Fraud detection in credit-card based payment system.

Nishant Kumar Bundela
IIIT-Delhi
Delhi, India
nishant17171@iiitd.ac.in

Abstract- **The rapid growth in the E-Commerce industry has lead to an exponential increase in the use of credit cards for online purchases, and consequently, they have been a surge in the fraud related to it. Machine learning plays a vital role in detecting credit card fraud in transactions. For predicting these transactions, banks make use of various machine learning methodologies, past data has been collected, and new features have been used for enhancing predictive power. While many supervised classification techniques have been applied in the past to model fraud detection, yet very few of them take into consideration that the dataset being used is highly skewed and imbalanced. This project aims to design a model that could handle the skewness of the dataset and classify the transaction as a fraud or a legit transaction with a high detection rate using semi-supervised classification techniques. Semi-Supervised techniques basically refer to the process of modifying the dataset using unsupervised methods and then classifying using supervised methods. We are using Autoencoder neural networks as the unsupervised method for all of our approaches to changing the dataset. The primary goal is to catch most of the true positives, where positive refer to fraudulent transactions.**

Keywords—Credit-card fraud detection, E-commerce, Semi-Supervised learning, Autoencoders Neural Network, Automated fraud detection, neural network, Imbalanced dataset.

I. PROBLEM STATEMENT AND MOTIVATION

Credit card fraud detection is a developing risk with full results in the money business, enterprises, and government. Frauds can be characterized as criminal duplicity with the aim of procuring monetary benefit. As credit cards turn into the most well-known technique for payment methods for both on the web and offline exchange, the fraud rate likewise quickens. The major reason behind credit card frauds is because of the absence of security, which includes the utilization of stolen credit cards to get money from a bank through authentic access or to get access to the security credentials of the owner associated with a credit card. Necessary prevention measures can be taken to stop this abuse, and the behavior of such fraudulent practices can be studied to minimize it and protect against similar occurrences in the future. Fraud detection involves monitoring the activities of populations of users in order to estimate, perceive or avoid objectionable behavior, which consists of fraud, intrusion, and defaulting. This is a very challenging problem that requires the expertise of people working in the machine learning and data science sector where the solution to this problem can be automated. This problem becomes particularly difficult because of its dataset is highly imbalanced. The number of non-fraud transactions outnumbers the fraudulent ones by a large amount. Also, the transaction patterns tend to change over time, which leads to the failure of models already trained with a specific distribution. Another problem which researches face is the real-time detection of these fraudulent payments. By real-time, I mean the detection of payment at the time of its happening. All these problems significantly reduce the effectiveness of binary classifiers like Logistic Regression, Decision tree classifier undesirably biasing the results towards the prevailing class, while we are interested in the minority class. While there is nothing wrong with the classifiers, the faulty thing here is the imbalanced dataset. In the past, people have tried classification by under-sampling the dataset to make an equal number of fraud and non-fraud cases in the training process, but under-sampling has its own cons that it leads to a great deal of information loss. This project focuses on working with Semi-Supervised learning approaches to train the dataset and then using a simple binary classification technique to classify the transaction as fraud or not.

Semi-Supervised Learning: It is a combination of supervised and unsupervised learning processes in which the unlabeled data is used for training a model as well. In this approach, the properties of unsupervised learning are used to learn the best possible representation of data, and the properties of supervised learning are used to learn the relationships in the representations, which are then used to make predictions. To learn the best possible representation of the data I am using Traditional Autoencoders and Denoising Autoencoder neural network.

Autoencoders: The aim of the Autoencoder is to learn representations to reconstructs features for a set of data, typically for the purpose of dimensionality reduction. The simplest form of an autoencoder is a feedforward, non-recurrent neural network which is similar to the multilayer perceptron

Denoising Autoencoders: This is a variation of traditional Autoencoder, which aims to make autoencoder neural network learn how to remove the noise and reconstruct undisturbed input with as minimum loss as possible.

Having a high recall and accuracy rate with a decent precision rate would remain the major objective of this project as a fraud transaction being classified as non-fraud is more harmful than classifying a non-fraud one as a fraud.

II. LITERATURE REVIEW

There is a long history of using machine learning for fraud detection in the payments industry. Bhattacharyya et al. note that while fraud algorithms are actively used by banks and payment companies, the breadth of studies on the use of machine learning techniques for payment fraud detection is limited [1], possibly due to the sensitive nature of the data. Their study concluded that random forests, though not widely deployed, may outperform more traditional methods. Roy et al. found that network size is the strongest driver of a neural network's performance for fraud classification [2]. Chaudhary et al. note that no single algorithm is Pareto optimal across all performance metrics; each has a unique set of strengths and weaknesses [3]. A comprehensive survey conducted by Clifton Phua and his associates has revealed that techniques employed in this domain include data mining applications, automated fraud detection, adversarial detection. [4]. A similar research domain was presented by Wen-Fang YU and Na Wang where they used Outlier mining, Outlier detection mining, and Distance sum algorithms to accurately predict fraudulent transactions in an emulation experiment of credit card transaction data set of one certain commercial bank. Outlier mining is a field of data mining which is basically used in monetary and internet fields. It deals with detecting objects that are detached from the main system i.e., The transactions that aren't genuine. They have taken attributes of customer's behavior and based on the value of those attributes, they've calculated that distance between the observed value of that attribute and its predetermined value.[6]. Unconventional techniques such as hybrid data mining/complex network classification algorithm is able to perceive illegal instances in an actual card transaction data set, based on network reconstruction algorithm that allows creating representations of the deviation of one instance from a reference group has proved efficient typically on medium-sized online transaction. There have also been efforts to progress from a completely new aspect. Attempts have been made to improve the alert feedback interaction in case of the fraudulent transaction. Cui et al. highlight research is suggesting that oversampling can lead to overfitting, while added noise from synthetic data generation can reduce predictive performance [7]. Maniraj explored this problem with the help of traditional classifiers (local outlier factor and isolation forest algorithm) and found that although isolation forest performed better than a local outlier, both produced poor results when it came to recalling score [9].

III. DATASET AND FEATURES

The dataset being used is collected from Kaggle. It contains 31 variables and nearly 300,000 credit card transactions labeled as either legitimate or fraudulent [9]. Table 1 provides descriptive information about the dataset. An important attribute of the dataset is that it has been processed to protect cardholder privacy. In particular, it contains 28 non-descriptive numerical variables (V1, …, V28) that are the result of a principal component analysis (PCA) transformation of several variables of interest that could not be publicly disclosed.[8].

Table 1: Dataset Description.

Variable	Description
Time	Time elapsed since first transaction
V1 ⋮ V28	Non-descriptive variables resulting from a PCA dimensionality reduction to protect sensitive data
Amount	Transaction Amount
Class	Classification as fraud or not.

Table 2: Class distribution

	Number	Percent
All transactions	284,807	100
Fraud	492	0.173
Non-Fraud	284,135	99,827

Fig 1: No. of samples in each class

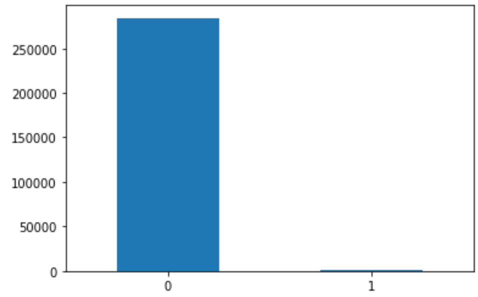
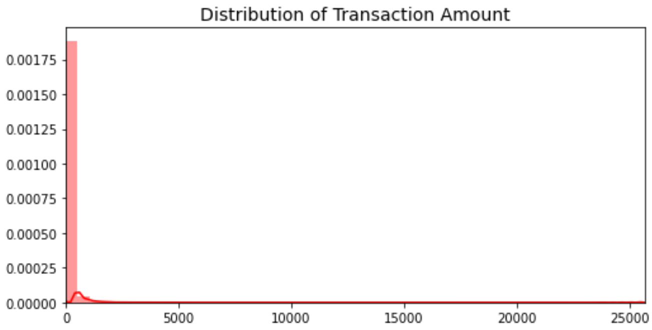


Table 3: Class vs Amount

	Total value	Average value
All transactions	25,162,590	88.35
Fraud	60,128	88.29
Non-Fraud	25,102,462	122.21

Fig 2

Fig 3



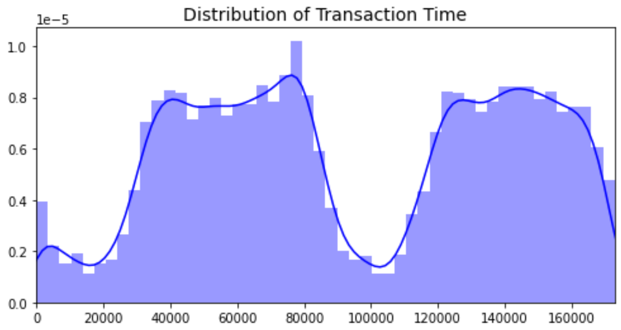
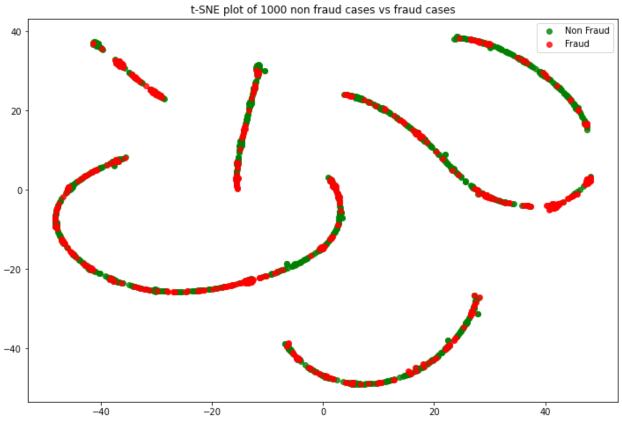


Fig 4

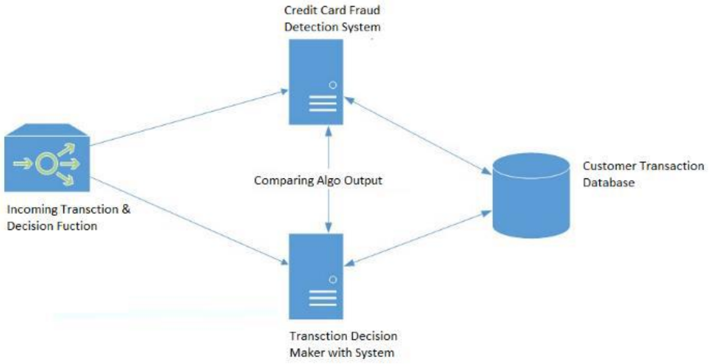


A preliminary analysis reveals several interesting features of the data. Table 2 shows the split of fraudulent and legitimate transactions. Only 492 transactions, or less than 0.2 percent, are fraudulent, highlighting a stark class imbalance. Table 3 shows that fraudulent transactions have greater transaction value on average than legitimate transactions. Further, the PCA variables are uncorrelated with each other, consistent with the fact that they are orthogonal and have been standardized to have zero mean. Fig 2 shows the distribution of the transaction amount. Fig 3 shows the distribution of transaction time. Most of the transactions are quite small as you would expect in everyday transactions. The time is recorded in a number of seconds elapsed since the first transaction in the dataset. The transactions are over a period of 2 days. The time feature has been dropped in the analysis as I don’t have much info on the exact time of the transaction. I could speculate most of the fraudulent is at night, but I have decided to drop the variable in the initial analysis. I have also normalized the amount variable due to the large variance. Rest all other features has been kept intact. Fig 4 shows the t-SNE plot of randomly picked 1000 non-fraud cases along with all the fraud cases. It shows that with the current feature set, we can hardly distinguish the fraud and non-fraud cases. So, either we have to find a better representation of the dataset or we have to find a better classification algorithm that can classify these transactions irrespective of their features being highly similar.

IV. Proposed Architecture

The basic rough architecture of the fraud detection with respect to the banking system would be like Fig 5:

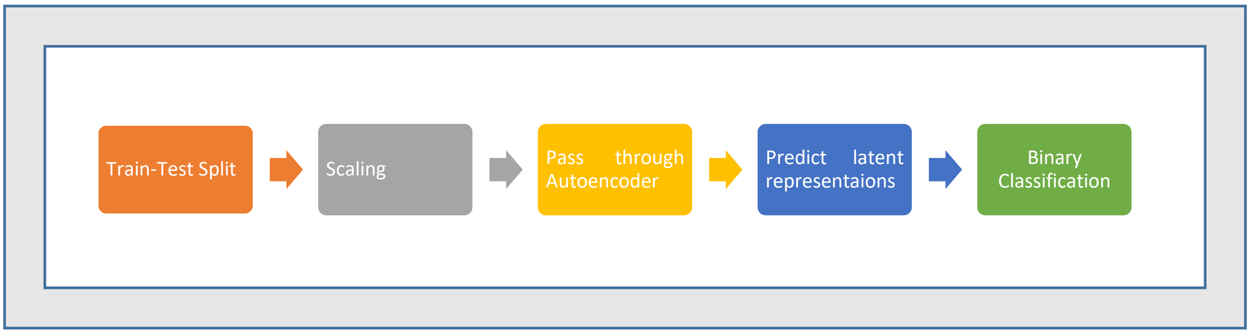
Fig 5



Inside the fraud detection system would reside our main algorithm. This project proposes two different semi-supervised models for the overall task, which are:

A. *Model 1:*

Using the autoencoder neural network to predict the latent representations and then classifying the latent representations using simple binary classifiers. A rough architecture can be represented as.



What is latent representations and why to use it: The latent space representation of our data contains all the critical information needed to represent our original data point. This representation must then serve the features of the original data. In other words, the model learns the data features and simplifies its representation to make it easier to analyze.

The Autoencoder will only be shown the non-fraud data, and it would be given the task to predict the latent representations of both fraud and non-fraud data. Since in this way, the latent representations would be different for most of the fraud and non-fraud transactions; hence for this algorithm, we don't need a lot of training data that solves the imbalanced data problem.

Autoencoder architecture:

Layers	Activation	Nodes
Input Layer	----------	29
Layer 1: Fully connected	Tanh	100
Layer 2: Fully connected	Tanh	75
Layer 3: Fully connected	Tanh	50
Layer 4: Fully connected	Tanh	20
Layer 5: Fully connected	Tanh	20
Layer 6: Fully connected	Tanh	50
Layer 7: Fully connected	Tanh	75
Layer 8: Fully connected	Tanh	100
Output layer	Tanh	29

Optimizer used: Adam

Loss: MSE

Framework used: Keras

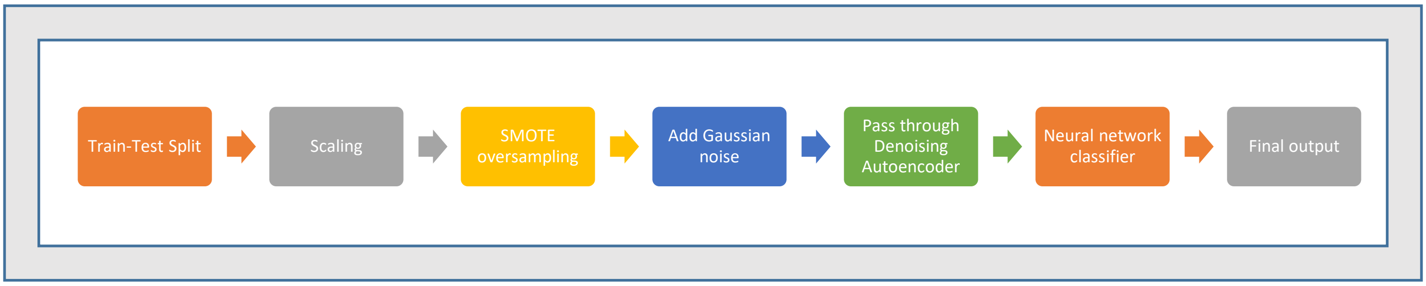
Hidden representations: Hidden representations were predicted from the bottleneck layer of the Autoencoder (i.e. Layer 5).

Binary classifiers: Binary classifiers used are *Logistic Regression* and *Random Forest classifiers* from the scikit-learn package.

B. *Model 2:*

Using SMOTE oversampling method to augment synthetic samples, then adding Gaussian noise and using denoising autoencoder to learn the denoised inputs, which would be more robust than the original feature set and then using fully connected neural network to classify the transaction as fraud or not.

A rough architecture of the model can be represented as



What is SMOTE oversampling and why to add noise: SMOTE Oversampling is a technique used to deal with an imbalanced dataset, its subject to create specific class sample so the class distribution of the original dataset can be balanced. SMOTE is one of the most popular oversampling techniques. Though it has been seen that it adds noise to the dataset, to tackle this, we are deliberately adding more Gaussian noise to the entire training dataset and then using denoising autoencoder to predict the original data. By doing this, we are making the Autoencoder to do more work to predict the denoised input by which it tends to create more robust features hence tackling the data skewness issue as well as the noise issue.

SMOTE sampling strategy=’not majority’.

Denoising Autoencoder architecture:

Layers	Activation	Nodes
Input layer	-----------	29
Layer 1: Fully connected	LeakyRelu	22
Layer 2: Fully connected	LeakyRelu	15
Layer 3: Fully connected	LeakyRelu	10
Layer 4: Fully connected	LeakyRelu	15
Layer 5: Fully connected	LeakyRelu	22
Output layer	LeakyRelu	29

Optimizer used: Adam (learning rate=0.001)

Loss: MSE

Framework used: Keras

Neural network classifier architecture:

Layers	Activation	Nodes
Input layer	-----------	29
Layer 1: Fully connected	LeakyRelu	22
Layer 2: Fully connected	LeakyRelu	15
Layer 3: Fully connected	LeakyRelu	10
Layer 4: Fully connected	LeakyRelu	5
Output layer	SoftMax	2

Optimizer used: Adam

Loss: Binary cross-entropy

The framework used: Keras.

Why SoftMax The softMax function is often used for probability distribution transformation for the classification results as the output of this functio is within range 0 to 1, which adds up to 1. Final output can be given by taking the maximum probability.

V. Evaluation and Results

The methodology and architecture of each model was discussed in the previous part. This part will present the evaluation of the results obtained.

.

A. *Model-1*

Before the train test split, the T-SNE plot of the entire dataset was obtained in Fig 6. It can be seen from this plot that initially, the non-fraud and fraud data were completely mixed up giving us the intuition that we need to get a more specific feature set, which could help in distinguishing both the classes.

1) Evaluation of training data

Further after train-test split that the T-SNE plot of the first 3000 non-fraud samples and fraud samples is shown in Fig 7. A similar conclusion can be drawn from this too.

Fig 6

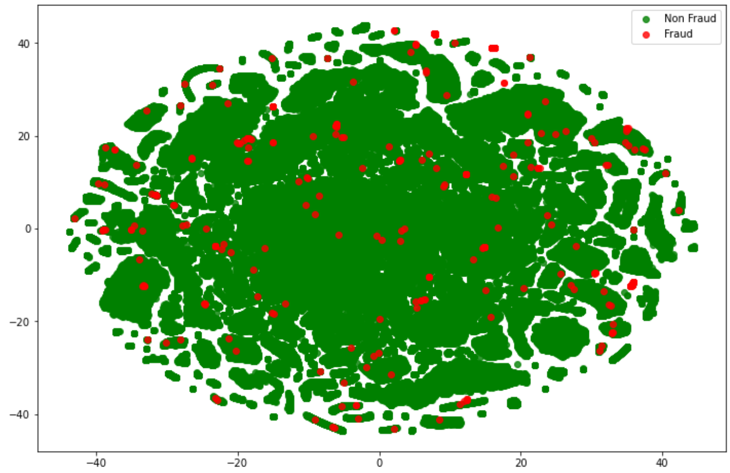


Fig 8 on the other hand shows the t-SNE plot of the latent representations of the training samples in Fig 7 obtained from the bottleneck layer of the autoencoder. It can be clearly shown that the autoencoder was able to differentiate between the latent representations of the non-fraud and fraud samples. Hence, it can be concluded that the latent representations are now ready for classification.

Fig 7.

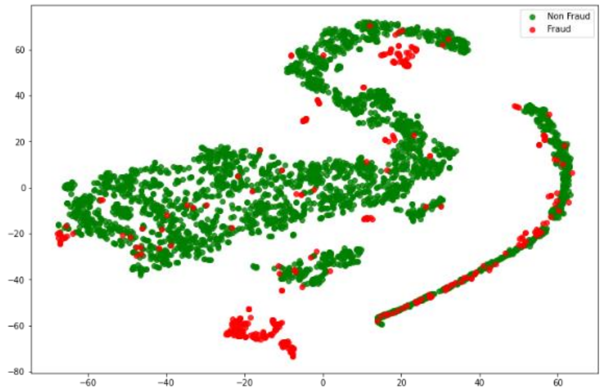
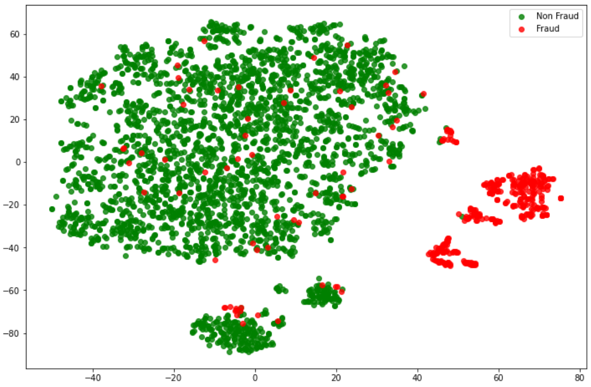


Fig 8.

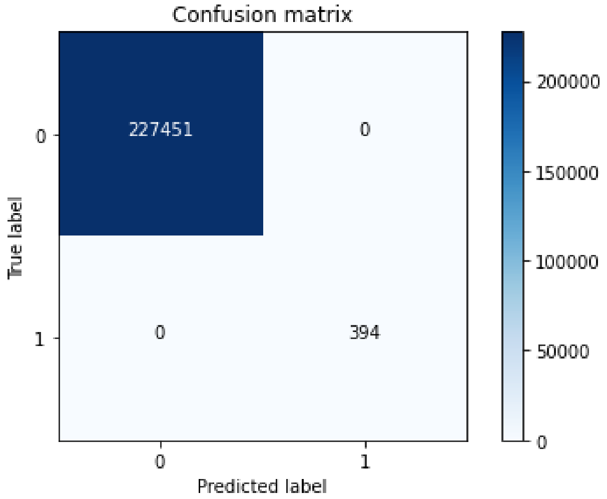


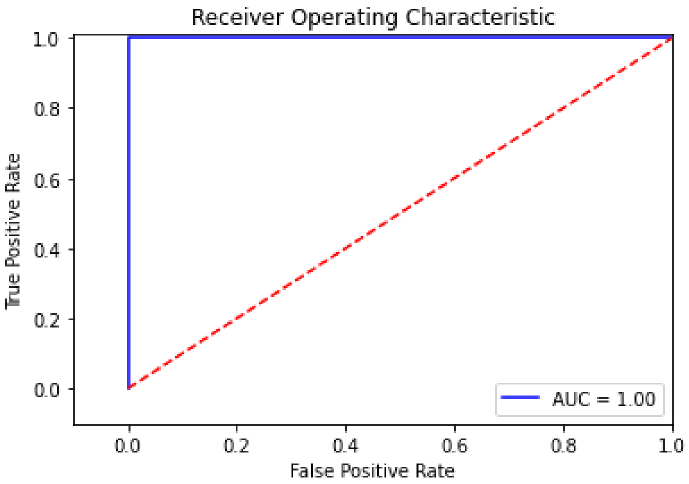
Evaluation result on training data:

Fig 9 shows the confusion matrix of the training data. Fig 10 shows the ROC plot and the AUC score.

Fig 9

Fig 10





Classifier = Random Forest

Class	Precision	Recall	F1
0	1.00	1.00	1.00
1	1.00	1.00	1.00

Overall accuracy: 1.0

Classifier= Logistic Regression

Class	Precision	Recall	F1
0	1.00	1.00	1.00
1	0.52	0.86	0.64

Overall accuracy= 0.999

2) Evaluation result on Test data

T-SNE plots similar to fig 7 and fig 8 can also be shown for test data. Fig 9 shows the initial distribution. Fig 10 shows the distribution of the latent representations of the test data.

Fig 11

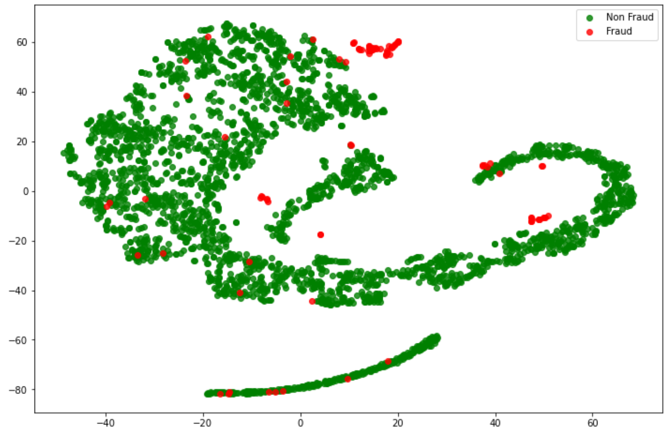
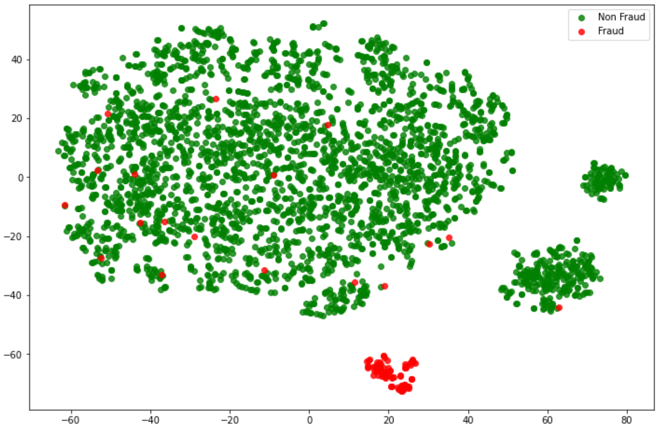


Fig 12.



Evaluation result on Test data:

Fig 13 shows the confusion matrix of the test data. It can be seen that False positives are more than false negatives, which will lead to slightly low precision than the recall score, but the major objective of this model was to have as minimum false negatives as possible. Fig 14 shows the ROC plot and the AUC score.

Fig 13

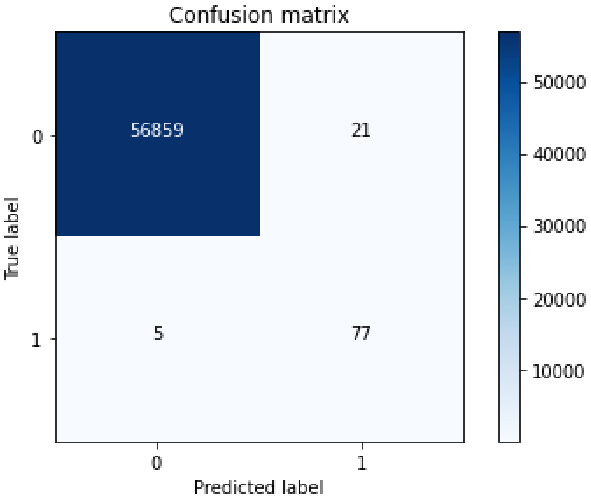
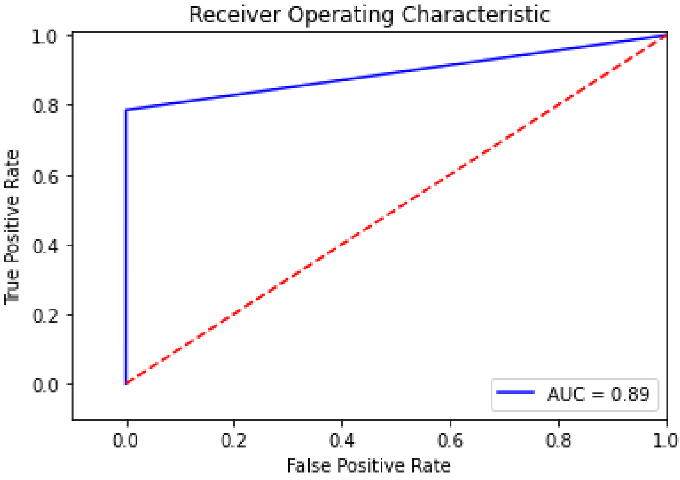


Fig 14



Classifier = Random Forest

Class	Precision	Recall	F1
0	1.00	1.00	1.00
1	0.81	0.94	0.87

Overall accuracy: 0.9995

Classifier= Logistic Regression

Class	Precision	Recall	F1
0	1.00	1.00	1.00
1	0.54	0.88	0.67

Overall accuracy: 0.99908

B. Model-2

1) Evaluation on Train data

Fig 15 shows the initial distribution of the training data prior to oversampling. It can be clearly seen that the data is highly skewed. Fig 16 shows the final distribution of the traning data after oversampling.

Fig 15.

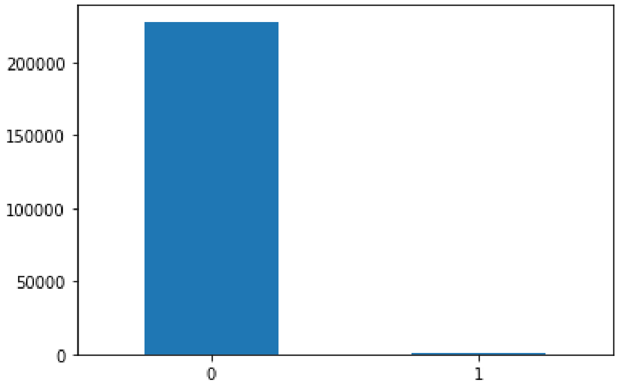


Fig 16.

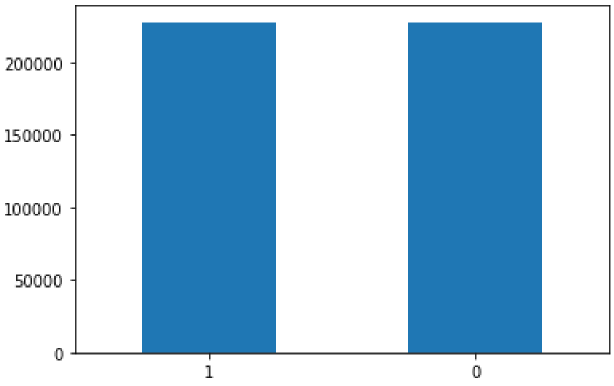


Fig 17 shows the confusion matrix after classification of training data by the neural network classifier and Fig 18 shows the ROC curve and the AUC score.

Fig 17

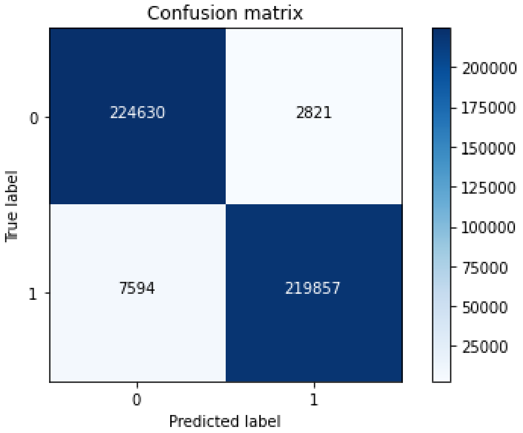
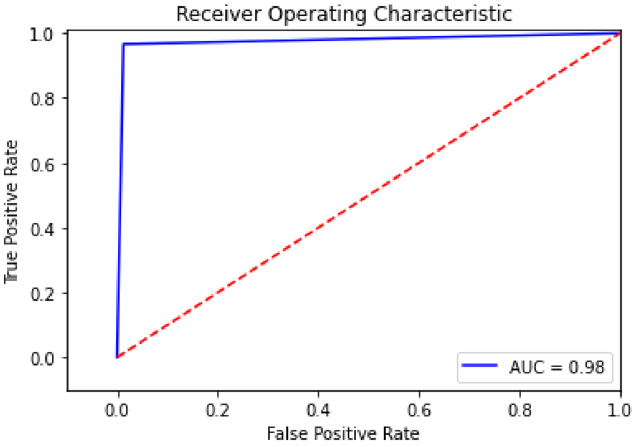


Fig 18



Classification report at threshold =0.5:

Class	Precision	Recall	F1
0	0.99	0.97	0.98
1	0.99	0.96	0.98

Overall accuracy score: 0.977

2) *Evaluation on Test data*

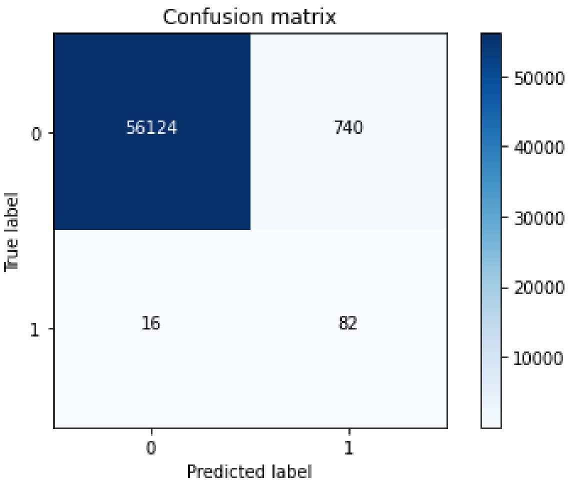
Classification report at threshold =0.5:

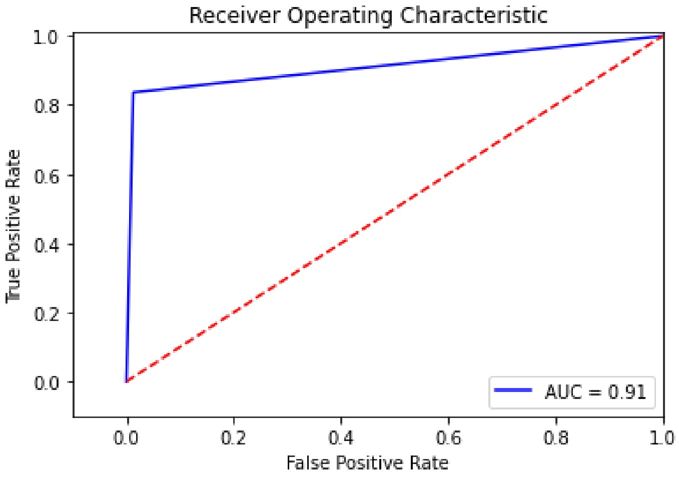
Class	Precision	Recall	F1
0	0.99	1.00	0.99
1	0.11	0.88	0.18

Overall accuracy score : 0.9876

Fig 19.

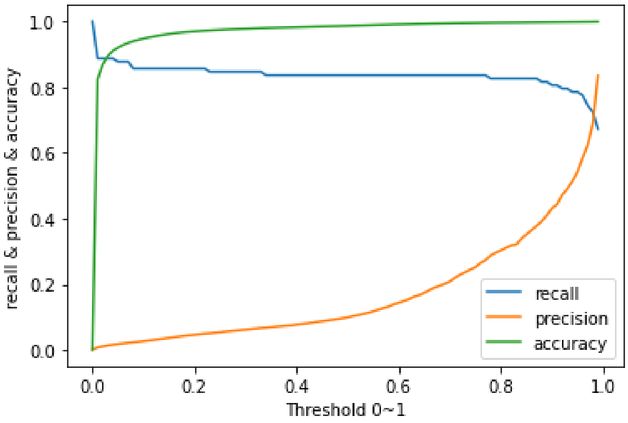
Fig 20.





SoftMax threshold vs accuracy vs precision vs recall curve is shown in Fig 21. It can be seen that in this model high recall will lead to low precision.

Fig 21.



As we would have expected, increasing threshold leads to increased precision and decreased recall value. As our major goal was to have excellent recall and accuracy scores along with a decent precision rate, an ideal threshold would be somewhere between 0.5-0.6.

CONCLUSION

In the machine learning area, imbalance data classification receives increasing attention as big data become popular. Detection of credit card fraud is an intentional part of testing for the researchers over a long time and will be an interesting part of testing in the coming time. This project presents two different fraud detection systems for credit-cards using semi-supervised learning techniques and training our machine using these algorithms with the transaction records we have. We saw that model-1 had overall better results in recall and accuracy alongside a decent precision score. Model-2, however, also had good accuracy and recall, but its precision rate was not acceptable. Future work includes designing a model that could also provide an excellent precision rate alongside recall and accuracy scores. These algorithms show us that the given transaction tends to be a type of fraud or not; these algorithms were selected using experimentation, literature review from previous papers, and from discussion among peers. Real-time detection of fraud transactions with high precision will still remain a problem to be more research on. While both of our models gave a high accuracy on test data but our best precision was still 81 %, which still needs to be increased in order to deem any model as a market-ready model.

REFERENCES

[1] Siddhartha Bhattacharyya, Sanjeev Jha, Kurian Tharakunnel, J. Christopher Westland, Data mining for credit card fraud: A comparative study, Decision Support Systems, v.50 n.3, p.602-613, February 2011

[2] Roy, A., Sun, J., Mahoney, R., Alonzi, L., Adams, S., and Beling, P. Deep learning detecting fraud in credit card transactions. Systems and Information Engineering Design Symposium (SIEDS) (2018), pp. 129–134.

[3] Chaudhary, Khyati & Yadav, Jyoti & Mallick, Bhawna. (2012). A review of Fraud Detection Techniques: Credit Card. International Journal of Computer Applications. 45.

[4] CLIFTON PHUA1, VINCENT LEE1, KATE SMITH1 & ROSS GAYLER2 " A Comprehensive Survey of Data Mining-based Fraud Detection Research" published by School of Business Systems, Faculty of Information Technology, Monash University, Wellington Road, Clayton, Victoria 3800, Australia

[5] Survey Paper on Credit Card Fraud Detection by Suman", Research Scholar, GJUS&T Hisar HCE, Sonepat published by International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 3 Issue 3, March 2014.

[6] Research on Credit Card Fraud Detection Model Based on Distance Sum – by Wen-Fang YU and Na Wang" published by the 2009 International Joint Conference on Artificial Intelligence.

[7] Y. Cui, M. Jia, T.-Y. Lin, Y. Song and S. Belongie. Class-balanced loss based on an effective number of samples. CoRR, abs/1901.05555, 2019.

[8] Worldline and the Machine Learning Group. Credit Card Fraud Detection. Retrieved from https://www.kaggle.com/mlg-ulb/creditcardfraud.

[9] Maniraj, S & Saini, Aditya & Ahmed, Shadab & Sarkar, Swarna. (2019). Credit Card Fraud Detection using Machine Learning and Data Science. International Journal of Engineering Research and. 08. 10.17577/IJERTV8IS090031.

[10] Zou, Junyi & Zhang, Jinliang & Jiang, Ping. (2019). Credit Card Fraud Detection Using Autoencoder Neural Network.