Deep Learning Approach for Detecting Credit Card Fraud Transactions

Shinjon Ghosh

Department of Mathematics, Illinois State University

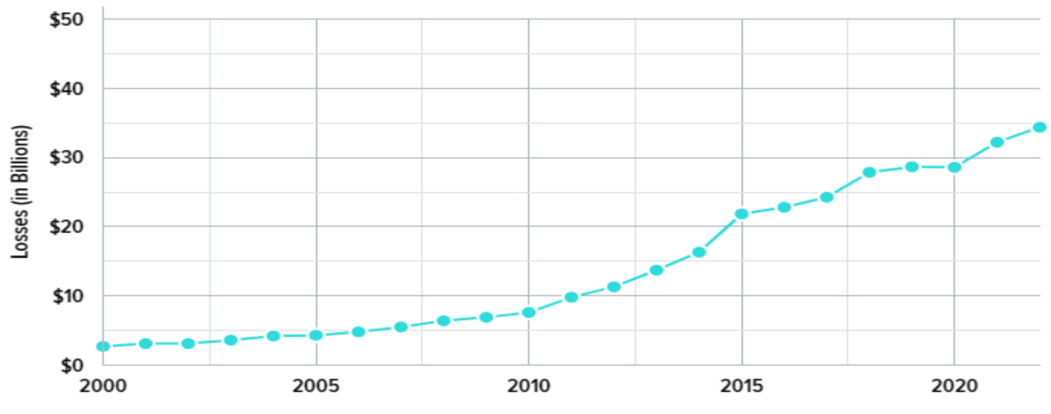
Abstract:

Currently, credit card fraud detection is a one of the major challenges for financial institutions. It needs robust and adaptive techniques to minimize losses and protect consumers. In this project, I propose a deep learning approach for detecting credit card fraud transactions using an autoencoder model. Autoencoders, a type of unsupervised feed forward neural network, are well-suited for anomaly detection. I trained the autoencoder model on a large dataset of credit card transactions by legal transactions and determine threshold by validation set. The performance of the autoencoder model was evaluated in terms of accuracy, precision, recall, and F1-score. Our findings suggest that autoencoders can effectively indicate credit card fraud with reduced false positives and enhanced adaptability to new fraud tactics.

Keywords: deep learning, credit card, fraud detection, autoencoder, ROC curve, AUC value, precision, recall, confusion matrix, classification report.

Introduction:

Nowadays credit card payments are most popular modern financial transactions. It provides a convenient way to make purchases all over the world and financially flexible for users. In the meantime, the North American institutions losses $246 million and globally almost $35 billion by credit cards fraud. Currently, a lot of financial security-based institutions try to find out how to protect fraud credit card transactions.



(Source: Nilson Report, December 2021)

Among deep learning methods, autoencoders offer a better approach to anomaly detection. Autoencoders are feed forward neural networks designed to learn efficient representations of data, typically for the purpose of reconstruction the output. This project explores of an autoencoder model to detect credit card fraud transactions. Our approach involves several steps. First, I preprocess the credit card transaction data to train and test autoencoder model. Then I determine threshold point by validation set and use mean squared error as a loss function. Later, I evaluate the model performance using confusion metrics, accuracy, precision, recall, and F1-score. This approach provides a flexible and adaptive solution to credit card fraud detection.

Background & Literature Review:

Credit card fraud in the financial market is a growing concern, leading to significant losses for financial organizations and merchants. The challenges in detecting fraudulent transactions due to data imbalance, requiring the use of machine learning and deep learning anomaly detection algorithms [3]. Global credit card fraud detection is crucial, and a model combining deep learning techniques with hyperparameter optimization is proposed to detect patterns in transactions. The model uses AE, CNN, and LSTM deep learning techniques with random search and Bayesian optimization, showing superior performance in accuracy, AUC, and processing times [4]. XGBoost outperformed other baseline models (Logistic regression, decision tree, random forest, and neural network) in credit card fraud detection. Ensemble methods like XGBoost are effective for handling class-imbalanced problems. Oversampling methods (such as VAEGAN) improve performance in imbalanced classification [6]. AE-PRF shows promise for handling imbalanced datasets without resampling, achieving competitive performance compared to existing methods [2].

Objectives:

- Build an optimum deep learning model specifically based on Autoencoder to identify credit card fraud transections.
- Split the dataset to train & test the model. Moreover, create a validation set to determine threshold point.
- Draw confusion matrix to detect model performance.
- Evaluate model accuracy to improve model productivity.

Methods & Materials:

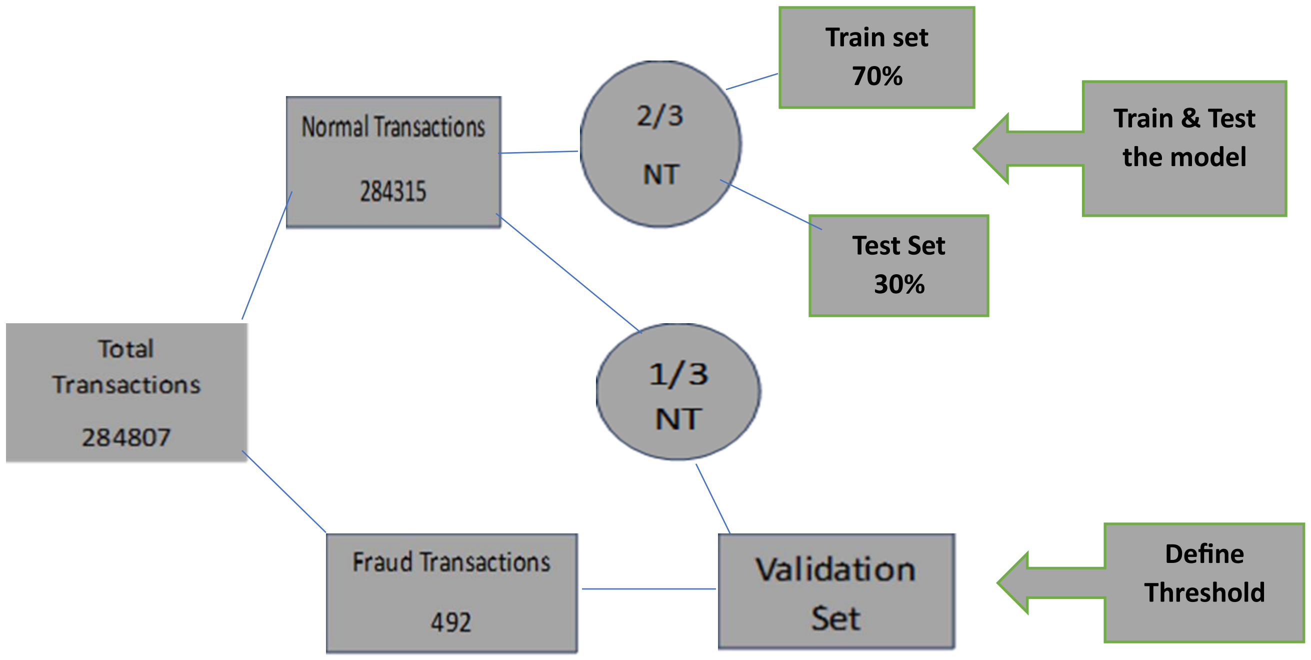
Data Source & Overview

The credit card fraud detection dataset collects from Kaggle (secondary data resources) which provides credit card transactions data that occurred during a period of two days. This data set comes from a European bank card holders credit card transactions performed in September 2013. The dataset contains 284807 transactions and 31 features. Among these transactions only 492 are fraudulent and 284315 transactions are normal. All variables of this dataset are numerical. The data has undergone Principal Component Analysis transformations for privacy considerations. The

Time and Amount features remain unchanged. Time represents the duration in seconds between each transaction and the initial transaction in the dataset.

Data Preparations

At first, I check the null value on the dataset and there is not any null value. Then I exclude the Time column from the analysis as it will not be utilized, and instead, apply the StandardScaler from the scikit-learn library to standardize the Amount feature. This scaling technique involves eliminating the mean from the data and adjusting the values to unit deviance. According to the dataset, it is highly imbalanced data. That's why I train and test our model on the normal transactions only. Moreover, I create a validation set based on fraudulent transactions and 1/3 of legal transactions to define threshold point. If the distance between the original and reproduced transactions falls below a specified threshold, the transaction is authentic. However, if the distance exceeds the threshold, the transaction is flagged as a potential fraud candidate. I draw a diagram below to understand the data processing techniques.



Result & Discussions:

Exploratory Data Analysis

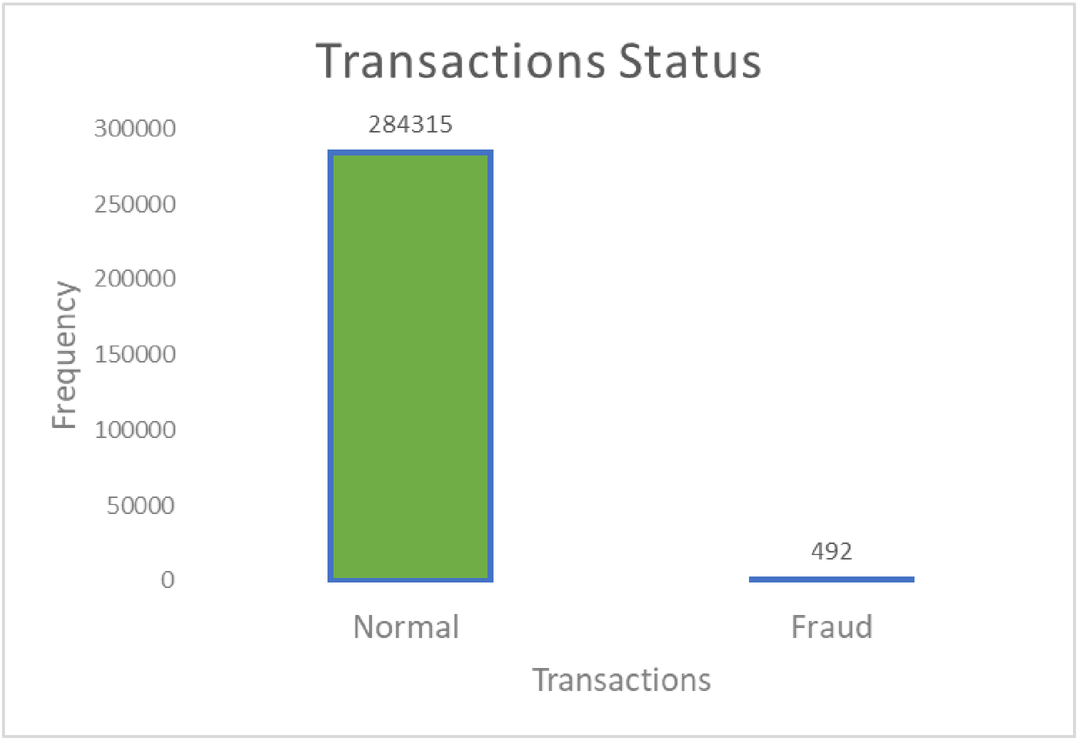


Fig 1: Transactions Status

The information from Fig 1 we can say that, the normal transactions (284315) outperformed the fraud transactions (492). Overall, we find only 0.2% data are fraudulent. So, the data is highly imbalanced.

Fraud Transactions	**Normal Transactions**
count 492.000000 mean 122.211321 std 256.683288 min 0.000000 25% 1.000000 50% 9.250000 75% 105.890000 max 2125.870000	count 284315.000000 mean 88.291022 std 250.105092 min 0.000000 25% 5.650000 50% 22.000000 75% 77.050000 max 25691.160000

Table 1: Descriptive Statistics of the Amount variables

From Table 1 we illustrate that the average fraud amount is $122 and the maximum amount is almost 2126 USD. On the other hand, the average normal transactions are $88 and maximum amount is 25691 USD. Moreover, the median fraud transactions amount is more than 9 US dollars and legal transactions median amount is exact 22 USD.

Deep Learning Model

Autoencoder

Anomalies data are challenging to identify. The values of the selected characteristics subtly deviate from the expected distribution. Moreover, the deviation from the norm becomes apparent only upon considering a sequence of occurrences and time characteristics. Furthermore, our data is highly imbalanced. In this case, autoencoder model performs well.

An autoencoder is characterized as a feed-forward multilayer neural network designed to replicate the input data at the output layer. Consequently, the quantity of output units must be similar with that of the input units. Moreover, the training of an autoencoder involves the application of the backpropagation algorithm with a specified loss function, such as the mean squared error (MSE). In an autoencoder model, the encoder and decoder are two essential components that work together to reconstruct input data. The explanation of encoder, decoder, backpropagation and loss function (mean squared error) are given below:

Encoder: The encoder part compresses the input data into a latent-space representation. It takes the input data and reduced the dimension of the dataset features. This reduced dimension typically captures the most important characteristics of the input data.

Decoder: The decoder is the opposite part of the encoder. It takes the compressed representation produced by the encoder and reconstructs the original input data from it. The main part of the decoder is to generate output data which is closely resembles the input data.

The encoder and decoder part of an autoencoder trained to minimize the reconstruction error between the input data and the output data. The encoder-decoder structure allows autoencoders to learn efficient representations of data while also being capable of generating new data similar to the input.

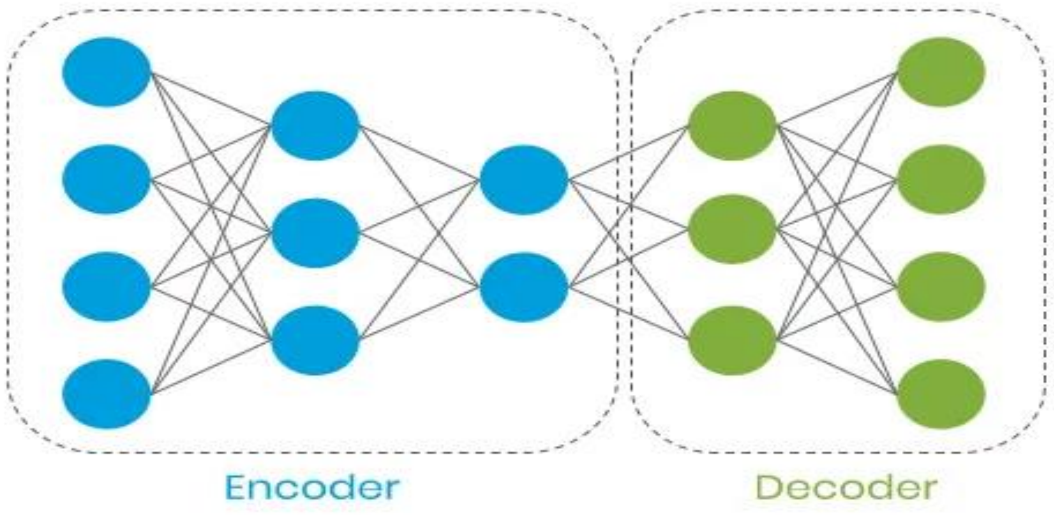


Fig 3: Graphical representation of sample autoencoder model

Backpropagation: Backpropagation in an autoencoder model is a key part of the training process. It is allowing the model to learn from its reconstruction errors. Backpropagation calculate gradients and update the weights in the neural network to minimize the loss. After the computing loss, it involves propagating the gradients backward through the network to determine how each weight performed to the error.

Loss function (mean squared error): The loss function in an autoencoder model is guiding the training process by measuring the difference between the original input data and the reconstructed output from the autoencoder. One of the most commonly used loss functions for autoencoders is Mean Squared Error (MSE). Mathematically, the MSE is,

$$L = \frac{1}{n} \sum_{i=1}^{n} (y_i - \widehat{y_i})^2$$

In this case study, I built an autoencoder model with three hidden layers, with the number of units 30–14–7–14–30 (here 30-14-7 is encoder and 14-30 is decoder part) and tanh. Then I introduce reLu as activation functions. The autoencoder was trained with Adam — an optimized version of backpropagation — on just legal transactions. After that, I draw 50 epochs & 22 batch sizes, against the MSE as a loss function.

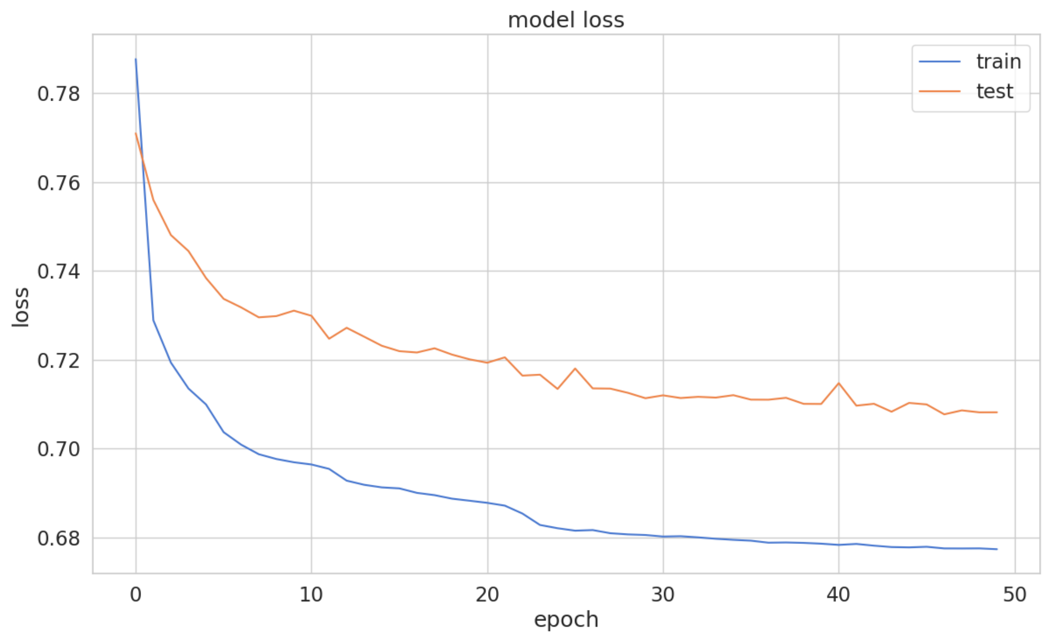


Fig 4: Loss function calculation

From Fig 4, we depict the mean squared error of training and test data seems to converge nicely.

Now, I draw precision vs recall plot to describe model performance because precision and recall is most common metrics to denote model evaluation when the response data is binary. Precision measures the relevancy of obtained results. On contrary, recall measures how many relevant results are returned. These are calculated as:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

High recall but low precision means many results, most of which has low or no relevancy. On the other hand, high precision but low recall means few returned results with very high relevancy. A high area under the curve represents both high recall and high precision, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate.

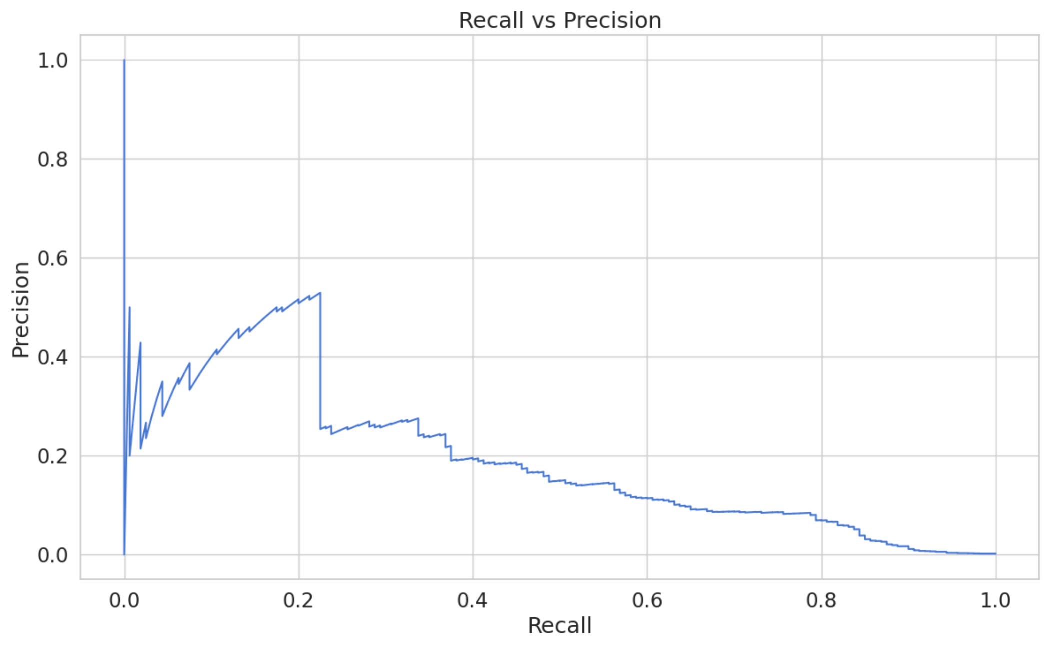


Fig 5: Precision vs Recall Curve

From Fig 5, we see the recall is high but precision is gradually low down. For identify clear concept of model performance, I draw ROC curve and produce AUC value. ROC curve is very useful tool to detect performance of binary classifier. If the ROC curve will be the top-left corner of the plot, then it indicates good performance. Moreover, a higher AUC value which is close to 1 identify better model.

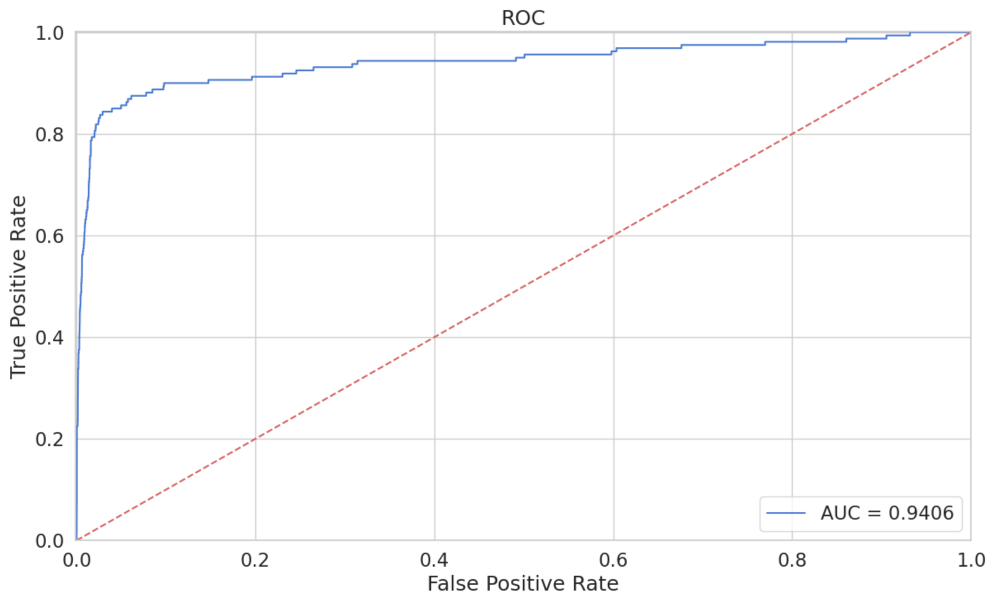


Fig 6: ROC curve with AUC value

From Fig 6, we say that ROC curve is very close to left corner and the AUC is 0.94 which is close to 1. It indicates the autoencoder fraud detection model perform well.

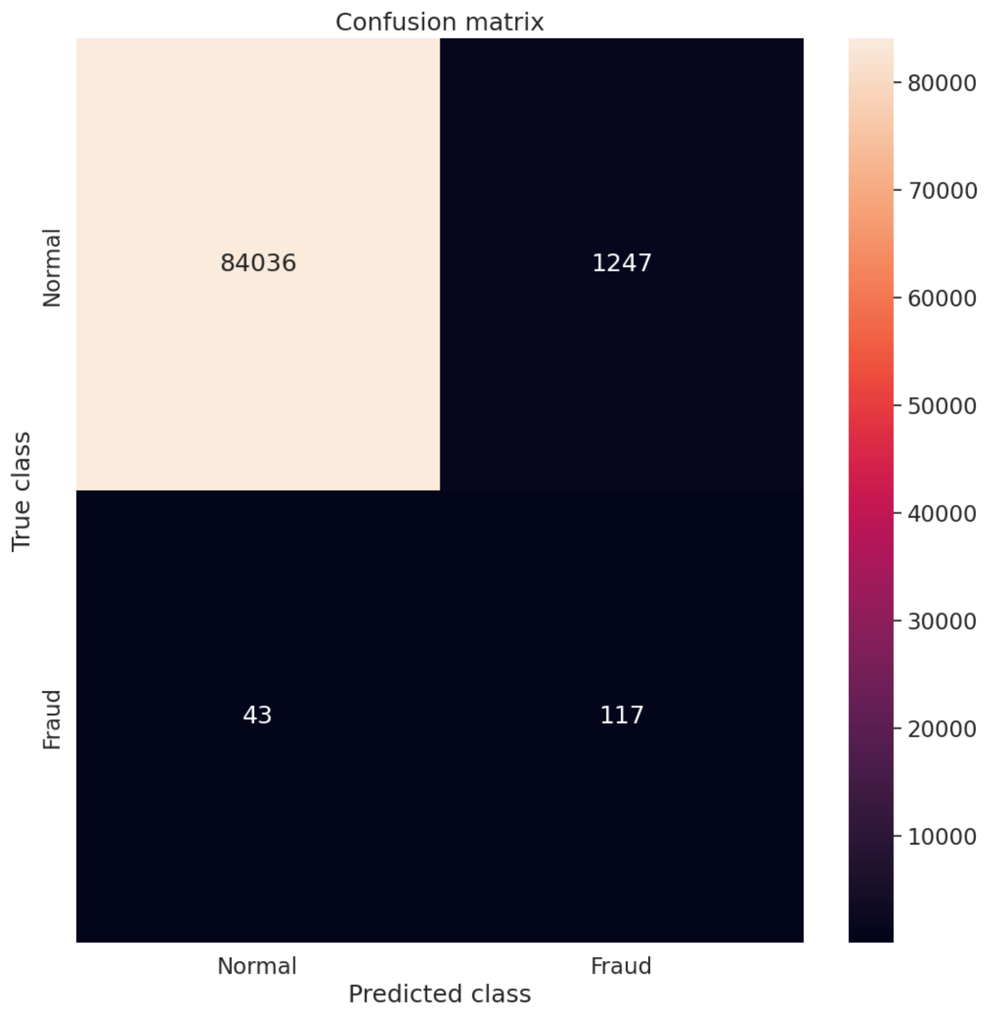


Fig 7: Confusion Matrix

In the confusion matrix we found, true positive rate is so high which is 84036. On contrary, false positive rate is very low (43) and false negative rate is little bit high (1247). Now we calculate the classification report according to confusion matrix.

	Equation	Result
Accuracy	$\frac{TP+TN}{TP+FP+TN+FN}$	0.98
Precision	$\frac{TP}{TP+FP}$	0.99

Recall	$\frac{TP}{TP+FN}$	0.98
F1 score	$2 \times \frac{Precision\ \times Recall}{Precision + Recall}$	0.99

Table 2: Classification Report

According to Table 2, the accuracy and F1 score of the fraud detection model is very high. It indicates model identify the fraud detection very well.

Limitations & Improvement:

Almost, all of the data are Normal transaction data. We can address more fraud data-based dataset that the model trained and predict more accurately. The network parameters could be optimized, including experimentation with different activation functions and regularization parameters, a higher number of hidden layers and units per hidden layer.

Conclusion:

The aim of this project is drawing an optimum credit card fraud transactions model which identifies fraud transactions and send a suspicious email to credit card holders. I use autoencoder model because of anomalies imbalanced data train and test performance is better than other traditional model. The loss function plot (Fig 4) indicates train and test data converge nicely. Moreover, the ROC curve and AUC value describes the model performance is very well. Subsequently, the confusion matrix and classification report indicates the model accuracy is very high. In this case, the model detects fraud transactions and send a suspicious email to credit card holders.

Reference:

[1] Roy, A., Sun, J., Mahoney, R., Alonzi, L., Adams, S., Beling, P., & University of Virginia. (2018). Deep learning Detecting fraud in credit card transactions. IEEE, 129.

[2] Lin, T. W., & Jiang, J. (2021). Credit Card Fraud Detection with Autoencoder and Probabilistic Random Forest. Mathematics, 9(21), 2683. https://doi.org/10.3390/math9212683

[3] Soni, J., Gangwani, P., Sirigineedi, S. S., Joshi, S., Prabakar, N., Upadhyay, H., & Kulkarni, S. A. (2023). Deep learning approach for detection of fraudulent credit card transactions. In Intelligent systems reference library (pp. 125–138). https://doi.org/10.1007/978-3-031-28581-3_13.

[4] Sulaiman, S. S., Nadher, I., & Hameed, S. M. (2024). Credit card fraud detection using improved deep learning models. Computers, Materials & Continua/Computers, Materials & Continua (Print), 78(1), 1049–1069. https://doi.org/10.32604/cmc.2023.046051.

[5] https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud.

[6] Ding, Y., Kang, W., Feng, J., Peng, B., & Yang, A. (2023). Credit card fraud detection based on improved variational autoencoder generative adversarial network. IEEE Access, 11, 83680–83691. https://doi.org/10.1109/access.2023.3302339.