function, number of each boosting tree, number of leaves in a tree, the learning rate, the tree's max depth, and our regularization term.

IV. Experiment and Result

In this section, we describe our model implementation in detail and the metrics we choose. The accuracy of a classifier is the total number of correct predictions by the classifier divided by the total number of predictions. This may be good enough for a well-balanced class but not ideal for an imbalanced class problem. The other metrics such as precision is the measure of how accurate the classifier's prediction of a specific class and recall is the measure of the classifier's ability to identify a class. In rare cases like money laundering detection or disease prediction, it is vital to identify the minority classes correctly. So the model we implemented should not be biased to detect only the majority class but should give equal weight or importance to the minority class too. For an imbalanced class dataset, F1 score is a more appropriate metric. We designed three scenarios based on the data and transformation function used. Under each scenario, different models are evaluated.

A. Scenario 1: Custinfo Data Based Table

The first baseline we tried to use only the attributes (occupation, risk rank, age, total asset) in the cust_info table, and the model used is random forest, XGBoost, and lightGBM, and the result in Table VI, Figure 9, 10, and 11.

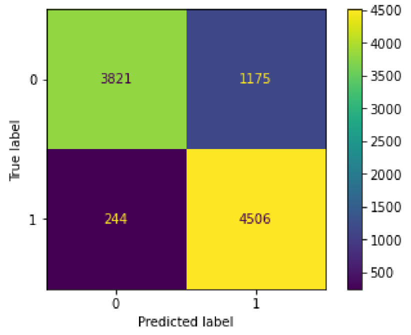

Fig. 9: Random Forest Confusion Matrix on Scenario 1

Model	Precision	Recall	F1-score
Random Forest	0.793	0.948	0.863
XGBoost	0.763	0.938	0.842
LightGBM	0.8	0.959	0.872

TABLE VI: Model Performance of Scenario 1

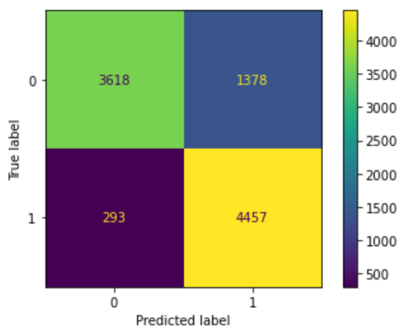

Fig. 10: XGBoost Confusion matrix on Scenario 1

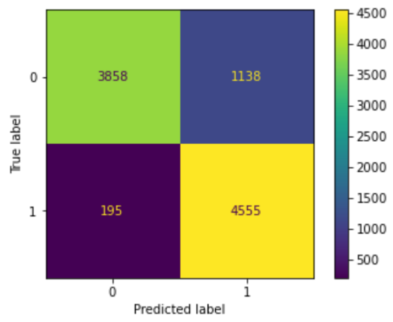

Fig. 11: LightGBM confusion matrix on Scenario 1

B. Scenario2: Customer Information Data Based Table with Time Series Features

The second attempt involves summarizing the time series data from other tables including the occurrence of cust_id in other tables and the statistical features (mean, std, sum) from other tables. And we also treat imbalance data to prevent it. This method got 0.008403 as the public score.

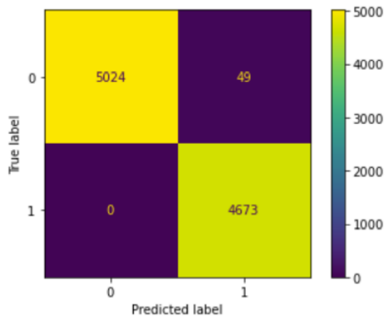

Fig. 12: Random Forest Confusion Matrix on Scenario 2

Model	Precision	Recall	F1-score
Random Forest	0.989	1.0	0.994
XGBoost	0.918	1.0	0.957
LightGBM	0.964	1.0	0.983

TABLE VII: Model Performance of Scenario 2

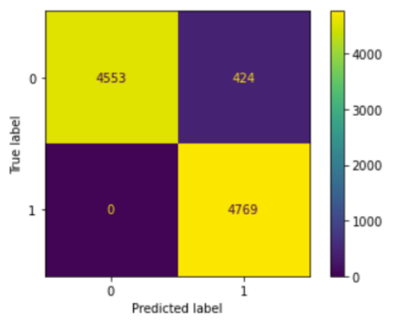

Fig. 13: XGBoost Confusion Matrix on Scenario 2

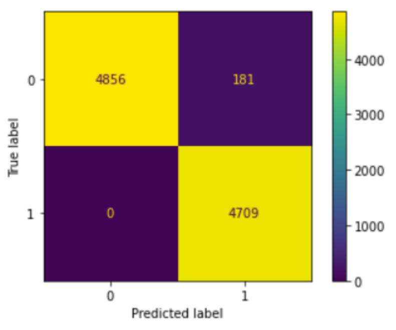

Fig. 14: LightGBM Confusion Matrix on Scenario 2

C. Scenario3: Use Time Series Transaction Data Based Table to Join Customer Information

We merged the new data which was announced on December 25, 2022 to the training dataset and implemented XGBoost and LightGBM these two models, the results were shown below.

The training result looks good in Table VIII, F1 score is very high since we use an imbalance function. But the actual result is not good because this model is already overfitting. And the training result looks good, the F1 score is very high since we use an imbalance function. But the actual result is not good because this model is already overfitting. This situation is the same as XGBoost model.

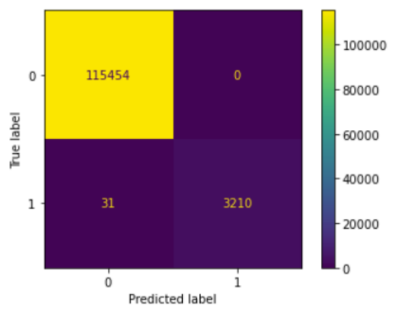

Fig. 15: XGBoost Confusion Matrix on Scenario 3

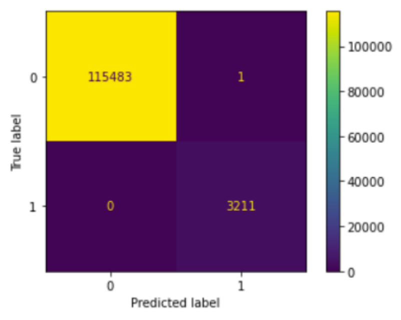

Fig. 16: LightGBM Confusion Matrix on Scenario 3

Model	Precision	Recall	F1-score
XGBoost	1.0	0.99	0.99
LightGBM	0.99	1.0	0.99

TABLE VIII: Model Performance of Scenario 3

D. Overall Performance

Under all the scenarios, the performance of XGBoost and LightGBM is actually comparable. However, LightGBM still slightly outperforms other models. Therefore, it has been chosen as our best prediction model and results in 0.026525 on the public board. Since the answer of the public board has already been revealed, there is no way to get an accurate rank estimation.

V. Discussion And Future Plan

In this session, we will discuss some methodologies we have tried but did not work well, including the sampling strategy and feature encoding. We also want to provide some insights and suggestions toward data and modeling aspects. Hopefully, these findings can be used for future AML algorithm development.

A. Data Imbalanced issue

As mentioned above, the data is highly imbalanced. Since there is no routine pattern of the transaction, it is hard to simulate new data for training. Therefore, we tried upsampling but it resulted in overfitting. The model can only learn the sample it has seen. We have tried the one-class model. The one-class model was trained only by normal transaction data. However, it results in good recall but low precision. We speculate that this is because the good transaction data is very diverse, which makes unseen good data look suspicious to the model.

B. Feature Encoding

We tried to apply a LSTM-autoencoder to learn the representation of the time series data. Since the transaction amount deviates among customers, a normalization of transaction value is applied. Figure 18 and 19 shows that the model reconstructs the data poorly. This indicates the model can not learn either normal or abnormal patterns effectively. We speculate that is also because there is no routine pattern of normal transactions. Even though we postponed this idea, we still believe a proper feature encoding method might be beneficial for improving prediction.

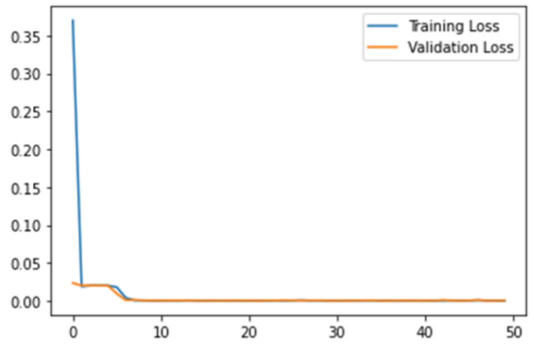

Fig. 17: Training Loss of Building LSTM-Autoencoder

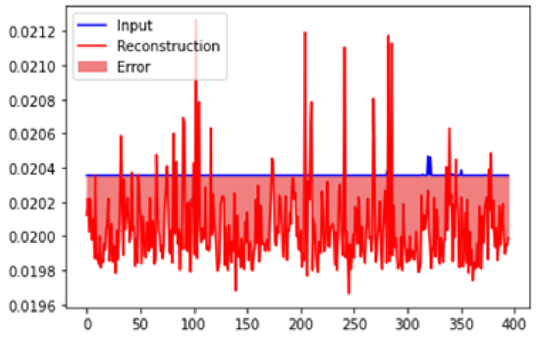

Fig. 18: Data Reconstruction Example 1

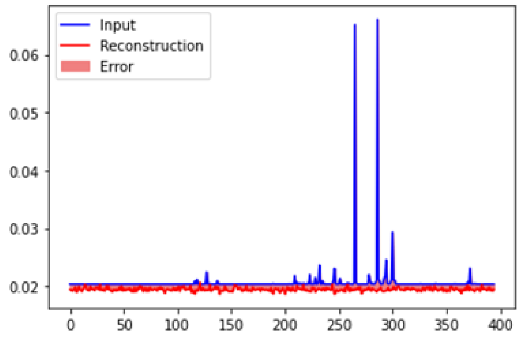

Fig. 19: Data Reconstruction Example 2

C. *Lessons Learned And Future Plan*

First, we spent a lot of time working and experimenting with prediction models based on "cust_id" instead of the alert_key-based method, so in the end, we don't have enough time to formulate a time series-based solution. Secondly, the precision of recall@N-1 provided by E-SUN should be used as the main performance evaluation tool instead of precision and recall; some of the models we tried that have reasonable precision and recall still got a bad score based on the metrics provided by E-SUN. Lastly, we first duplicated the data with SAR=1 to avoid the effect of data imbalance, but the result in performance metrics still has not improved. We need to consider more methods to handle data imbalance.

References

[1] D. V. Kute, B. Pradhan, N. Shukla, and A. Alamri, "Deep learning and explainable artificial intelligence techniques applied for detecting money laundering–a critical review," *IEEE Access*, vol. 9, pp. 82300–82317, 2021.

[2] D. Vassallo, V. Vella, and J. Ellul, "Application of gradient boosting algorithms for anti-money laundering in cryptocurrencies," *SN Computer Science*, vol. 2, no. 3, pp. 1–15, 2021.

[3] M. Jullum, A. Løland, R. B. Huseby, G. Ånonsen, and J. Lorentzen, "Detecting money laundering transactions with machine learning," *Journal of Money Laundering Control*, vol. 23, no. 1, pp. 173–186, 2020.

[4] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, 2017.

[5] D. A. Harris, K. L. Pyndiura, S. L. Sturrock, and R. A. Christensen, "Using real-world transaction data to identify money laundering: Leveraging traditional regression and machine learning techniques," *STEM Fellowship Journal*, vol. 7, no. 1, pp. 21–32, 2022.