

Abstract :	5
1. Introduction.	6
1.1. Types of credit card fraud.	6
1.2. Global cyberattacks report.	8
1.3. Problem statement.	9
2. Literature review.	10
3. Objective of the study.	11
3.1. Specific objective.	11
4. Methodology.	11
4.1. Dataset and dataset collection.	11
4.2. Dataset attribute.	11
5. Algorithms.	12
5.1. Decision Tree.	12
5.2. Logistic Regression.	13
5.3. Artificial Neural Networks (ANN)	13
6. Isolated virtual environment creation.	15
7. Libraries and dataset importation.	15
8. Dataset preparation	15
9. Data cleaning	16
10. Column-by-column Analysis	16
10.1. Descriptive statistics analysis.	16
10.2. Skewness analysis.	16
10.3. Outlier analysis.	16
10.4. Transformation.	16
10.5. Normalization.	17
10.6. Visualization.	17
11. Time column analysis.	17
12. V1 column analysis.	18
13. V2 column analysis	19

14. V3 column analysis. ....	20
15. V4 column analysis. ....	21
16. V5 column analysis. ....	22
17. V6 column analysis. ....	23
18. V7 column analysis. ....	24
19. V8 column analysis. ....	25
20. V9 column analysis. ....	26
21. V10 column analysis. ....	27
22. V11 column analysis. ....	28
23. V12 column analysis. ....	29
24. V13 column analysis. ....	30
25. V14 column analysis. ....	31
26. V15 column analysis. ....	32
27. V16 column analysis. ....	33
28. V17 column analysis. ....	34
29. V18 column analysis. ....	35
30. V19 column analysis. ....	36
31. V20 column analysis. ....	37
32. Amount column analysis. ....	38
33. Exploratory data analysis. ....	39
33.1 Linear relationship among the variables. ....	39
33.2. Feature engineering analysis. ....	40
33.3. Up sampling. ....	40
34. Modularity. ....	41
35. Model development. ....	41
35.1. Dataset split. ....	41
35.2. Training. ....	41
35.3. Validation. ....	42
35.4. Testing. ....	42

36. Model performance evaluation. ....	42
37. Result /Discussion. ....	43
37.1 Decision Tree. ....	43
37.2. Logistic Regression. ....	44
37.3. Artificial Neural Network (ANN). ....	45
38. Receivers Operating Characteristic Curve. ....	46
39. Flask APP development. ....	49
40. Flask APP deployment. ....	49
41. Conclusion. ....	51
42. Reference. ....	52

**Credit Card Fraud Detection Software**

**Reference number:(51 - 45)**

**Think Pacific Foundation**

**Eze, Jonas Chinweike**

Email: [Q2156726@live.tees.ac.uk](mailto:Q2156726@live.tees.ac.uk)

**Abstract :**

*Credit card is a small plastic (or metal) card issued by financial institutions to users, allowing them to borrow funds to purchase goods and services while credit card fraud refers to any deceptive activity aimed at obtaining financial gain with the card or its information without the knowledge of the cardholder or the issuing financial institution. This study developed machine learning models that can detect credit card fraud using machine learning algorithms: Decision Tree, Logistic Regression and Artificial Neural Network(ANN). Python programming language was use for the project. The dataset was gotten from kaggle thorough the link provided by the Think Pacific Foundation. It was cleaned, preprocessed and split into three sets for models development; 70% for training, 15% for validation and 15% for test. The algorithms were trained, cross validated with grid search method and 4-folds. They were also validated and tested. The models' performance was evaluated with confusion metrics parameters: Accuracy, Precision, Recall and F1 score. Their performance was further evaluated using Receivers Operating Characteristic(ROC) Curve and Area Under the Curve(AUC). The performance evaluation results demonstrated that the models achieved a good fit, well generalization ability to new and unseen data and excellent discriminative ability with Accuracy, Precision, Recall, F1 score and AUC values above 90% across all the phases of development(Training, Cross Validation, Validation and Test). The modes were saved and used to develop a Flask APP to examine their predictive ability. The APP was deployed locally as an Application Programming Interface(API) and tested with postman as client for simulation. It was further deployed locally on the web where a user interactive form was created using HTML. Additional user interactivity was added to the form using JavaScript and it was styled with CSS. Results from the APP demonstrated that the models' predictions are as expected(1 for fraud data and 0 for non fraud data). While there are limitations to be considered, the models' findings provided valuable insights for financial institutions. Also, the study provides avenues for future researchers.*

## **1. Introduction.**

According to Raj and Portia (2011), a credit card is a small plastic (or metal) card issued by financial institutions to users, allowing them to borrow funds to purchase goods and services. The cardholder agrees to repay the borrowed amount, usually with interest, according to the terms of the issuer. Credit card security depends on both the physical protection of the card and the privacy of sensitive information like the credit card number, expiration date, and security code. The use of credit cards has become increasingly common, even in developing countries. People use them for shopping, paying bills, and conducting online transactions. However, with the rise in the number of credit card users, instances of credit card fraud have also increased. Globally, credit card fraud causes losses amounting to billions of dollars. (Jain et al.2019). From Delamaire et al. (2009) perspective, credit card fraud refers to any deceptive activity aimed at obtaining financial gain without the knowledge of the cardholder or the issuing bank. Fraud can occur in various ways, including through lost or stolen cards, the production of fake or counterfeit cards, cloning legitimate websites, erasing or modifying the card's magnetic strip (which contains the cardholder's information), phishing, skimming, or stealing data from merchants.

### **1.1. Types of credit card fraud.**

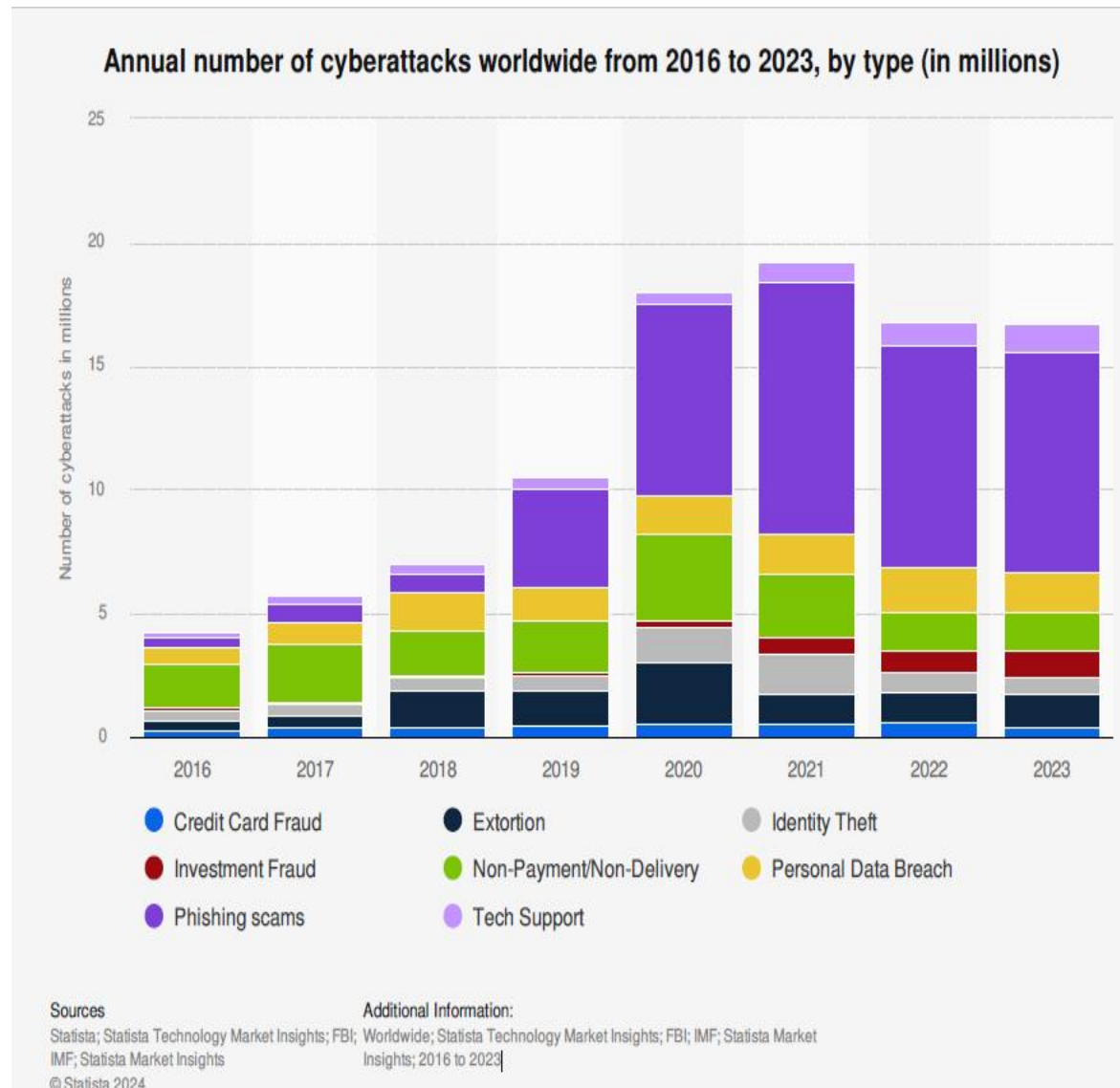
Credit card fraud includes, but is not limited to, the following:

- **Application Frauds:** Application fraud occurs when a fraudster uses stolen personal information, such as a victim's name, address, and Social Security number, to apply for credit or a new credit card in the victim's name.
- **Electronic or Manual Credit Card Imprints:** Fraudsters may skim or copy the information from the magnetic strip of the card, which contains confidential data. This allows them to replicate the card or use the data for future fraudulent transactions.
- **CNP (Card Not Present):** Card-Not-Present fraud occurs when a fraudster knows the card's account number and expiry date, enabling them to make online or phone transactions without physically possessing the card.
- **Counterfeit Card Fraud:** Counterfeit card fraud involves creating a fake card that contains the original cardholder's details, often through a process called skimming, where the fraudster secretly copies data from the card's magnetic strip.
- **Lost and Stolen Card Fraud:** When a credit card is lost or stolen, fraudsters may attempt to use it for purchases, especially in online transactions that do not require a PIN.
- **Card ID Theft:** In card ID theft, a fraudster steals personal details to either use an existing card or open a new account in the victim's name. This form of fraud is particularly difficult to detect.
- **Mail Non-Receipt Card Fraud:** Mail non-receipt fraud occurs when a credit card is stolen during the mailing process, before the legitimate cardholder receives it. Fraudsters may activate the card and use it for purchases.

- Account Takeover: In account takeover fraud, a criminal gains access to the victim's account information and contacts the credit card company, pretending to be the cardholder. The fraudster may request changes to the account, such as updating the address, so they can receive a new card and use it for purchases.
- False Merchant Sites: Fraudsters create fake merchant websites that closely mimic legitimate ones, often offering unrealistic discounts. When customers make purchases, their card details are captured for fraudulent transactions.
- Merchant Collusion: Merchant collusion occurs when a business deliberately passes cardholder information to fraudsters, allowing them to use the details for unauthorized transactions.
- Phishing: Phishing involves tricking a victim into voluntarily providing their sensitive information, such as credit card details, passwords, or account numbers. This is typically done through fake emails, messages, or websites that appear to be from legitimate sources (like a bank or online store).

## 1.2. Global cyberattacks report.

According to Statista (2024), nearly nine million cases of phishing scams were registered worldwide in 2023, making it the most frequently reported type of cybercrime. Personal data breach ranked second, with 1.66 million incidents, followed by 1.5 million non-payment/non-delivery cases as shown in figure below.





### **1.3. Problem statement.**

Global e-commerce losses due to online payment fraud were estimated at 41 billion U.S. dollars in 2022, marking an increase from the previous year. This figure is projected to grow further, reaching 48 billion U.S. dollars by 2023 (Statista, 2022). As fraudulent strategies continue to evolve, there is a critical need to develop effective machine learning models capable of predicting credit card fraud before it occurs. Such models could significantly reduce both the global fraud rate and the associated financial losses. This study aims to address this issue.

## 2. Literature review.

The scientific community is currently involved in developing methods and tools to identify and predict different credit card fraud, which have significant implications for people's welfare. Several researchers have utilized machine learning methods to forecast credit card fraud, as exemplified by:

Kumar et al. (2020) in their research to predict credit card fraud developed machine learning models using three different traditional machine learning algorithms: Logistic Regression, Naive Bayes and Decision Tree. In addition, they applied a deep learning algorithm: Artificial Neural Network (ANN). According to their results, ANN achieved the best result with accuracy value of 98.69%, followed by logistic regression with accuracy value of 94.84% while Naive Bayes and Decision Tree achieved 91.62% and 92.88% respectively.

Also, Khan et al. (2022) in their analysis to detect credit card fraud, built machine learning models using two traditional machine learning algorithms: Support Vector Classifier, logistic regression and a deep learning algorithm: Artificial Neural Network (ANN). The models' performance was analyzed using Receiver Operating Characteristic (ROC) curve and Area under the curve. From the result they obtained, Logistic Regression achieved a better result with a threshold point value of (0.38,1) and Area under the curve value of 97%. This was followed by support vector classifier with a threshold point value of (0.22,1) and Area under the curve value of 94% while ANN achieved the least result with a threshold point value of (0.24,1) and Area under the curve value of 90%.

Another analysis by Afriyie et al. (2023) developed machine learning models that can predict credit card fraud using three different machine learning algorithms: Decision Tree, Random Forest and Logistic Regression. The models were evaluated using confusion matrix; Accuracy and Receivers Operating Characteristic (ROC) curve and the Area under the curve. According to their results, Random Forest yielded the best result with an accuracy value of 0.96 and area under the curve value of 98.9%, followed by Decision Tree with an accuracy value of 0.92 and area under the curve of 94.5% while Logistic Regression achieved the least result with an accuracy value of 0.92 and area under the curve value of 87.9%.

Furthermore, Awoyemi et al. (2017) in the analysis to forecast credit card fraud, built machine learning models using different machine learning algorithms; Naïve Bayes, K-nearest Neighbor and Logistic Regression. The models were evaluated using accuracy and from their performance results, Naïve Bayes yielded a better result with an accuracy value of 97.92%, followed by K-nearest Neighbor with an accuracy value of 97.69% whereas Logistic Regression achieved the least result with an accuracy value of 54.86%.

### 3. Objective of the study.

The main objective of this project is to develop machine learning models that can detect credit card fraud using algorithms: decision tree, logistic regression and artificial neural network(ANN).

#### 3.1. Specific objective.

Specific objective of this project is to use the built models to develop a Flask APP.

- The Flak APP would be deployed locally as an Application Programming Interface(API) and test with postman.
- Also, the APP would be deployed locally on the web where a user interactive form would be created.

### 4. Methodology.

#### 4.1. Dataset and dataset collection.

The dataset was sourced from Kaggle through the link provided by the Think Pacific Foundation. The dataset contains two days transactions made by credit cards in September 2013 by European cardholders.

It contains 284,807 rows and 31 columns. Apart from the Time, Amount and Class variables, other variables have under gone principal component analysis (PCA) and the principal components have been encoded into (V<sub>1</sub>.....V<sub>28</sub>) for a confidential reason.

#### 4.2. Dataset attribute.

S/N	Variables	Data Type	Meaning
1.	Time	Numeric	Seconds elapsed between each transaction and the first transaction
2.	Amount	Numeric	Transaction Amount.
3.	Class	Numeric	Fraud/No Fraud
4.	V <sub>1</sub> ...V <sub>28</sub>	Numeric	Encoded for confidential reasons

The dataset is a supervised dataset because it has both dependent and independent variables; in other words, it a labeled dataset. Based on this, it is a classification tax. As a result, classification machine learning algorithms were used to develop the models.

Python programming language was used to analyze the dataset and developed the models. Jupyter notebook environment was used for column-by-column analysis, Spyder environment was used for encapsulation logic within functions coding and models development while pycharm environment was used for Flask APP development and deployment.

## 5. Algorithms.

The following algorithms were used for this project as recommended by the Think Pacific Foundation:

### 5.1. Decision Tree.

A decision tree is a computational tool used for classification and prediction tasks. It consists of internal nodes that represent tests on attributes, branches that denote the outcomes of these tests, and leaf nodes (or terminal nodes) that hold class labels. The algorithm recursively partitions a dataset using either a depth-first or breadth-first greedy approach, stopping when each subset consists predominantly of a single class.

To ensure an efficient partition, the algorithm uses a criterion to measure the "purity" of the subsets created at each split. Commonly used splitting criteria include Information Gain and Gini Impurity.

Information Gain (used in ID3, C4.5 algorithms): It measures the reduction in entropy (uncertainty) after a dataset split based on attribute A. Information Gain is given by:

$$\text{Information Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$$

Where:

S = original dataset.

$S_v$  = subset of S for which attribute A has value v

Entropy(S) is calculated as:

$$\text{Entropy}(S) = - \sum_{i=1}^c p_i \log_2(p_i)$$

Where:

$p_i$  is the proportion of examples in class i, and c is the number of classes.

**Gini Impurity** (used in CART algorithm): It measures the impurity of a subset, with lower values indicating greater purity. The Gini Impurity for a subset SSS is calculated as:

$$\text{Gini}(S) = 1 - \sum_{i=1}^c p_i^2$$

Where:

$p_i$  is the proportion of instances belonging to class  $i$  in subset  $S$ , and  $c$  is the number of classes.

The goal is to find a split that minimizes impurity, ensuring that data is divided into groups where one class predominates in each group. This allows the decision tree to create well-defined, disjoint subsets. (Gaikwad, et al,2015)

## 5.2. Logistic Regression.

Logistic regression addresses the limitations of linear regression, particularly when predictions fall outside the  $[0,1]$  range. Despite its name, logistic regression is primarily used for classification problems, predicting binomial (two-category) and multinomial (multiple-category) outcomes. Its goal is to estimate the values of model parameters (coefficients) using the sigmoid function, which maps predictions to a probability range between 0 and 1.

The logistic regression model uses the following sigmoid function to model the probability  $P(y = 1 | x)$  of a binary outcome  $y$  given a set of predictors  $x$ :

$$P(y = 1 | x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n)}}$$

Where:

$P(y = 1 | x)$  = probability of the positive class (e.g., a fraudulent transaction)

$\beta_0$  = intercept term.

$\beta_1, \beta_2, \dots, \beta_n$  = coefficients for each predictor  $x_1, x_2, \dots, x_n$

$-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n)$  = linear combination of the input variables, mapped to a probability between 0 and 1 by the sigmoid function.

Logistic regression is commonly used in applications like fraud detection, where it examines the attributes of a transaction and outputs the probability that it should proceed or be flagged. ( Sahin, et al, 2011).

## 5.3. Artificial Neural Networks (ANN).

An artificial neural network (ANN) is a computational model inspired by the structure of the human brain. It consists of *neurons* (nodes) that act as processing units and *edges* (connections) between neurons, each with an associated weight that represents the strength of the connection. These weights determine the influence of each neuron in the previous layer on the neuron in the current layer, ultimately contributing to the network's decisions and outputs.

### Mathematically:

In an ANN, the output of each neuron is typically calculated using a weighted sum of its inputs followed by an activation function to introduce non-linearity. For a neuron  $j$  in a hidden layer, the output  $a_j$  is computed as:

$$z_j = \sum_{i=1}^n w_{ij}x_i + b_j$$
$$a_j = f(z_j)$$

Where:

$x_i$  represents the input from neuron  $i$  in the previous layer.

$w_{ij}$  is the weight of the connection between neuron  $i$  and neuron  $j$ .

$b_j$  is the bias term for neuron  $j$ .

$f(z_j)$  is the activation function applied to the weighted sum  $z_j$ , commonly a sigmoid, ReLU, or tanh function, which introduces non-linearity to the model.

### Activation Function Example

A popular activation function is the sigmoid function, which maps the output to a range between 0 and 1:

$$f(z) = \frac{1}{1 + e^{-z}}$$

This function is often used in binary classification tasks, allowing the output to be interpreted as a probability.

### Training an ANN

ANNs are trained by adjusting weights and biases to minimize the difference between the predicted and actual outcomes. In **supervised training**, this involves calculating the error (loss) using a function such as mean squared error for regression tasks or cross-entropy loss for classification:

$$\text{Loss} = - \sum_i (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$$

Where:

$y_i$  = actual label.

$\hat{Y}_1$  = predicted output.

The ANN adjusts weights and biases using backpropagation, where gradients of the loss with respect to each weight are calculated and used to update the weights through an optimization algorithm like gradient descent. ( Murli, et al. 2014).

## 6. Isolated virtual environment creation.

To ensure reproducibility and prevent dependency conflicts that may arise from using global environments, this project was developed within an isolated conda environment(action\_project\_env) using python 3.10.13. By isolating the environment, I aimed at avoiding version mismatches and interference from globally installed packages, thus creating a controlled setting for dependency management and ensuring that all required libraries are specific to this project.

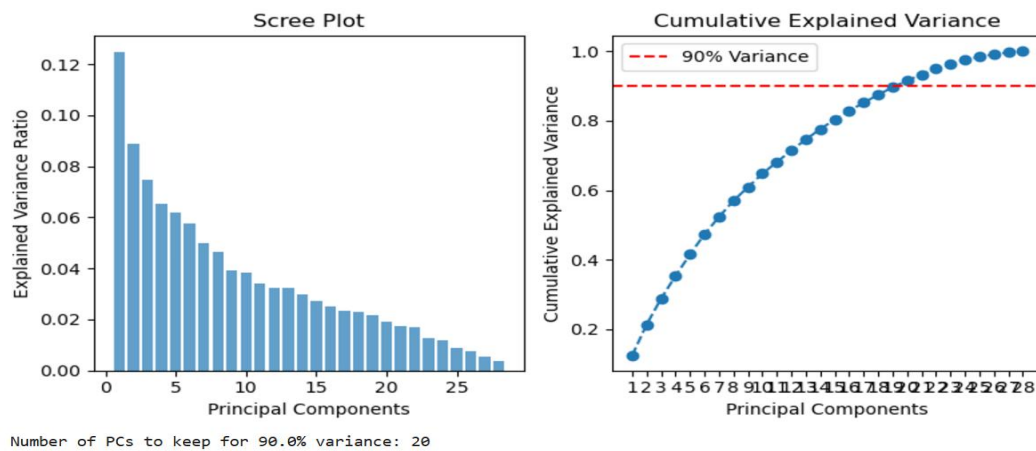
## 7. Libraries and dataset importation.

The required libraries and the dataset were imported into jupyter notebook.

## 8. Dataset preparation.

Because some of the features (v1...v28) of the dataset have under gone principal component analysis(PCA), there is a room for dimensionality reduction. As a result, scree plot and cumulative explained variance were applied to find out number of v1...v28 that would provide 90% information of the original dataset. From my analysis, it was shown that 20 of v1...v28 would provide this information. Because principal components are arranged in descending order according to the amount of information they can provide, I kept v1...v20 and dropped v21...v28. This reduced the dataset's number of columns from 31 to 23. Figure 1 below shows the scree plot and cumulative explained variance.

Fig.1



## **9. Data cleaning .**

This is a process in data analysis that involve analyzing a dataset to detect corrupt data such as wrong data type, missing value, completely empty row, presence of outliers, duplicated rows and handle them accordingly. So, in this project, the dataset was examined for wrong data type, missing value, completely empty row and duplicated rows. Based on the analysis result, there was no wrong data type, missing value, completely empty rows, but there were 1081 duplicated rows. In order to prevent redundant data, these rows were dropped and this reduced the number of the rows from 284807 to 283726.

## **10. Column-by-column Analysis.**

Column -by-column analysis is a process in data processing that involve analyzing each column of a dataset one after another in order to get insight about the data and make informed decision. In this project, the following analytics methods were applied on each column:

### **10.1. Descriptive statistics analysis.**

This was applied to examine its statistical values like mean, median, standard deviation, maximum value, minimum value, 25th and 75th percentile of the column. This is important in analysis because minimum and maximum value of the data determine the type of transformation method that should be applied on the data. Also, mean and standard deviation values determine whether the data follows standard normal distribution that is assumed by many machine learning algorithms or not. A mean value of 0 and standard deviation value of 1 means that the data follows standard normal distribution.

### **10.2. Skewness analysis.**

This was applied to determine each column's asymmetry. This is very important in data analysis because its value does not only show how it is distributed, but also determines the choice of normalization. Skewness value of 0 indicates normal distribution, skewness value greater than 0 indicates positively skewed while skewness value less than 0 indicates negatively skewed.

### **10.3. Outlier analysis.**

Outliers are extreme low and high values in data. The presence of outliers or improperly handled outliers can lead to computational error during analysis. This error can affect the ability of a machine learning algorithm to learn. It can lead to over-fitting; a condition where by an algorithm captured both the error and the pattern in the data. Under this condition, the model performs well during training, but performs poorly during testing. It can also lead to under-fitting; a condition where by an algorithm did not capture any information about the data. Under this condition, the model will not perform well during training as well as during testing. As a result, interquartile range method was applied to each column to determine the number and percentage of outliers it has.

### **10.4. Transformation.**

This is the application of mathematical function on data to reduce its skewedness as heavily skewed data can lead to biased prediction. In this project, each column was transformed based on its analytics outcome.



### 10.5. Normalization.

This is the application of mathematical function on data to either make it have a comparable range (min. value of 0 and max. value of 1) since many machine learning algorithms are sensitive to data range or make it follow a standard normal distribution (mean value of 0 and standard deviation of 1) since many machine learning algorithms assume standard normal distribution data. In this project, each column was normalized based on its analytics outcome.

### 10.6. Visualization.

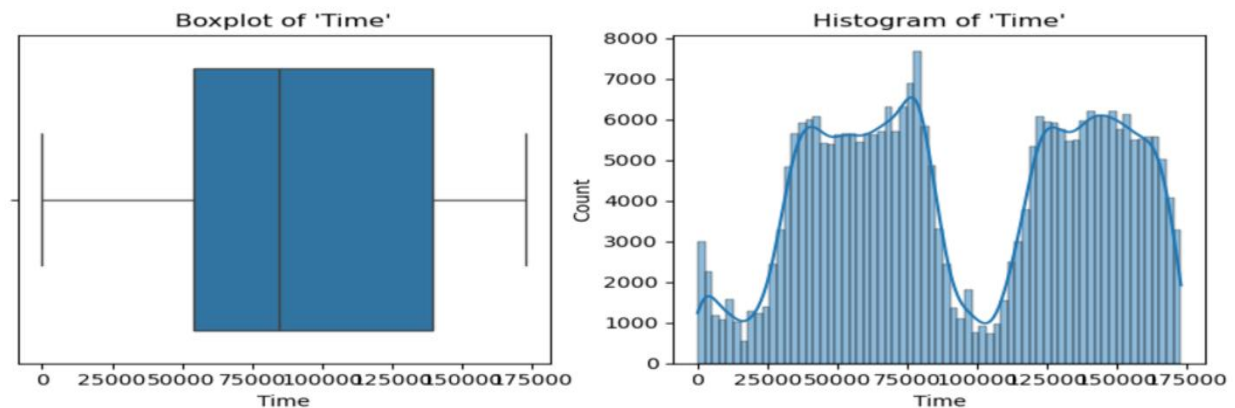
To have a better view of the data, each column was visualized using a box plot and histogram.

### 11. Time column analysis.

From the analysis result, it was shown that time column has:

- Zero (0) number of outliers.
- Skewness value of -0.0356. This value is very close to 0. As a result, it was normalized using standardization method. figure 2 below shows visual representation of the time column.

Fig.2



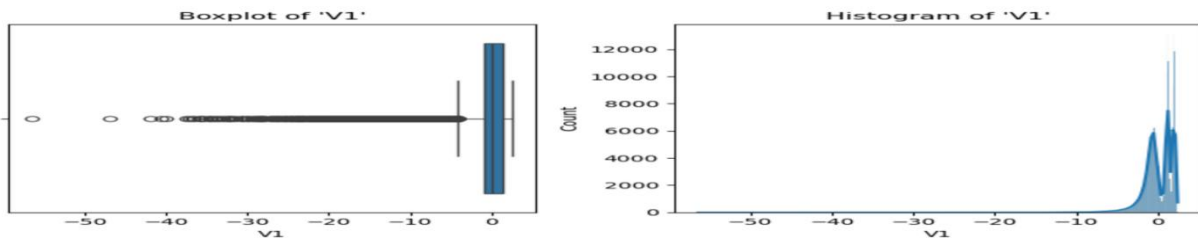
## 12. V1 column analysis.

From the analysis result, it was shown that V1 column has:

- 6948 number of outliers
- Skewness value of -3.27
- Min value of -56.41 and max value of 2.45

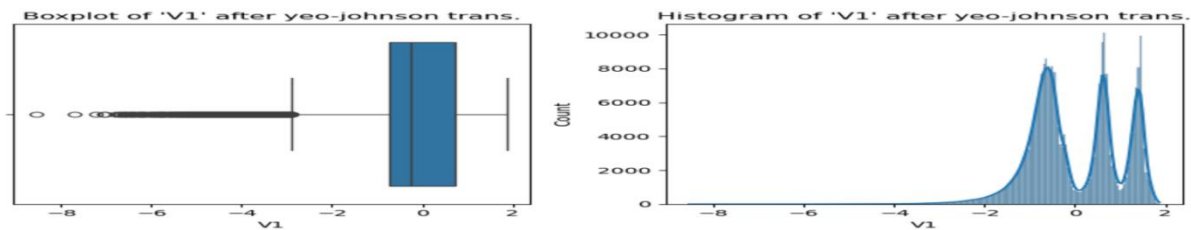
Figure 3 bellow shows its visual representation.

Fig.3



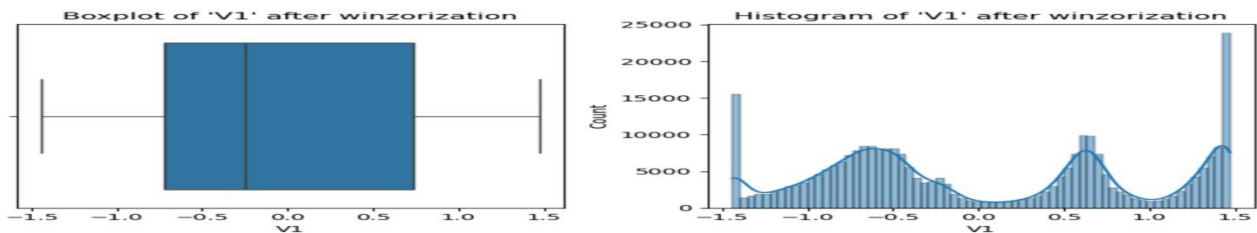
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 4 bellow shows V1 column after transformation.

Fig.4



In other to handle the remaining outliers in V1 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 5 bellow shows V1 after winsorization.

Fig.5



After winsorization, V1 skewness value became 0.17. This value is close to zero and because of this, V1 was normalized using standardization method.

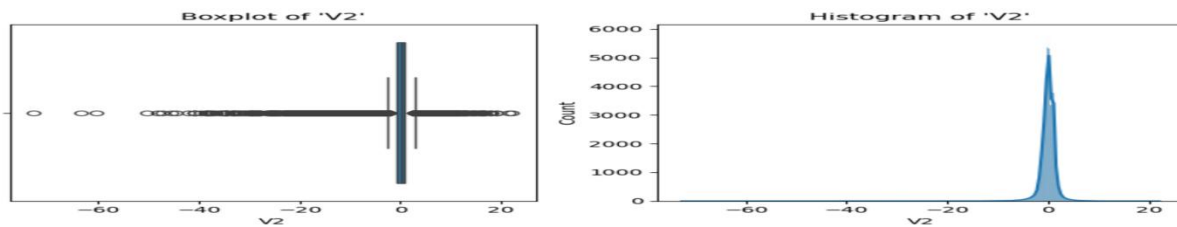
### 13. V2 column analysis.

From the analysis result, it was shown that V2 column has:

- 13390 number of outliers
- Skewness value of -4.70
- Min value of -72.73 and max value of 22.06

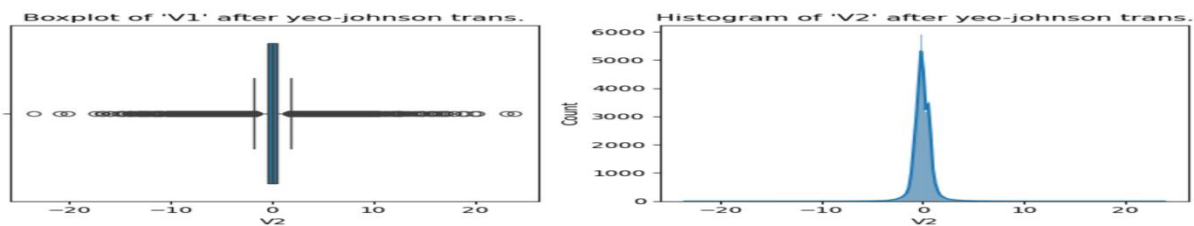
Figure 6 bellow shows its visual representation.

Fig.6



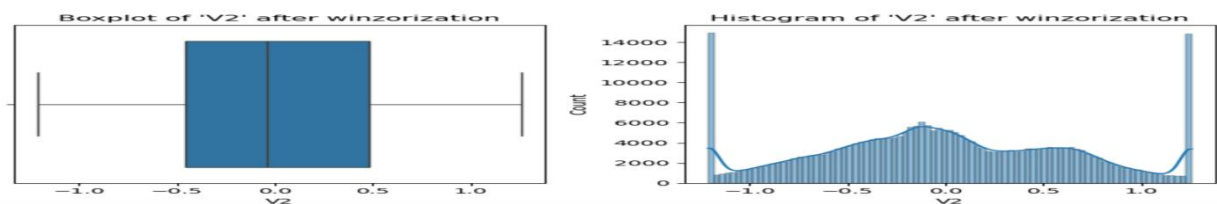
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 7 bellow shows V2 column after transformation.

Fig.7



In other to handle the remaining outliers in V2 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 8 bellow shows V2 after winsorization.

Fig.8



After winsorization, V2 skewness value became 0.08. This value is close to zero and because of this, V2 was normalized using standardization method.

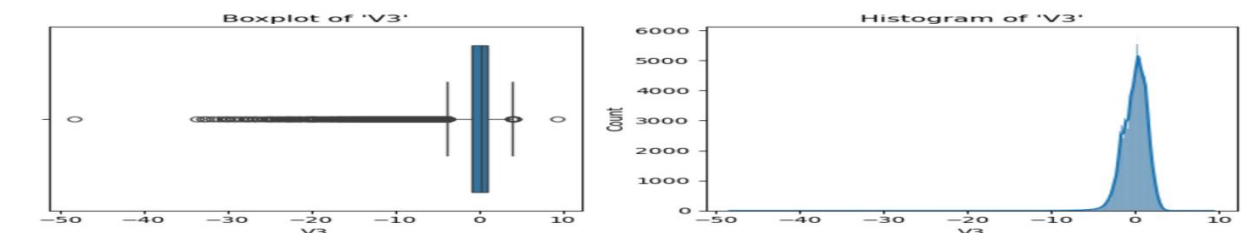
#### 14. V3 column analysis.

From the analysis result, it was shown that V2 column has:

- 3306 number of outliers
- Skewness value of -2.15
- Min value of -48.33 and max value of 9.38

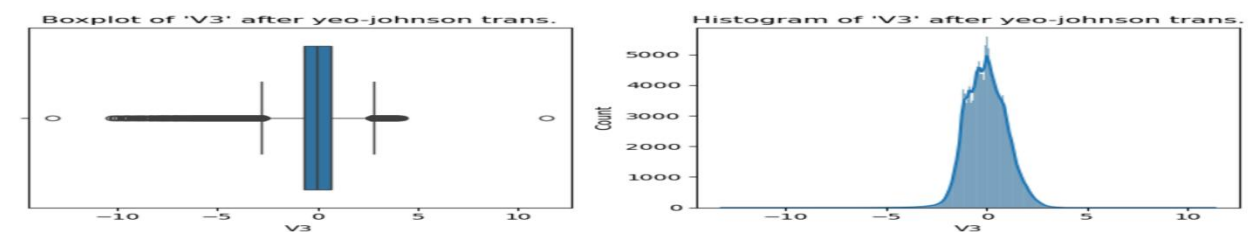
Figure 9 bellow shows its visual representation.

Fig.9



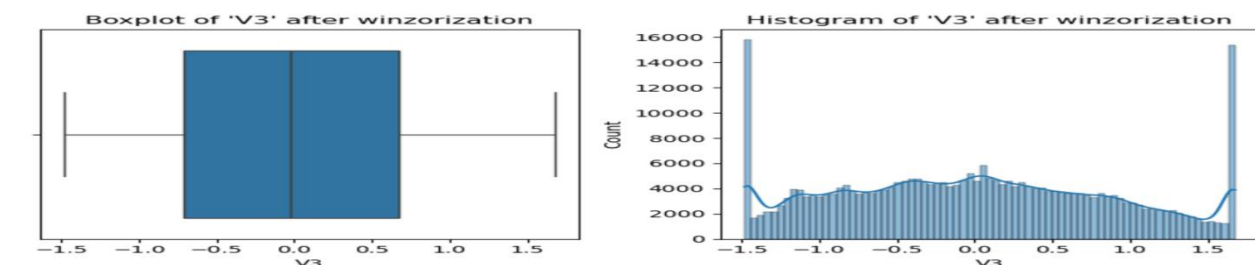
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 10 bellow shows V3 column after transformation.

Fig.10



In other to handle the remaining outliers in V3 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 11 bellow shows V3 after winsorization.

Fig.11



After winsorization, V3 skewness value became 0.15. This value is close to zero and because of this, it was normalized using standardization method.

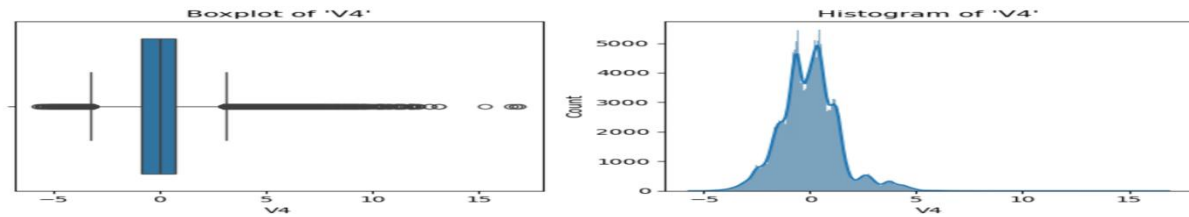
## 15. V4 column analysis.

From the analysis result, it was shown that V4 column has:

- 11094 number of outliers
- Skewness value of 0.67
- Min value of -5.68 and max value of 16.88

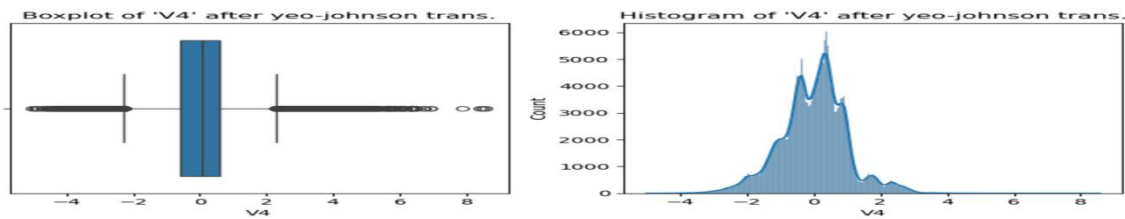
Figure 12 bellow shows its visual representation.

Fig.12



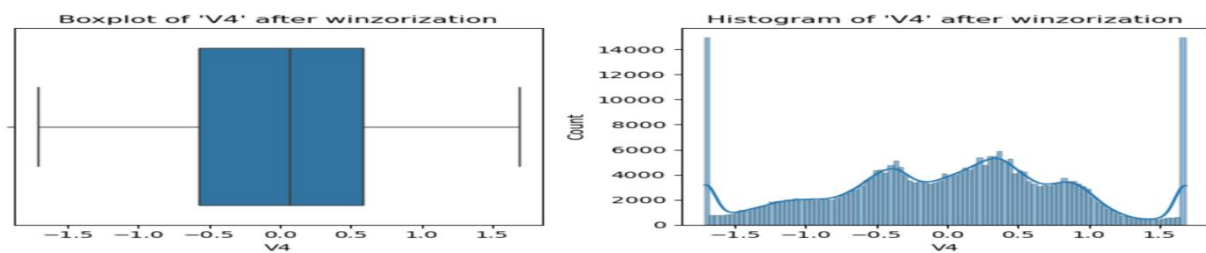
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 13 bellow shows V4 column after transformation.

Fig.13



In other to handle the remaining outliers in V4 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 14 bellow shows V4 after winsorization.

Fig.14



After winsorization, V4 skewness value became -0.09. This value is close to zero and because of this, it was normalized using standardization method.

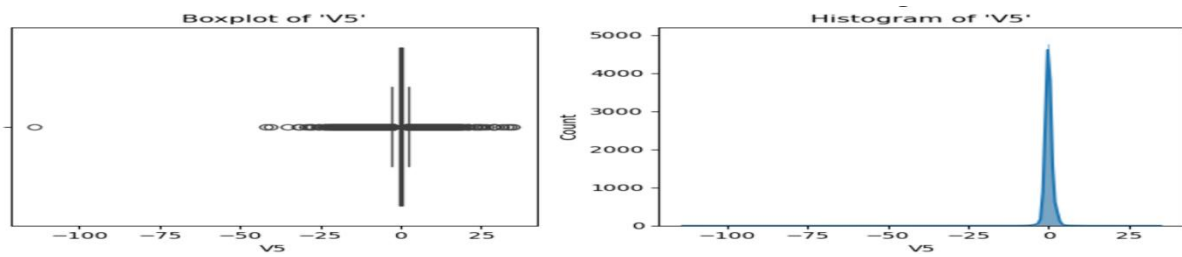
## 16. V5 column analysis.

From the analysis result, it was shown that V4 column has:

- 12221 number of outliers
- Skewness value of -2.41
- Min value of -113.74 and max value of 34.80

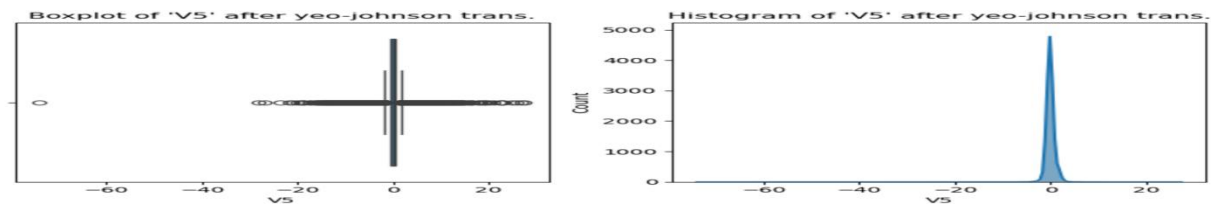
Figure 15 bellow shows its visual representation.

Fig.15



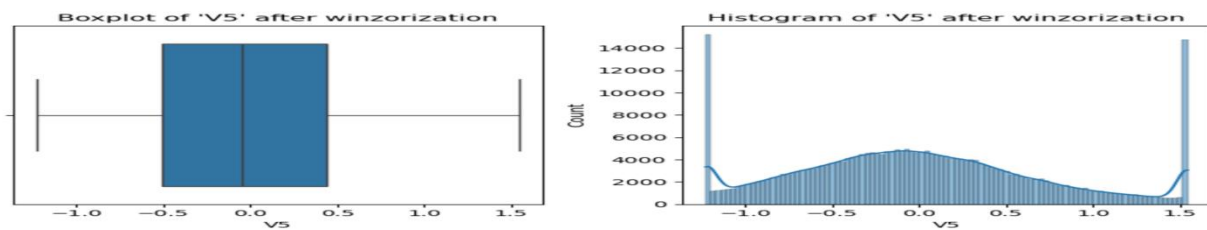
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 16 bellow shows V5 column after transformation.

Fig.16



In other to handle the remaining outliers in V5 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 17 bellow shows V5 after winsorization.

Fi.17



After winsorization, V5 skewness value became 0.34. This value is close to zero and because of this, it was normalized using standardization method.

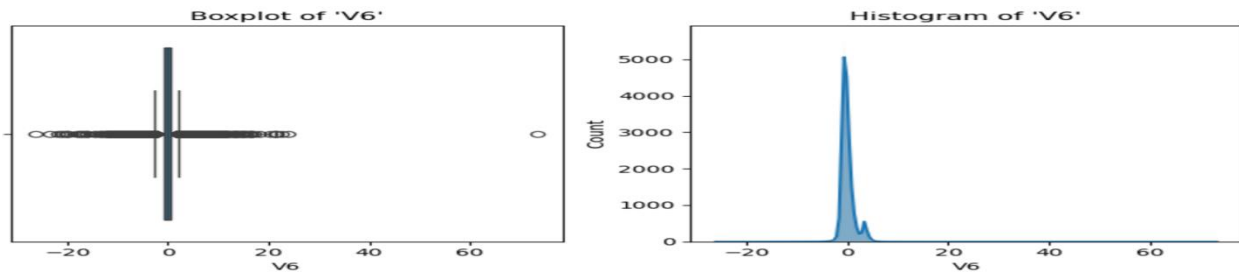
## 17. V6 column analysis.

From the analysis result, it was shown that V6 column has:

- 22886 number of outliers
- Skewness value of 1.83
- Min value of -26.16 and max value of 73.30

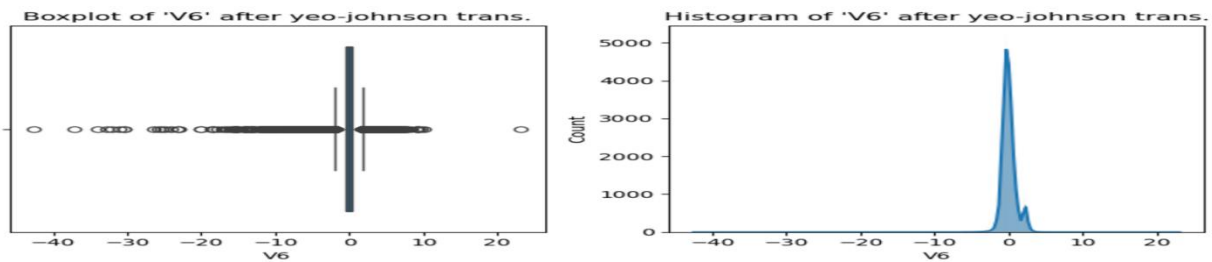
Figure 18 bellow shows its visual representation.

Fig.18



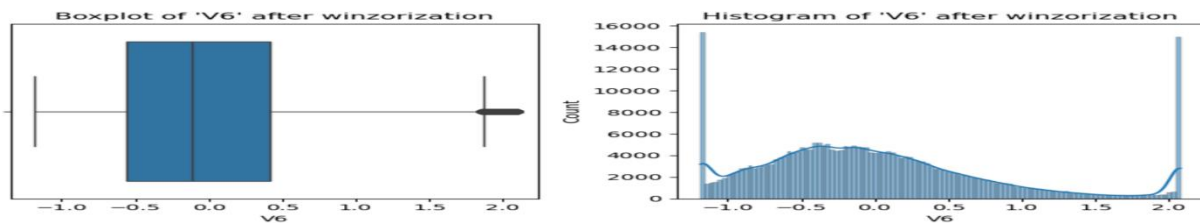
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 18 bellow shows V6 column after transformation.

Fig.18



In other to handle the remaining outliers in V6 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 19 bellow shows V6 after winsorization.

Fig.19



After winsorization, the number of outliers in V6 reduced from 22886 to 17624. Because of this, V6 was normalized using robust scaling to set its interquartile range value to 1 and its median value to 0. This method suppresses the impact of the remaining outliers on the data.

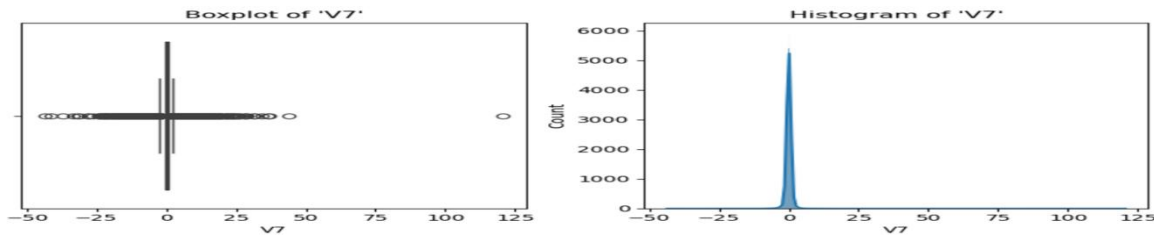
## 18. V7 column analysis.

From the analysis result, it was shown that V7 column has:

- 8839 number of outliers
- Skewness value of 2.89
- Min value of -43.56 and max value of 120.59

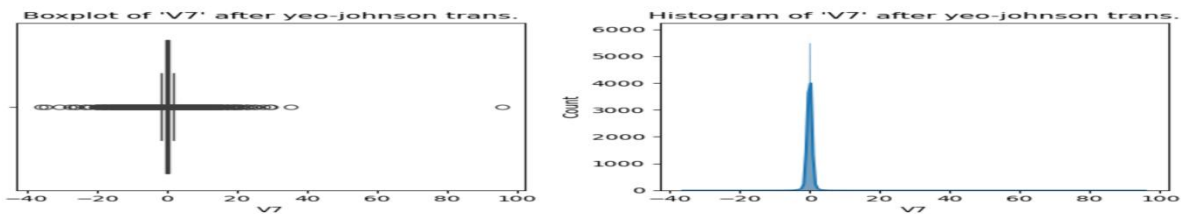
Figure 20 bellow shows its visual representation.

Fig.20



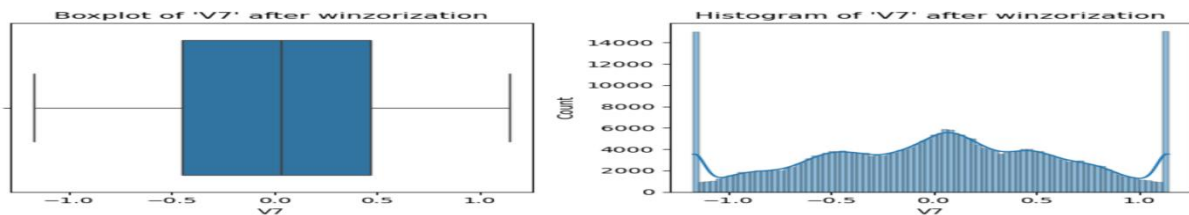
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 21 bellow shows V7 column after transformation.

Fig.21



In other to handle the remaining outliers in V7 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 22 bellow shows V7 after winsorization.

Fig.22



After winsorization, V7 skewness value became -0.07. This value is close to zero and because of this, it was normalized using standardization method.



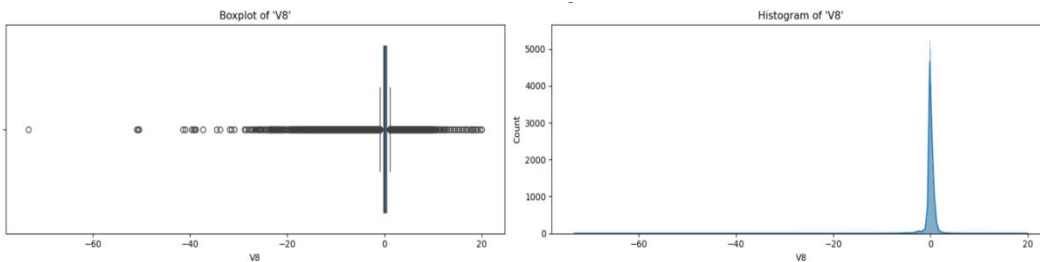
## 19. V8 column analysis.

From the analysis result, it was shown that V8 column has:

- 23904 number of outliers
- Skewness value of -8.31
- Min value of -73.22 and max value of 20.01

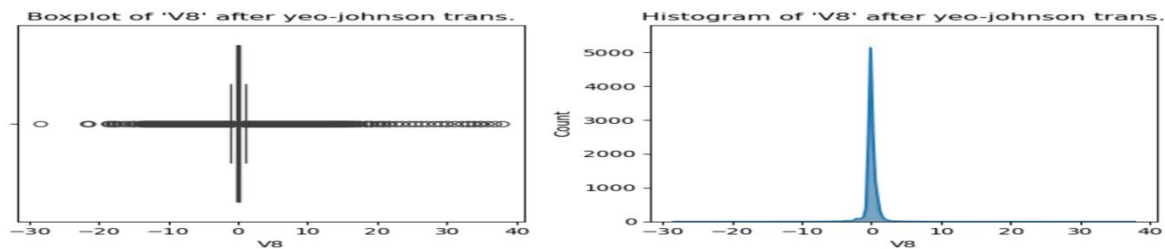
Figure 23 bellow shows its visual representation.

Fig.23



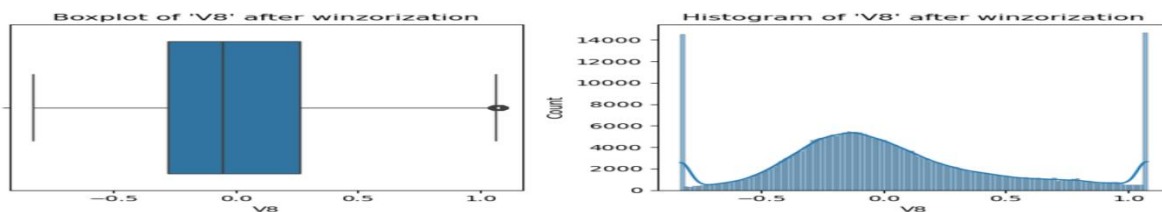
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 24 bellow shows V8 column after transformation.

Fig.24



In other to handle the remaining outliers in V8 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 25 bellow shows V8 after winsorization.

Fig.25



After winsorization, the number of outliers in V6 reduced from 23904 to 14685. Because of this, V6 was normalized using robust scaling to set its interquartile range value to 1 and its median value to 0. This method suppresses the impact of the remaining outliers on the data.

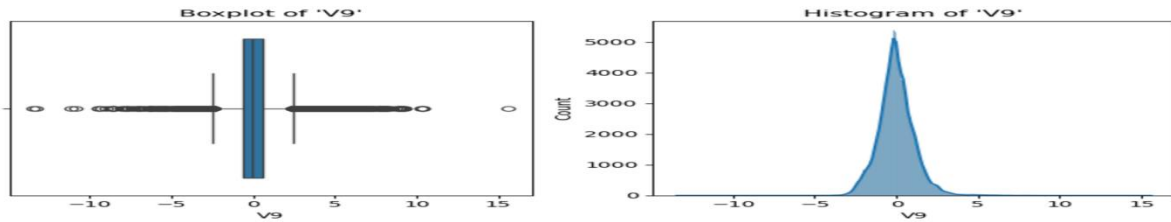
## 20. V9 column analysis.

From the analysis result, it was shown that V9 column has:

- 8199 number of outliers
- Skewness value of 0.54
- Min value of -13.43 and max value of 15.59

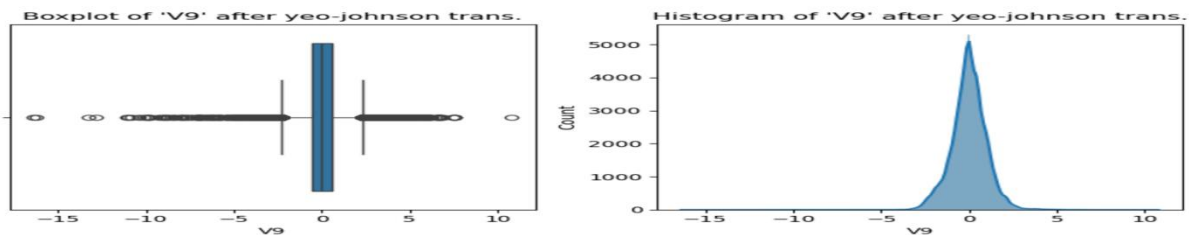
Figure 26 bellow shows its visual representation.

Fig.26



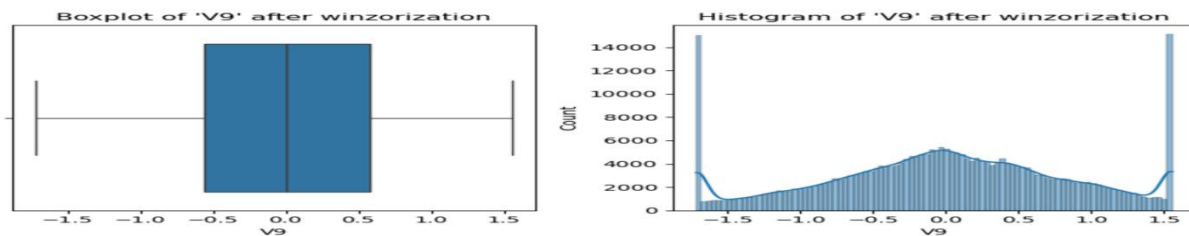
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 26 bellow shows V9 column after transformation.

Fig.26



In other to handle the remaining outliers in V9 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 27 bellow shows V9 after winsorization.

Fig.27



After winsorization, V9 skewness value became -0.13. This value is close to zero and because of this, it was normalized using standardization method.

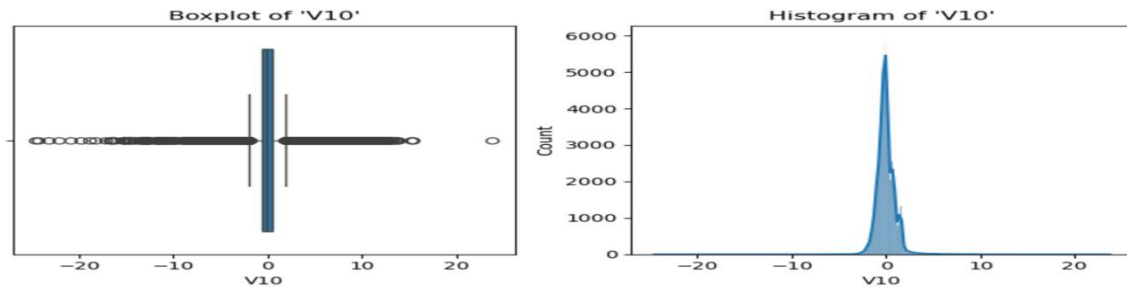
## 21. V10 column analysis.

From the analysis result, it was shown that V10 column has:

- 8199 number of outliers
- Skewness value of 0.54
- Min value of -13.43 and max value of 15.59

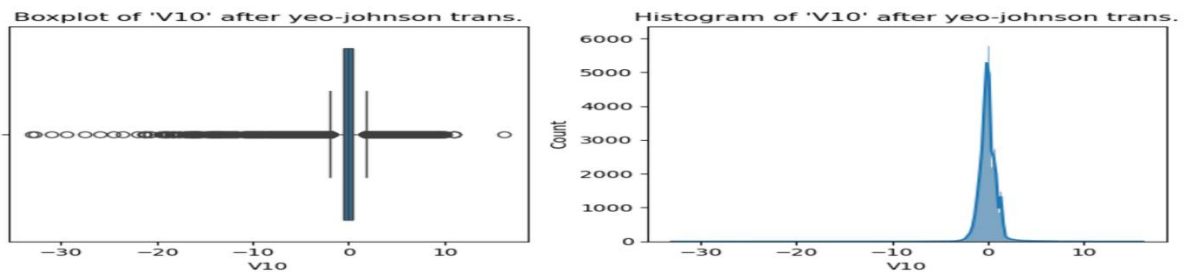
Figure 28 bellow shows its visual representation.

Fig.28



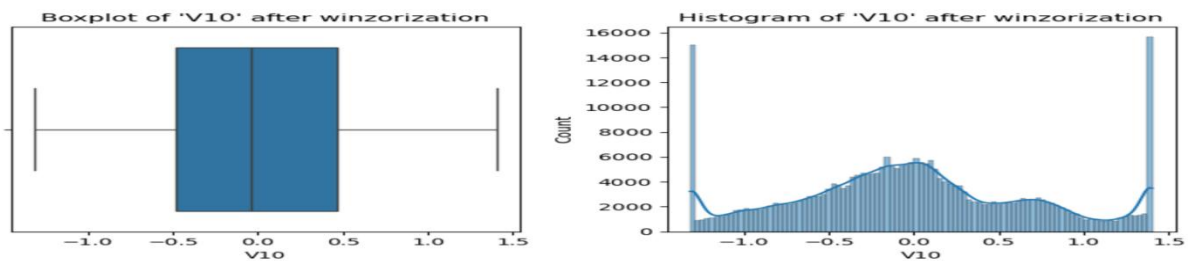
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 29 bellow shows V9 column after transformation.

Fig.29



In other to handle the remaining outliers in V10 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 30 bellow shows V10 after winsorization.

Fig.30



After winsorization, V10 skewness value became 0.17. This value is close to zero and because of this, it was normalized using standardization method.

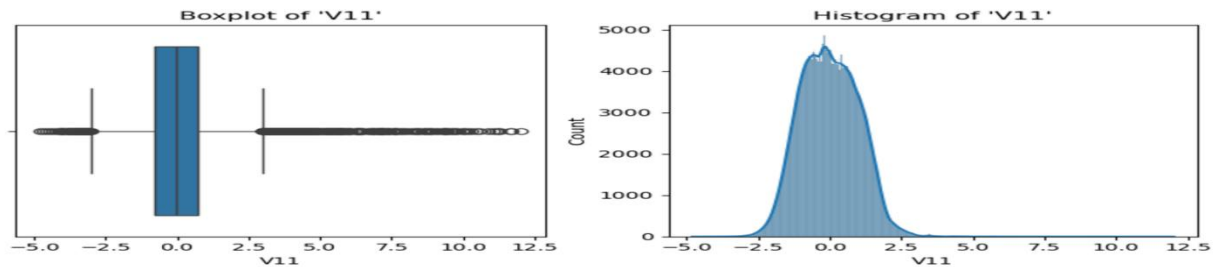
## 22. V11 column analysis.

From the analysis result, it was shown that V11 column has:

- 735 number of outliers
- Skewness value of 0.34
- Min value of -4.79 and max value of 12.01

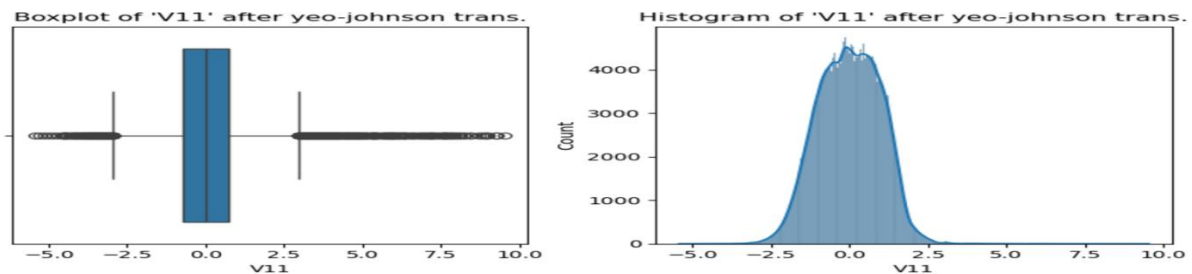
Figure 31 bellow shows its visual representation.

Fig.31



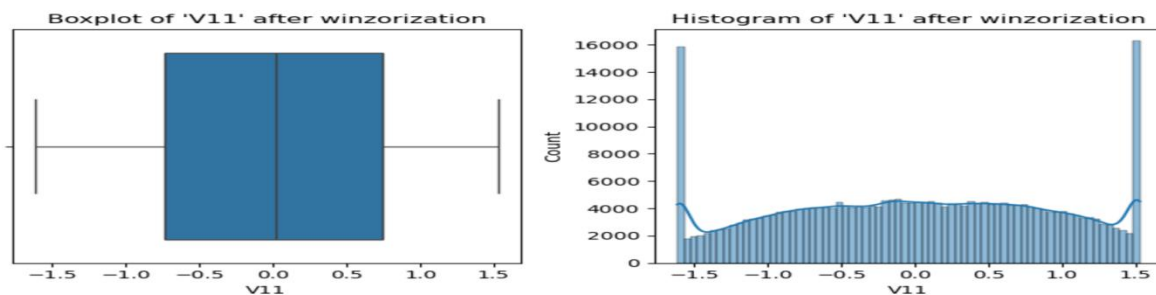
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 32 bellow shows V11 column after transformation.

Fig.32



In other to handle the remaining outliers in V11 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 33 bellow shows V11 after winsorization.

Fig.33



After winsorization, V11 skewness value became -0.06. This value is close to zero and because of this, it was normalized using standardization method.

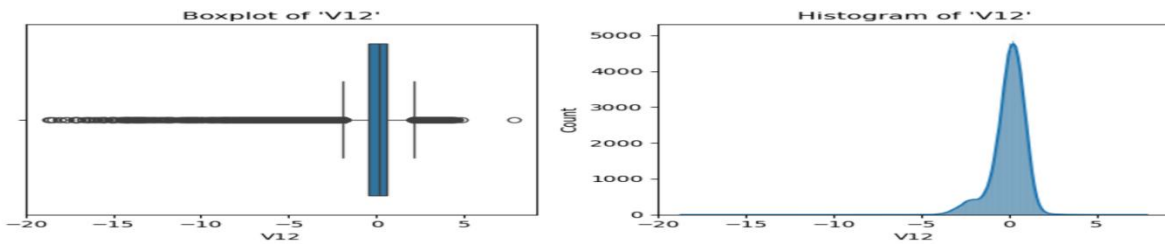
### 23. V12 column analysis.

From the analysis result, it was shown that V12 column has:

- 15282 number of outliers
- Skewness value of -2.19
- Min value of -18.68 and max value of 7.85

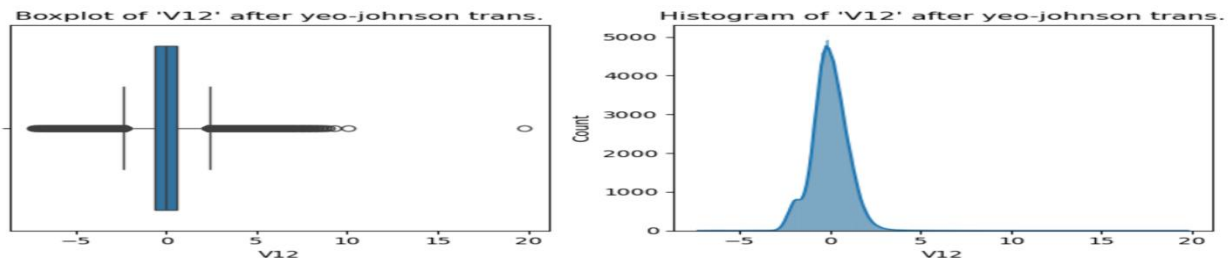
Figure 34 bellow shows its visual representation.

Fig.34



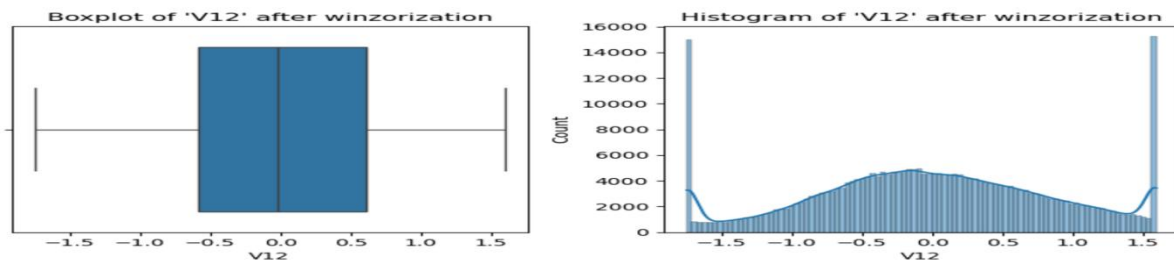
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 34 bellow shows V12 column after transformation.

Fig.34



In other to handle the remaining outliers in V12 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 35 bellow shows V11 after winsorization.

Fig.35



After winsorization, V12 skewness value became -0.07. This value is close to zero and because of this, it was normalized using standardization method.

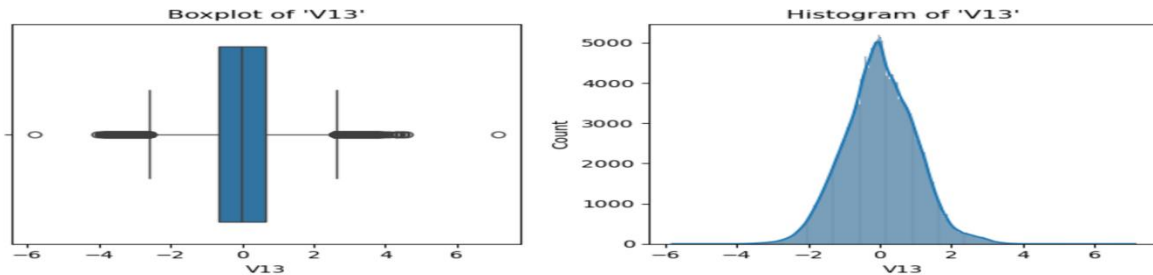
## 24. V13 column analysis.

From the analysis result, it was shown that V13 column has:

- 3362 number of outliers
- Skewness value of 0.06
- Min value of -5.79 and max value of 7.13

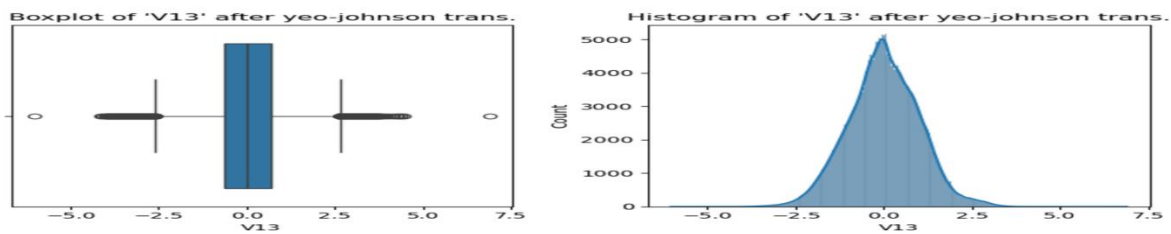
Figure 36 bellow shows its visual representation.

Fig.36



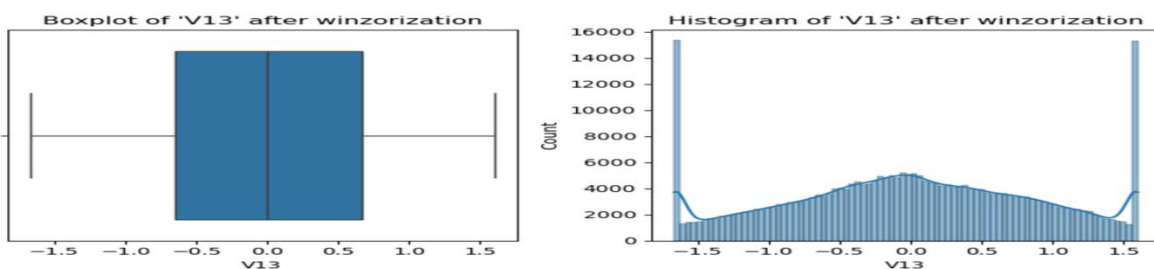
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 36 bellow shows V13 column after transformation.

Fig.36



In other to handle the remaining outliers in V13 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 37 bellow shows V13 after winsorization.

Fig.37



After winsorization, V13 skewness value became -0.05. This value is close to zero and because of this, it was normalized using standardization method.

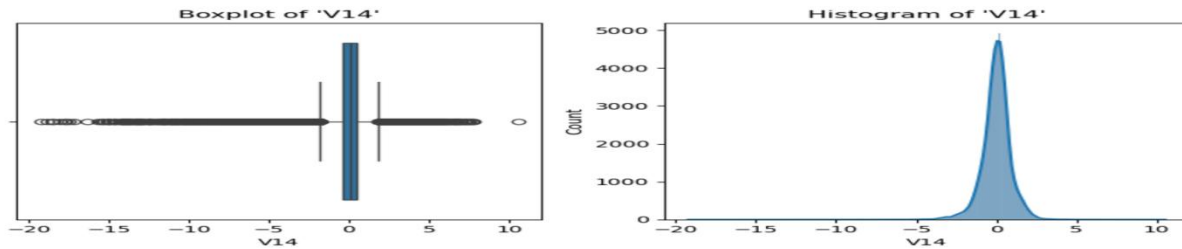
## 25. V14 column analysis.

From the analysis result, it was shown that V14 column has:

- 14060 number of outliers
- Skewness value of -1.92
- Min value of -19.21 and max value of 10.53

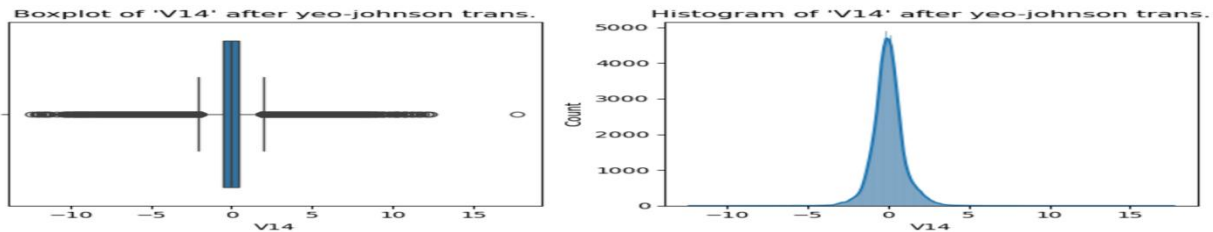
Figure 38 bellow shows its visual representation.

Fig.38



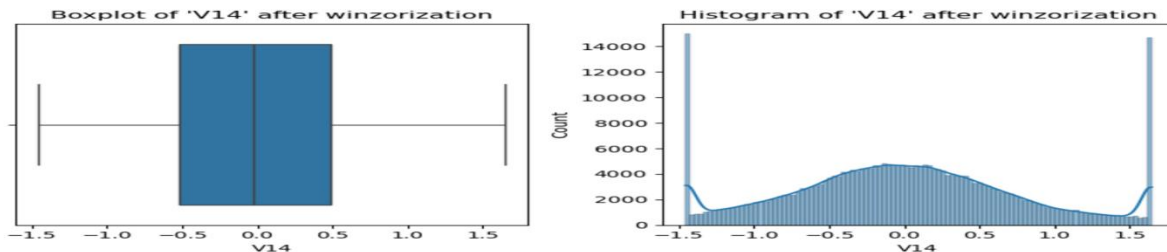
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 38 bellow shows V14 column after transformation.

Fig.38



In other to handle the remaining outliers in V14 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 39 bellow shows V14 after winsorization.

Fig.39



After winsorization, V14 skewness value became 0.17. This value is close to zero and because of this, it was normalized using standardization method.

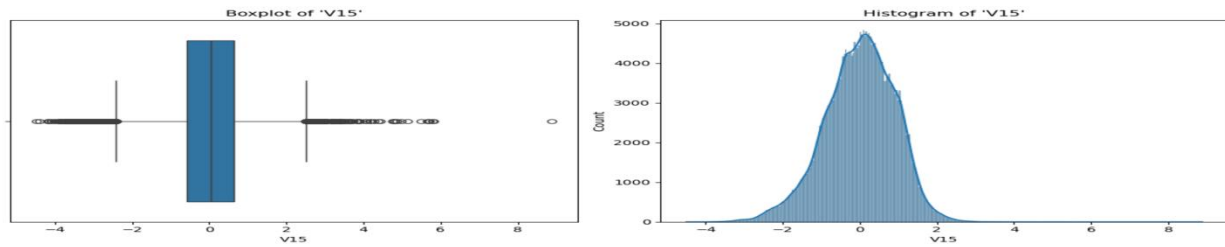
## 26. V15 column analysis.

From the analysis result, it was shown that V15 column has:

- 2884 number of outliers
- Skewness value of -0.31
- Min value of -4.49 and max value of 8.88

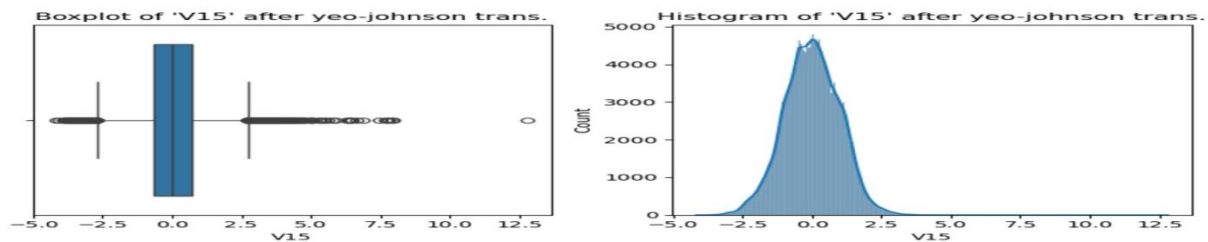
Figure 40 bellow shows its visual representation.

Fig.40



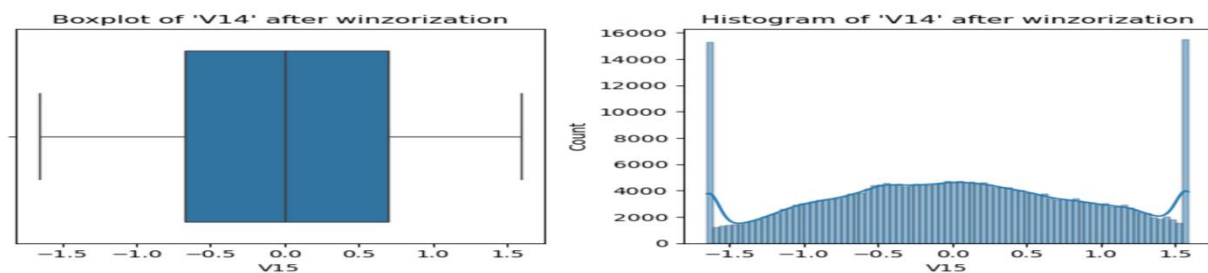
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 41 bellow shows V15 column after transformation.

Fig.41



In other to handle the remaining outliers in V15 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 42 bellow shows V15 after winsorization.

Fig.42



After winsorization, V15 skewness value became 0.02. This value is close to zero and because of this, it was normalized using standardization method.



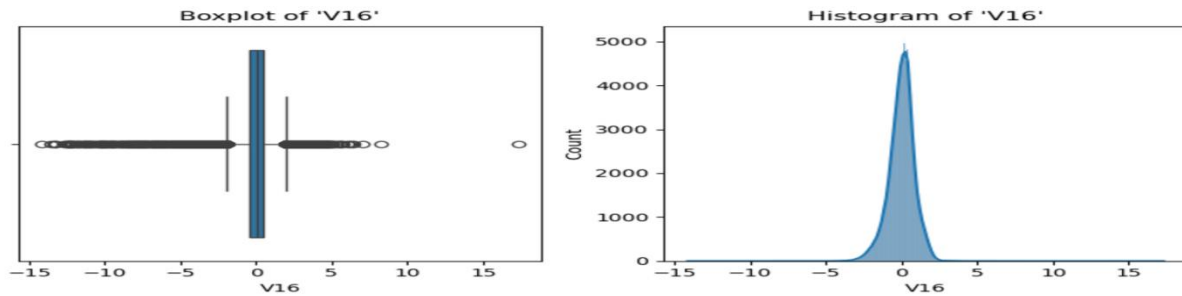
## 27. V16 column analysis.

From the analysis result, it was shown that V15 column has:

- 8180 number of outliers
- Skewness value of -1.05
- Min value of -14.13 and max value of 17.32

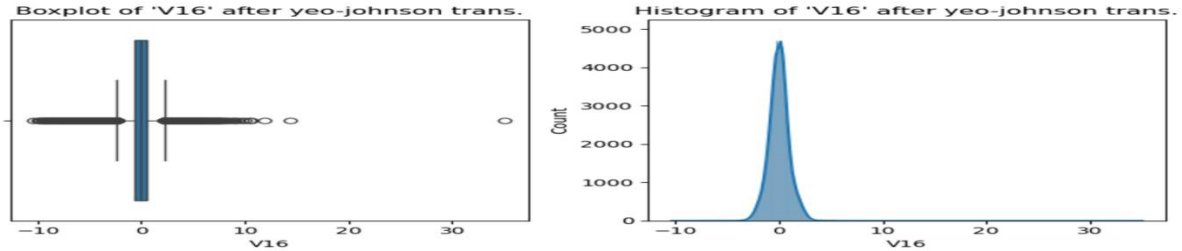
Figure 43 bellow shows its visual representation.

Fig.43



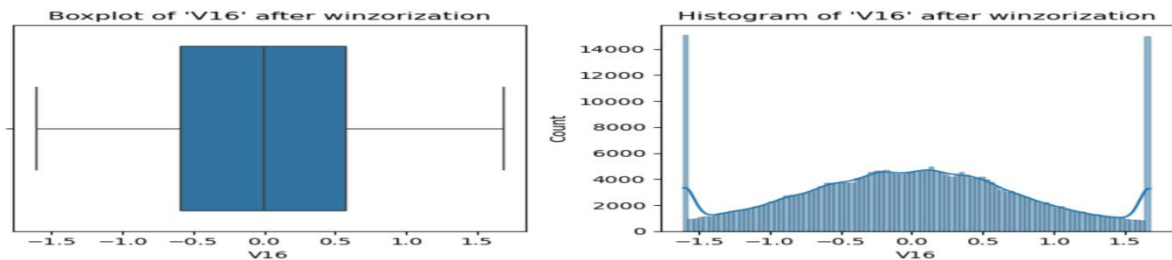
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 44 bellow shows V16 column after transformation.

Fig.44



In other to handle the remaining outliers in V16 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 45 bellow shows V16 after winsorization.

Fig.45



After winsorization, V16 skewness value became 0.06. This value is close to zero and because of this, it was normalized using standardization method.

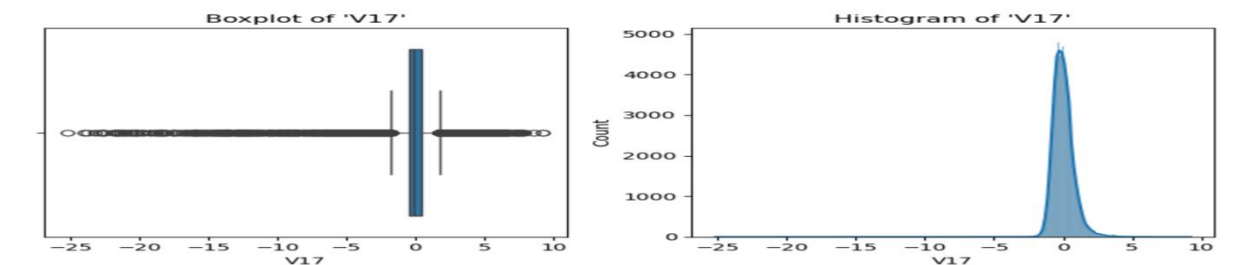
## 28. V17 column analysis.

From the analysis result, it was shown that V17 column has:

- 7353 number of outliers
- Skewness value of -3.69
- Min value of -25.16 and max value of 9.25

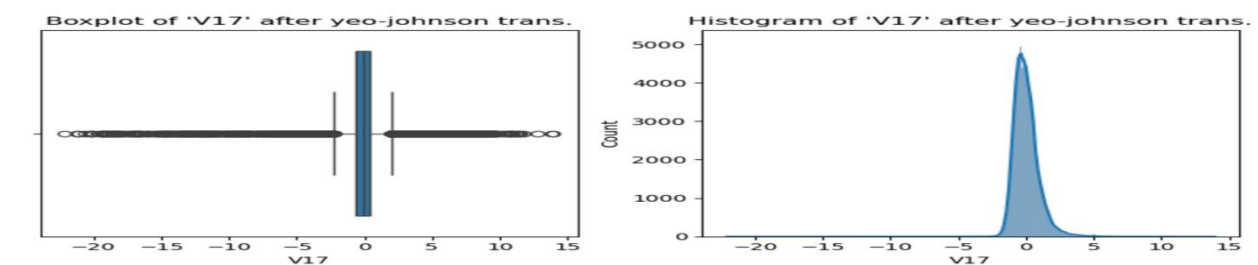
Figure 46 bellow shows its visual representation.

Fig.46



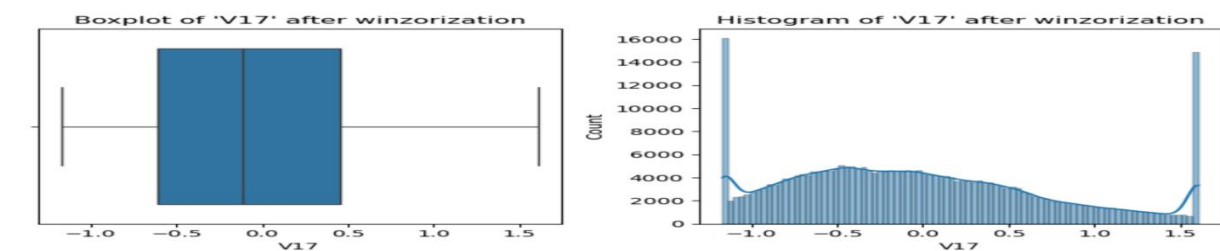
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 47 bellow shows V17 column after transformation.

Fig.47



In other to handle the remaining outliers in V17 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 48 bellow shows V17 after winsorization.

Fig.48



After winsorization, V16 skewness value became 0.49. This value is close to zero and because of this, it was normalized using standardization method.

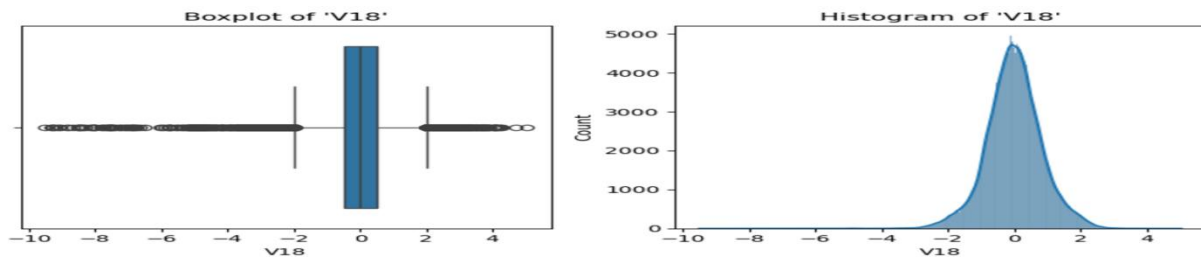
## 29. V18 column analysis.

From the analysis result, it was shown that V18 column has:

- 7468 number of outliers
- Skewness value of -0.25
- Min value of -9.49 and max value of 5.04

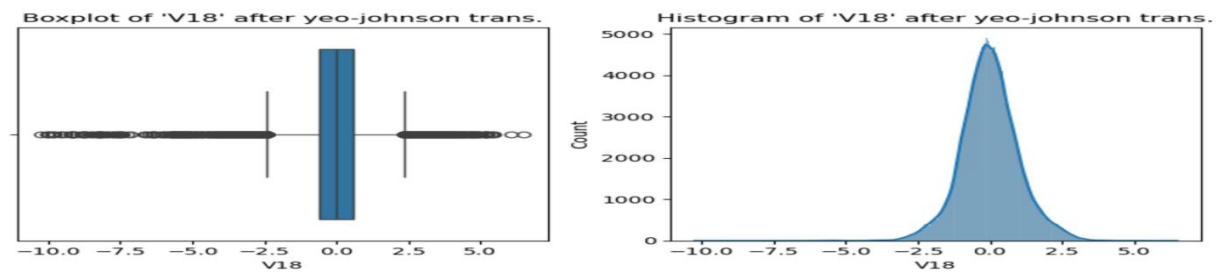
Figure 49 bellow shows its visual representation.

Fig.49



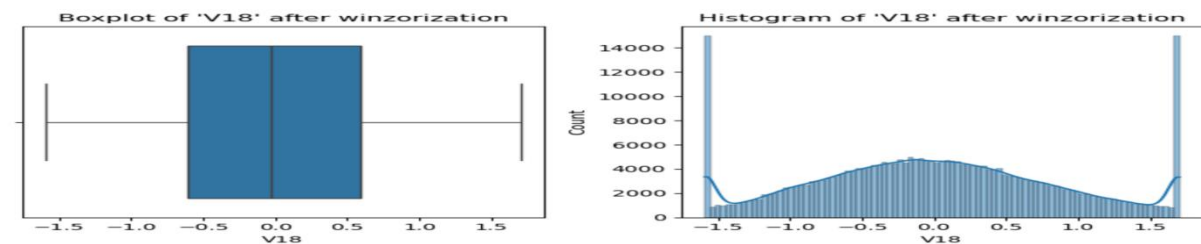
Because it has both positive and negative values, it was transformed using Yeo-Johnson transformation method that is capable of handling both values. Figure 50 bellow shows V18 column after transformation.

Fig.50



In other to handle the remaining outliers in V18 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 50 bellow shows V18 after winsorization.

Fig.50



After winsorization, V18 skewness value became 0.11. This value is close to zero and because of this, it was normalized using standardization method.

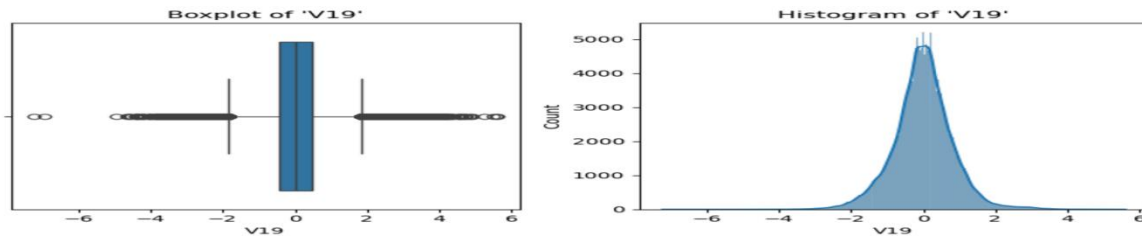
### 30. V19 column analysis.

From the analysis result, it was shown that V19 column has:

- 10150 number of outliers
- Skewness value of 0.11
- Min value of -7.21 and max value of 5.59

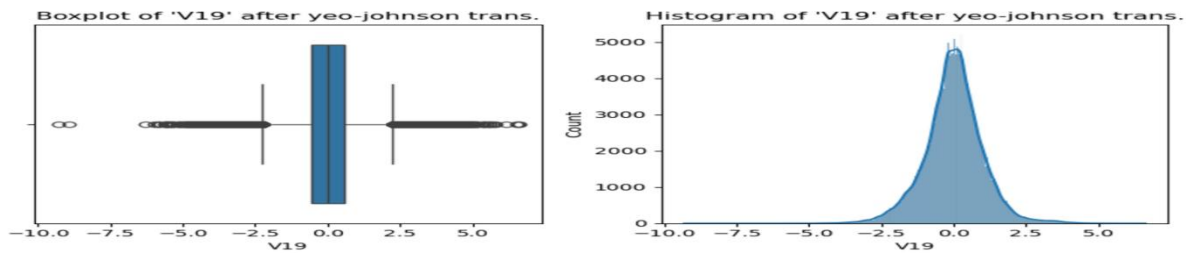
Figure 51 bellow shows its visual representation.

Fig.51



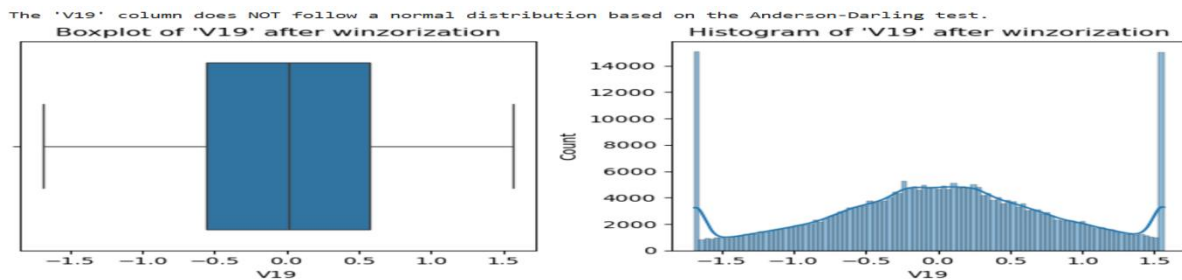
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 52 bellow shows V19 column after transformation.

Fig.52



In other to handle the remaining outliers in V19 after transformation, winsorization method at limit [0.05,0.05] was applied. Figure 53 bellow shows V18 after winsorization.

Fig.53



After winsorization, V19 skewness value became -0.10. This value is close to zero and because of this, it was normalized using standardization method.

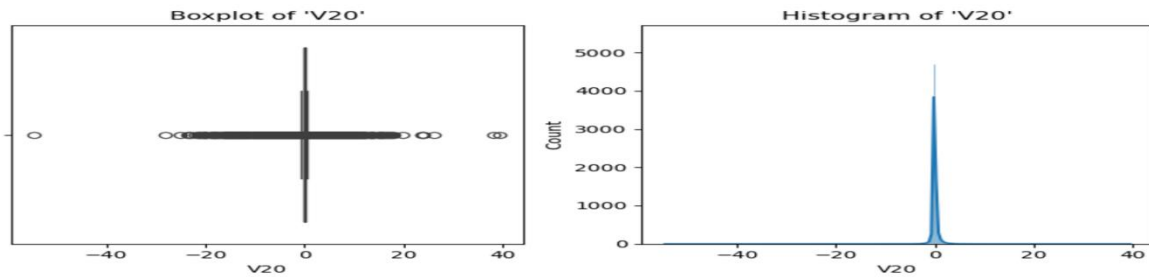
### 31. V20 column analysis.

From the analysis result, it was shown that V20 column has:

- 27553 number of outliers
- Skewness value of -2.04
- Min value of -54.50 and max value of 39.42

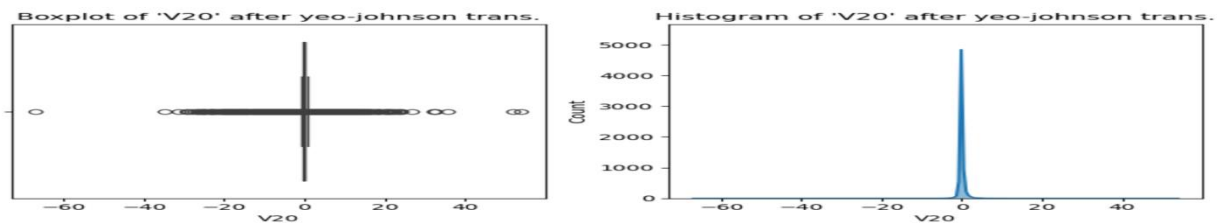
Figure 54 bellow shows its visual representation.

Fig.54



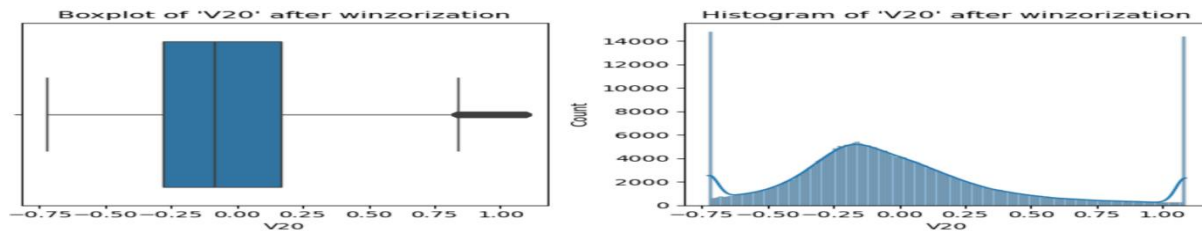
Because it has both positive and negative values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 55 bellow shows V20 column after transformation.

Fig.55



In other to handle the remaining outliers in V20 after transformation, winsorization method at limit [0.05, 0.05] was applied. Figure 56 bellow shows V20 after winsorization.

Fig.56



After winsorization, the number of outliers in V6 reduced from 27553 to 19085. Because of this, V6 was normalized using robust scaling to set its interquartile range value to 1 and its median value to 0. This method suppresses the impact of the remaining outliers on the data.

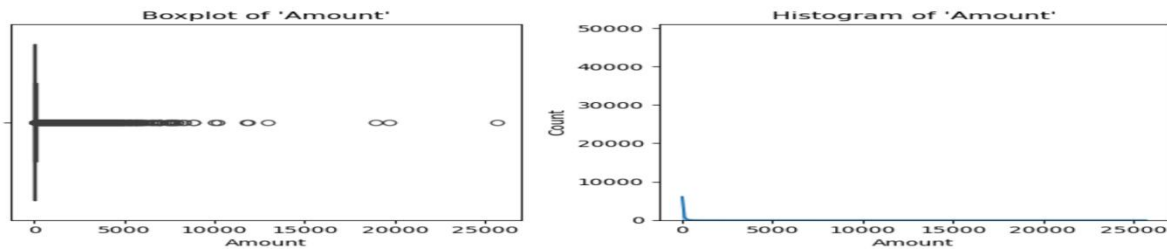
### 32. Amount column analysis.

From the analysis result, it was shown that Amount column has:

- 31685 number of outliers
- Skewness value of 16.98
- Min value of 0.00 and max value of 25691.16

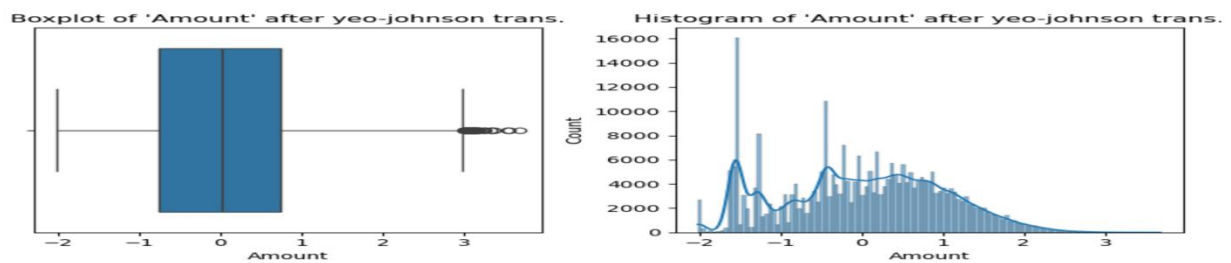
Figure 57 bellow shows its visual representation.

Fig.57



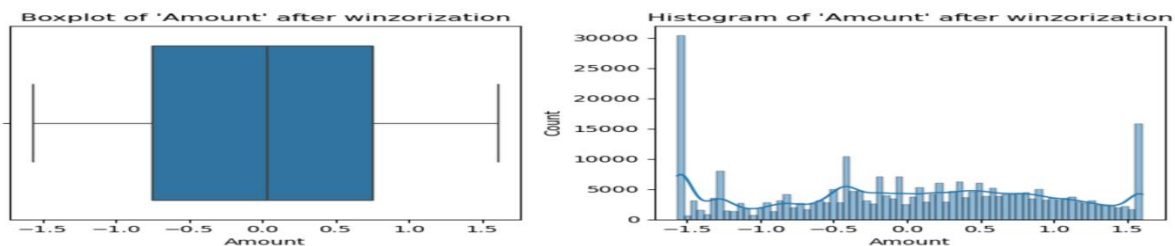
Because it has both positive and zero (0) values, it was transformed using Yoe-Johnson transformation method that is capable of handling both values. Figure 58 bellow shows Amount column after transformation.

Fig.58



In other to handle the remaining outliers in Amount after transformation, winsorization method at limit [0.05, 0.05] was applied. Figure 59 bellow shows Amount after winsorization.

Fig.59



After winsorization, Amount column skewness value became -0.09. This value is close to zero and because of this, it was normalized using standardization method.

### 33. Exploratory data analysis.

This very important phase in data analysis. It focuses on examining variables distribution and how they related to each other.

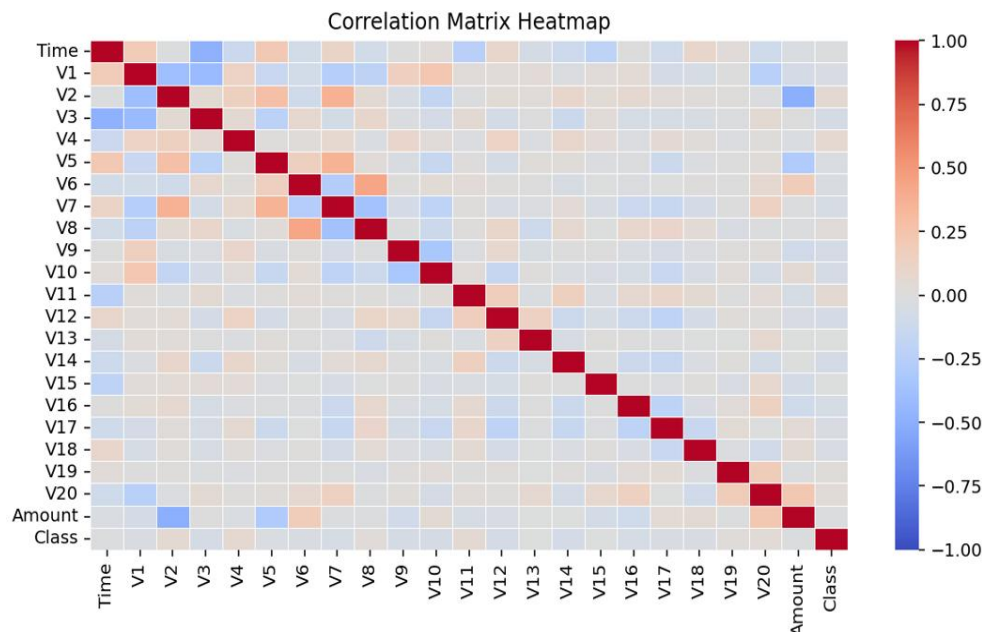
In this study, the distribution of the variables has been examined during data cleaning.

#### 33.1 Linear relationship among the variables.

Linear relationship among variables is a very important attribute of variables in data analysis. This is because it determines which variables that should be kept for model development and ones that should be dropped. It ranges from -1 to 1. A value of -1 or 1 is said to be perfect linear relationship. If linear relationship between two variables is -1, 1 or very close to these values, one of the variables would be dropped to avoid collinearity effect; a condition where by the model cannot differentiate between the effect of the variables.

In this project, the linear relationship among the variables was analysed using correlation Metrix and visualized using heatmap as shown in figure 60 bellow.

Fig.60

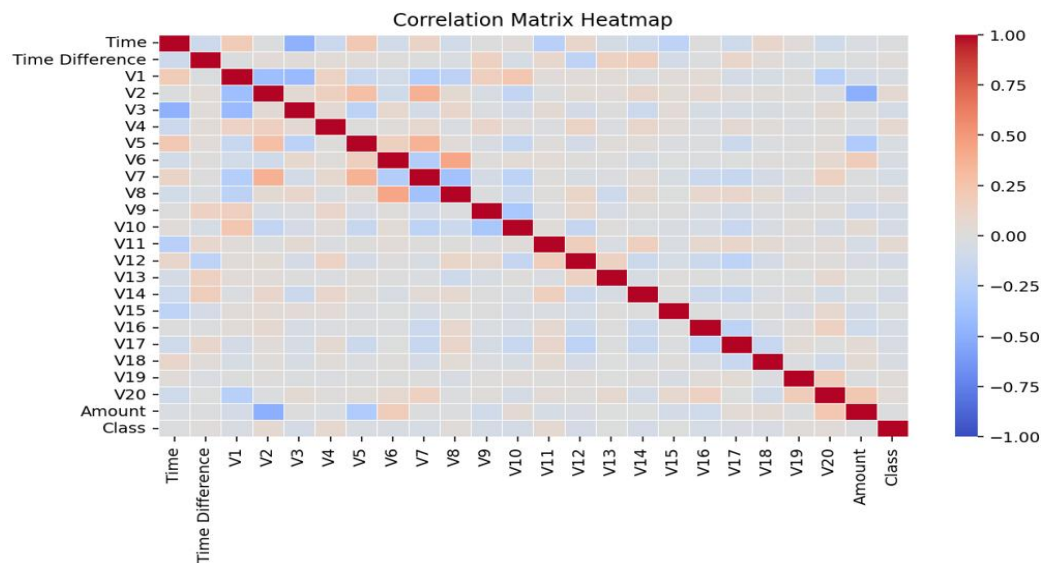


From the heatmap above, it is shown that none of the variable has a strong relationship with one another. As a result, all of them were kept for model development.

### 33.2. Feature engineering analysis.

This a phase in data analysis that focuses on addition of variables and balancing unbalanced class (dependent variable).

In this project, Time Difference variable was added from Time variable. This is because multiple transaction at small interval may be an act of a fraudster trying to make transactions before he is detected. After adding this variable, the linear relationship among the variables was examined again to check if it is strongly related to any of the variables. From the examination result, it does not strongly related to any of the variable as shown in figure 61 bellow. So, it was kept for model development.



### 33.3. Up sampling.

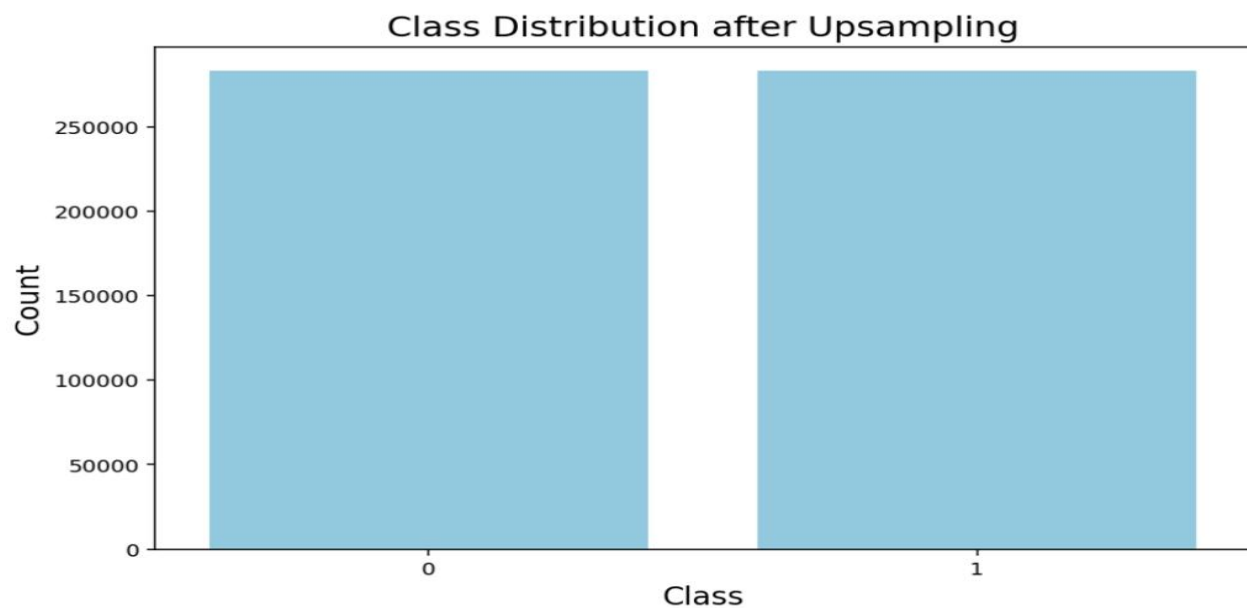
This is a method that is used to balance an unbalanced class. From my analysis, it was shown that the class (target variable) is not balanced. So, up sampling method was applied to balance it. Figure 62 and 63 bellow show the class before and after up sampling respectively.

Fig.62





Fig.63



### 34. Modularity.

After performing a column-by-column analysis in the Jupyter Notebook environment, the dataset was imported into the Spyder environment. Here, the code was organized using encapsulation logic within functions. This structure ensures that each task is assigned to a specific function, facilitating easy code maintenance, bug fixing, and reusability, as this project is intended to be ongoing. Additionally, the use of this structure enables detailed explanations of each function through docstring comments. Furthermore, it incorporates precautionary measures to prevent system crashes.

### 35. Model development.

#### 35.1. Dataset split.

The dataset was split into three sets: 70%, 15% and 15% for training, validation and testing respectively.

#### 35.2. Training.

The models were trained using 70% portion of the dataset. This is to ensure that the models have enough data to learn the underlying pattern among the variables. At this phase, the models were also cross validated using grid search method and 4-folds; ensuring that the algorithm's hyperparameters were tuned as well as examining how the models perform on unseen data.

### 35.3. Validation.

At this phase, 15% of the dataset was used to further fine tune the algorithms' hyperparameters to prevent overfitting.

### 35.4. Testing.

After training, cross validation and validation, the models were tested using 15% of the dataset to examine how the models will perform on new and unseen data.

### 36. Model performance evaluation.

After training, the models' performance was evaluated using Confusion Matrix parameters: Accuracy, Precision, Recall and F1 Score.

- Accuracy is a model's performance evaluation matrix that measures the proportion of correct predications (TP and TN) among all the predictions (TP, FP, TN and FN) made by the model. Mathematically,  $\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$

Where:

TP = TRUE POSITIVE

TN = TRUE NEGATIVE

FP = FALSE POSITIVE

FN = FALSE NEGATIVE

- Precision is a model performance evaluation matrix that measures the proportion of trues positive (TP) predictions among all the positive predictions (TP and FP) made by the model

Mathematically,  $\text{Precision} = \frac{TP}{TP + FP}$

- Recall is a model's performance evaluation matrix that measures the proportion of trues positive (TP) predictions among all actual positive prediction (TP and FN) made by the model.

Mathematically,  $\text{Recall} = \frac{TP}{TP + FN}$ .

- F1-Score is the harmonic mean of precision and recall.

Mathematically,  $\text{F1-Score} = 2 \times \frac{(\text{precision} \times \text{recall})}{(\text{precision} + \text{recall})}$ .

### 37. Result /Discussion.

#### 37.1 Decision Tree.

	Accuracy	Precision	Recall	F1 Score
Training	1	1	1	1
Cross Validation	0.9997	0.9994	1	0.9997
Validation	0.9997	0.9994	1	0.9997
Testing	0.9997	0.9994	1	0.9997

During training, decision tree achieved:

- Accuracy of 1; indicating that it perfectly classified all the instances (positive and negative classes) in the dataset.
- Precision of 1; indicating that it perfectly identified true positive class in the dataset, no false positive. So, it did not miss classify any negative class.
- Recall of 1; indicating that it perfectly identified actual positive class in the dataset, no false negative. So, it did not miss classify any positive class.
- F1 score of 1; indicating that it perfectly balanced precision and recall.

During cross validation, validation and test, it achieved:

- Average Accuracy of 0.9997; indicating that it had 99.97% correct classification of all the instances in the dataset; making only 0.03% error.
- Average Precision of 0.9994; indicating that it had 99.94% correct identification of true positive class in the dataset. So, it misidentified 0.06% negative class as positive class.
- Average Recall of 1; indicating that it had 100% correct identification of actual positive class in the dataset. So, it had no false negative.
- Average F1 score of 0.9997; indicating that it had 99.97% correct balance between precision and recall; making only 0.03% error (false positive and false negative).

From the discussion above, a uniform Recall value of 1 across all phases of development (training, cross-validation, validation, and test) is a good sign.

Also, a slight drop in Accuracy, Precision, and F1 score from 1 during training to 0.9997, 0.9994, and 0.9997 during cross-validation, validation, and test, respectively, is expected because of parameter optimization during training.

So, Decision Tree can be said to have achieved a good fit during training and generalizes well to new data during cross validation, validation and test.

### 37.2. Logistic Regression.

	Accuracy	Precision	Recall	F1 Score
Training	0.9471	0.9648	0.9281	0.9461
Cross Validation	0.9471	0.9649	0.9281	0.9461
Validation	0.9481	0.9650	0.9298	0.9471
Testing	0.9465	0.9631	0.9285	0.9455

During training, logistic regression achieved:

- Accuracy of 0.9471; indicating that it had 94.71% correct classification of all the instances (positive and negative classes) in the dataset; making only 5.29% error.
- Precision of 0.9648; indicating that it had 96.48% correct in identifying true positive class in the dataset, making only 3.52% error (false positive).
- Recall of 0.9281; indicating that it had 92.81% in identifying actual positive class in the dataset, making only 7.19 error (false negative).
- F1 score of 0.9461; indicating that it had 94.61% in balancing precision and recall, making only 5.39% error (false positive and false negative).

During cross validation, validation and test it achieved:

- Average Accuracy of 0.9472; indicating that it had 94.72% correct classification of all the instances (positive and negative classes) in the dataset; making only 5.28% error.
- Average Precision of 0.9643; indicating that it had 96.43% correct in identifying true positive class in the dataset, making only 3.57% error (false positive).
- Average Recall of 0.9288; indicating that it had 92.88% in identifying actual positive class in the dataset, making only 7.12% error (false negative).
- Average F1 score of 0.9462; indicating that it had 94.62% in balancing precision and recall, making only 5.38% error (false positive and false negative).

From the discussion above, a slight drop in Precision from 0.9648 during training to average Precision value of 0.9643 during cross validation, validation and test is expected.

However, a slight increase in Accuracy, Recall and F1 score from 0.9471, 0.9281 and 0.9461 during training to average of 0.9472, 0.9288 and 0.9462 respectively during cross validation, validation and test is a good sign.

As result, Logistic Regression can be said to have achieved a good fit during training and generalizes well during cross validation, validation and test to new data.

### 37.3. Artificial Neural Network (ANN).

	Accuracy	Precision	Recall	F1 Score
Training	0.9997	0.9993	1	0.9997
Cross Validation	0.9993	0.9987	1	0.9993
Validation	0.9995	0.9990	1	0.9995
Testing	0.9995	0.9991	1	0.9995

During training, logistic regression achieved:

- Accuracy of 0.9997; indicating that it had 99.97% correct classification of all the instances (positive and negative classes) in the dataset; making only 0.03% error.
- Precision of 0.9993; indicating that it had 99.93% correct in identifying true positive class in the dataset, making only 0.07% error (false positive).
- Recall of 1; indicating that it had a perfect identification of actual positive class in the dataset, no error made.
- F1 score of 0.9997; indicating that it had 99.97% in balancing precision and recall, making only 0.03% error (false positive and false negative).

During cross validation, validation and test it achieved:

- Average Accuracy of 0.9994; indicating that it had 99.94% correct classification of all the instances (positive and negative classes) in the dataset; making only 0.06% error.
- Average Precision of 0.9989; indicating that it had 99.89% correct in identifying true positive class in the dataset, making only 0.11% error (false positive).
- Average Recall of 1; indicating that it had a perfect identification of actual positive class in the dataset, no error made.
- Average F1 score of 0.9994; indicating that it had 99.94% in balancing precision and recall, making only 0.06% error (false positive and false negative).

From the discussion above, a uniform Recall value of 1 across all the phases of development is a good sign.

However, a slight drop in Accuracy, Precision and F1 Score from 0.9997, 0.9993 and 0.9997 respectively during training to average Accuracy, average Precision and average F1 Score values of 0.9994, 0.9989 and 0.9994 respectively during cross validation, validation and test is expected because of parameter's optimization during training.

As a result, Artificial Neural Network(ANN) can be said to have achieved a good fit during training and generalizes well during cross validation, validation and test.

The models' performance was further evaluated using Receivers Operating Characteristic Curve and Area under the curve

### 38.Receivers Operating Characteristic Curve.

This is a graph that shows the trade-off between True Positive Rate (TPR) and False Positive Rate (FPR); where TPR measures the proportion of true positive class that was correctly identified as actual positive class by a model and FPR measures the proportion of negative class that was wrongly identified as positive by a model.

Area under this curve is a value that shows the ability of a model to assign a higher predicted probability value (a value close to 1 or 1) to positive instance and low predicted probability value (a value close to 0 or 0) to the negative instance if the instances were chosen at random.

Fig.64

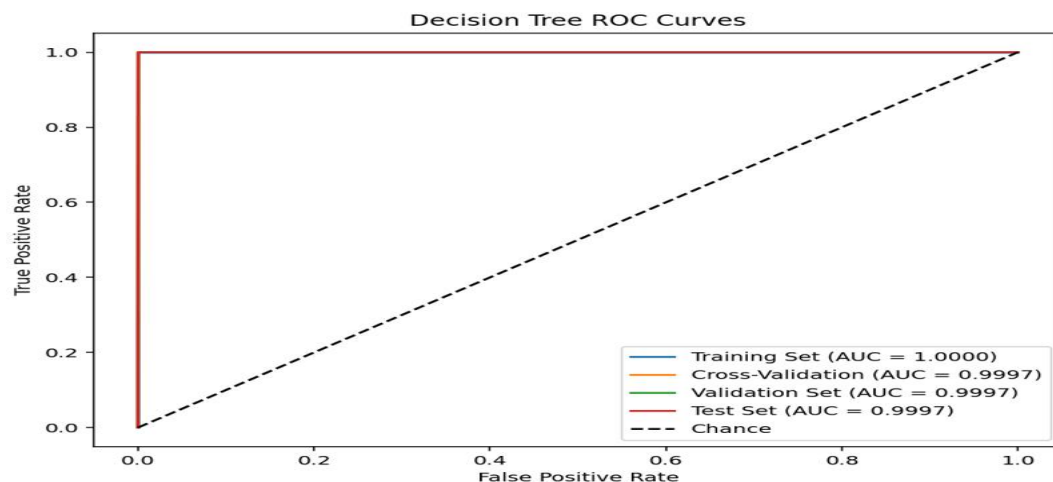


Figure 64 above shows the Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC) for the Logistic Regression model across different phases. The results indicate the model's performance as follows:

- Training: AUC = 1; indicating a ability to rank a randomly chosen positive instance higher than a randomly chosen negative instance based on predicted probabilities.
- Cross validation: AUC = 0.9997, slight decrease compared to training phase which may result from parameter optimization during training. But this still show outstanding discriminative ability.
- Validation: AUC = 0.9997, showing consistent performance with the cross validation phase.
- Test: AUC = 0.9997, showing consistent performance with the cross validation phase.

AUC = 1 during training and a consistent AUC = 0.9997 across cross validation, validation and test remonstrate that the model generalizes well to new data.

In summary, the model has achieved outstanding discriminative ability across all phases of development. The consistently high AUC values indicate a robust and well-generalized model.

Fig.65

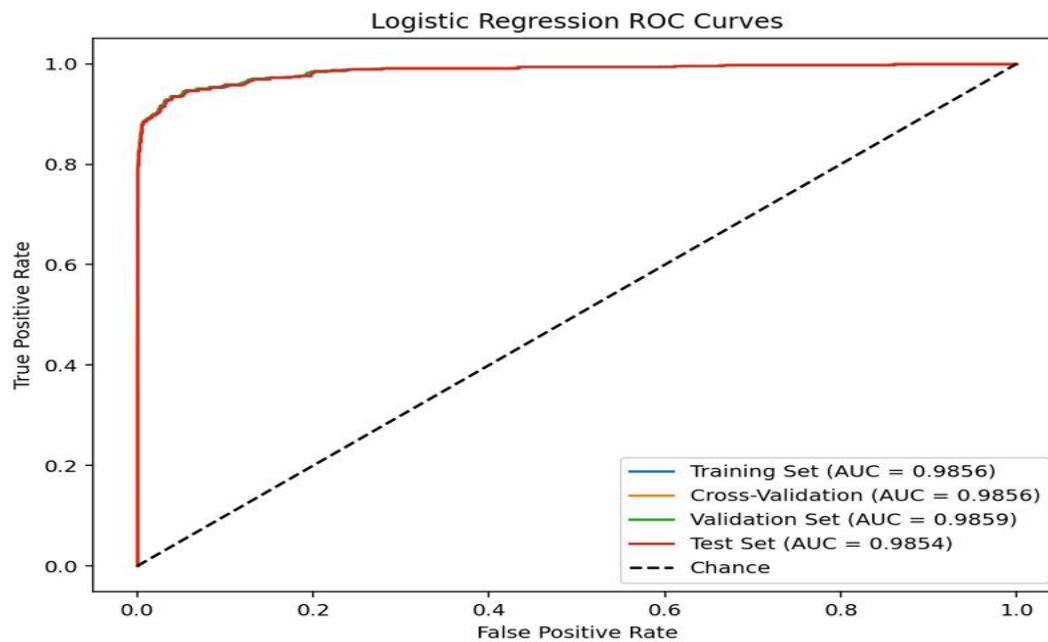


Figure 65 above shows the Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC) for the Logistic Regression model across different phases. The results indicate the model's performance as follows:

- Training: AUC = 0.9856; indicating an excellent ability to rank a randomly chosen positive instance higher than a randomly chosen negative instance based on predicted probabilities.
- Cross validation: AUC = 0.9856: showing consistent performance with the training phase.
- Validation: AUC = 0.9859, slightly higher than the training phase value of 0.9856. This indicates that the model maintains excellent discriminative performance.
- Test: AUC = 0.9854; a slight decrease compared to the training phase, likely due to parameter optimization. This still demonstrates outstanding discriminative ability.

The consistency of AUC = 0.9856 across the training and cross validation phases and a slight increase to AUC = 0.9859 during validation suggests that the model generalizes well to unseen data. The slight drop in AUC during test (from 0.9856 to 0.9854) is expected and reflects the optimization process during training.

In summary, the model has achieved excellent discriminative ability across all phases of development. The consistently high AUC values indicate a robust and well-generalized model.

Fig.67

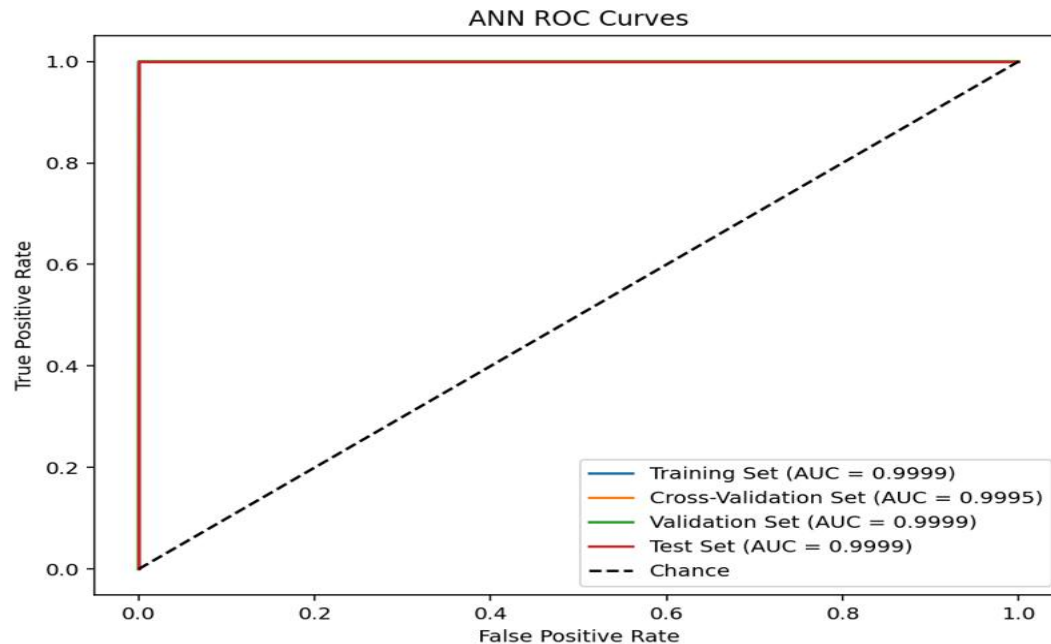


Figure 67 above shows the Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC) for the ANN model across different phases. The results indicate the model's performance as follows:

- Training Phase: AUC = 0.9999, indicating an excellent ability to rank a randomly chosen positive instance higher than a randomly chosen negative instance based on predicted probabilities.
- Cross-Validation Phase: AUC = 0.9995, a slight decrease compared to the training phase, likely due to parameter optimization. This still demonstrates outstanding discriminative ability.
- Validation Phase: AUC = 0.9999, showing consistent performance with the training phase.
- Test Phase: AUC = 0.9999, confirming the model's robustness and excellent generalization ability.

The consistency of AUC = 0.9999 across the training, validation, and test phases suggests that the model generalizes well to unseen data. The minor drop in AUC during cross-validation (from 0.9999 to 0.9995) is expected and reflects the optimization process during training.

In summary, the model has achieved exceptional discriminative ability across all phases of development. The consistently high AUC values indicate a robust and well-generalized model.



### 39.Flask APP development.

After evaluating the performance of the models, they were saved for later use. These saved models were then utilized to develop Flak APP to examine the predictive ability of the models.

### 40.Flask APP deployment.

- The Flask APP was subsequently deployed locally as an API and tested using Postman as the client for simulation. The result demonstrated that the models functioned as expected; predicting 1 for fraud data and 0 for non fraud data.

Figure 68 an 69 bellow show ANN predictions with the different rows of the cleaned data.

Fig.68

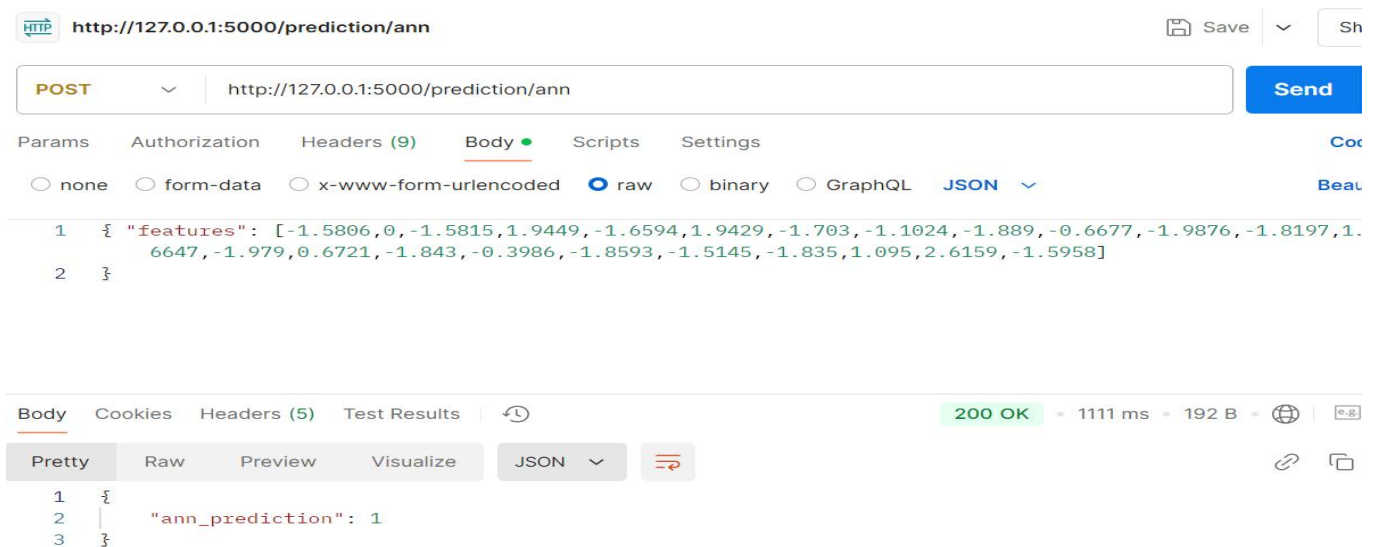
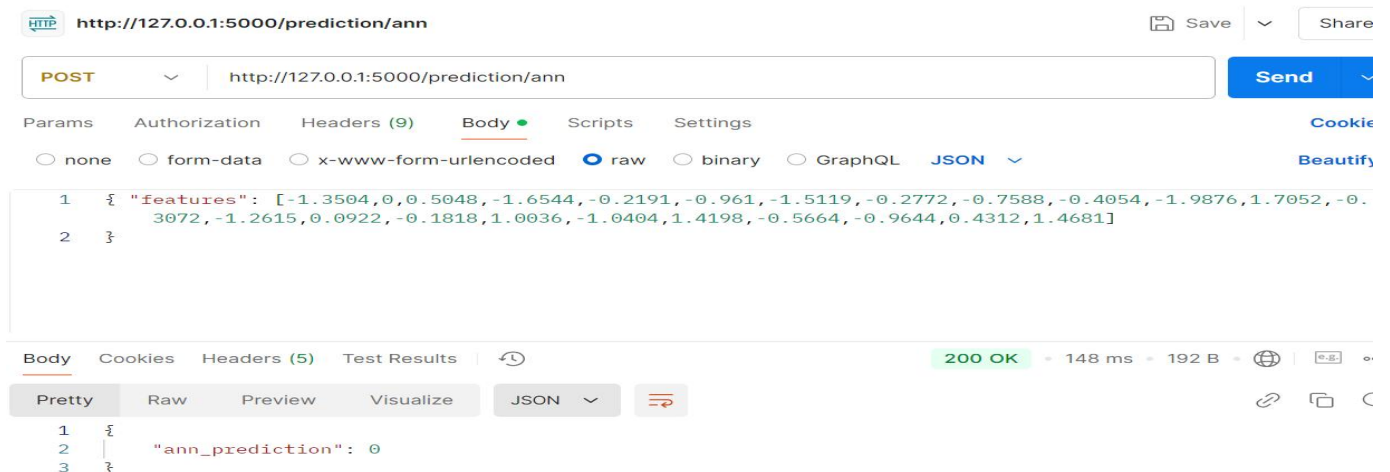
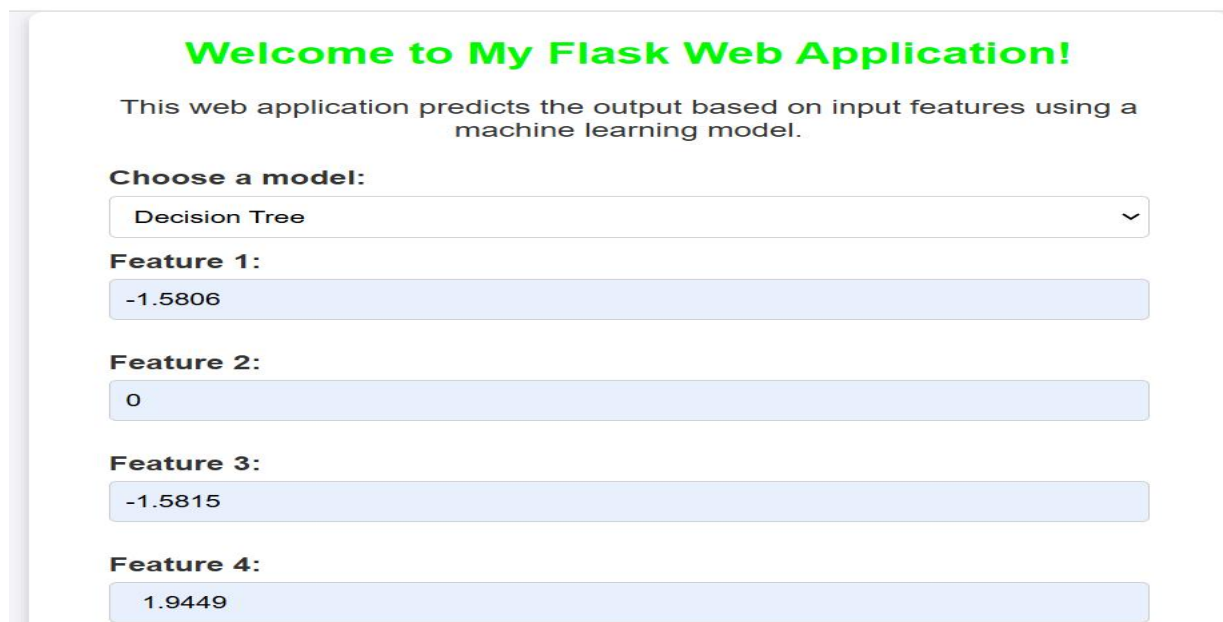


Fig.69



- Furthermore, the Flask APP was deployed on the web where I created a user-interactive form using HTML. additional interactivity was incorporated into the form using javaScript and it was styled with CSS. The result demonstrated that the models functioned as expected ; predicting 1 for fraud data and 0 for non fraud data. Figures 70 and 71 bellow show parts of the user interactive form and the decision tree model's prediction.

Fig.70



**Welcome to My Flask Web Application!**

This web application predicts the output based on input features using a machine learning model.

**Choose a model:**

Decision Tree

**Feature 1:**

-1.5806

**Feature 2:**

0

**Feature 3:**

-1.5815

**Feature 4:**

1.9449

Fig.71



**Feature 19:**

-1.5145

**Feature 20:**

-1.835

**Feature 21:**

1.095

**Feature 22:**

2.6159

**Feature 23:**

-1.5958

**Get Prediction**

**Prediction Result: 1**

#### **41.Conclusion.**

In summary, this research effectively tackled the prediction of credit card fraud through machine learning models(Decision Tree, Logistic Regression and Artificial Neural Network(ANN)), achieving Accuracy, Precision, Recall,F1 score and AUC values above 90% across all the phases of development(Training, Cross Validation, Validation and Test). Despite acknowledging certain limitations, the results offer valuable insights for financial institutions, it presents opportunities for future investigation, including, but not limited to deploying the Flask APP to the cloud.

## 42.Reference.

- Afriyie, J. K., Tawiah, K., Pels, W. A., Addai-Henne, S., Dwamena, H. A., Owiredun, E. O., Ayeh, S. A., & Eshun, J. (2023). A supervised machine learning algorithm for detecting and predicting fraud in credit card transactions. *Decision Analytics Journal*, 6, 100163. <https://doi.org/10.1016/j.dajour.2023.100163>
- Awoyemi, J. O., Adetunmbi, A. O., & Oluwadare, S. A. (2017). Credit card fraud detection using machine learning techniques: A comparative analysis. *Federal University of Technology, Akure, Nigeria*. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8123782>
- Delamaire, L., Abdou, H., & Pointon, J. (2009). Credit card fraud and detection techniques: A review. *Banks and Bank Systems*, 4(2), 57-68. <https://salfordrepository.worktribe.com/output/1465545>
- Gaikwad, J. R., Deshmame, A. B., Somavanshi, H. V., Patil, S. V., & Badgujar, R. A. (2014). Credit card fraud detection using decision tree induction algorithm. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, 4(6), 66. <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=e96f2dc84f124100dc321b188a1103426ceb9672>
- Jain, Y., Tiwari, N., Dubey, S., & Jain, S. (2019). A comparative analysis of various credit card fraud detection techniques. *International Journal of Recent Technology and Engineering (IJRTE)*, 7(5S2), 402-407. <https://doi.org/10.35940/ijrte.E52073.017519>
- Khan, S., Alourani, A., Mishra, B., Ali, A., & Kamal, M. (2022). Developing a credit card fraud detection model using machine learning approaches. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 13(3), 411-417. <https://doi.org/10.14569/IJACSA.2022.0130348>
- Kumar, V. K. S., Kumar, V. G., Shankar, A., & Pratibha, K. (2020). Card fraud detection using machine learning algorithms. *International Journal of Engineering Research & Technology (IJERT)*, 9(7), 649-654. <https://doi.org/10.17577/IJERTV9IS070649>
- Murli, D., Jami, S., Jog, D., & Nath, S. (2014). Credit card fraud detection using neural networks. *International Journal of Students Research in Technology & Management*, 2(02), 84-88. [https://d1wqtxts1xzle7.cloudfront.net/33454229/CREDIT\\_CARD\\_FRAUD\\_DETECTION\\_USING\\_NEURAL\\_NETWORKS-libre.pdf?1397349297=&response-content-disposition=inline%](https://d1wqtxts1xzle7.cloudfront.net/33454229/CREDIT_CARD_FRAUD_DETECTION_USING_NEURAL_NETWORKS-libre.pdf?1397349297=&response-content-disposition=inline%)
- Raj, S. B. E., & Portia, A. A. (2011). Analysis on credit card fraud detection methods. In *Proceedings of the International Conference on Computer, Communication and Electrical Technology (ICCCET 2011)* (pp. 152-157). IEEE. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5762457>
- Sahin, Y., & Duman, E. (2011). *Detecting credit card fraud by ANN and logistic regression*. Marmara University. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5946108>
- Statista. (2022). *Value of e-commerce losses to online payment fraud worldwide from 2020 to 2023(in billion U.S. dollars)*. Statista Technology Market Insights; FBI; IMF; Statista Market Insights. <https://www.statista.com/statistics/1273177/ecommerce-payment-fraud-losses-globally/>

Statista. (2024). *Annual number of cyberattacks worldwide from 2016 to 2023, by type (in millions)*. Statista Technology Market Insights; FBI; IMF; Statista Market Insights.  
<https://www.statista.com/forecasts/1485016/cyberattacks-annual-worldwide-by-type>