Fraud Risk Detection and Adversarial Attack Defense

Jing Xia, Siyuan Liu, Yunyu Guo, Yuqing Qiao

Implementation Code:

CS5100 final proj.ipynb

ShadowModel - Privacy ML.ipynb

Introduction

Fraud detection is essential for financial institutions, but traditional rule-based systems often miss sophisticated patterns. Machine learning models can identify complex fraud behaviors more effectively. This project aims to enhance fraud detection accuracy and efficiency using machine learning models, focusing on features like transaction amounts, transaction categories, age, gender, merchants and user behavior. Additionally, transaction data is highly sensitive, and clients—such as banks, financial technology companies, and buyers—are often reluctant to trust third-party environments due to privacy concerns. Even within the same banking institution, it is crucial to control and limit access to information across departments to mitigate risk. Therefore, a secure method to protect data privacy while maintaining model performance is necessary.

Our goal is to develop a machine learning model that detects fraudulent transactions by analyzing transaction details and patterns. To ensure privacy, the model must be capable of processing encrypted data received from users and responding to client requests with encrypted outputs. Only the client will be able to decrypt the response they receive. Thus, no sensitive information can be revealed outside the client environment.

Related Works

There are several methods that are commonly used for fraud detection, such as Decision Tree, Random Forest, Support Vector Machine, Neural Network and Naive Bayes.

Dataset

BankSim is an agent-based simulator of bank payments based on a sample of aggregated transactional data provided by a bank in Spain. The main purpose of BankSim is the generation of synthetic data that can be used for fraud detection research.

The FHE library https://github.com/OpenMined/TenSEAL and https://openfhe-development.readthedocs.io/en/latest/ will be researched to implement the security component for the model training and inference.

Exploratory Data Analysis (EDA)

See Colab for full data features

Data Features

- no NAs
- Step: represents the day when the transaction happened.
- Customer: represents the unique ID of the person who initialized the transaction. It is formed by the letter C, followed by a unique sequence of 10 numbers. There are a total of 4,109 unique customers available in the dataset.
- Age: this variable is split in age intervals, starting from 0 to 6 and the letter U which stands for Unknown. Age is Unknown only for transactions that have the gender equal to Enterprise. The coding for the numbers is:
  0: less than 18 years old
  1: between 19 and 25 years old
  2: between 26 and 35 years old
  3: between 36 and 45 years old
  4: between 46 and 55 years old
  5: between 56 and 65 years old
  6: older than 65 years old
- Gender: this variable is coded as F for Female, M for Male, E for Enterprise and U for Unknown. The Unknown group has around 170 customers aged in groups 1, 2 and 3.
- Merchant: this variable represents the unique ID of the party which receives the transaction. Similar to customer ID, the sequence is formed by the letter M, followed by a series of 9 numbers. There are a total of 50 unique merchants in the dataset.
- Category: there are 15 unique categories that label the general type of the transaction: transportation, food, health, wellness and beauty, fashion, bars and restaurant, hyper, sports and toys, tech, home, hotel services, other services, contents, travel, leisure.
- Amount: represents the value of the transaction. There are only 52 values equal to 0 and no negative values.
- Fraud: a flag column coded with 0 if the transaction was clean and with 1 if the transaction was fraudulent.

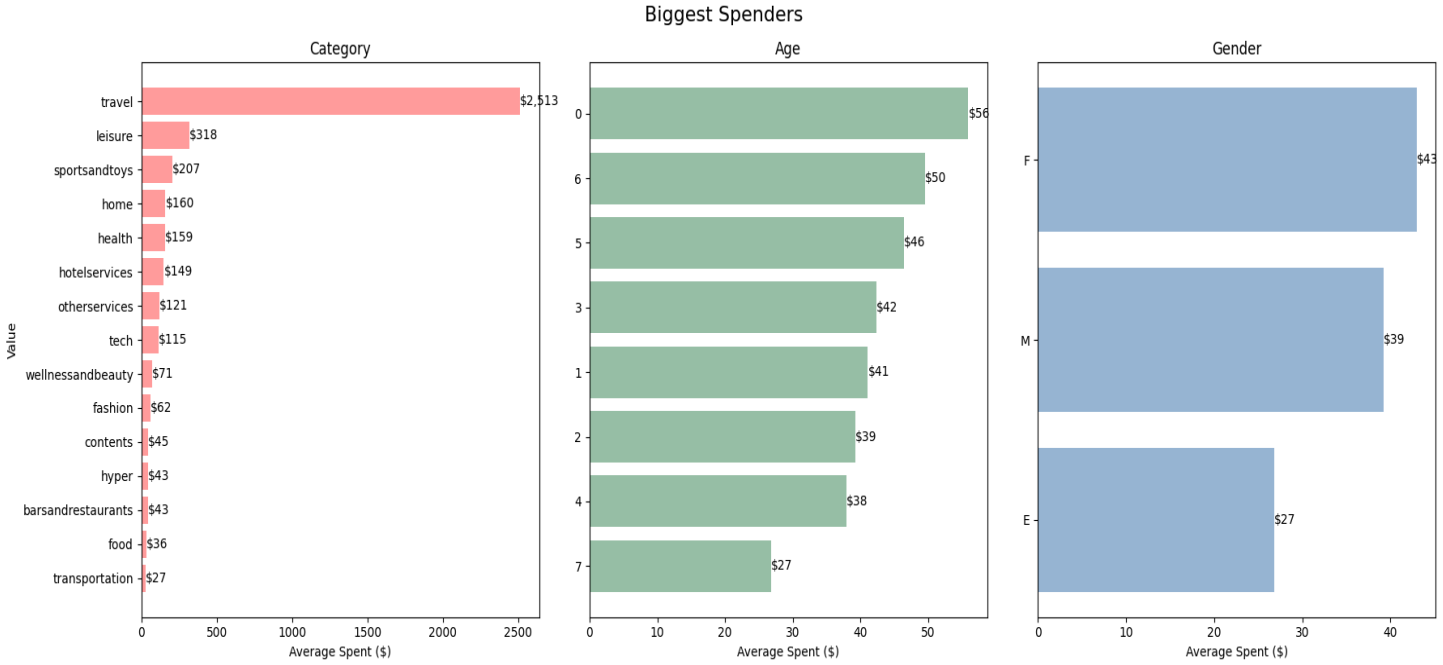
- zipcodeOri and zipMerchant: these two features were removed from the dataset, as they contained a constant value of 28007, which is a postal code in Ansonville, North Carolina, United States. Therefore, the amount will be from now on expressed in US dollars.

All Unknown for Age will be kept. It appears that only Enterprise gender doesn't have its age, so the missingness is systematic. The U will be replaced with number 7.
All Unknown for Gender will be erased. It appears that the age for these people are in 1, 2 and 3 intervals, which each has an average distribution of F = 55%, M = 44%, U = 1%.

Data Processing

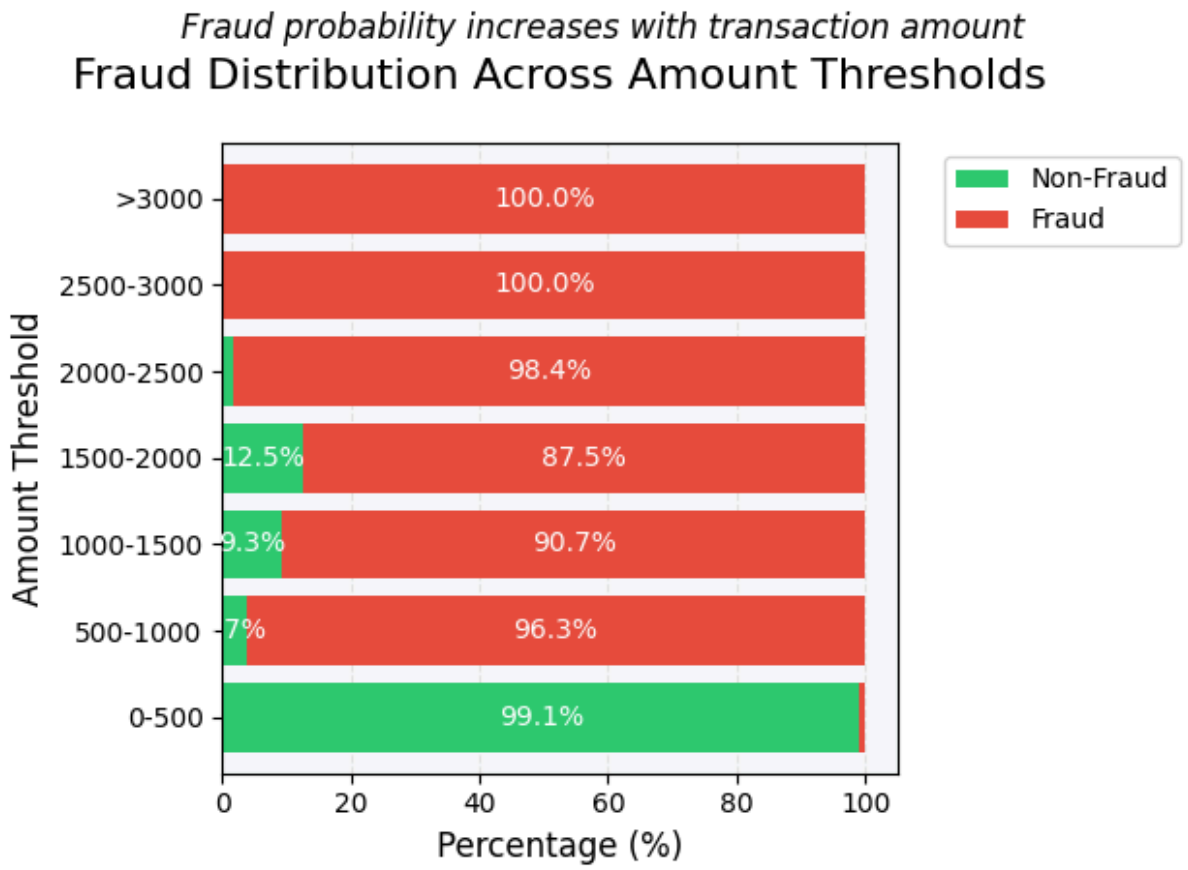
Clean the data: drop unnecessary columns(step, zipcodeOri and zipMerchant), remove punctuations and replace Unknown data in age with 7. Encode features to prepare for machine learning model training.

Average spending by category, age and gender



Fraud Percentage for Spent Amount Thresholds

There are close to no fraudulent cases in transactions below $500. The percentage jumps to almost 90% or more when the amounts are beyond the $500 threshold.

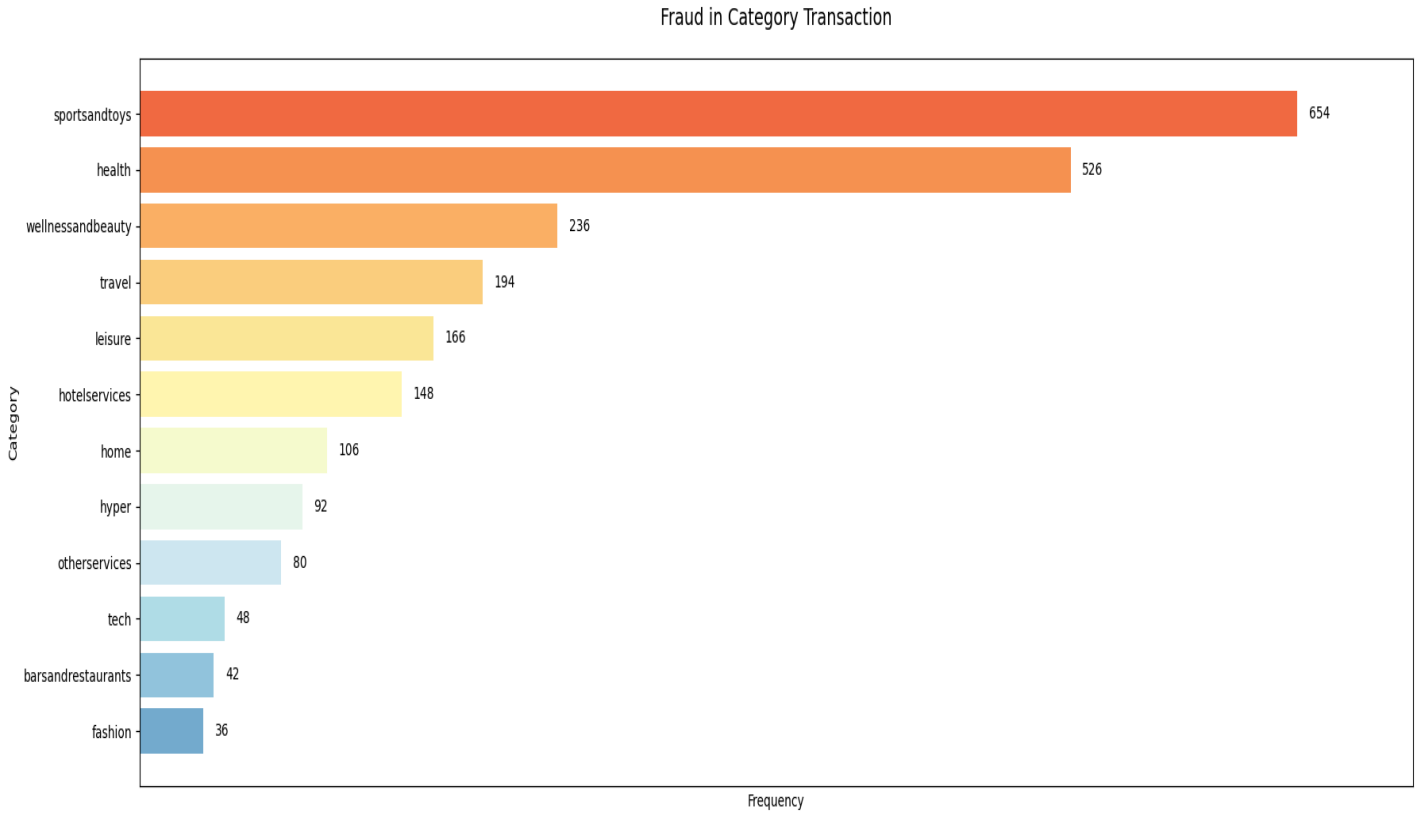


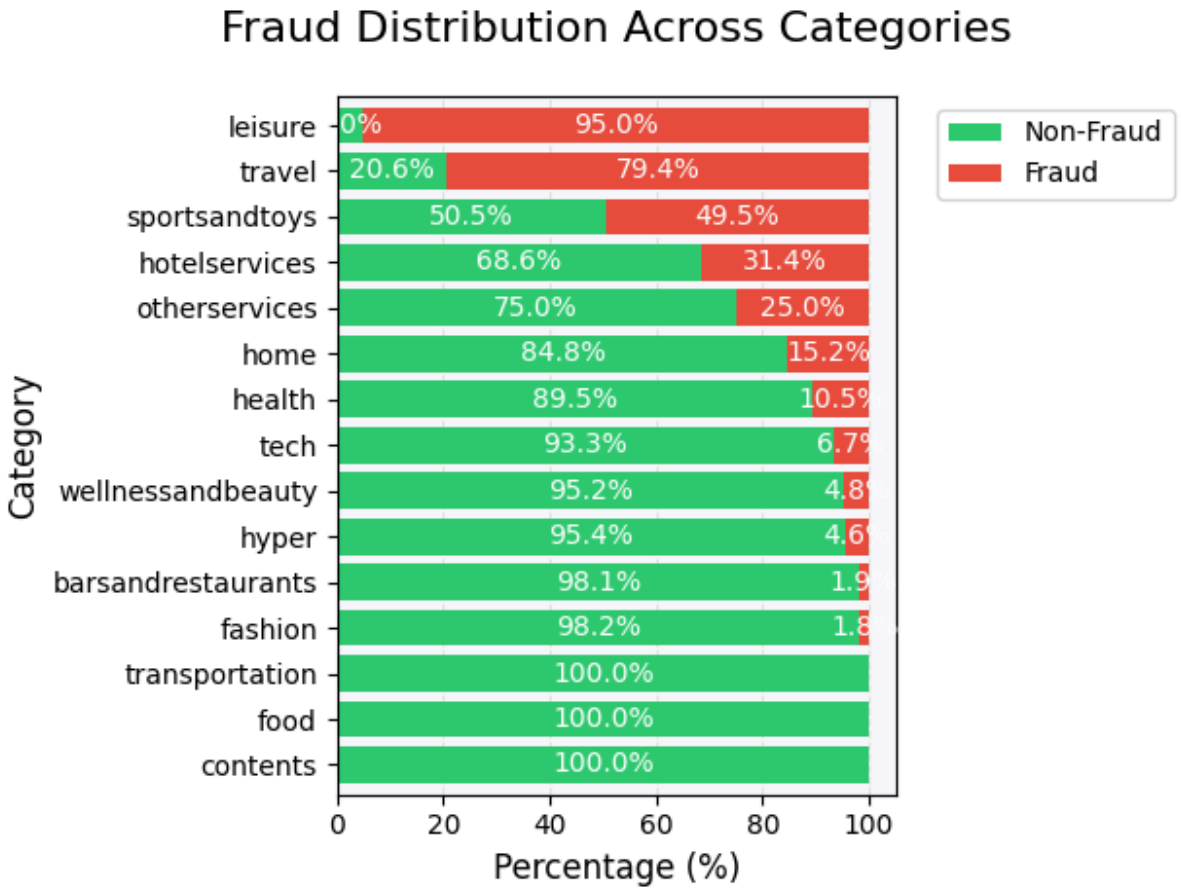
amount_thresh	Non-Fraud	Fraud
0-500	99.13	0.87
500-1000	3.74	96.26
1000-1500	9.35	90.65
1500-2000	12.50	87.50
2000-2500	1.56	98.44
2500-3000	NaN	100.00

>3000 NaN 100.00

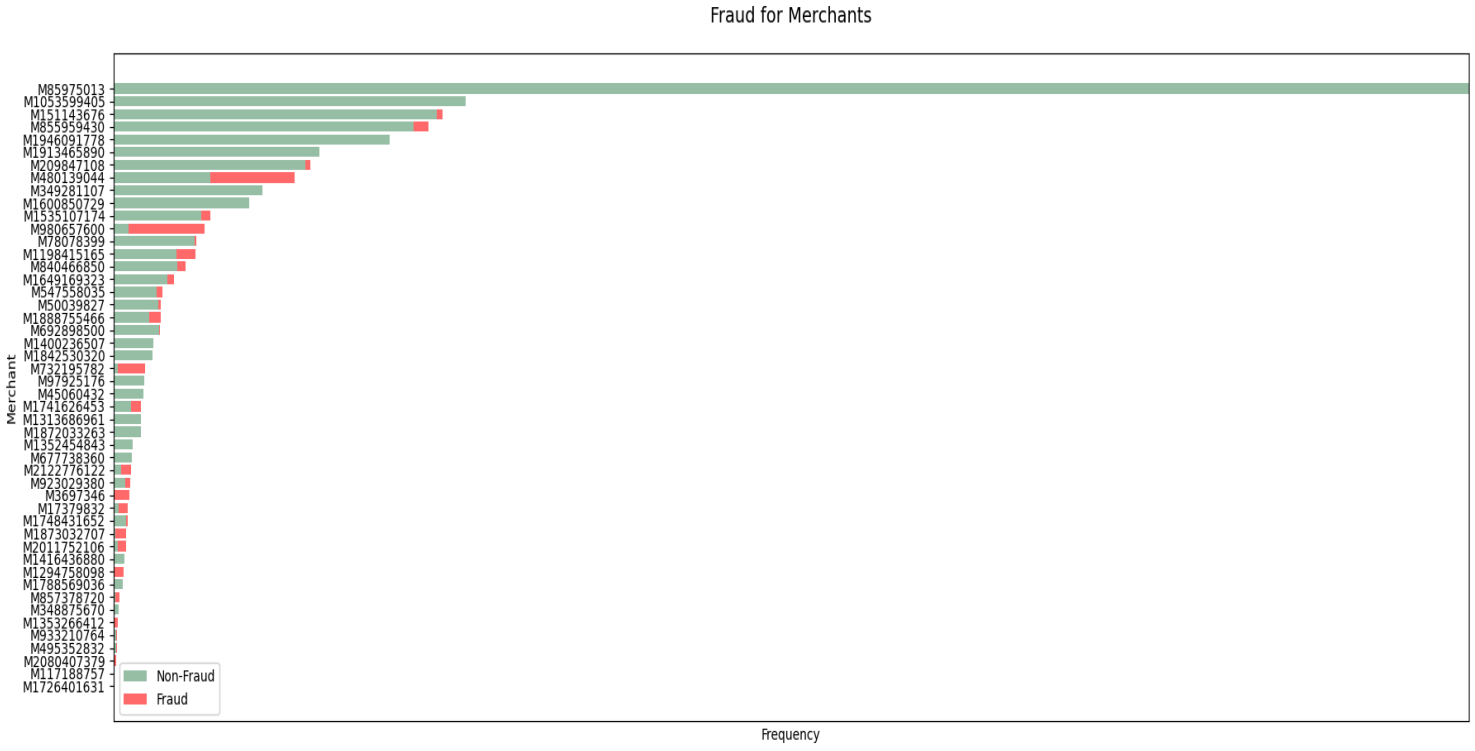
Fraud in Spending Categories

The categories that have the most fraud cases are in areas like sports and toys, health, wellness and leisure related categories. Leisure, travel, sports and toys have more than 50% of the transactions flagged as fraud, while transportation, food and contents have 100% clean cases.





Merchants and Fraud



Using Oversampling to Balance the Data

In our dataset, the fraud and non-fraud cases are extremely imbalanced, with only 1.2% out of 586,928 total cases being fraud. In a classification problem, this would create difficulties for a model to correctly identify the fraud label, because it is so scarce throughout the dataset.

This data structure issue can be solved by using the Synthetic Minority Oversampling Technique (SMOTE). Unlike Random Over Sampling, SMOTE does not create exact copies of observations, but creates new, synthetic, samples that are quite similar to the existing observations in the minority class.

AI Methods

In this project, we aim to identify the most effective machine learning model for fraud transaction detection. We have selected five distinct models based on their versatility and effectiveness in classification tasks. These models are:

- **Random Forest:** An ensemble method known for its robustness and ability to handle complex, high-dimensional datasets.
- **K Neighbors Classifier**: A simple, instance-based learning algorithm that performs well on smaller, less noisy datasets.
- **Logistic Regression**: A linear model commonly used for binary classification tasks, offering interpretability and efficiency.
- **XGBoost Classifier**: A gradient boosting model that often provides state-of-the-art performance by combining multiple weak learners.
- **Multilayer Perceptron (MLP)**: A deep learning model capable of capturing nonlinear relationships in the data, particularly useful for complex patterns.

To determine the best-performing model, we will evaluate their performance using metrics such as precision, recall, F1 score, accuracy, ROC Curve and AUC, Precision-Recall Curve and AUC. These metrics will help assess each model's ability to correctly identify fraud while minimizing false positives and negatives.

XGBoost

XGBoost is an implementation of Gradient Boosted decision trees(GBDT). XGBoost models majorly dominate in many Kaggle Competitions.

In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed

into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and these variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.

Results

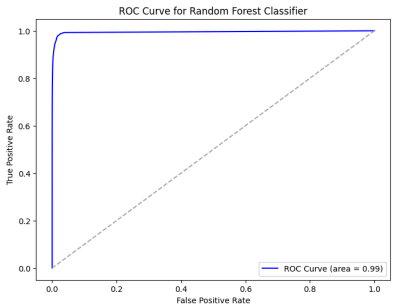
Models performance (without resampling):

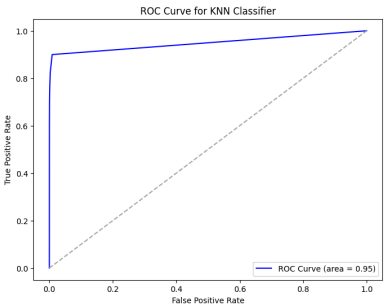
Precision, recall and F1-score:

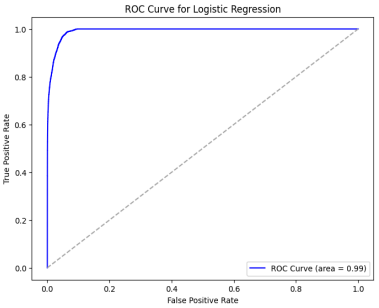
Model	Precision	Recall	F1-score
Random Forest	0.95	0.83	0.88
KNN	0.93	0.79	0.85
Logistic Regerssion	0.58	0.93	0.62
XGBoost	0.93	0.86	0.89
MLP	0.94	0.71	0.79

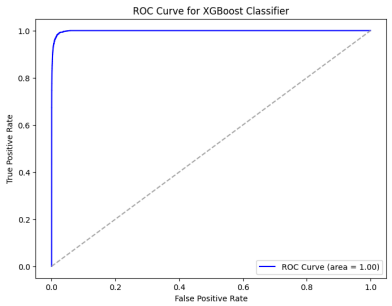
XGBoost and Random forest have the best performance based on precision, recall and F1-score.

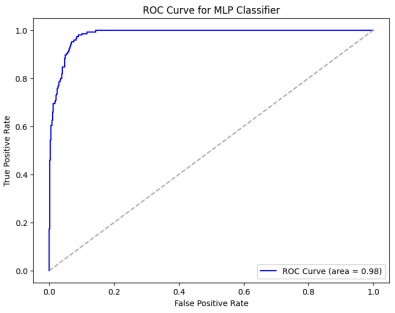
Receiver Operating Characteristic (ROC) Curve and Area Under the Curve (AUC):





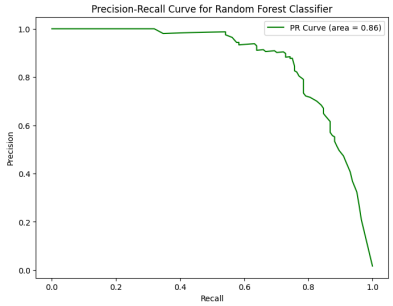


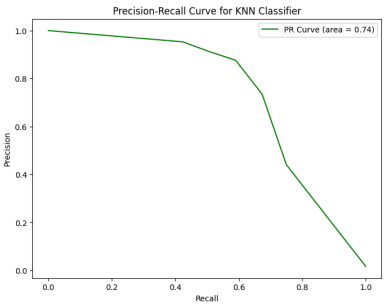


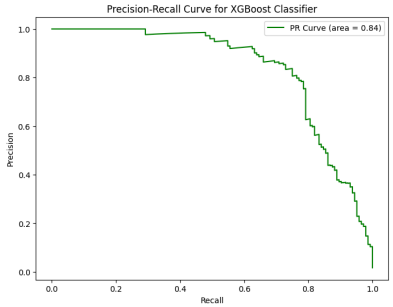


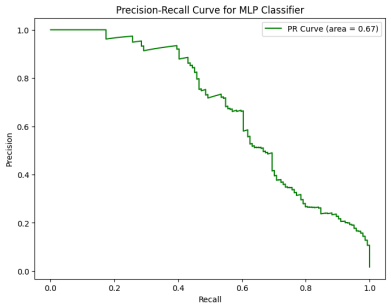
XGBoost has the best AUC, which means it is better at discriminating between classes.

Precision-Recall Curve and Area Under the Curve (AUC):









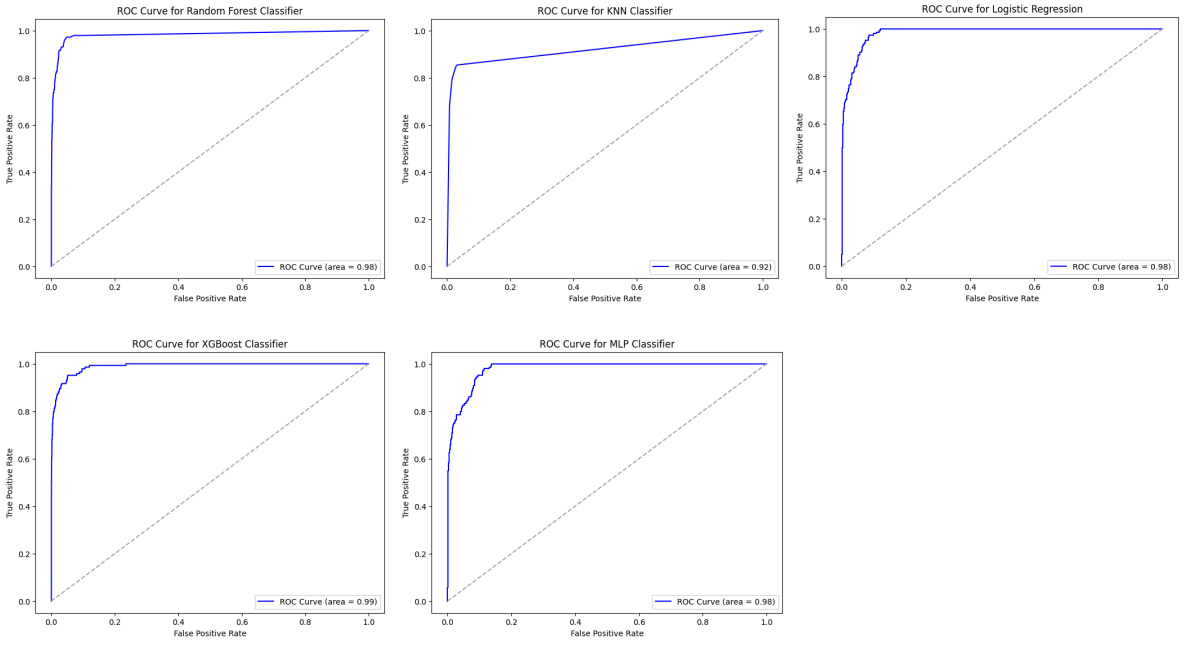
XGBoost and Random forest have the best AUC, which means they are better at distinguishing the minority class (fraudulent transactions).

Models performance (after resampling):

Model	Precision	Recall	F1-score
Random Forest	0.84	0.86	0.85
KNN	0.70	0.90	0.76
Logistic Regerssion	0.60	0.93	0.64
XGBoost	0.85	0.88	0.87
MLP	0.60	0.89	0.65

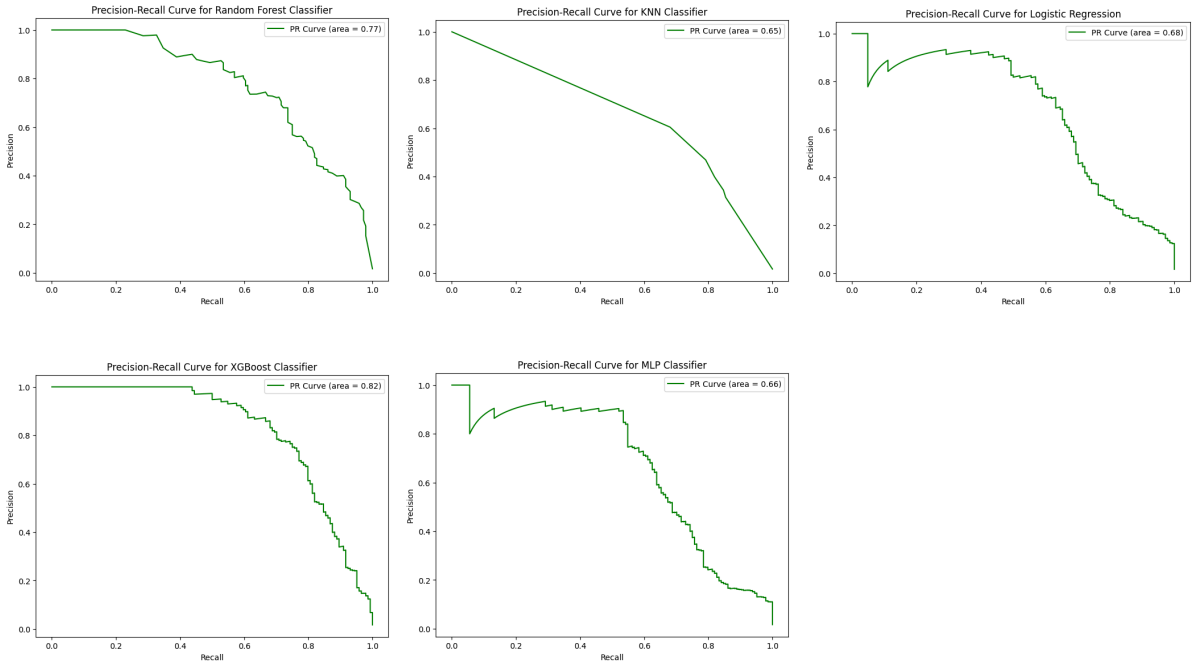
XGBoost has the best performance based on precision, recall and F1-score.

ROC Curve and Area Under the Curve (AUC):



XGBoost has the best AUC, which means it is better at discriminating between classes.

Precision-Recall Curve and Area Under the Curve (AUC):



XGBoost has the best AUC, which means it is better at distinguishing the minority class (fraudulent transactions).

Final Model

According to the evaluations of five models, the XGBoost classifier has the best performance. To fine-tune the model, we apply K-Fold cross-validation to find the better hyperparameters. Then train the model with the whole training dataset and predict with the test dataset to evaluate the overall performance.

Potential hyperparameters:

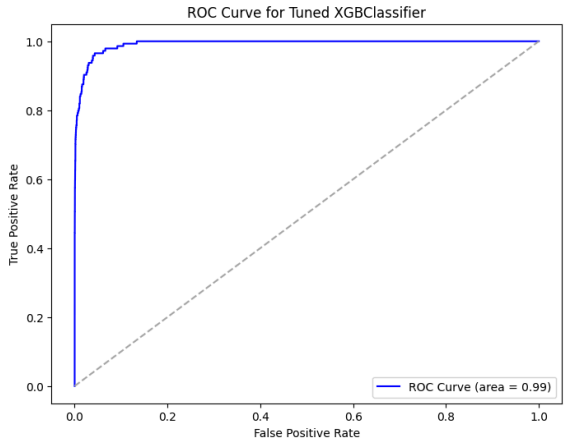
- n_estimators: 100, 200, 300
- learning_rate: 0.01, 0.1, 0.2
- max_depth: 3, 5, 7

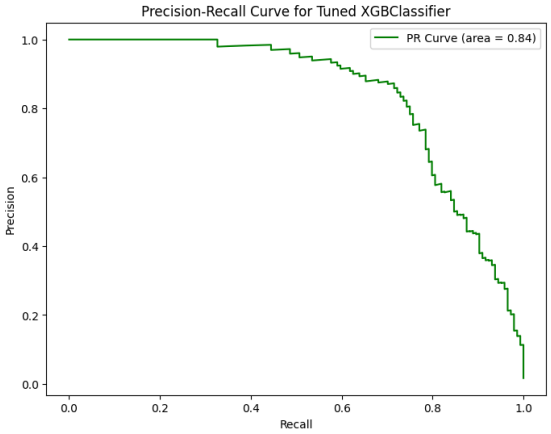
Best hyperparameters:

- n_estimators: 200
- learning_rate: 0.2
- max_depth: 7

Final results

Model	Precision	Recall	F1-score
XGBoost	0.93	0.86	0.89





Discussion of Results

After applying the models on the data that had been fitted by SMOTE, the performance decreased. This might be because SMOTE generates synthetic samples by interpolating between minority class samples. If the model becomes too focused on these synthetic points, it might overfit the minority class. Also, if the minority class has noisy data points, SMOTE may create synthetic samples from these noisy points, effectively amplifying the noise.

Among all models, XGBoost and Random Forest show the most promising results for this classification problem. Fraud detection datasets are typically highly imbalanced, with fraudulent transactions being rare compared to legitimate ones. Both XGBoost and Random Forest have mechanisms to address this issue:

XGBoost:

- Allows for weighted loss functions, enabling the model to penalize misclassifications of the minority (fraud) class more heavily.
- Supports early stopping and fine-grained parameter tuning to prevent overfitting to the majority class.

Random Forest:

- Creates bootstrapped samples during training, ensuring that minority class samples are present in at least some trees, leading to better representation of the minority class.
- Can use balanced class weights to address the imbalance.

Defend Adversarial Attack

Machine learning (ML) models are integral to modern applications but face security threats like model extraction attacks. In such attacks, an adversary queries the model and uses the responses to infer its internal parameters or replicate its functionality. This not only risks intellectual property theft but also enables unauthorized or malicious use of the model. To combat these threats, Fully Homomorphic Encryption (FHE) offers a practical solution by securing computations through encryption.

FHE allows ML models to process encrypted inputs without ever decrypting them. This ensures that sensitive user data and the model's parameters remain secure during interactions. In an FHE-enabled system, users encrypt their inputs before sending them to the model. The server processes the encrypted inputs, and the results are returned in encrypted form. Only the user, with the decryption key, can interpret the output. This approach protects the model from adversaries, as they cannot gain meaningful insights even with repeated queries.

The use of FHE fundamentally changes the attack surface for model extraction. Responses to queries remain encrypted and useless to attackers without the decryption key, significantly reducing the risk of model reconstruction. This makes FHE a robust defense against adversaries seeking to exploit query-response pairs.

Despite its strong security guarantees, FHE introduces computational overhead and is slower than traditional operations. However, advancements in encryption techniques are making FHE more efficient and feasible for real-world use. As these improvements continue, FHE has the potential to become a cornerstone technology for securing ML models against threats like model extraction while safeguarding user privacy.

Conclusion

In this study, we evaluated five machine learning models—Random Forest, KNN, Logistic Regression, XGBoost, and MLP for fraudulent transaction detection. Among these models, XGBoost and Random Forest demonstrated the most promising results, effectively addressing the challenges posed by the highly imbalanced nature of the dataset.

XGBoost achieved the best performance overall, making it the most suitable model for this classification problem. Through K-fold cross-validation, we fine-tuned the model to ensure optimal results, confirming its effectiveness for fraud detection tasks.

References

[1] https://trenton3983.github.io/posts/fraud-detection-python/

[2] Khaled Gubran Al-Hashedi, Pritheega Magalingam,Financial fraud detection applying data mining techniques: A comprehensive review from 2009 to 2019,Computer Science Review,Volume 40,2021,100402,ISSN 1574-0137, https://doi.org/10.1016/j.cosrev.2021.100402.

[3] https://www.kaggle.com/datasets/ealaxi/banksim1/data

[4] FHE library https://github.com/OpenMined/TenSEAL

[5] https://openfhe-development.readthedocs.io/en/latest/