Money Laundering Detection Using Machine Learning Project Report

By Alex Pham, CPA

Problem Statement

Money laundering poses a significant threat to the integrity and stability of financial systems worldwide. Detecting money laundering activities is challenging as criminals are employing increasingly complex methods. Traditional rule-based systems often fall short in detecting sophisticated laundering schemes due to their static nature and inability to adapt to evolving patterns.

The aim of this project is to develop and implement machine learning models to effectively identify suspicious transactions within a synthetic dataset designed to emulate real-world money laundering activities. This can then help improve detection accuracy and efficiency in identifying suspicious transactions, thereby enhancing anti-money laundering efforts.

Data Source

The Anti Money Laundering Transaction Data (SAML-D) dataset provided on Kaggle (see the References section) includes 9,504,852 transactions, of which 0.1039% are suspicious. It has 12 features and 28 typologies (split between 11 normal and 17 suspicious), incorporating diverse geographic locations, high-risk countries, and high-risk payment types. Additionally, it includes 15 graphical network structures that represent transaction flows, adding complexity to the detection challenge.

Features of the dataset:

- Transaction Time and Date
- Sender and Receiver Account Numbers
- Transaction Amount (converted to British pound sterling)
- Payment Type (e.g., credit card, debit card, cash, ACH transfers, cross-border, and cheque)
- Sender and Receiver Bank Location (e.g., the UK, Mexico, Turkey, Morocco, and the UAE)
- Payment and Receiver Currency
- 'Is Suspicious' (binary indicator differentiating normal from suspicious transactions) – the target variable
- Typology (Type of normal or suspicious transaction)

Literature Review

Assessing existing anti-money laundering (AML) literature and transaction datasets, in addition to leveraging the knowledge and experience of an AML specialist’s input, Oztas et al. (2023) synthesized the SAML-D dataset to aid researchers in evaluating their AML models and develop more advanced monitoring methods. The dataset creators also compared the SAML-D dataset to publicly available AML datasets and concluded that the SAML-D dataset provides a detailed and robust resource for AML transaction monitoring.

To establish the suitability, purpose, and applicability of their own synthesized dataset, Oztas et al. (2023) conducted statistical data analysis on it with varied machine learning models to detect suspicious transactions. They used Support Vector Machines (SVM), Naïve Bayes (NB), Decision Trees, and Random Forest as they deemed these approaches the most utilized techniques in relevant literature.

Their data preprocessing included the conversion of categorical features into numerical form via one-hot encoding and label encoding, the split of the date feature into year, month, and day, the removal of redundant columns, and the standardized rescaling of numerical variables. To build the models, they used a 70-15-15 stratified train-validation-test split. In building the models, they faced the constraint of a single GPU and thus chose to conduct the experiments on a representative subset of the SAML-D dataset. However, in the research paper there is no mention of which relevant data features were finally selected for the machine learning models or how much proportion of the data population the representative sample was.

To evaluate and assess the performance of the models, Oztas et al. (2023) used the True Positive Rate (TPR), True Negative Rate (TNR), False Positive Rate (FPR), False Negative Rate (FNR), and Area Under the Curve (AUC) score. These metrics are mentioned as widely used in AML literature. Among the metrics, the TPR is crucial since banks can face significant financial and reputational consequences if they miss suspicious transactions. The FPR, on the other hand, shows the quantification of transactions mislabeled as suspicious transactions which may incur high operational costs for banks.

Methodology

1. Machine Learning Models

To address the defined problem and introduce new modeling ideas for the SAML-D dataset, I use the following machine learning models:

- Ensemble methods: Gradient Boosting (GB), Extreme Gradient Boosting (XGBoost), and Categorical Boosting (CatBoost), in addition to Random Forest to compare my results with those experimented by the dataset creators.
- Decision Tree (to compare the results with that of the dataset creators)
- Logistic Regression

This is a slightly different approach compared to the one I mentioned in the project proposal as deep learning models such as Recurrent Neural Networks (RNN) and Graph Neural Networks (GNN) are no longer explored due to my current limits of GPU and deep learning knowledge. I may revisit the idea of using deep learning for this dataset in my future CS 7643: Deep Learning course at Georgia Tech. Also, SVM (due to its computational intensiveness) is replaced by CatBoost and Decision Tree.

In terms of boosting methods, I choose not to use AdaBoost since it does not handle the non-linearity and complexity of the dataset as effectively as gradient boosting techniques such as GB, XGBoost, and CatBoost.

2. Data Preprocessing

After checking for missing values and data duplicates, I performed one-hot encoding on the categorical variables: 'Payment_currency', 'Received_currency', ‘Sender_bank_location', 'Receiver_bank_location', and 'Payment_type' and standard-scaled the transaction ‘Amount’.

As the dataset is larger and as my computer has limited computing power, I used stratified random sampling on the dataset, creating a representative sample equivalent to 10% of the population. For simplicity, I performed stratified train-test split on the representative sample with the ratio of 75:25 instead of performing a train-validation-test split.

The training dataset was then processed further to counter the imbalance of the minority class and the majority one as only 0.1039% of the data points are labeled are suspicious. This data imbalance hinders the machine learning models’ performance.

To address the data imbalance, different techniques were used for the minority class such as the Synthetic Minority Over-sampling Technique (SMOTE), Adaptive Synthetic Sampling (ADASYN), and SMOTE-NC (Nominal Continuous) as the dataset contains both categorical and numerical features. For the majority class (non-suspicious), the Random Under Sampler was used. These combinations helped produce more balanced training sets and prevented biased model outputs.

Evaluation Strategies

To evaluate the performance of my implemented models, I use similar strategies mentioned in the literature review: True Positive Rate (TPR), True Negative Rate (TNR), False Positive Rate (FPR), and False Negative Rate (FNR) which can be inferred from confusion matrices. For feature selection, only the most relevant features will be kept for final models.

The Accuracy score is calculated for reference, but it is less crucial than the TPR and the FPR, given the business setting/implication. In addition to the Recall (TPR), Precision and F1-Score are calculated to provide more insights into the metrics of the classes – suspicious and non-suspicious. Area Under the ROC Curve (AUC-ROC) may be shown to better visualize the trade-off between the TPR and FPR.

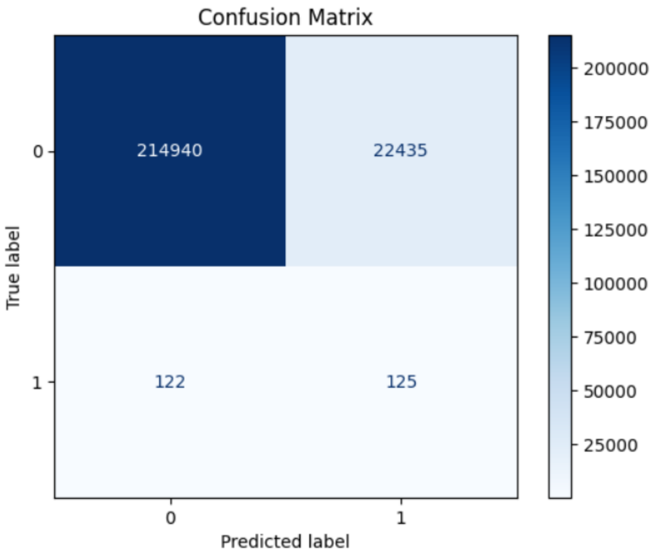
Due to time constraints and limited computing power, K-fold cross-validation is not performed although it can help ensure model robustness and generalizability.

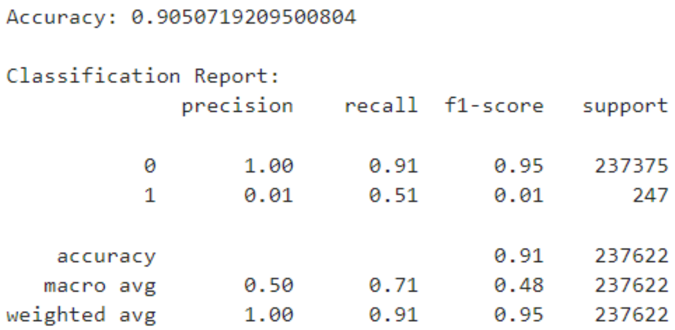
Final Results

1. More balanced (between the suspicious and non-suspicious) training datasets result in better True Positive Rate (TPR) model performance than less balanced training datasets do.

   For example, below is the comparison for the Gradient Boosting (GB) model with SMOTE and Random Under Sampler being used.

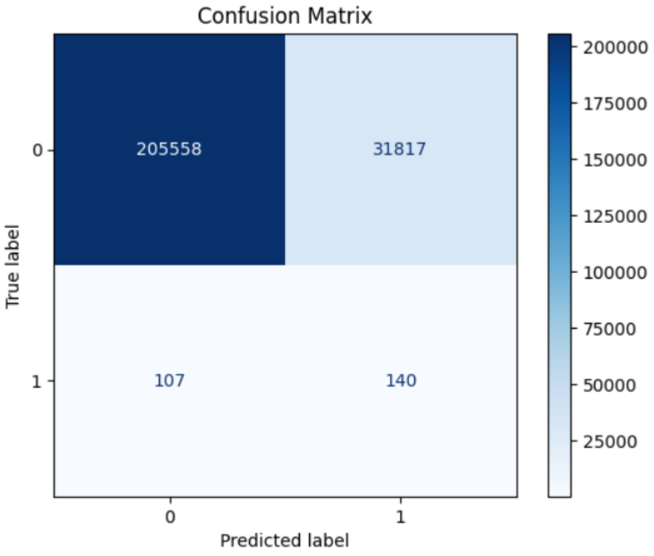
   Results with the 1:2 ratio between suspicious and non-suspicious in the training set.

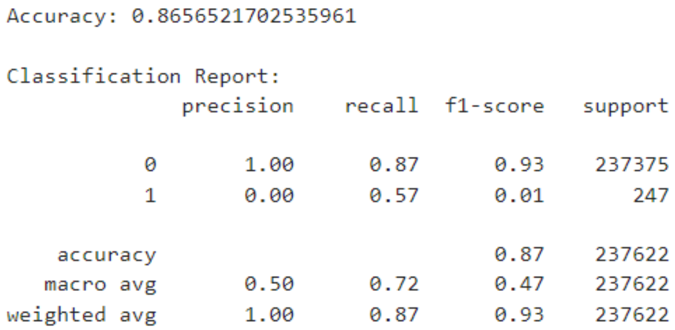




```
Accuracy: 0.9050719209500804

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.91      0.95    237375
           1       0.01      0.51      0.01       247

    accuracy                           0.91    237622
   macro avg       0.50      0.71      0.48    237622
weighted avg       1.00      0.91      0.95    237622
```

Results with the 1:1 ratio between suspicious and non-suspicious transactions in the training set.



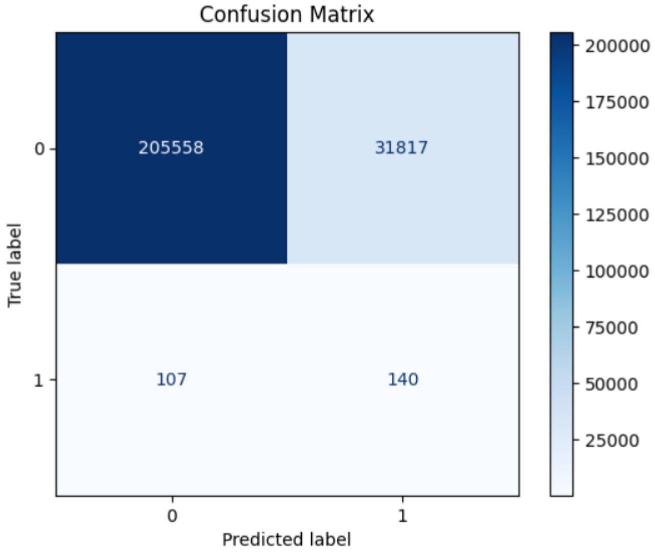


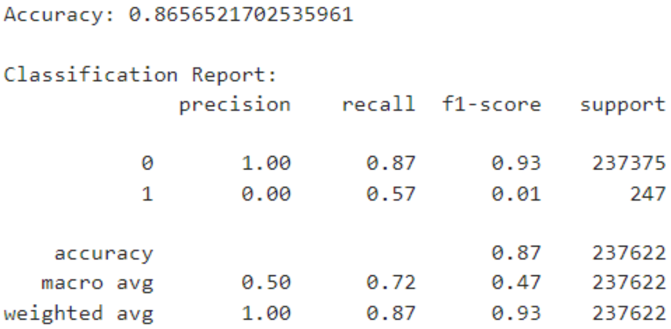
```
Accuracy: 0.8656521702535961

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.87      0.93    237375
           1       0.00      0.57      0.01       247

    accuracy                           0.87    237622
   macro avg       0.50      0.72      0.47    237622
weighted avg       1.00      0.87      0.93    237622
```

2. The Gradient Boosting (GB) model outperforms the other models.

   The results below are based on models built from the training set with the 1:1 ratio between suspicious and non-suspicious data points. The metrics are calculated from the test set.

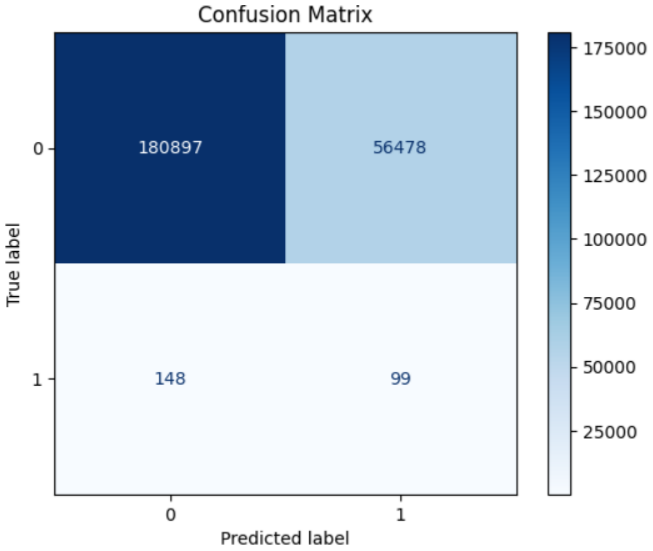
   GB model performance

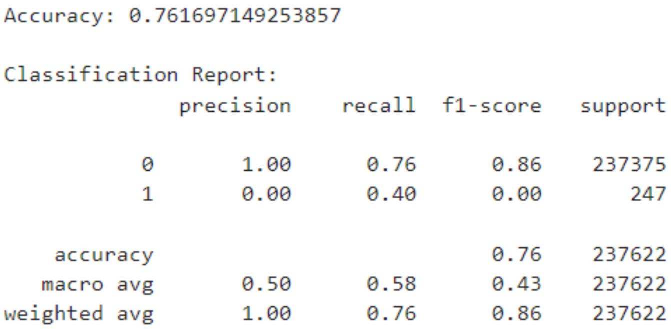




```
Accuracy: 0.8656521702535961

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.87      0.93    237375
           1       0.00      0.57      0.01       247

    accuracy                           0.87    237622
   macro avg       0.50      0.72      0.47    237622
weighted avg       1.00      0.87      0.93    237622
```

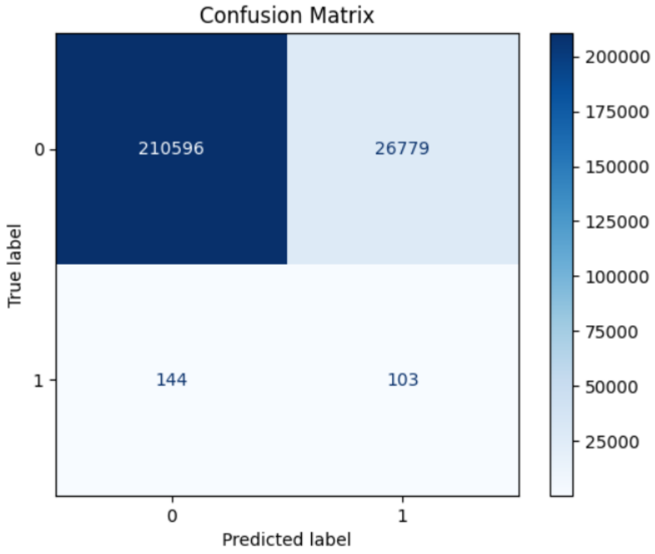
   Random Forest model performance

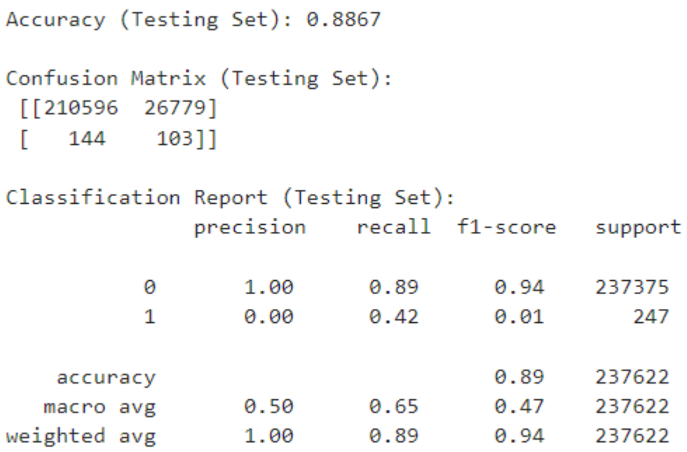




```
Accuracy: 0.761697149253857

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.76      0.86    237375
           1       0.00      0.40      0.00       247

    accuracy                           0.76    237622
   macro avg       0.50      0.58      0.43    237622
weighted avg       1.00      0.76      0.86    237622
```

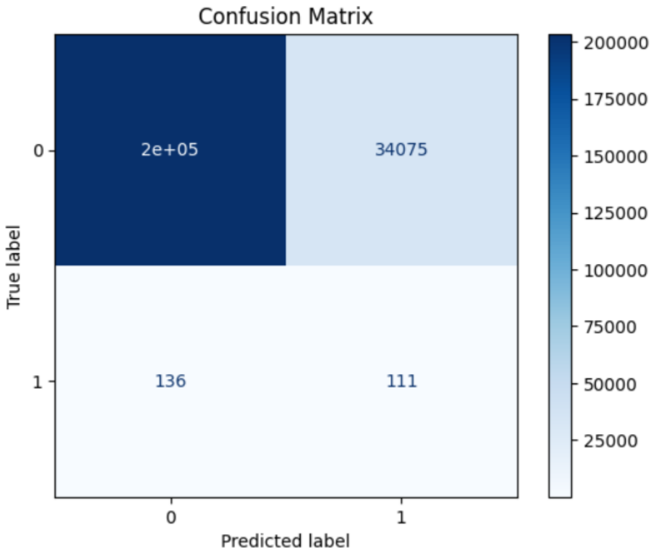
XGBoost model performance

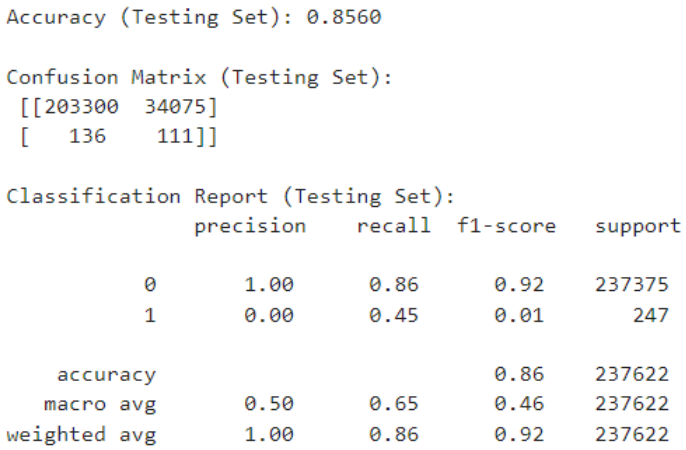




```
Accuracy (Testing Set): 0.8867

Confusion Matrix (Testing Set):
 [[210596  26779]
 [   144    103]]

Classification Report (Testing Set):
               precision    recall  f1-score   support

           0       1.00      0.89      0.94    237375
           1       0.00      0.42      0.01       247

    accuracy                           0.89    237622
   macro avg       0.50      0.65      0.47    237622
weighted avg       1.00      0.89      0.94    237622
```

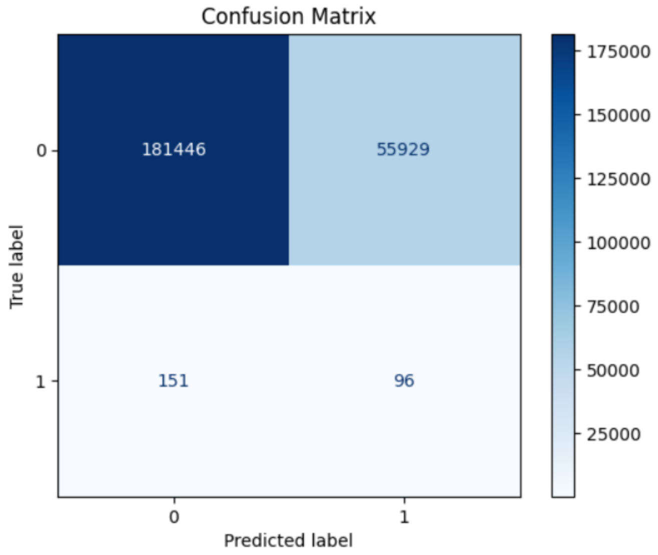
CatBoost model performance

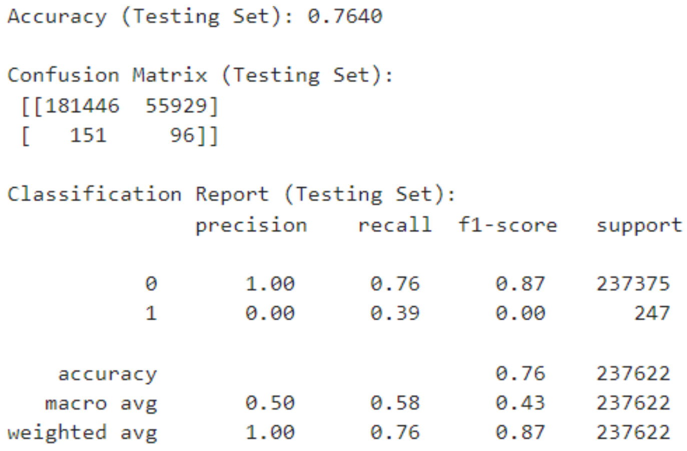




```
Accuracy (Testing Set): 0.8560

Confusion Matrix (Testing Set):
 [[203300  34075]
 [   136    111]]

Classification Report (Testing Set):
               precision    recall  f1-score   support

           0       1.00      0.86      0.92    237375
           1       0.00      0.45      0.01       247

    accuracy                           0.86    237622
   macro avg       0.50      0.65      0.46    237622
weighted avg       1.00      0.86      0.92    237622
```

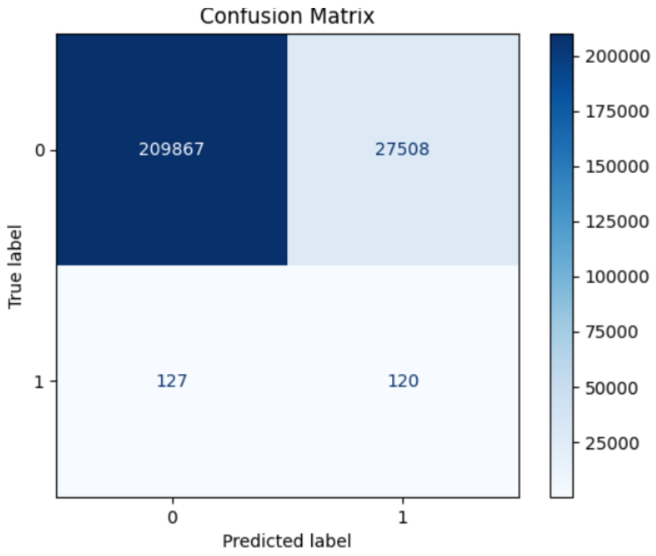
Decision Tree model performance

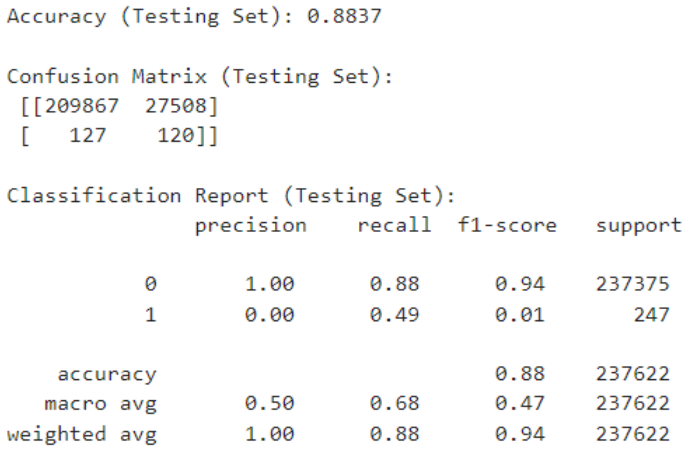




```
Accuracy (Testing Set): 0.7640

Confusion Matrix (Testing Set):
 [[181446  55929]
 [   151     96]]

Classification Report (Testing Set):
              precision    recall  f1-score   support

           0       1.00      0.76      0.87    237375
           1       0.00      0.39      0.00       247

    accuracy                           0.76    237622
   macro avg       0.50      0.58      0.43    237622
weighted avg       1.00      0.76      0.87    237622
```

Logistic Regression model performance

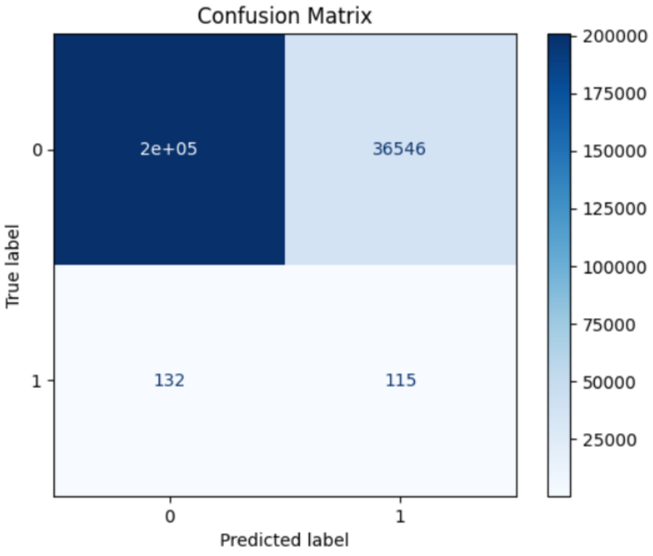


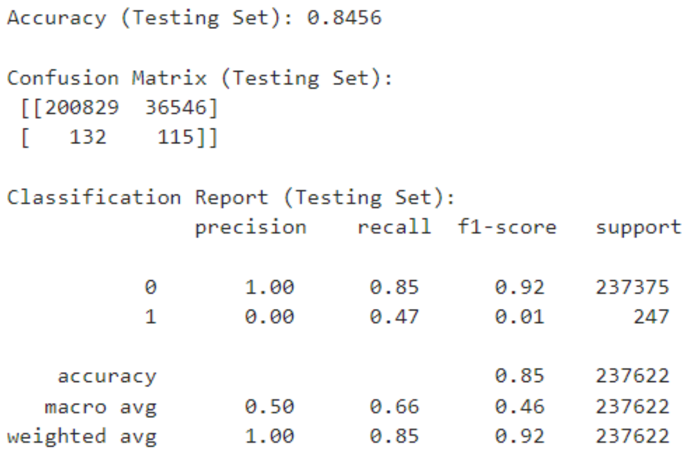


```
Accuracy (Testing Set): 0.8837

Confusion Matrix (Testing Set):
 [[209867  27508]
 [   127    120]]

Classification Report (Testing Set):
              precision    recall  f1-score   support

           0       1.00      0.88      0.94    237375
           1       0.00      0.49      0.01       247

    accuracy                           0.88    237622
   macro avg       0.50      0.68      0.47    237622
weighted avg       1.00      0.88      0.94    237622
```

3. Models built on training sets with the combination of ADASYN and Random Under Sampler being used or with the combination of SMOTE-NC and Random Under Sampler being used result in slightly better performance results for TPR than those with the combination of SMOTE and Random Under Sampler being used to address the data imbalance. This is probably because the ADASYN and SMOTE-NC better capture the density of the suspicious data points during the minority oversampling process.

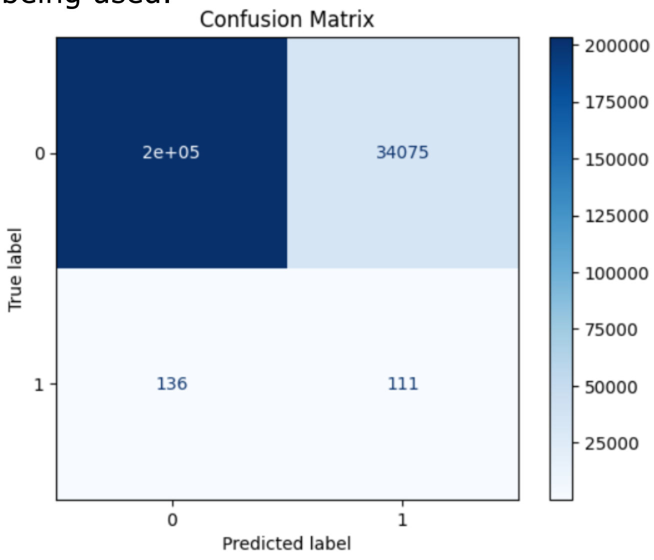
   For example, CatBoost results with the combination of ADASYN SMOTE and Random Under Sampler being used

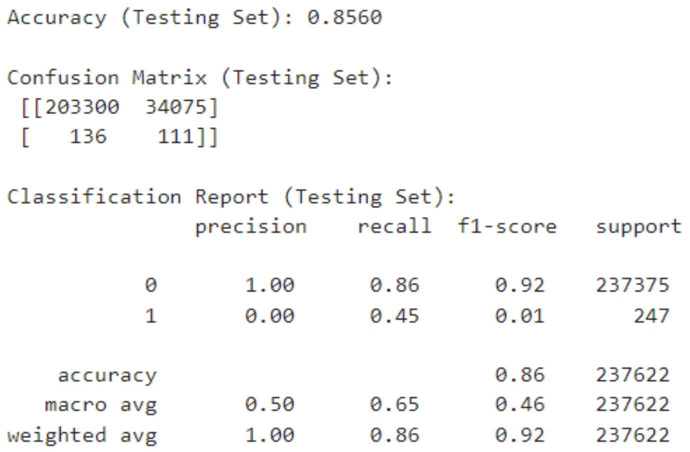




```
Accuracy (Testing Set): 0.8456

Confusion Matrix (Testing Set):
 [[200829  36546]
 [   132    115]]

Classification Report (Testing Set):
              precision    recall  f1-score   support

           0       1.00      0.85      0.92    237375
           1       0.00      0.47      0.01       247

    accuracy                           0.85    237622
   macro avg       0.50      0.66      0.46    237622
weighted avg       1.00      0.85      0.92    237622
```

versus SMOTE results with the combination of ADASYN SMOTE and Random Under Sampler being used.

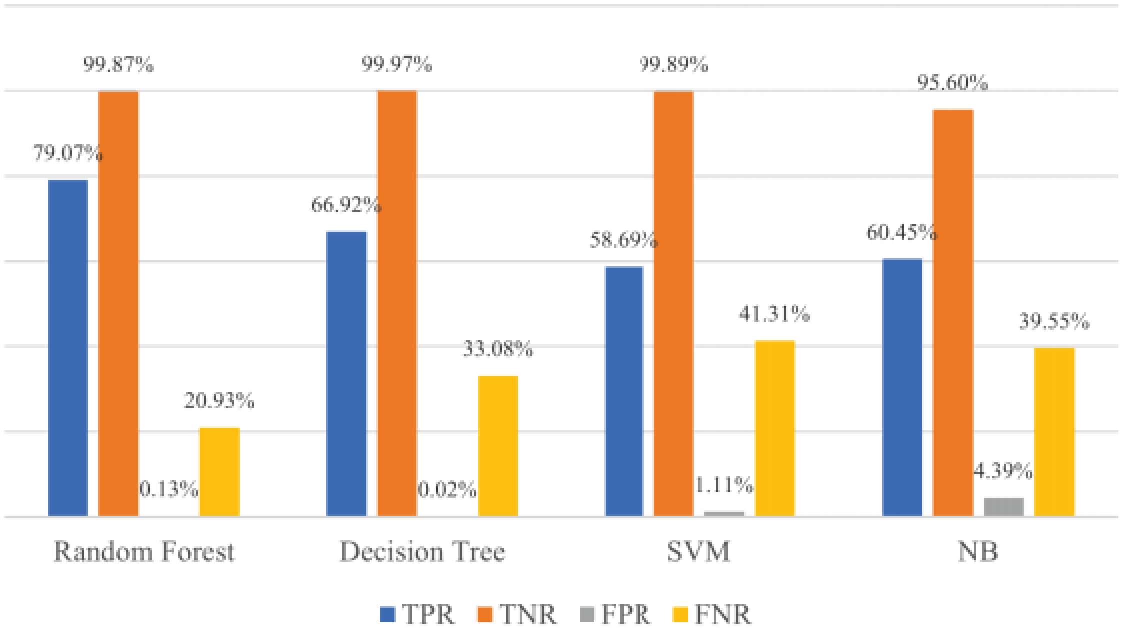




```
Accuracy (Testing Set): 0.8560

Confusion Matrix (Testing Set):
 [[203300  34075]
 [   136    111]]

Classification Report (Testing Set):
              precision    recall  f1-score   support

           0       1.00      0.86      0.92    237375
           1       0.00      0.45      0.01       247

    accuracy                           0.86    237622
   macro avg       0.50      0.65      0.46    237622
weighted avg       1.00      0.86      0.92    237622
```

4. My implemented machine learning model results, however, do not outperform those experimented by the dataset creators.

   Below is the visualization (in the research paper) of the model performance of the machine learning models that the dataset creators used on the dataset. The differences in model performance (especially in the cases of Random Forest and Decision Tree) can be attributed to sampling error, feature selection, and hyperparameter tuning.



5. Recommendations for future research

Although via stratified random sampling, the representative 10% sample that I used may not best capture the patterns of the full population. Increasing sample size can help reduce sampling error and lead to better model performance. Ideally, the full population can be used given sufficient computing power.

In addition, I would suggest deep learning models be explored for the SAML-D dataset to capture the complexity patterns in the data.

References

(1) B. Oztas, D. Cetinkaya, F. Adedoyin, M. Budka, H. Dogan and G. Aksu, "Enhancing Anti-Money Laundering: Development of a Synthetic Transaction Monitoring Dataset," 2023 IEEE International Conference on e-Business Engineering (ICEBE), Sydney, Australia, 2023, pp. 47-54, doi: 10.1109/ICEBE59045.2023.00028.

Dataset download via https://www.kaggle.com/datasets/berkanoztas/synthetic-transaction-monitoring-dataset-aml/data <accessed June 23, 2024>