

Anomaly Detection in Credit Card Transactions Using Deep Learning Techniques

Project progress report submitted

to

**MANIPAL ACADEMY OF HIGHER
EDUCATION**

For Partial Fulfillment of the Requirement for

the Award of the Degree

of

Bachelor of Technology

in

Information Technology

by

Harsha Vardhan Kumar Khandyana

Reg. No. 190911234

Under the guidance of

Mrs. Vibha

Assistant Professor Senior Scale

Department of I & CT

Manipal Institute of Technology

Manipal, India



MANIPAL INSTITUTE OF TECHNOLOGY

MANIPAL

(A constituent unit of MAHE, Manipal)

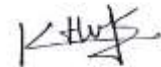
May 2023

DECLARATION

I hereby declare that this project work entitled **Anomaly Detection in credit card transactions using deep learning techniques** is original and has been carried out by me in the Department of Information and Communication Technology of Manipal Institute of Technology, Manipal, under the guidance of **Vibha Prabhu, Assistant Professor Senior Scale**, Department of Information and Communication Technology, M. I. T., Manipal. No part of this work has been submitted for the award of a degree or diploma either to this University or to any other Universities.

Place: Manipal

Date :10-05-23



Harsha Vardhan Kumar Khandyana

Place: Hyderabad

Date: 24/05/2023

TO WHOM IT MAY CONCERN

*This is to certify that **Mr. Harsha Vardhan Kumar Khandyana** has successfully completed **Internship** from 5th January 2023 to 5th May 2023. During the period of his Internship with us he was found punctual, hardworking and inquisitive.*

We wish him every success in life.



Authorized Signatory



+91 97007 79739



info@semicoretech.com
www.semicoretech.com



5th Floor, Durga Bhavani Complex,
Satyam Theatre Rd, Gayatri Nagar,
Ameerpet, Hyderabad,
Telangana 500016



MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

CERTIFICATE

This is to certify that this project entitled **Your Project Title** is a bonafide project work done by **Mr. Harsha Vardhan Kumar Khandyana (Reg.No.:190911234)** at Manipal Institute of Technology, Manipal, independently under my guidance and supervision for the award of the Degree of Bachelor of Technology in Information Technology.

Mrs. Vibha
Assistant Professor Senior Scale
Department of I & CT
Manipal Institute of Technology
TechnologyManipal, India

Dr. Smitha N Pai
Professor & Head
Department of I & CT
Manipal Institute of
Manipal, India

ACKNOWLEDGEMENTS

It gives me a great sense of pleasure to present the report of the Project Work undertaken during R. Tech. Final Year. I owe special debt of gratitude to my Project Guide Mrs. Vibha, Assistant Professor Senior Scale, Manipal Institute of Technology, Manipal for her constant support and guidance throughout the course of my work. It is only her cognizant efforts that my endeavors have seen light of the day.

My deepest thanks to my Project Coordinator Mrs. Manjula Belagavi, Assistant Professor Senior Scale, and Head of the Department Dr. Smitha Pai Department of Information Technology, Manipal Institute of Technology, for guiding and correcting various documents of mine with attention and care.

Along with my family and friends, I would want to thank everyone who helped me and contributed to the successful completion of this internship.

Harsha Vardhan Kumar Khandyana,
190911234,
Information Technology,
MIT, Manipal.

ABSTRACT

One of the most popular methods of payment in both online and offline transactions is the use of a credit card. As the number of transactions increase the chances for fraud also increases. Due to this type of fraud the banks and people both lose their hard-earned money to thieves and criminals. To identify this type of fraud transactions we are using dataset provided by the European Banks and training various deep learning models to predict new fraud transactions.

The dataset is preprocessed before feeding it to the model. The preprocessing includes filling the null values, rectifying the skewness of the attributes, dividing the dataset into training and test datasets, plotting graphs between attributes to determine the correlation between them. The Random Forest, Vanilla Autoencoder, LSTM Autoencoder and XGBclassifier models were used to predict the outputs. Evaluation metrics AUC score, F1 score, Precision, Recall, Accuracy is used to compare the models.

The AUC score measures the capacity of binary classifiers to distinguish between classes, which is the same exact thing we need as we want differentiate between the normal and fraud transactions. Taking the AUC score values as the main evaluation metric for comparing the models.

These are the AUC scores obtained for different models in this project, Random Forest model got 0.8877 score, Vanilla Autoencoder model got 0.9036, LSTM Autoencoder model got 0.9352, XGBoost Classifier model got 0.992.

[Computing Methodology] - Search methodologies - Discrete space search

[Computing Methodology] - Machine learning - Anomaly detection

[Computing Methodology] - Machine learning algorithms- Spectral methods - Feature selection

[Computing Methodology] - Machine learning algorithms- Spectral methods - Regularization

[Computing Methodology] - Machine learning algorithms- Cross-validation

Table of Contents

DECLARATION	2
CERTIFICATE	4
ACKNOWLEDGEMENTS	5
ABSTRACT	6
Chapter-1.....	9
Chapter-2.....	12
Chapter-3.....	14
3.1 Project goals	14
3.2 Objective	15
Chapter-4.....	16
4.1 Pre-processing	17
4.2 Feature extraction	20
4.3 SMOTE	21
4.3 Training.....	21
4.4 Evaluation	22
4.5 Metrics.....	23
Chapter-5.....	25
5.1 Random Forest.....	25
5.2 Autoencoder	27
5.3 LSTM Autoencoder:	29
5.4 XGBoost Classifier	31
Scope	34
Conclusion	35
References	39

Table of Figures

Figure 4.1 - Methodology Flowchart	16
Figure 4.2 - Scatter Plot of Amount vs Time	18
Figure 4.3 - Distribution Curves for all Attributes.....	19
Figure 4.4 - Histograms for all attributes.....	20
Figure 4.5 - Scatter Plots before and after Oversampling.....	21
Figure 5.1 - Random Forest tree generated	25
Figure 5.2 - ROC Curve for Random Forest Model.....	26
Figure 5.3 - Confusion Matrix for Random Forest	26
Figure 5.4 - Autoencoder model architecture	27
Figure 5.5 - ROC curve for Autoencoder model.....	28
Figure 5.6 - Confusion Matrix for Autoencoder.....	28
Figure 5.7 - LSTM Autoencoder model.....	29
Figure 5.8 - ROC curve for LSTM Autoencoder	30
Figure 5.9 - Confusion Matrix for LSTM Autoencoder	30
Figure 5.10 - ROC curve for XGBoost Classifier.....	31
Figure 5.11 - Output from GridSearchCV.....	31
Figure 5.12 - Best Hyperparameters	32
Figure 5.13 - Confusion Matrix for XGBoost Classifier	32

Chapter-1

Introduction



Credit Card Fraud Detection

Using the Machine Learning Classification Algorithms to detect Credit Card Fraudulent Activities

dataaspirant.com

[Source](#)

The usage of a credit card is one of the most widely used payment methods for both online and offline transactions. The crooks were located using the credit card numbers of the victims. This strategy might reduce the frequency of fraudulent transactions. Each credit user receives a unique code comprising information about the transaction, the payment amount, and the most recent purchase date. Any departure from this pattern is regarded as a fraud instance. Unauthorized people are more adaptable and unafraid to adjust their behavior to reduce the probability of becoming trapped. Anti-fraud solutions must also be adaptable because this is essential. The models Random Forest, Autoencoders, and LSTM Autoencoders were employed in this project to detect anomalies and compare their performance on the test dataset.

Credit card information and data are regularly used fraudulently to make transactions and cash withdrawals [1].

This is the most prevalent scam technique. Credit cards and networked banking are becoming more and more popular, as are internet shopping and other

online activities. The cardholder only learns about fraud after it has already occurred. Currently, it is impossible to recognize a fraudulent transaction due to the lack of a computer model and software. Users, cardholders, and corporate assets are all at danger from credit card theft. Money lost to fraud increased by 16.2 percent between 2017 and 2018, according to the Nelson Report 2019 on Card Users, Merchandisers, and ATM Acquirers of Card Deals at ATMs [4].

Since fraudulent transactions are less frequent than legitimate ones, we may classify fraudulent transactions as anomalies and apply anomaly detection techniques to identify transactions that deviate from legitimate ones and categorize them as fraudulent in our test scenario.

We may utilize unsupervised learning algorithms like K-Nearest Neighbors, other clustering techniques, or even algorithms like Support Vector Machine to find anomalies.[1]

Shen found that neural networks beat both SVM and LR in the detection of credit card fraud. According to Bhattacharyya, decision trees (DTs) outperform SVMs, LRs, and the pair of LRs in terms of fraud detection and false positives. Therefore, decision trees are used in this project.

Fraud detection has become a costly issue due to automated methods that generate a high number of false positives, which has a negative impact on revenue. A system that accurately recognises fraud in its early stages is essential. Machine learning models are effective in detecting anomalies, as has been obvious in recent years. Autoencoders have been in great demand because of their capacity to gather knowledge about usual behaviour and utilise that recognition to detect aberrations that are not thought to be normal. It is possible to employ an event-based or sequence-based strategy when looking for abnormalities in transaction data. In contrast to the sequence-based method, the event-based approach evaluates each data point separately rather than as a part of a subsequence. Similar to this, a recent study examined the use of an oversampling approach and found that the LSTM performed well in terms of accuracy.

But there are still substantial flaws in the LSTM approach. When the LSTM is initially trained, it is difficult for the discriminator and generator to achieve theoretical Nash equality. Convergence of the model is a challenge. Cases will be reproduced since the provided data show little variation. Due to its loss function and the peculiarities

of text data, the LSTM is mostly inadequate for discrete data processing [6]. As a result, we are employing the XGBoost classifier to address LSTM's drawbacks.

To find the ideal XGBoost classifier hyperparameters, GridSearchCV is employed.

Chapter-2

Literature Survey

The identification of fraud in credit card transactions is studied using Artificial Neural Networks (ANN) and Bayesian Belief Networks, two popular anomaly detection techniques. Huge uses credit card transaction data and ROC as two of the most important variables in their analysis. It has been demonstrated in this study that ANN and BBN are effective in detecting credit card fraud. Additionally, they come to the conclusion that breakthroughs are possible.[2]

In the identification of credit card fraud, Shen discovered that neural networks outperformed both SVM and LR. In terms of fraud detection and false positives, Bhattacharyya claims that decision trees (DTs) perform better than SVMs, LR, and the pair of LR.[3]

Hawkins recommended employing autoencoders for anomaly detection in 2002. Since then, auto coders have been used for a variety of purposes, including anomaly detection.[4]

Graves and colleagues demonstrate how LSTMs may be used to build complex sequences by predicting one data point at a time. Graves et al. argue that the network needs a centralised server (LSTM) to accomplish this purpose. These predictions have little margin for error because they are entirely based on prior inputs, and the network's ability to correct prior errors is extremely unlikely.[5]

According to Dai et al., LSTM layers employed in autoencoders may be used for both sequence prediction and perform better than regular LSTM networks. Park et al. in these publications use the Adam Optimizer and a learning rate of 0.001 for their LSTM-based Autoencoder. Their model beat a number of competing techniques, including support vector machines and standard Autoencoders, in the detection of abnormalities [6].

For the IEEE-CIS Fraud Detection Competition, Yixuan Zhang et al.'s [7] XGBoost classifier was created using the Kaggle dataset. When XGBoost's performance was compared to other methods like Support Vector Machine, Random Forest, and Logistic Regression, it performed admirably and had an accuracy of 97.1.[8] XGBoost and data oversampling with SMOTE have both been used in combination

with several feature engineering techniques.[9]

Halvaiee and Akbari have researched the usage of an artificial immune framework for fraud detection [10]. In the author's work, the artificial recognition structure method is given certain enhancements that increase the model's accuracy while lowering its cost and training time. The comparison of several techniques for detecting fraud by Izotova et al. [11]. The research determines the possibility for fraud prediction using both the homogeneous and heterogeneous Poisson processes. Additionally, it used boosting and machine learning algorithms as part of ensemble techniques to categories issues Page 5/14, and the outcomes were compared. To handle unbalanced classes in the dataset, Victoria Priscilla [12] presented an OXGBoost (optimized XGBoost) strategy without the need of resampling techniques. He employed the 98% accurate RamdomizedSearchCV optimization approach to find the OXGBoost' s optimum parameters.[14]

Chapter-3

Problem Definition

Anomaly detection is required here because the dataset is highly imbalanced with very few fraud transactions compared to the normal transitions. Hence, we can treat the fraud transactions as outliers and train our model on the normal transactions makes our model to detect fraud transactions more efficiently. The efficiency comes because when we use our autoencoders trained on normal transactions to predict fraud transactions the reconstruction error is very high making it easy to classify them as fraud.

In this project, you will analyse customer-level data that has been collected and analysed during a research collaboration of Worldline and the Machine Learning Group.

3.1 Project goals

- To determine whether autoencoders might be utilised to look for anomalies in credit card transaction records. This issue was treated as a classification problem by introducing various categories of anomalies and methods for locating each anomaly. Detecting anomalies helps us identifying fraud transactions.
- Apply suitable pre-processing techniques to reduce the skewness of the attributes.
- Check if time and amount of transaction plays a role in determining the type of transaction.
- Trying out different oversampling methods like SMOTE to overcome the problem of imbalanced dataset.
- Find other methods and compare the results between different models.
- Identifying the best hyperparameters for training.

3.2 Objective

To identify unusual patterns that do not conform to expected behavior,
Applying suitable preprocessing techniques to determine which suits best to train the autoencoder model. It was identified Standardize the data as helped the model detect anomalies

Develop a deep learning model using autoencoders to find anomalies.
Identify suitable data visualization technique to detect the anomalies.
Identify suitable data visualization technique to visualize the dataset and draw conclusions from them.

Training a Machine Learning model on a dataset consisting for 29 attributes obtained by applying principal component analysis to credit card transaction data to identify fraudulent transactions. To do this we are using Random Forest, Autoencoder and LSTM Autoencoders models.

Three different types of fashions random forest, vanilla autoencoder, and the LSTM autoencoder had been developed. Choosing the best model is the main objective of this project.

Check if time and amount of transaction plays a role in determining the type of transaction.

Oversample the imbalanced dataset and check if it effects the results of the models.

Find Suitable methods to find the best hyperparameters for the model training process to show the best results that the model can offer

Methodology



4.1 Pre-processing

Pre-processing is the initial step. At this point, the specified dataset is imported together with all required Python libraries. The dataset is carefully examined to determine whether any values are missing, and the class distribution of the

The target variable is finished. Two categories have been created for easier understanding; Class 0 stands for legitimate transactions, while Class 1 stands for fraudulent transactions. Additionally, it has been noted that the dataset mostly consists of legitimate transactions, with very few fraudulent ones.

Fraudulent transactions make up just 0.17% of the entire dataset. The dataset is very unbalanced as a result. Scaling the dataset in which Principal Component Analysis (PCA) has been used marks the conclusion of the preprocessing stage.

Dimensionality reduction of large datasets is accomplished via PCA. This is accomplished by downsizing a larger dataset while maintaining the integrity of the data. PCA makes the data simple to grasp for machine learning algorithms because smaller datasets are better for visualization and exploration [23]. Therefore, its main goal is to reduce the dataset's number of variables while retaining as much of the data as possible. The V1-V28 attributes, in contrast to column Amount and Time, are zero-centered, as may be seen after studying statistics. Histograms have been plotted for the aforementioned attributes to provide a more in-depth perspective.

The below scatter plot is plotted having time, amount differentiated by the types of transaction to check if there is any noticeable pattern. As we can see below, we couldn't identify any useful pattern because the fraud transaction is neither concentrated at a particular time nor at a specific transaction amount.

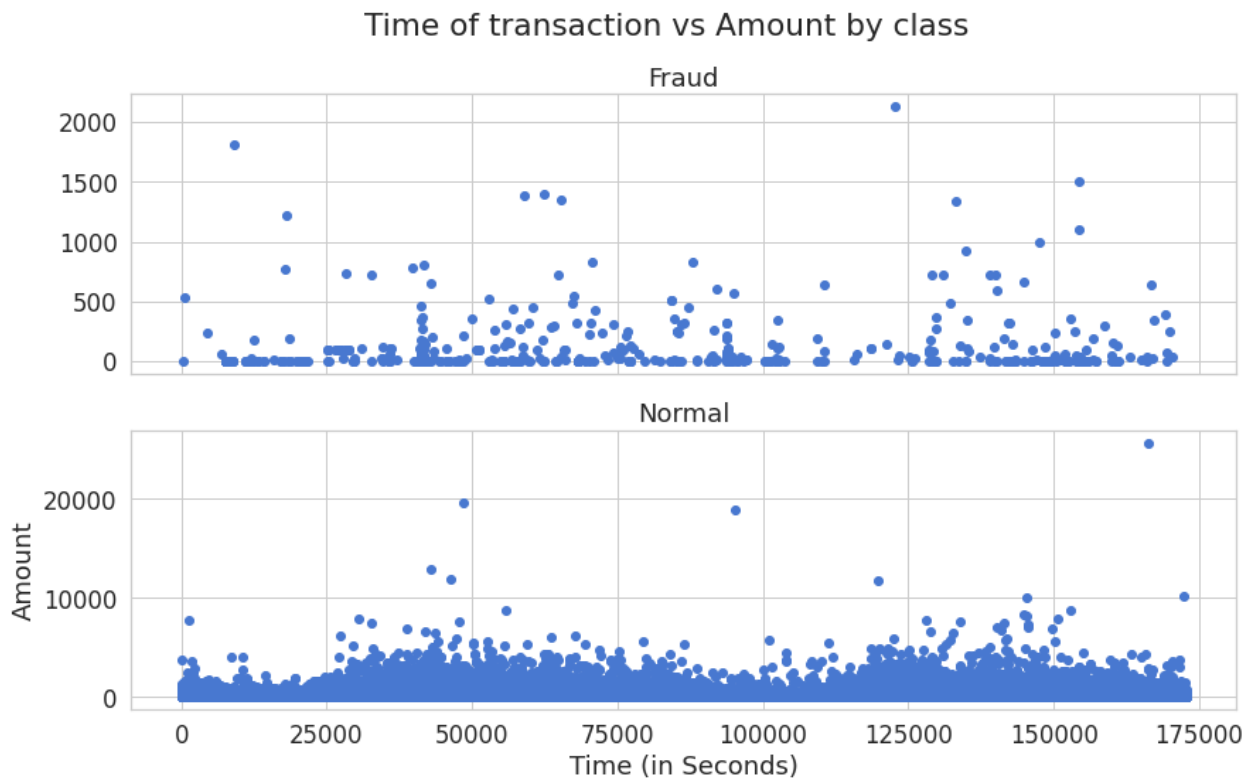


Figure 4.1 - Scatter Plot of Amount vs Time

Distribution curves for each attribute is plotted where fraud and normal transactions are plotted differently to see the difference of the distribution for normal and fraud transactions if the distribution is same for normal and fraud then we cannot use that attribute to differentiate them. From the above graphs we can see for V15, V17, V24, V27 have the same distribution for both fraud and normal transactions hence these can be removed.

The remaining attributes are standardized.

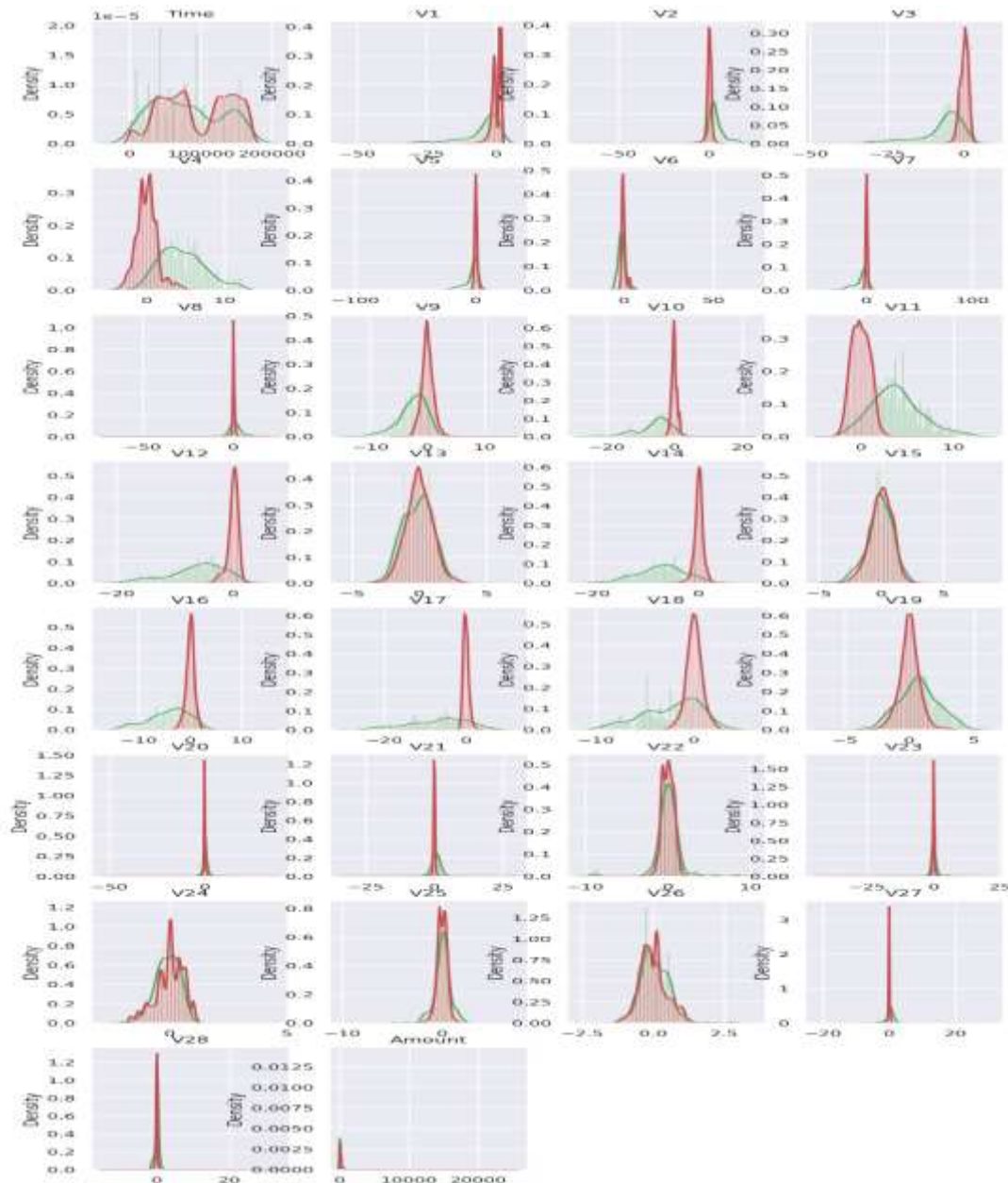


Figure 4.2 - Distribution Curves for all Attributes

The main transaction in the data set totals \$25,691.16, whereas the mean of all transactions is \$88.35. However, as shown in Figure below, the distribution of the monetary value of all transactions is strongly right-skewed. Most transactions are quite modest and only a small percentage are large.

Only a small percentage of transactions are close to the limit. To reduce this skewness, the used Power Transform method by Sklearn. This method applies zero-mean, unit-variance normalization to the transformed output.

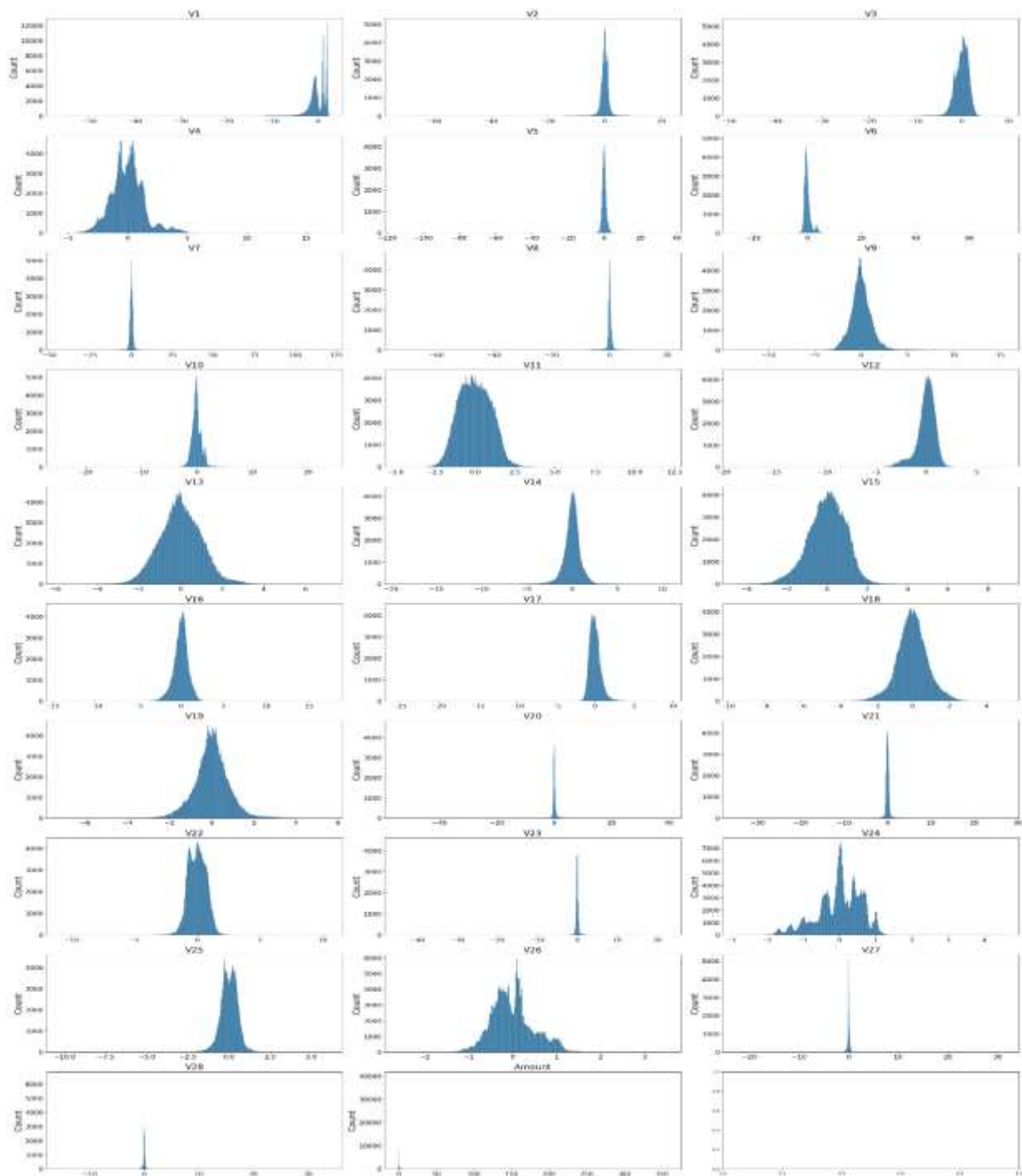


Figure 4.3 - Histograms for all attributes

4.2 Feature extraction

Feature extraction comes next. The appropriate characteristics are chosen at this point, given the necessary transformations, and then extracted for use. Due to the dataset's extreme imbalance, the SMOTE technique is used to oversample the data. Oversampling lowers the class size.

4.3 SMOTE

SMOTE oversamples the marginal class data by adding fresh synthetic examples to the count discrepancy. SMOTE is a method of oversampling that creates artificial samples from the minority class. In order to train the classifier, it is utilized to create a training set that is artificially class-balanced or nearly class-balanced.

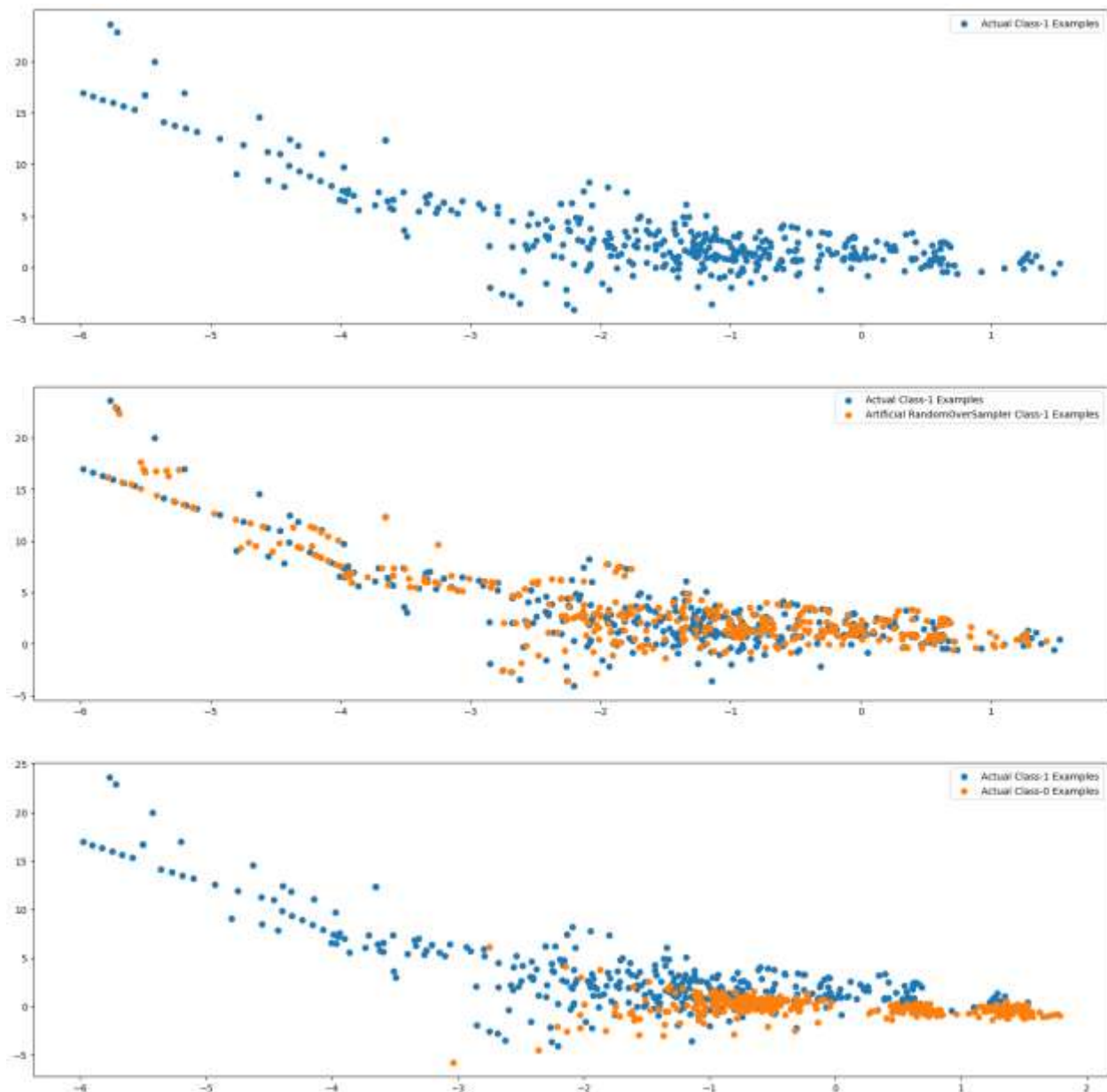


Figure 4.4 - Scatter Plots before and after Oversampling

4.4 Training

The model **training** step comes after the oversampled data has been trained using:

- Random Forest Model is trained on the dataset
- Autoencoder with 3 layers

- LSTM Autoencoder
- The XGBoost classifier and cross-validation has been done using a 3-fold stratified k-fold for each of 8 candidates, totaling 24 fits.

4.3.1 Training of XGBoost classifier:

GridSearchCV is a method for adjusting hyperparameters to find the best values for a particular model. As was already noted, a model's performance is strongly influenced by the value of its hyperparameters. Noting that there is no way to determine the best values for hyperparameters in advance, it is ideal to explore every conceivable value before deciding what the best ones are. We utilize GridSearchCV to automate the tweaking of hyperparameters because doing it manually could take a lot of time and resources.

4.3.1.1 Pseudocode for GridSearchCV in this model:

For each learning rate in learning rates:

 For each max depth in max depths:

 For each subsample size in sizes:

 k-fold cross validation is done for model with the hyperparameters

all the results for each combination of hyperparameters is stored and they can be accessed using built in methods. The best hyperparameters are chosen using the metrics mentioned below and the classifier trained using them is used make predictions.

4.4 Evaluation

Evaluation process for Autoencoder Models:

The model trained on the normal transaction dataset is used to predict the test dataset. As mentioned above as the encoder has never seen the fraud transactions before the reconstruction error for the fraud transactions is high.

The threshold for the reconstruction error separating normal and fraud transactions is determined by taking the 99th percentile of the losses obtained from reconstruction of the training dataset by the model. 99th percentile is taken because I have assumed that 1% of the dataset has fraud transactions. After

predicting the results for test dataset, the losses are taken and the predictions are made using the threshold. The following metrics AUC Score, Precision, Recall, F1 score and Confusion matrix, later these metrics are used to compare different models and determine the best model.

Evaluation process for Random Forest and XGBclassifier:

The classes for the test set are predicted using the classifier and the results are evaluated using the metrics AUC Score, Precision, Recall, F1 score and Confusion matrix. Later these metrics are used to compare different models and determine the best model.

4.5 Metrics

Accuracy is defined as the ratio of correct predictions to total forecasts.

The proportion of true positives to the total of true positives and false positives is known as precision.

The ratio of true positives to the total of both true positives and false negatives is known as recall.

The correlation value defines whether or not the traits are dependent on one another. The traits are favorably associated if the correlation value is positive, not related if it is zero, and negatively related if it is negative.

Precision-Recall is a useful measure of success of prediction when the classes are very imbalanced. Since our data is highly imbalanced, we use this. In information retrieval, precision is a measure of result relevancy, while recall is a measure of how many truly relevant results are returned.

The Receiver Operator Characteristic (ROC) curve is a measuring tool for binary classification problems. Plotting the TPR versus the FPR at different threshold levels essentially distinguishes the "signal" from the "noise" In other words, it demonstrates the overall effectiveness of a classification scheme. The ability of a binary classifier to discriminate between classes is measured by the Area Under the Curve (AUC), which summarizes the ROC curve.

True Positive: The outcome was as expected and is actual.

accurate Negative: A prediction that turns out to be accurate.

A false positive is one that was anticipated to be positive.
False bad: a bad outcome that was falsely expected.

The confusion matrix includes each of these variables.

Fraud detection is complicated by the cost of misclassifying a transaction. It is still acceptable to label a routine transaction as fraudulent, but the consequences of failing to catch a fraudulent transaction are severe. It is a viable method if we can reduce the number of false negatives while raising the number of false positives by offsetting the threshold.

Chapter-5

Results

The following are the results obtained after training Random Forest, Autoencoder, LSTM Autoencoder and XGBoost classifier models on the oversampled training dataset. Predictions for the test dataset are made using these models and the metrics AUC Score, Precision, Recall, F1 score and Confusion matrix are used to compare different models and determine the best model.

5.1 Random Forest Tree generated by training with given dataset

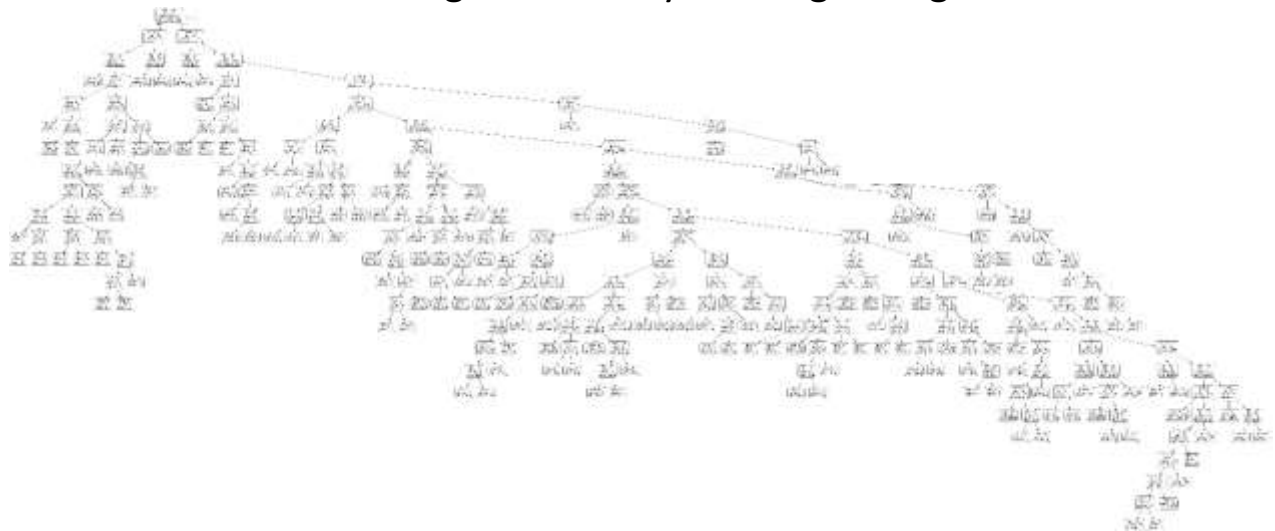


Figure 5.1 - Random Forest tree generated

Random Forest Model Evaluation results:

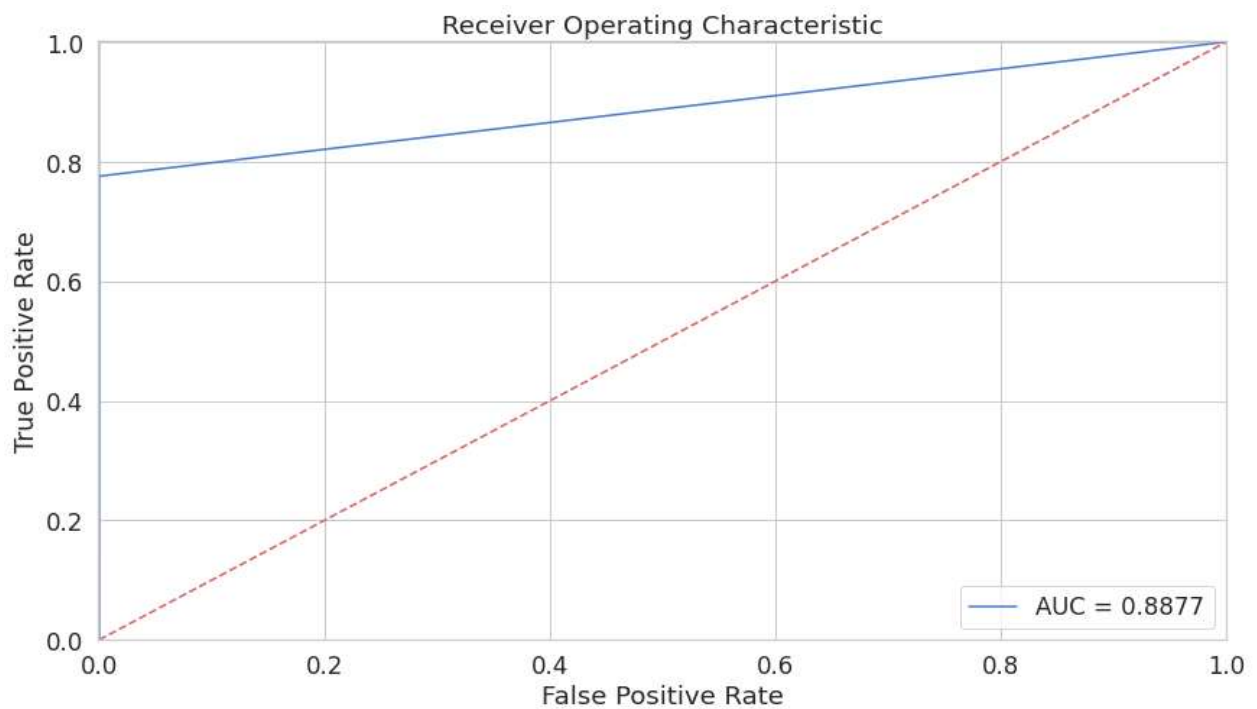


Figure 5.2 - ROC Curve for Random Forest Model

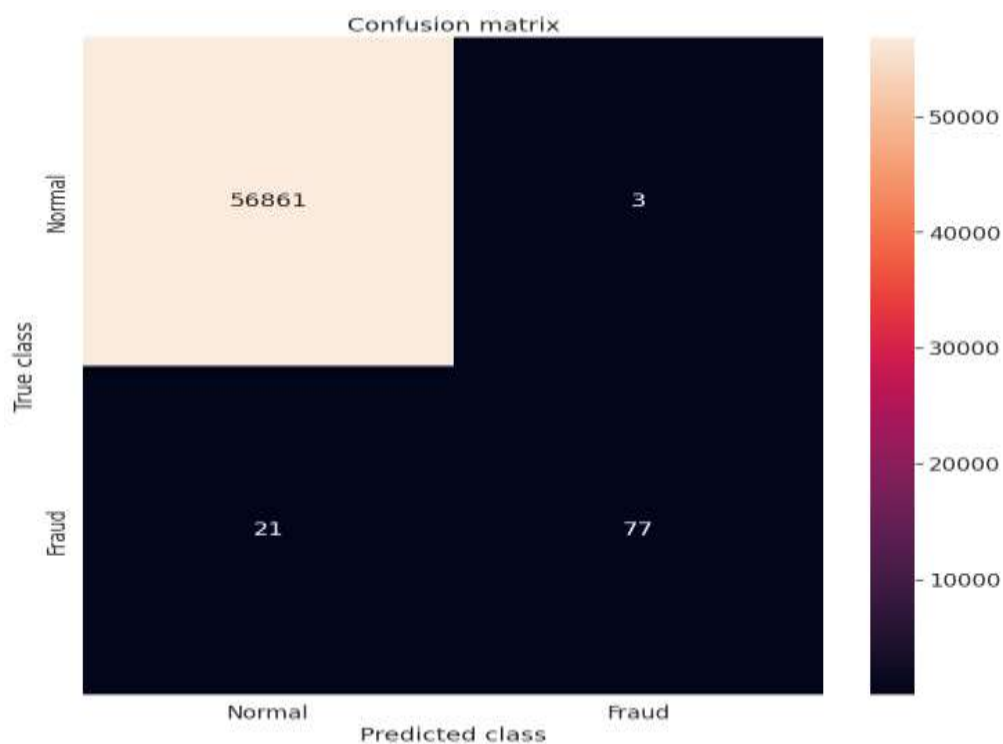


Figure 5.3 - Confusion Matrix for Random Forest

The model used is Random Forest classifier

The accuracy is 0.9995786664794073

The precision is 0.9625

The recall is 0.7857142857142857

The F1-Score is 0.8651685393258427

The Matthews correlation coefficient is 0.8694303688259544

5.2 Autoencoder

Autoencoder which has 3 encoding layers of units 100,50,25 and 2 decoding layers 50,100 and one output layer.

Autoencoder which has 3 encoding layers of units 100,50, dropout layer,25 and 2 decoding layers 50, dropout layer ,100 and one output layer.

[] Model: "encoder"		
Layer (type)	Output Shape	Param #
dense_6 (Dense)	(None, 100)	3000
dropout (Dropout)	(None, 100)	0
dense_7 (Dense)	(None, 50)	5050
dense_8 (Dense)	(None, 25)	1275
=====		
Total params: 9,325		
Trainable params: 9,325		
Non-trainable params: 0		
=====		
▶ decoder.summary()		
[] Model: "decoder"		
Layer (type)	Output Shape	Param #
dense_9 (Dense)	(None, 50)	1300
dense_10 (Dense)	(None, 100)	5100
dropout_1 (Dropout)	(None, 100)	0
dense_11 (Dense)	(None, 29)	2929
=====		
Total params: 9,329		
Trainable params: 9,329		
Non-trainable params: 0		
=====		

Figure 5.4 - Autoencoder model architecture

Autoencoder model evaluation results:

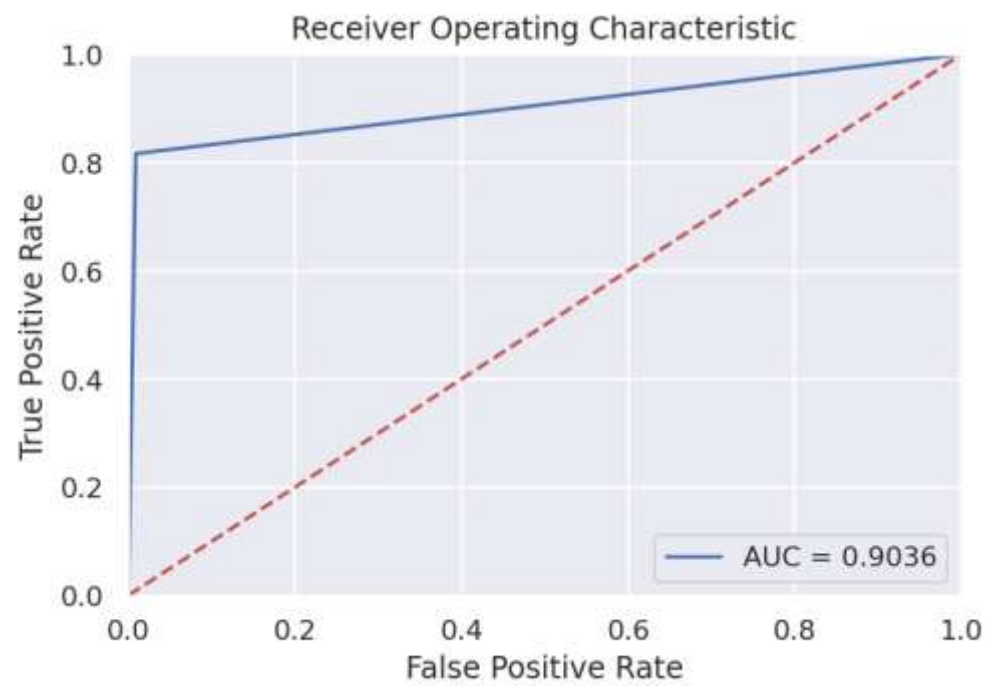


Figure 5.5 - ROC curve for Autoencoder model

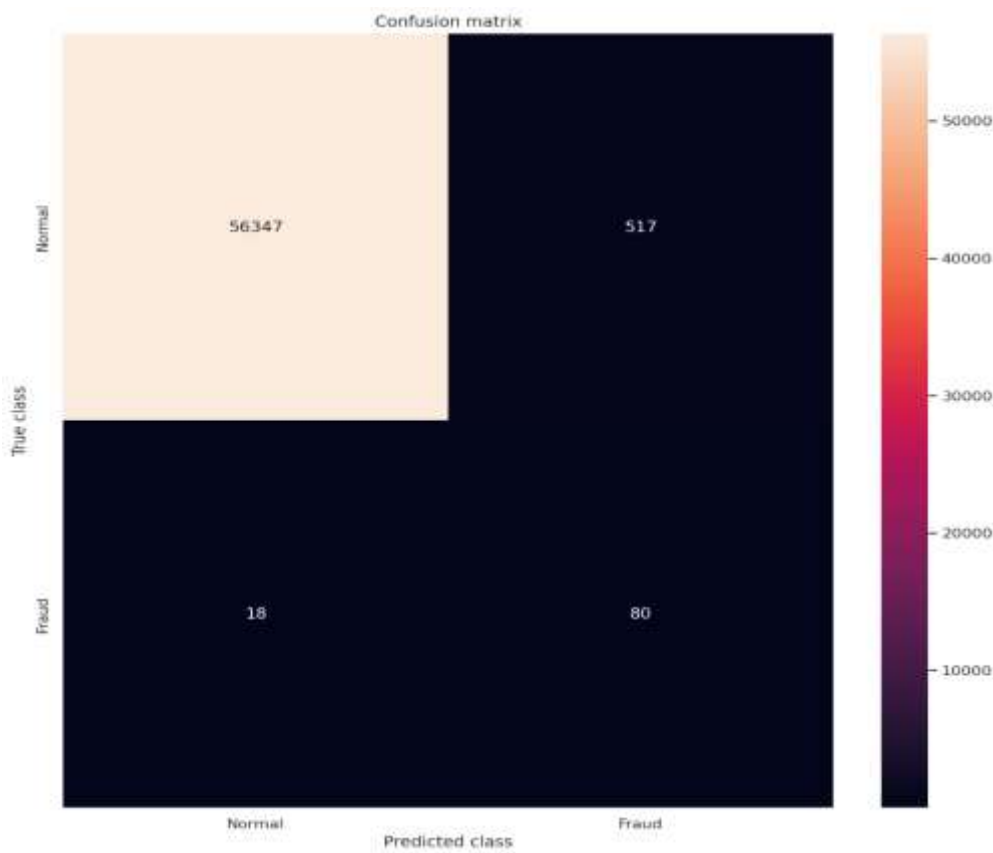


Figure 5.6 - Confusion Matrix for Autoencoder

Implementation of Simple Autoencoder is done. The Evaluation of predictions of the model on the test dataset are shown above. I used ROC Curve, AUC and confusion matrix as main metrics to measure the performance of the model because they give a clear picture of the number of false positives and false negatives which are important in our case.

Other results

	precision	recall	f1-score	support
0	1.00	0.99	1.00	56864
1	0.13	0.82	0.23	98
accuracy			0.99	56962

5.3 LSTM Autoencoder:

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 100)	44400
repeat_vector (RepeatVector)	(None, 1, 100)	0
lstm_1 (LSTM)	(None, 1, 100)	80400
time_distributed (TimeDistributed)	(None, 1, 10)	1010
=====		
Total params: 125,810		
Trainable params: 125,810		
Non-trainable params: 0		

Figure 5.7 - LSTM Autoencoder model

LSTM Autoencoder Evaluation Results:

The accuracy is 0.9350954633651642

The precision is 0.9895151411372647

The recall is 0.8796707221396035

The F1-Score is 0.9313653777011865

The Matthews correlation coefficient is 0.8756321155383204

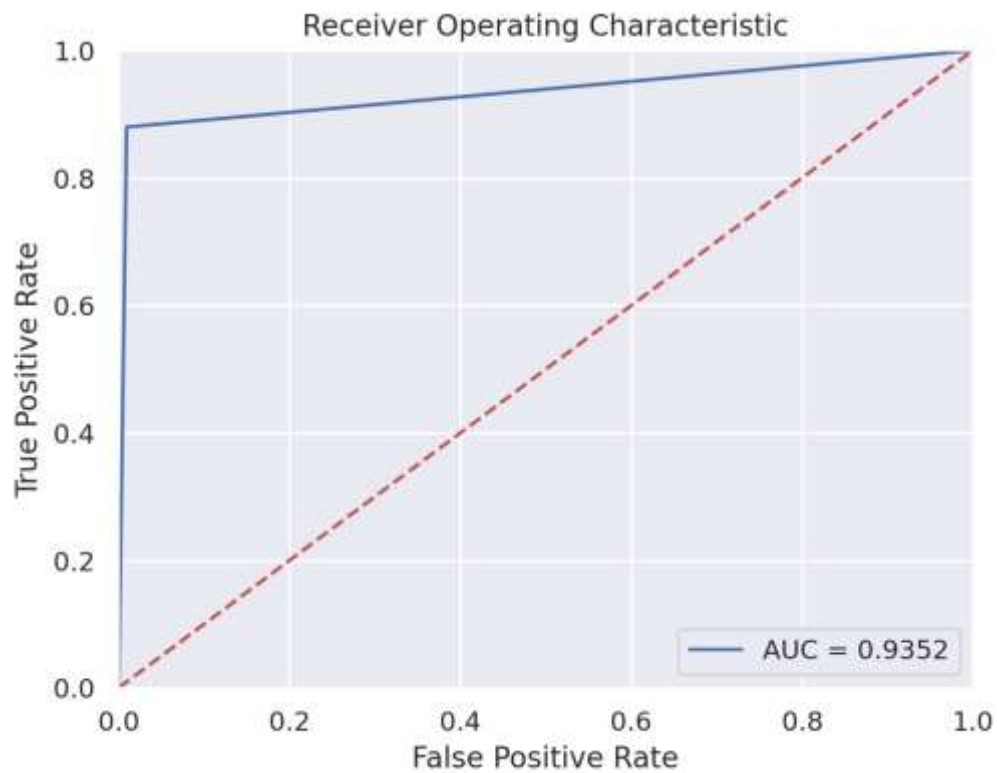


Figure 5.8 - ROC curve for LSTM Autoencoder



Figure 5.9 - Confusion Matrix for LSTM Autoencoder

5.4 XGBoost Classifier

XGBoost Classifier Evaluation Results:

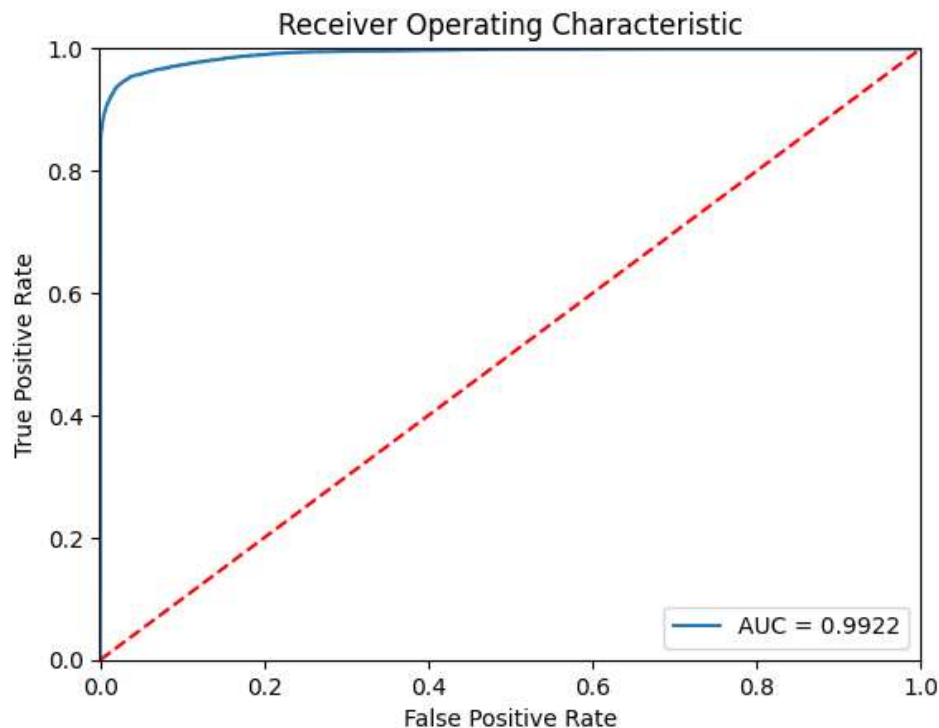


Figure 5.10 - ROC curve for XGBoost Classifier

Output from GridSearchCV:

Best hyperparameters are

Learning rate = 0.8

Max depth = 5

Subsample = 0.9

```
Mean test scores:
{'learning_rate': 0.8, 'max_depth': 5, 'subsample': 0.9} 0.9999932782930695
{'learning_rate': 1, 'max_depth': 5, 'subsample': 0.9} 0.9999878416511785

Rank of each hyperparameter combination:
1 {'learning_rate': 0.8, 'max_depth': 5, 'subsample': 0.9}
2 {'learning_rate': 1, 'max_depth': 5, 'subsample': 0.9}

Standard deviation of test scores:
{'learning_rate': 0.8, 'max_depth': 5, 'subsample': 0.9} 4.101474115165894e-06
{'learning_rate': 1, 'max_depth': 5, 'subsample': 0.9} 8.711287160329478e-06
```

Figure 5.4.2 - Output from GridSearchCV

Best ROC AUC score: 0.9999932782930695
Best hyperparameters: {'learning_rate': 0.8, 'max_depth': 5, 'subsample': 0.9}

Figure 5.11- Best Hyperparameters

XGBOOST Classifier ROC-AUC Score on Test Set = 0.9921570289753232

XGBOOST Classifier F1-Score on Test Set = 0.9248408605806464

XGBOOST Classifier Precision on Test Set = 0.9988575625280509

XGBOOST Classifier Recall on Test Set = 0.8610368598761958

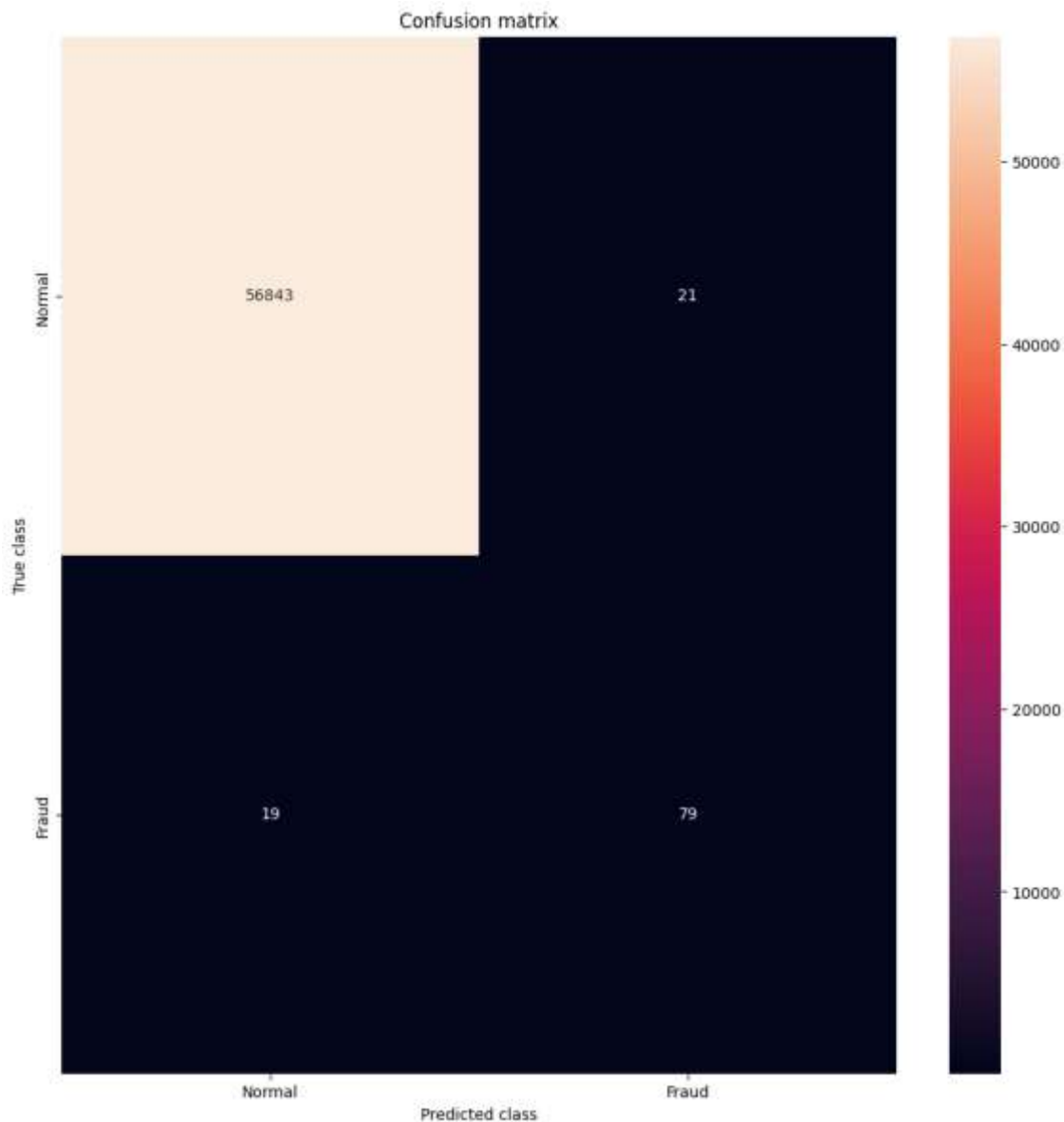


Figure 5.12 - Confusion Matrix for XGBoost Classifier

Inference of the Results:

MODEL	AUC SCORE
RANDOM FOREST	0.8877
VANILLA AUTOENCODER	0.9036
LSTM AUTOENCODER	0.9352
XGBCLASSIFIER	0.992

Since XGBoost classifier performs well with the unbalanced dataset, hence it has the highest AUC score. Additionally, XGBoost is a cost-efficient boosting ensemble method with a higher prediction level. And as a result, it will effectively detect the fraud.

Scope

A number of methods exist today for detecting credit card fraud, but none of them are totally effective at doing so before it occurs. It typically recognizes when the fraud has already happened and the banks are unable to stop the transaction. The banks have suffered as a result of this issue.

The **primary objective** of this research is to accurately and precisely identify fraudulent transactions.

Its helps both the banks and their credit card users at it helps identifying the fraudulent transactions thereby preventing the loss caused by theft. Banking Sector. This project can also be used in our fraud detection testcases and outlier/anomaly detection testcases also, because we are using Autoencoders which try reconstruct the original data from dataset given hence it can used to predict the general cases more efficiently since the fraud cases will have high reconstruction error.

Conclusion

Since the dataset was highly imbalanced i.e. The number of fraud transactions compared to the normal transactions is very less, hence finding the fraud was treated as anomalies and anomaly detection methods are used for fraud detection. This way of treating the fraud transactions as anomalies worked and the results are highly promising.

To deal with the imbalanced dataset SMOTE technique was used to oversample the minority class. This helped the models increase their AUC scores.

The AUC score measures the ability of a model to distinguish between positive and negative classes by calculating the area under the curve of the receiver operating characteristic curve. Hence it was taken as the main criteria for comparison.

The XGBoost classifier was used with GridSearchCV method to find the best hyperparameters gave the best AUC score among the other three.

The Simple Autoencoder and LSTM autoencoder took comparatively more time to train than the XGBoost and also had less AUC score.

The Random Forest has less training time compared to the XGBoost performed well the training data but has less AUC score on the test data.

Taking all the parameters into consideration XGBoost classifier performed the best with an AUC score of 0.992.

Appendices

Appendix A(Code)

```
1 fraud = df[df.isnull()
2 total = len(df)
3 fraud_percentage = (fraud/total)*100
4 fraud_percentage
5 # so lets say that 2% of transactions are fraud so lets find the threshold
6 0.177205439620014

7 # Calculating the error in the whole train data
8 reconstructed = autoencoder.predict(train_data_normalised)
9 train_loss = losses.mean(reconstructed, train_data_normalised).numpy()

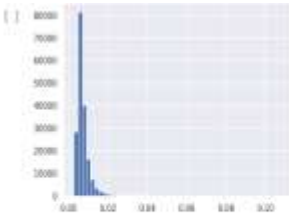
10 5607/5000 [=====] - 0s 1ms/step

11 # we are assuming that we have 2% of fraud transaction here
12 cut_off_1 = np.percentile(train_loss, 98)
13 cut_off_1
14 0.0289171720508726

15 if error is greater than this value we would say fraud transaction if lesser than cut-off means legitimate transaction

16 time for prediction on our TEST data


17 reconstructed = autoencoder.predict(test_data_normalised)
18 errors = losses.mean(reconstructed, test_data_normalised)
19 len(errors)
20 predicted = []
21 for error in errors:
22     if error > cut_off_1:
```



```
1 # calculating the train error - in fraud transactions
2 reconstructed = autoencoder.predict(fraud_train_data)
3 train_fraud_loss = losses.mean(reconstructed, fraud_train_data).numpy()
4 print(np.mean(train_fraud_loss))
5 plt.hist(train_fraud_loss, bins = 30)
6 plt.title('error fraud train loss')
7 plt.show()

8 10/10 [=====] - 0s 1ms/step
9 0.0185294594273301

10 error fraud train loss
```

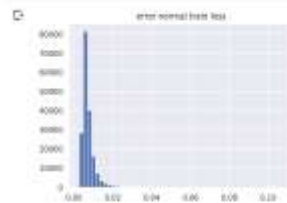


- Calculating error(loss) for normal transaction train data

```
1 # calculating the train error for legit transactions
reconstructed = autoencoder.predict(normal_train_data)
train_normal_loss = losses.mse(reconstructed, normal_train_data).numpy()
print(np.mean(train_normal_loss))
```

```
568/568 [=====] - 6s 1ms/step
0.00726497548338811
```

```
2 plt.plot(train_normal_loss, bins = 50)
plt.title('error normal train loss')
plt.show()
```



- lets find the train fraud loss

```
1 # calculating the train error for fraud transactions
reconstructed = autoencoder.predict(fraud_train_data)
train_fraud_loss = losses.mse(reconstructed, fraud_train_data).numpy()
print(np.mean(train_fraud_loss))
```

```
3 # train model
es = EarlyStopping(monitor='val_loss', min_delta=0.0001, patience=5, restore_best_weights=True)
history = autoencoder.fit(normal_train_data, y_normal_train_data, epochs=100, verbose=1, validation_data=(normal_validation_data, y_normal_validation_data), callbacks=[es])
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

```
Epoch 1/100
568/568 [=====] - 18s 2ms/step - loss: 0.8808 - mean_squared_error: 0.8808 - val_loss: 1.4956e-04 - val_mean_squared_error: 1.4956e-04
Epoch 2/100
568/568 [=====] - 13s 2ms/step - loss: 4.2237e-04 - mean_squared_error: 4.2237e-04 - val_loss: 2.5543e-04 - val_mean_squared_error: 2.5543e-04
Epoch 3/100
568/568 [=====] - 13s 2ms/step - loss: 3.3896e-04 - mean_squared_error: 3.3896e-04 - val_loss: 2.1185e-04 - val_mean_squared_error: 2.1185e-04
Epoch 4/100
568/568 [=====] - 14s 2ms/step - loss: 2.8695e-04 - mean_squared_error: 2.8695e-04 - val_loss: 1.5635e-04 - val_mean_squared_error: 1.5635e-04
Epoch 5/100
568/568 [=====] - 14s 2ms/step - loss: 2.6275e-04 - mean_squared_error: 2.6275e-04 - val_loss: 1.6686e-04 - val_mean_squared_error: 1.6686e-04
Epoch 6/100
568/568 [=====] - 13s 2ms/step - loss: 2.4875e-04 - mean_squared_error: 2.4875e-04 - val_loss: 1.4686e-04 - val_mean_squared_error: 1.4686e-04
Epoch 7/100
568/568 [=====] - 14s 2ms/step - loss: 2.4674e-04 - mean_squared_error: 2.4674e-04 - val_loss: 1.5648e-04 - val_mean_squared_error: 1.5648e-04
Epoch 8/100
568/568 [=====] - 14s 2ms/step - loss: 2.3387e-04 - mean_squared_error: 2.3387e-04 - val_loss: 1.2608e-04 - val_mean_squared_error: 1.2608e-04
Epoch 9/100
568/568 [=====] - 14s 2ms/step - loss: 2.2748e-04 - mean_squared_error: 2.2748e-04 - val_loss: 1.3343e-04 - val_mean_squared_error: 1.3343e-04
Epoch 10/100
568/568 [=====] - 13s 2ms/step - loss: 2.2384e-04 - mean_squared_error: 2.2384e-04 - val_loss: 1.3962e-04 - val_mean_squared_error: 1.3962e-04
Epoch 11/100
.....
```

```
1 # Feature importance
V25 -0.436292
V26 0.574060
V27 0.000209
V28 0.070409
Amount 18.120343
```

```
4 # Filtering the features which has skewness less than -2 and greater than +2
skewed = top_idx[(top["Skewness"] > 2) | (top["Skewness"] < -2)].index
skewed.to_list()
```

```
['V1',
 'V2',
 'V3',
 'V4',
 'V5',
 'V6',
 'V7',
 'V8',
 'V9',
 'V10',
 'V11',
 'V12',
 'V13',
 'V14',
 'V15',
 'V16',
 'V17',
 'V18',
 'V19',
 'V20',
 'V21',
 'V22',
 'V23',
 'V24',
 'V25',
 'V26',
 'V27',
 'V28',
 'Amount']
```

```
1 # Apply preprocessing.PipelineTransformer(cpp=False) to fit & transform the train & test data
pts.preprocessing.PipelineTransformer(methods=['no-joblib'], cpp=False) # creates an instance of the PipelineTransformer class.
pt.fit(X_train)

X_train_pt = pt.transform(X_train)
X_test_pt = pt.transform(X_test)
```

```

import time
import joblib as job
from sklearn.model_selection import GridSearchCV, StratifiedKFold

params = {
    'learning_rate': [0.8, 1],
    'max_depth': [3, 5],
    'subsample': [0.9, 0.8],
}

# Define the XGBoost classifier
xgb_classifier = job.XGBClassifier(objective='binary:logistic', eval_metric='auc') # As the number of classes are 2

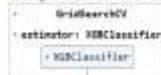
start_time = time.time()

# Create a GridSearchCV object with stratified cross validation
model_GridSearch = GridSearchCV(xgb_classifier,
                                param_grid=params,
                                scoring='roc_auc',
                                cv=skf,
                                n_jobs=-1,
                                verbose=0,
                                return_train_score=True)

# Fit the GridSearchCV object and perform hyperparameter tuning on the resampled data
model_GridSearch.fit(X_train, y_train)

```

Fitting 3 folds for each of 8 candidates, totalling 24 fits



```

cv_results = model_GridSearch.cv_results_

# Print the mean test scores for each hyperparameter combination
print("Mean test scores:")
for mean_score, params in zip(cv_results["mean_test_score"], cv_results["params"]):
    print(params, mean_score)

# Print the rank of each hyperparameter combination based on mean test score
print("\nRank of each hyperparameter combination:")
for rank, params in enumerate(cv_results["rank_test_score"]):
    print(rank+1, params)

# Print the standard deviation of test scores for each hyperparameter combination
print("\nStandard deviation of test scores:")
for std_score, params in zip(cv_results["std_test_score"], cv_results["params"]):
    print(params, std_score)

```

Mean test scores:

```

('learning_rate': 0.8, 'max_depth': 3, 'subsample': 0.9) 0.999932782594693
('learning_rate': 1, 'max_depth': 3, 'subsample': 0.9) 0.99993781651173

```

Rank of each hyperparameter combination:

```

1 ('learning_rate': 0.8, 'max_depth': 3, 'subsample': 0.9)
2 ('learning_rate': 1, 'max_depth': 3, 'subsample': 0.9)

```

Standard deviation of test scores:

```

('learning_rate': 0.8, 'max_depth': 3, 'subsample': 0.9) 4.38547411535894e-06
('learning_rate': 1, 'max_depth': 3, 'subsample': 0.9) 8.72128716852949e-06

```

```

# As PCA is already performed on the dataset from V1 to V18 features, we are scaling only 'Amount' field
scaler = RobustScaler()

# Transforming the test data
X_test_scaled[['Amount']] = scaler.fit_transform(X_test_scaled[['Amount']])

# Define the RandomOverSampler
smote = over_sampling.SMOTE(random_state=0)

# Resample the training data using RandomOverSampler
X_train_scaled_smote, y_train_scaled_smote = smote.fit_resample(X_train_scaled, y_train_scaled)

# Initialize the model with optimum hyperparameters
start_time = time.time()
clf = XGBClassifier(learning_rate=0.8, max_depth=3, subsample=0.9, objective='binary:logistic', eval_metric='auc')
clf.fit(X_train_scaled_smote, y_train_scaled_smote)
# Predict on test set to give probability
y_pred_proba = clf.predict_proba(X_test_scaled)
# Calculate the ROC-AUC score
roc_auc = roc_auc_score(y_test_scaled, y_pred_proba[:, 1])
print("XGBOOST Classifier ROC-AUC Score on Test Set = ", roc_auc)
# Predict on test set to get the class labels
y_pred = clf.predict(X_test_scaled)
# Calculate the F1-score, precision, and recall
F1 = f1_score(y_test_scaled, y_pred)
precison = precision_score(y_test_scaled, y_pred)
recall = recall_score(y_test_scaled, y_pred)
# Print the results
print("XGBOOST Classifier F1-Score on Test Set = ", F1)
print("XGBOOST Classifier Precision on Test Set = ", precison)
print("XGBOOST Classifier Recall on Test Set = ", recall)
end_time = time.time()
print("Time taken: {:.2f} seconds".format(end_time - start_time))

XGBOOST Classifier ROC-AUC Score on Test Set = 0.99237018975122
XGBOOST Classifier F1-Score on Test Set = 0.924849009380444
XGBOOST Classifier Precision on Test Set = 0.968857625288889
XGBOOST Classifier Recall on Test Set = 0.8918285861894

```

```
[ ] # Initialize the model with optimal hyperparameters
start_time = time.time()
clf = SGDClassifier(learning_rate=0.1, max_depth=1, subsample=0.5, objective='binary:logistic', eval_metric='auc')
clf.fit(X_train, y_train)
# predict on test set to give probability
y_pred_proba = clf.predict_proba(X_test_saved_data)
# calculate the ROC-AUC score
roc_auc = roc_auc_score(y_train_test_saved_data, y_score=y_pred_proba[:, 1])
print("ROC0057 Classifier ROC-AUC Score on Test Set = ", roc_auc)
# predict on test set to get the class labels
y_pred = clf.predict(X_test_saved_data)
# calculate the F1-score, precision, and recall
f1 = f1_score(y_test_saved_data, y_pred)
precision = precision_score(y_test_saved_data, y_pred)
recall = recall_score(y_test_saved_data, y_pred)
# print the results
print("ROC0057 Classifier F1-Score on Test Set = ", f1)
print("ROC0057 Classifier Precision on Test Set = ", precision)
print("ROC0057 Classifier Recall on Test Set = ", recall)
end_time = time.time()
print("Time taken: {:.2f} seconds".format(end_time - start_time))

ROC0057 Classifier ROC-AUC Score on Test Set = 0.992157020753232
ROC0057 Classifier F1-Score on Test Set = 0.824880805986664
ROC0057 Classifier Precision on Test Set = 0.958975621380989
ROC0057 Classifier Recall on Test Set = 0.8688848481048
```

```
[ ] X = tr_data.drop(['Class'], axis = 'columns')
label_data = tr_data['Class']
```

```
[ ] # Generate and plot imbalanced classification dataset
from collections import Counter
from matplotlib import pyplot
from numpy import where
# summarize class distribution
counter = Counter(tr_data['Class'])
print(counter)
# scatter plot of examples by class label
for label, _ in counter.items():
    row_ix = where(tr_data['Class'] == label)[0]
```

```
Counter({0: 284315, 1: 482})
```

```
1 # transform the dataset
from sklearn.over_sampling import SMOTE
oversample = SMOTE()
X_r, y_r = oversample.fit_resample(X, tr_data['Class'])
# summarize the new class distribution
counter = Counter(y_r)
print(counter)
# scatter plot of examples by class label
for label, _ in counter.items():
    row_ix = where(y_r == label)[0]
```

```
2 Counter({0: 284315, 1: 284315})
```

```
[ ] from sklearn.preprocessing import StandardScaler
# Standardizing the data
X_r2 = StandardScaler().fit_transform(X_r)
```

```
[ ] X_train, X_test, y_train, y_test = train_test_split(X_r2, y_r, test_size=0.3)
```

```
[ ] X_train.shape
```

```
[ ] fpr, tpr, thresholds = metrics.roc_curve(y_train_test_saved_data, y_score=y_pred_proba[:, 1])
threshold = thresholds[np.argmax(tpr-fpr)]
print(threshold)
```

```
0.00017706863
```

```
3 from sklearn.metrics import roc_curve, auc
```

```
[ ] roc_auc = auc(fpr, tpr)
```

```
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, label='AUC = %0.2f' % roc_auc)
plt.legend(loc='upper right')
plt.plot([0, 1], [0, 1], 'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



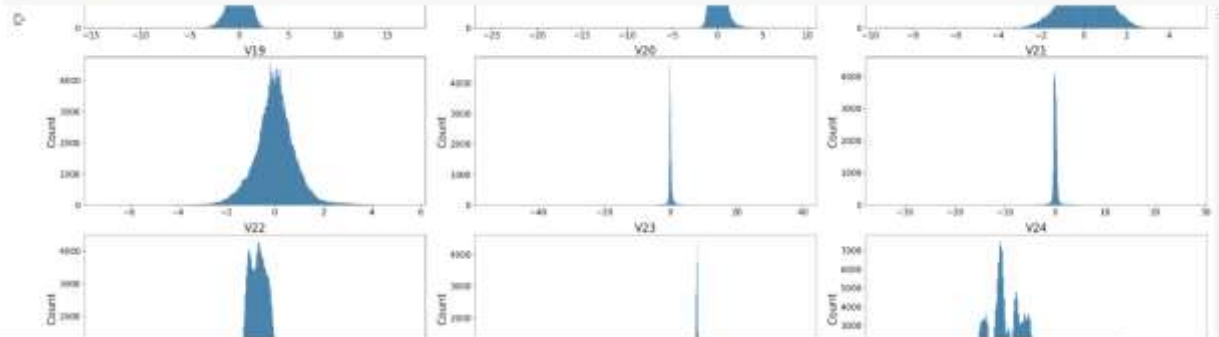
```

4 Fig, axes = plt.subplots(3, 3, figsize=(30, 40))
  axes = axes.flatten()

  for i, ax in enumerate(axes):
      if i < len(vars):
          ax.histplot(X_train(vars[i]), axes=axes)
          ax.set_title(vars[i], fontsize=10)
          ax.set_xlabel("Count", fontsize=20) # set label of the subplot
          ax.tick_params(axis="both", labelsize=10)
          ax.set_ylabel("") # set empty string as a label of the subplot

  plt.tight_layout()
  plt.show()

```



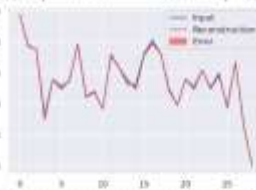
```

4 # Now let's define a function in order to plot the original timeseries and reconstructed ones and also show the error
def plot(data, n):
    dec_lag = autoencoder_predict(data) # This will decode or reconstruct
    plt.plot(data[n], 'b')
    plt.plot(dec_lag[n], 'r')
    plt.fill_between(np.arange(20), data[n], dec_lag[n], color = 'lightcoral')
    plt.legend(labels=['Input', 'Reconstruction', 'Error'])
    plt.show()

plot(normal_test_data, 4) # Here n shows the index of seg samples
plot(fraud_test_data, 4)

```

1777/1777 [-----] - 1s Seg/step



4/4 [-----] - 0s Seg/step



References

- [1] J. Gao, Z. Zhou, J. Ai, B. Xia, and S. Coggeshall, "Predicting credit card transaction fraud using machine learning algorithms," J. Intel. Learn. Syst.
- [2] A. Salazar, G. Safin, and L. Vergara, "Semi-supervised learning for imbalanced classification of credit card transaction," in Proc. IJCNN, Rio de Janeiro, Brazil, 2018.
- [3] Ghosh, S., Reilly, D.L., (1994). Credit card fraud detection with a neural network, in Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences, IEEE, Wailea, HI, USA. pp. 621–630.
- [4] Braise, R., Leinsdorf, T., Heap, M., (1999). Neural data mining for credit card fraud detection, in: Proceedings of the 11th International Conference on Tools with Artificial Intelligence, IEEE, Chicago, IL, USA. pp. 103–106.
- [5] V. V. Madhav and K. A. Kumari, "Analysis of Credit Card Fraud Data using PCA," 2020. [Online]. Available: www.iosrjen.org
- [6] A. Izotova and A. Vanillin, "Comparison of Poisson process and machine learning algorithms approach for credit card fraud detection," in Procedia Computer Science, 2021, vol. 186, pp. 721–726. Doi: [10.1016/j.procs.2021.04.214](https://doi.org/10.1016/j.procs.2021.04.214).
- [7] V. Shah, "Data Balancing for Credit Card Fraud Detection using Complementary Neural Networks and SMOTE Algorithm," 2020
- [8] S. F. Carcillo, Y. A. le Borgne, O. Caelen, Y. Kessaci, F. Oblé, and G. Bontempi, "Combining unsupervised and supervised learning in credit card fraud detection," Information Sciences, vol. 557, pp. 317–331, May 2021, Doi: [10.1016/j.ins.2019.05.042](https://doi.org/10.1016/j.ins.2019.05.042)
- [9] L. Seyedh Ossian and M. R. Hashem, "Mining information from credit card time series for timelier fraud detection," in 5th International Symposium on Telecommunications (IST'2010), 2010 © IEEE. Doi: 978-1-4244-8185-9/10/
- [10] Anomaly Detection in Credit Card Transaction using Deep Learning Techniques DOI: [10.1109/ICCES54183.2022.9835921](https://doi.org/10.1109/ICCES54183.2022.9835921)
- [11] Credit Card Fraud Detection using XGBoost Classifier with a Threshold Value DOI: <https://doi.org/10.21203/rs.3.rs-1722294/v1>
- [12] S. Lei, K. Xu, Y. Huang, and X. Sha, "An XGBoost based system for financial fraud detection," in E3S Web of Conferences, Dec. 2020, vol. 214. doi:10.1051/e3sconf/202021402042.
- [13] C. Meng, L. Zhou, and B. Liu, "A case study in credit fraud detection with SMOTE and XGBoost," in Journal of Physics: Conference Series, Aug. 2020, vol. 1601, no. 5. Doi: 10.1088/17426596/1601/5/052016.

- [14]C. V. Priscilla and D. P. Prabha, "Influence of optimizing xgboost to handle class imbalance in credit card fraud detection," in Proceedings of the 3rd International Conference on Smart Systems and Inventive Technology, ICSSIT 2020, Aug. 2020, pp. 1309–1315. Doi: [10.1109/ICSSIT48917.2020.9214206](https://doi.org/10.1109/ICSSIT48917.2020.9214206)

Project Details

<i>Student Details</i>			
Student Name	Harsha Vardhan Kumar Khandyana		
Register Number	190911234	Branch /Roll No.	IT-A
Email Address	Harshavk911@gmail.com	Phone No	7337429116
<i>Project Details</i>			
Project Title	Anomaly Detection using Deep Learning Techniques		
Project Duration	4 months	Start Date	5-01-2023
<i>Guide Details</i>			
Faculty Name	Mrs. Vibha		
Full contact address with pin code	Department of Information and Communication Technology, Manipal Institute of Technology, Manipal, Karnataka,576104		
Email address	vibha.prabhu@manipal.edu		

CO and PO Mapping

*[The COs for project work and the UG POs and PSOs are listed below. With respect to your project, map the COs to the POs and PSO's with the correlation levels as below **in consultation with your college guide:**

3 (if the CO has **HIGH EMPHASIS** to the POs and PSOs)

2 (If the CO has **MODERATE EMPHASIS** to the POs and PSOs)

1 (if the CO has **LOW EMPHASIS** to the POs and PSOs)

Put '–' (hyphen) if the CO is not related to the POs and PSOs]

CLOs		PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
ICT 4299.1	Assess the work available in the literature related to the project to identify the limitations and risks.	2	3	2	3	3	3	1	1	1	2	1	1
ICT 4299.2	Practice planning and time management in solving the problem.	2	3	3	1	1	2	1	2	1	2	3	1
ICT 4299.3	Demonstrate professional skills to work effectively in a team or individually.	2	2	2	3	3	3	2	3	2	3	2	3
ICT 4299.4	Develop the ability to adopt a methodological approach to solve societal problems..	2	3	2	3	2	3	3	2	3	3	2	3
ICT 4299.5	Conduct experimentation and testing to achieve the defined objectives through computing/coding/statistical analysis	2	3	2	2	2	3	3	3	3	2	2	2
ICT 4299.6	Compose the technical report with effective communication on incorporating ethical practices.	3	2	3	2	2	3	2	2	2	3	2	2
ICT 4299 (Avg. correlation level)		2.16	2.67	2.5	2.33	2.16	2.83	2	2.16	2	2.5	2	2

PROGRAM OUTCOMES (PO)

Engineering Graduates will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

CLOs		PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
ICT 4299.1	Assess the work available in the literature related to the project to identify the limitations and risks.	2	3	2	2	3	2	3	2	3
ICT 4299.2	Practice planning and time management in solving the problem.	3	2	3	2	2	3	3	2	2
ICT 4299.3	Demonstrate professional skills to work effectively in a team or individually.	3	2	2	2	2	2	2	2	3
ICT 4299.4	Develop the ability to adopt a methodological approach to solve societal problems.	2	3	2	3	2	2	3	2	3
ICT 4299.5	Conduct experimentation and testing to achieve the defined objectives through computing/coding/statistical analysis	3	2	3	2	3	2	3	2	3
ICT 4299.6	Compose the technical report with effective communication on incorporating ethical practices.	3	2	3	2	3	2	3	2	3
ICT 4299 (Avg. correlation level)		2.66	2.5	2.5	2.16	2.5	2.16	2.83	2	2.83

1. To identify, analyses and develop software systems using appropriate techniques and concepts related to information technology
2. To design an algorithm or process within realistic constraints to meet the desired needs through analytical, logical and problem-solving skills.
3. To apply state of the art IT tools and technologies, IT infrastructure management abilities in treading innovative career path as a prospective IT engineer
4. Apply the principles of science, maths and computer programming to solve complex problems related to information technology.
5. Apply knowledge of programming, computational intelligence, computer graphics and visualization, data analytics, software system design, cyber security to arrive at solutions to real world problems.
6. Apply IT knowledge to design and develop systems with respect to societal, user, customer needs, health and safety, diversity, inclusion, societal, environmental codes of practice and industry standard.
7. Integrate and interface industry relevant hardware and software components and technology to come up with innovative and creative solutions.
8. Use of industry standard software tools and platform to design and analyze IT systems.
9. Learn to function collaboratively as a member of leader in diverse teams in multidisciplinary settings to manage the process effectively and document, present and communicate with the engineering community.

COURSE Code	Course Title	PO 1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
ICT 4299	Project Work	2.16	2.67	2.5	2.33	2.16	2.83	2.6	2.1	2.5	2.6	2.5	2.5	2.1	2.5	2.6	2.1	2.83	2.83

*Kindly enter the average values from table A1.2 in the link provided (link for project details upload).

*Delete these lines when making the report submission.

IET (AHEP Mapping):

*[The CLOs for project work and the UG AHEP LOs are listed below. With respect to your project, map the CLOs to the AHEP LOs with the correlation levels as below **in consultation with your college guide:**

2 (If the CLO has HIGH EMPHASIS to the AHEP LO)

2 (if the CLO has MODERATE EMPHASIS to the AHEP LO) 1 (if the CLO has LOW EMPHASIS to the AHEP LO)

Put ‘–’ (hyphen) if the CLO is not related to the AHEP LO

Note: Make sure that all columns shown will have at least one entry.]

CLOs		C1	C2	C3	C4	C5	C6	C13	C16	C17
ICT 4299.1	Assess the work available in the literature related to the project to identify the limitations and risks.	2	3	2	3	2	3	2	2	3
ICT 4299.2	Practice planning and time management in solving the problem.	2	3	2	3	2	3	2	3	3
ICT 4299.3	Demonstrate professional skills to work effectively in a team or individually.	3	3	3	2	2	3	2	2	3
ICT 4299.4	Develop the ability to adopt a methodological approach to solve societal problems.	3	3	2	3	3	2	2	2	3
ICT 4299.5	Conduct experimentation and testing to achieve the defined objectives through computing/coding/statistical analysis	2	2	2	3	3	2	2	2	2
ICT 4299.6	Compose the technical report with effective communication on incorporating ethical practices.	2	2	3	2	3	3	3	3	2
ICT 4299 (Avg. correlation level)		2.33	2.66	2.33	2.66	2.5	2.66	2.16	2.33	2.66

IET (AHEP Mapping):

*[The CLOs for project work and the UG AHEP LOs are listed below. With respect to your project, map the CLOs to the AHEP LOs with the correlation levels as below **in consultation with your college guide:**

3 (If the CLO has HIGH EMPHASIS to the AHEP LO)

2 (if the CLO has MODERATE EMPHASIS to the AHEP LO) 1 (if the CLO has LOW EMPHASIS to the AHEP LO)

Put ‘–’ (hyphen) if the CLO is not related to the AHEP LO

Note: Make sure that all columns shown will have at least one entry.]

CLOs		C1	C2	C3	C4	C5	C6	C13	C16	C17
ICT 4299.1	Assess the work available in the literature related to the project to identify the limitations and risks.	2	3	2	3	2	3	2	2	3
ICT 4299.2	Practice planning and time management in solving the problem.	2	3	2	3	2	3	2	3	3
ICT 4299.3	Demonstrate professional skills to work effectively in a team or individually.	3	3	3	2	2	3	2	2	3
ICT 4299.4	Develop the ability to adopt a methodological approach to solve societal problems.	3	3	2	3	3	2	2	2	3
ICT 4299.5	Conduct experimentation and testing to achieve the defined objectives through computing/coding/statistical analysis	2	2	2	3	3	2	2	2	2
ICT 4299.6	Compose the technical report with effective communication on incorporating ethical practices.	2	2	3	2	3	3	3	3	2
ICT 4299 (Avg. correlation level)		2.33	2.66	2.33	2.66	2.5	2.66	2.16	2.33	2.66