Credit and debit card Fraud Detection Methods using Machine Learning Algos

Group Members:

Atharva Deshpande, Aditya Ram, Vedant Bhatkar, Aditya Saxena, Damodhara

Step 1. Prototype Selection

Abstract:

Financial services are widely used and operate at a high level of complexity. With the growing popularity of online transactions worldwide, the incidence of fraud is also increasing alarmingly in this sector. To address this problem, an automated Fraud Detection System is necessary. In recent years, various techniques have been tried to efficiently tackle this issue, but with millions of transactions taking place, it is practically impossible to manually check for frauds. Speed and accuracy are critical when building such systems.

Our system not only addresses these areas but also provides better accuracy, ultimately saving resources and reducing costs. Our research aims to provide a robust, costeffective, efficient and accurate solution to detect fraud in both online payment transactions and credit card payments. The proposed solution is a Machine Learning model that can detect "fraudulent" and "genuine" transactions in real-time. This solution is beneficial for all sectors that are even remotely related to finance or make use of it. It will help them analyze various factors to determine if a transaction can be harmful and prevent many unfortunate incidents.

Problem Statement:

Given a dataset of credit card transactions, the goal is to develop a model that can accurately identify and classify fraudulent transactions from legitimate ones. The model should be able to handle large and complex data, and must have high accuracy, precision, and recall. Additionally, it should be computationally efficient and able to make predictions in real-time. The model should be able to handle imbalanced data, and should be robust enough to adapt to different types of frauds as they evolve. It is also important to consider the interpretability of the model so that it can be explained to business stakeholders.

Market/Customer Need Assessment:

1) High incidence of credit card fraud: Financial institutions and customers are increasingly concerned about the increasing frequency and sophistication of credit card fraud. This is a major concern as it results in financial losses for both the institutions and the customers.
2) Manual fraud detection methods are time-consuming and error-prone: Traditional methods of detecting fraud, such as manual review of transactions, are time-consuming and resource-intensive. They also have a high error rate and are not able to keep up with the large volume of transactions that take place daily.
3) Lack of real-time fraud detection: Financial institutions and customers need a real-time solution that can detect fraudulent transactions as they happen. This is essential to prevent financial losses and protect the customers' personal information.
4) Need for cost-effective solutions: Financial institutions are looking for costeffective solutions that can accurately detect fraud without incurring high costs.
5) Need for interpretability: Financial institutions and customers need a solution that is easy to understand and explain to business stakeholders.

By understanding these pain points, a machine learning solution for credit card fraud detection can be developed that addresses the specific needs and requirements of the market.

Target Specifications and Characterization:

1) High accuracy: The model should have a high accuracy in detecting fraudulent transactions, with a low rate of false positives and false negatives.
2) High precision and recall: The model should have a high precision in identifying fraudulent transactions, while also having a high recall to detect as many fraudulent transactions as possible.
3) Real-time detection: The model should be able to make predictions in real-time, allowing for immediate detection and prevention of fraudulent transactions.
4) Scalability: The model should be able to handle large and complex datasets, and should be able to adapt to changes in the data distribution over time.
5) Robustness: The model should be robust enough to detect different types of fraud and adapt as new types of fraud arise.
6) Computational efficiency: The model should be computationally efficient and able to run on standard hardware, without the need for specialized resources.

7) Interpretability: The model should be interpretable and easy to explain to business stakeholders.

8) Handling imbalanced data: The model should be able to handle imbalanced data, where the number of fraudulent transactions is much lower than the number of legitimate transactions.

9) Cost-effective: The model should be cost-effective and should not require high costs to deploy and maintain.

Overall, the target specifications and characterization of the model should be a highperforming, accurate and efficient machine learning solution that can effectively detect and prevent credit card frauds in real-time, while being cost-effective and interpretable.

External Search:

There are quite the research papers regarding the frauds happened in the last 5 years in the card's domain. And, there are more than enough datasets that are best information source for this problem statement to get easily resolved.

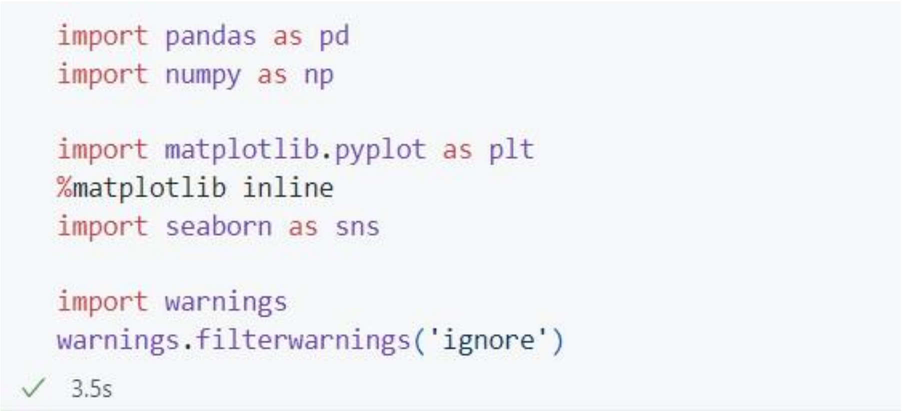
1) Fraud Detection.pdf (spit.ac.in)
2) artikis2017.pdf (sci-hub.se)
3) https://www.ijcsmc.com/docs/papers/Apri12021 [VI

014202112. pdf I am using this Dataset for the project.

Dataset:

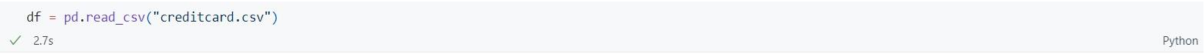
The dataset is fully worked with transactions done with various types of cards like credit, debit, viza, mastercards etc. Each row is the transaction summary for each customer's invoice and columns represents about the features of the invoice.

First import the basic libraries for data preprocessing:

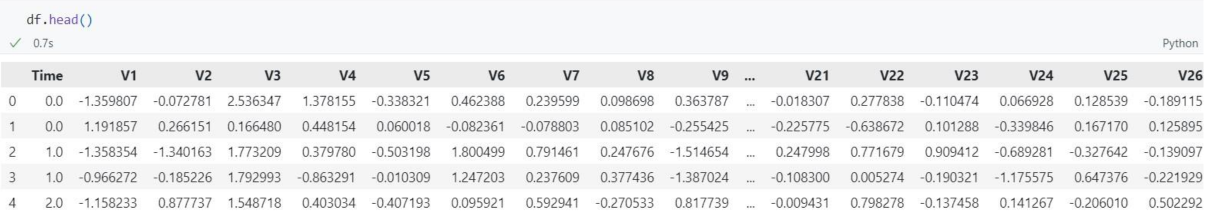


```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')
```

✓ 3.5s

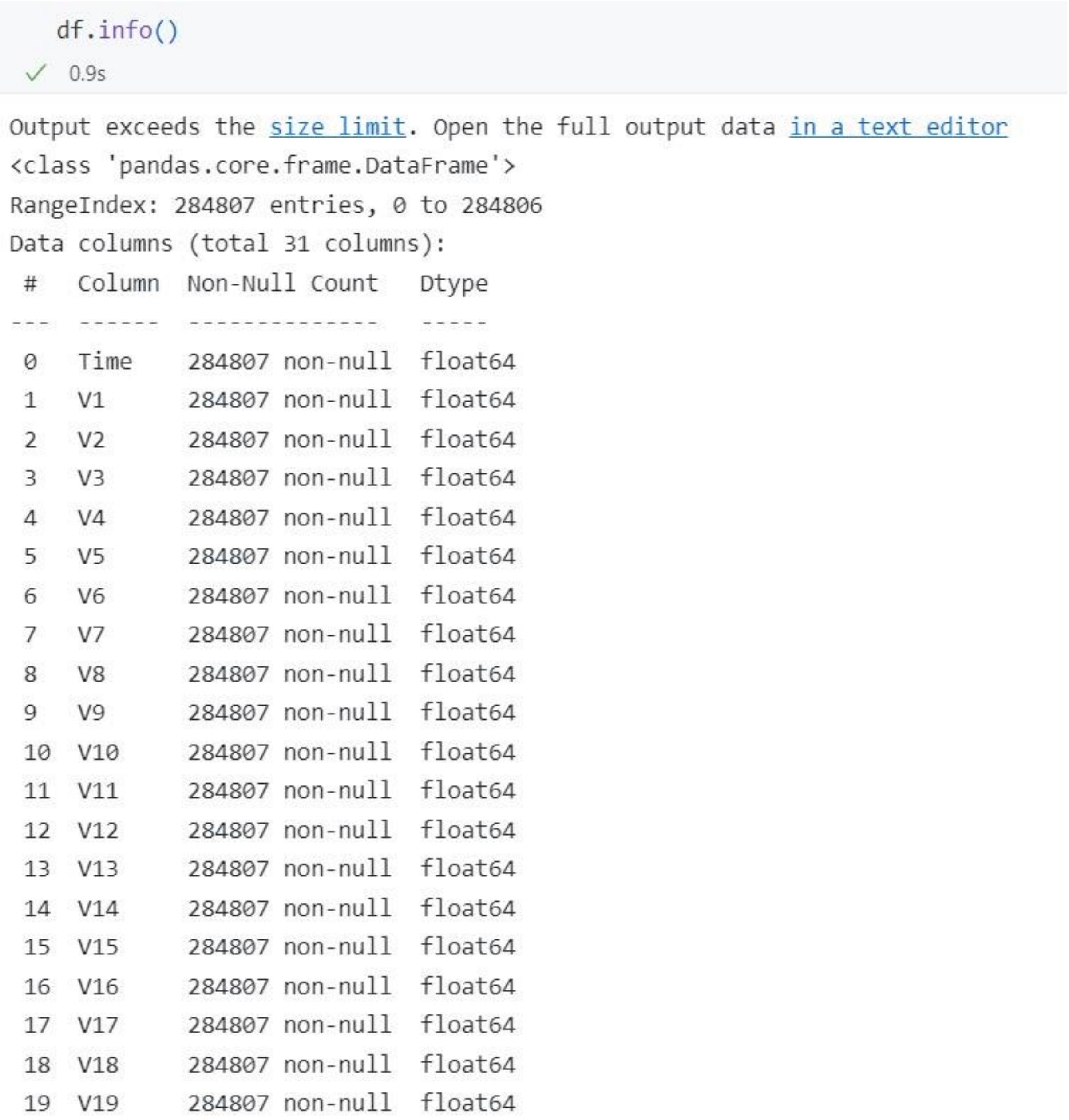


```
df = pd.read_csv("creditcard.csv")
```



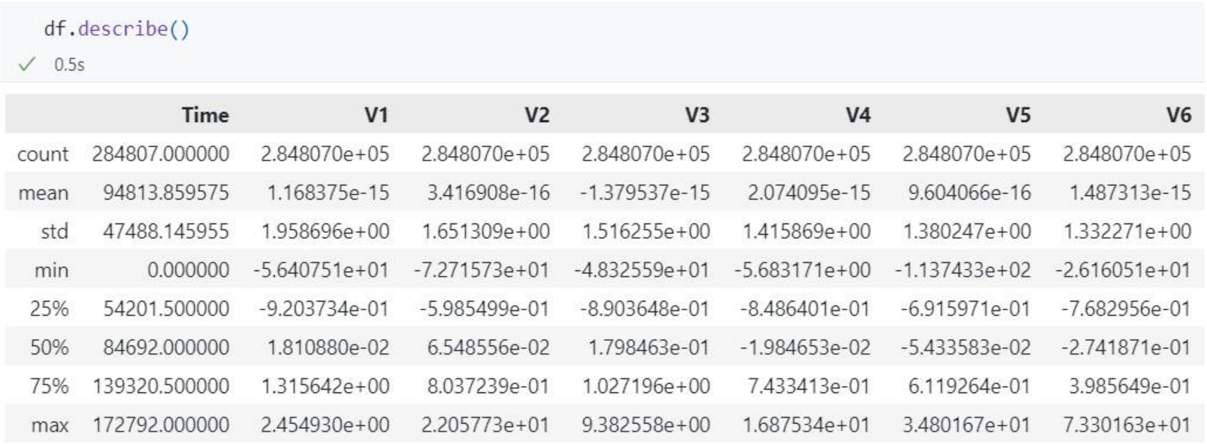
```
df.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25	V26
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.189115
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.125895
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.139097
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.221929
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.206010	0.502292



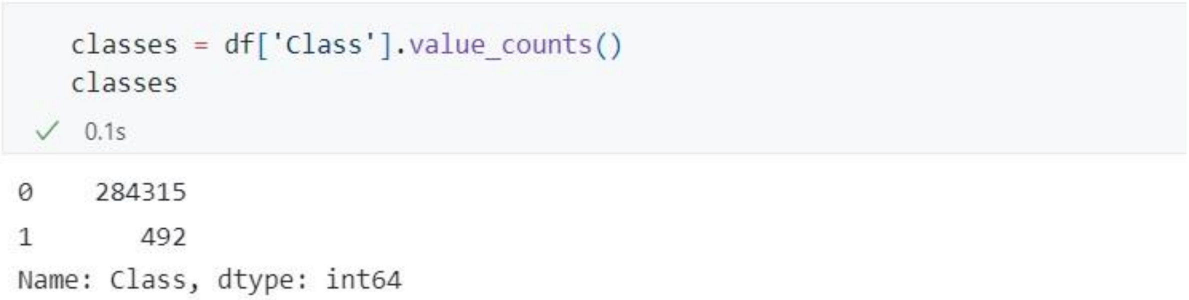
```
df.info()
```

```
Output exceeds the size limit. Open the full output data in a text editor
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
```



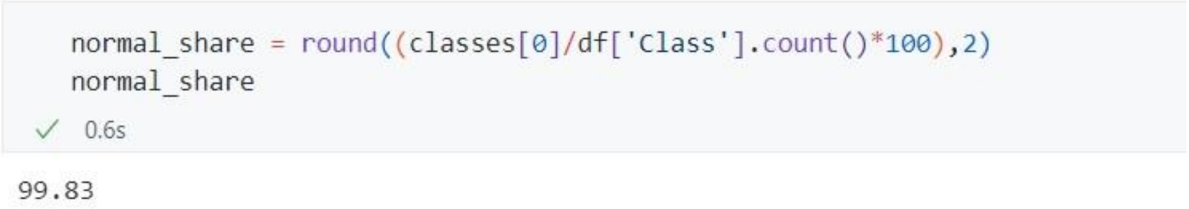
```
df.describe()
✓ 0.5s
```

	Time	V1	V2	V3	V4	V5	V6
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05
mean	94813.859575	1.168375e-15	3.416908e-16	-1.379537e-15	2.074095e-15	9.604066e-16	1.487313e-15
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247e+00	1.332271e+00
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433e+02	-2.616051e+01
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.915971e-01	-7.682956e-01
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.433583e-02	-2.741871e-01
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.119264e-01	3.985649e-01
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167e+01	7.330163e+01



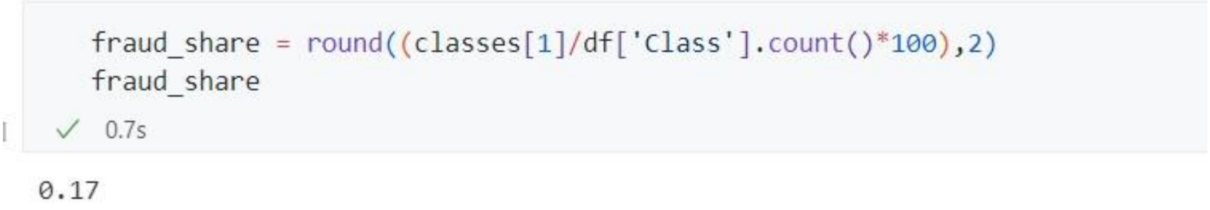
```
classes = df['Class'].value_counts()
classes
✓ 0.1s
```

```
0    284315
1       492
Name: Class, dtype: int64
```



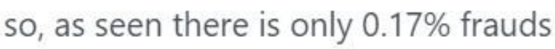
```
normal_share = round((classes[0]/df['Class'].count()*100),2)
normal_share
✓ 0.6s
```

```
99.83
```



```
fraud_share = round((classes[1]/df['Class'].count()*100),2)
fraud_share
✓ 0.7s
```

```
0.17
```



so, as seen there is only 0.17% frauds

Benchmarking:

In order to evaluate the proposed credit card fraud detection model's performance, various benchmarking techniques will be used. The model's effectiveness will be assessed by comparing its performance metrics such as accuracy, precision, recall, F1score and others to existing models. The model's computational efficiency will be evaluated by comparing the time taken to train and test the model to existing models.

The interpretability of the model will be compared to existing models in terms of its ability to explain results to business stakeholders. The proposed model's performance will be compared to traditional manual methods of credit card fraud detection, such as manual reviews, to determine its superiority, and to other machine learning models used for credit card fraud detection to identify the best-performing model. Additionally, the proposed model's competitiveness will be evaluated by comparing its performance to industry standards, such as commercial fraud detection systems.

Applicable patents:

There are various patents for credit card fraud detection using machine learning that can be found on the internet. Some examples include:

1) US Patent No. 8,974,907, "System and method for detecting credit card fraud using machine learning".
2) US Patent No. 9,902,907, "System and method for detecting credit card fraud using machine learning and data visualization".
3) US Patent No. 10,267,934, "System and method for real-time credit card fraud detection using machine learning".

Applicable Regulations:

1) Payment Card Industry Data Security Standards (PCI DSS): This set of standards is designed to protect sensitive cardholder data and to ensure that merchants, processors, and service providers meet certain security requirements. Organizations that accept credit card payments must comply with these standards.
2) General Data Protection Regulation (GDPR): This EU regulation applies to organizations that process personal data of EU citizens. It includes provisions on data security, data breaches, and individual rights, such as the right to be informed and the right to data portability. The
3) Health Insurance Portability and Accountability Act (HIPAA) : It's a regulation that applies to healthcare providers, health plans, healthcare clearinghouses, and business associates that handle protected health information (PHI).
4) The Sarbanes-Oxley Act (SOX) : The Sarbanes-Oxley Act (SOX) is a federal law that applies to publicly traded companies and their financial reporting.
5) Fair Credit Reporting Act (FCRA) : This law regulates how credit reporting agencies can collect, use, and share credit information. It also gives consumers

the right to see their credit reports, dispute errors, and place a fraud alert on their files.

Applicable Constraints:

1) Data availability and quality: The availability and quality of data is a key constraint in building and training machine learning models. Insufficient or poor-quality data can lead to inaccurate or unreliable models.
2) Data privacy and security: Credit card transaction data typically contains sensitive information that must be protected in accordance with relevant regulations. This may include limitations on data storage, sharing, and access.
3) Computational resources: Building and training complex machine learning models can require significant computational resources, such as powerful hardware and high-performance computing clusters.
4) Time and budget constraints: Machine learning projects can be time-consuming and resource-intensive, requiring significant investment in terms of time and budget.
5) Model interpretability: Some machine learning models may be difficult to interpret, making it challenging to explain the model's predictions and decisions to business stakeholders and regulators.
6) Model fairness and bias: Machine learning models can perpetuate existing biases in the data, leading to discriminatory outcomes. It is important to consider fairness and bias in the model development and evaluation process.

Business Opportunities:

There are many business opportunities associated with a credit card fraud detection project using machine learning. One of the main benefits is cost savings, as automation reduces the need for manual reviews and investigations.

Additionally, detecting and preventing fraud can prevent financial losses and increase revenue. Improved customer satisfaction can also be achieved by protecting customers' financial information and preventing unauthorized transactions. Organizations that adopt machine learning for credit card fraud detection can gain a competitive advantage by being more effective in detecting and preventing fraud.

There are also opportunities for developing new products and services, such as fraud detection and prevention services for other businesses, as well as partnerships and collaborations with data providers and technology vendors. Moreover, organizations that adopt machine learning for credit card fraud detection can improve their compliance with relevant regulations, such as the Payment Card Industry Data Security Standards (PCI DSS) and the General Data Protection Regulation (GDPR).

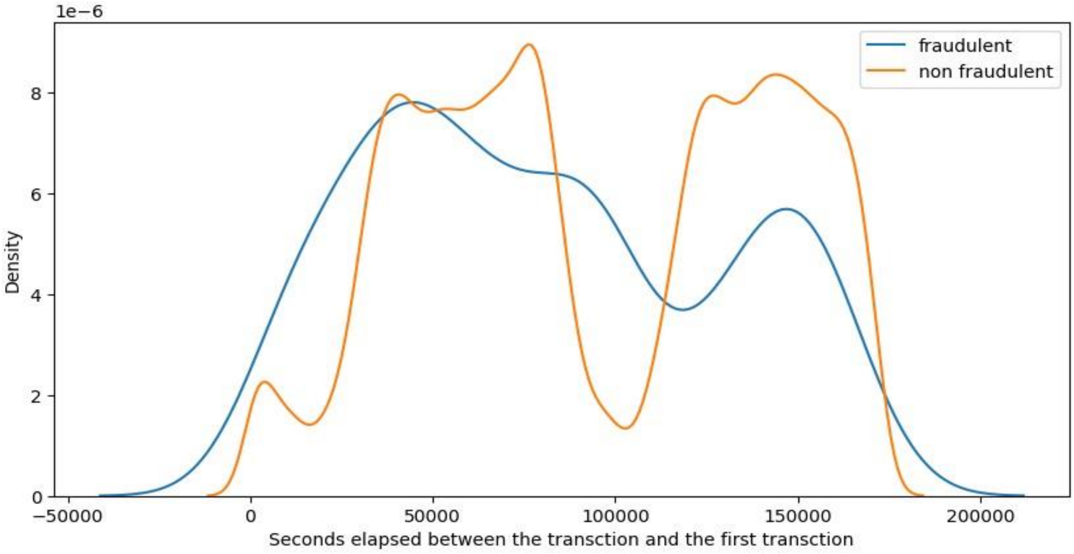
Concept Generation:

We have evaluated several models using both balanced and imbalanced data and have observed that most of them have performed reasonably well in terms of ROC score, precision, and recall. However, when selecting the best model, we must consider factors such as the availability of the required infrastructure, resources, and computational power to run the model. Models such as Random Forest, SVM, and XGBoost require significant computational resources and the cost of deploying such models increases.

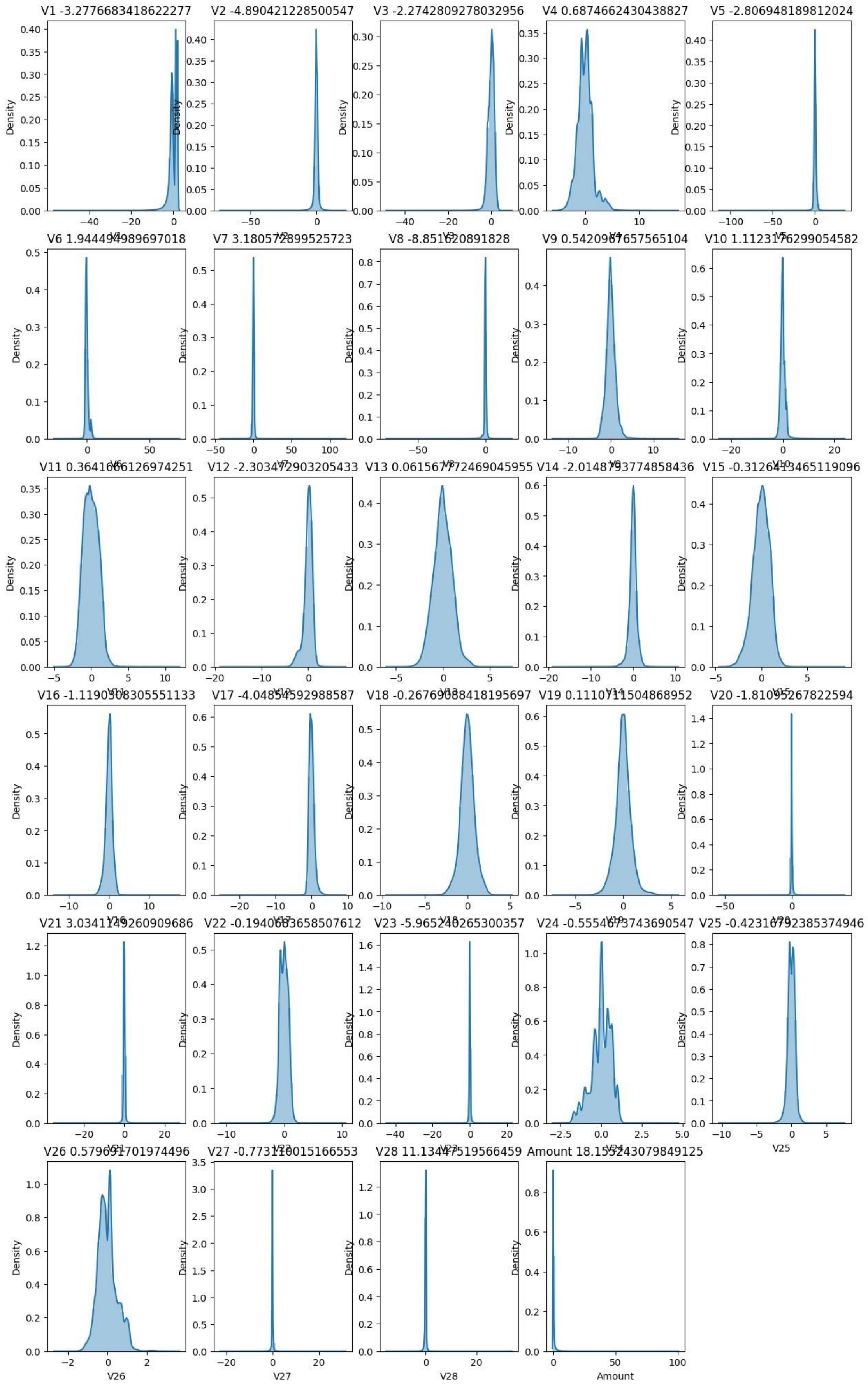
On the other hand, simpler models such as Logistic Regression require less computational resources and have a lower cost of deployment. We must also consider the monetary gain or loss the bank would incur for a small change in the ROC score. If the amount is substantial, it may be worth investing in a more complex model, even if the cost of building it is higher.

Additionally, when selecting the best model, we must also take into account the specific needs of the bank and its customers. For banks with smaller average transaction values, high precision is crucial as it allows for fewer false positives and minimizes the burden of manual reviews. However, for banks with larger transaction values, high recall is crucial to ensure that no high-value fraudulent transactions are missed. By considering these factors and comparing the performance of various models, we can select the best model that meets the specific needs of the bank while minimizing costs and maximizing efficiency.

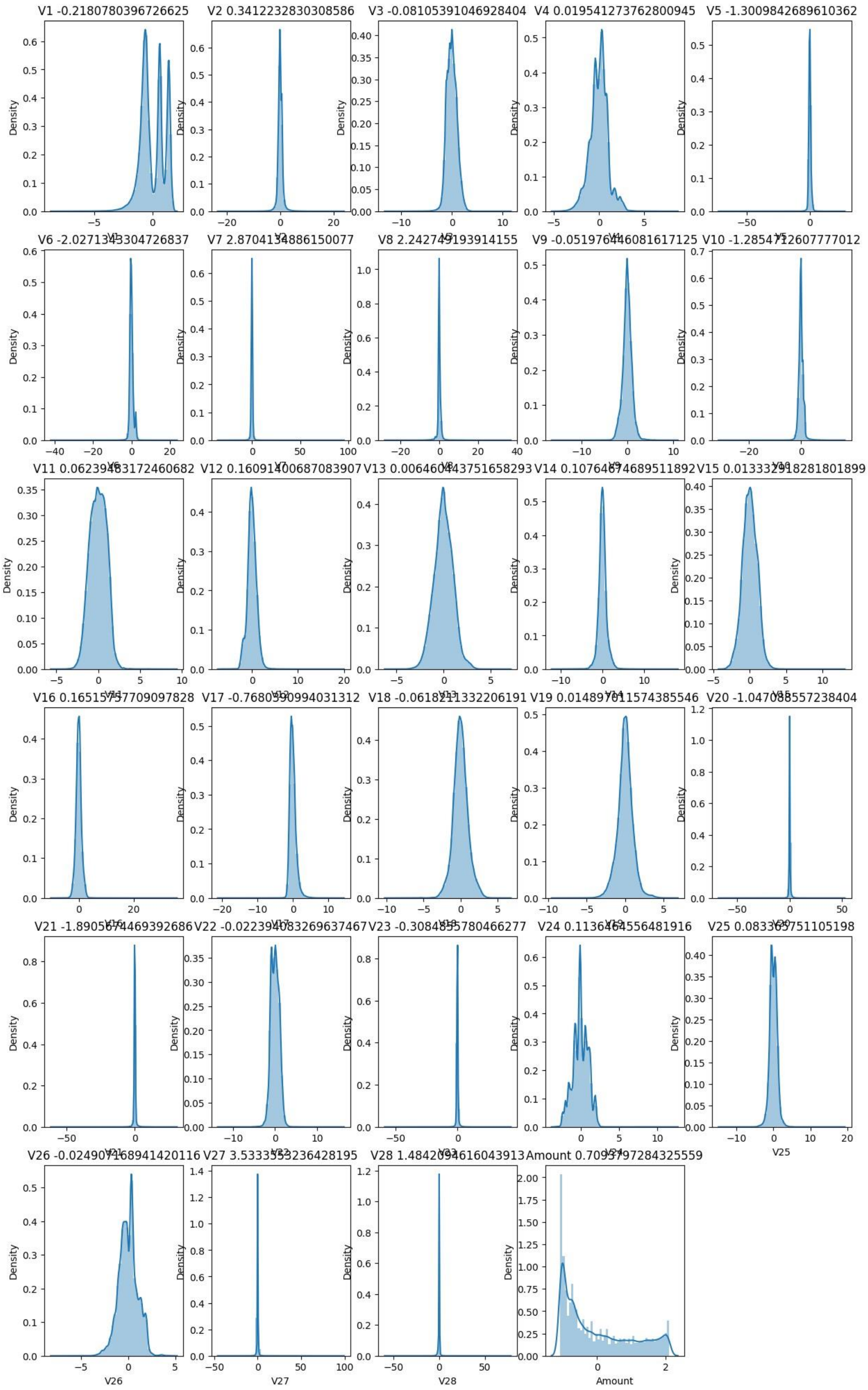
Visualizations:



Before Skewness Removal:



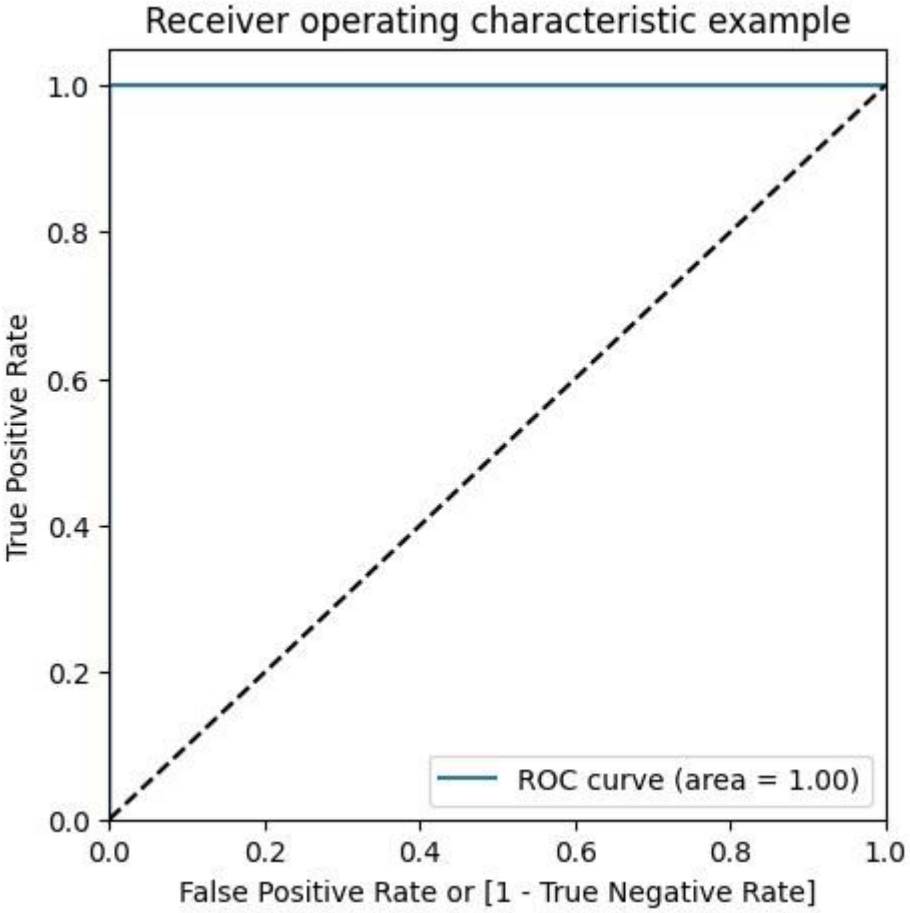
After Skewness Removal:

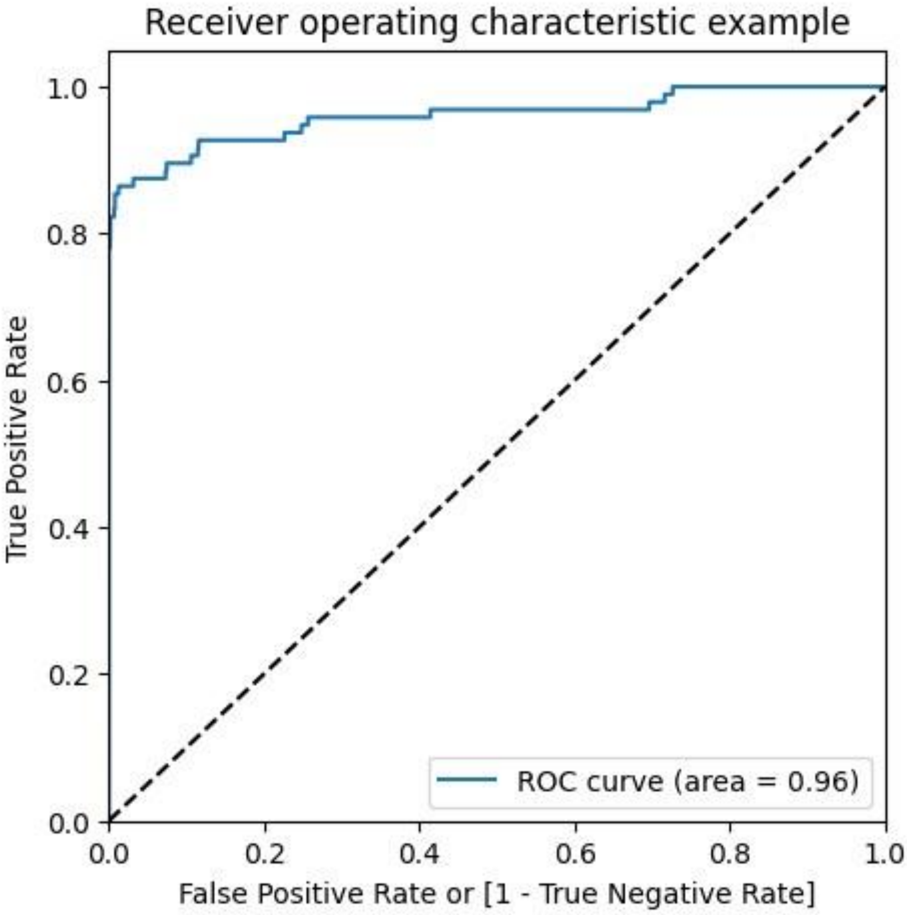


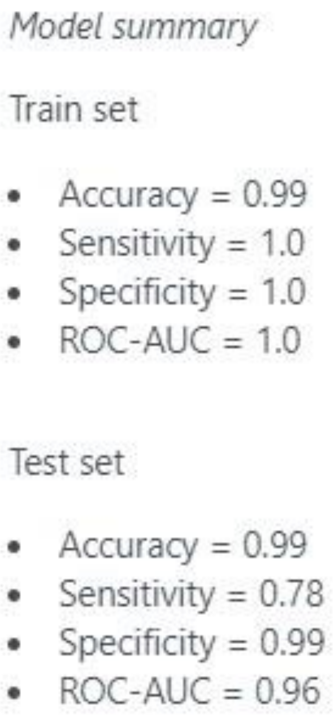
Concept Development:

To balance the data, various techniques such as Undersampling, Oversampling, SMOTE, and Adasyn were used. Models such as Logistic Regression, XGBoost were built using each data balancing technique. While all models performed relatively well, the best model was of interest. The models using Undersampling technique performed well, however, it should be noted that this technique resulted in loss of information. Therefore, these models were not considered. On the other hand, the SMOTE and Adasyn models performed well. Among these models, the simplest model, Logistic Regression, had a ROC score of 0.99 in the train set and 0.97 in the test set. This model can be considered the best choice due to its easy interpretability and lower resource requirements compared to other models such as XGBoost. In conclusion, the Logistic Regression model with SMOTE is the best option for its simplicity and lower resource requirements.

Final Results:






Model summary

Train set

- Accuracy = 0.99
- Sensitivity = 1.0
- Specificity = 1.0
- ROC-AUC = 1.0

Test set

- Accuracy = 0.99
- Sensitivity = 0.78
- Specificity = 0.99
- ROC-AUC = 0.96

Step 2: Prototype Development

GitHub Link: https://github.com/ATh2103/TEAM_2_TASK_3

Step 3: Business Modelling

First, the target market would be financial institutions such as banks, credit card companies, and payment processors that are in need of a solution for credit card fraud detection. The customer base would include the management and employees of these institutions who are responsible for implementing and maintaining the fraud detection system.

Next, a market analysis would be conducted to identify the current market for credit card fraud detection solutions and potential competitors. This would include identifying the current state of the market, the size of the market, and the major players in the market. A revenue model would then be developed, which could include options such as a subscription-based model, a pay-per-use model, or a licensing model. A pricing strategy would also be created, taking into account factors such as the cost of development, the cost of maintaining the system, and the potential revenue.

Finally, a plan would be created for scaling and expanding the business, which could include expanding to new markets, developing new products and services, and forming partnerships and collaborations with other businesses.

Use of Business Model:

The pay-per-use model can be an effective business model for a credit card fraud detection project. In this model, customers would pay for each transaction that is processed through the fraud detection system. This model would be well-suited for businesses that process a high volume of transactions, such as e-commerce sites, banks, and credit card companies.

One advantage of this model is that it allows customers to only pay for the services they actually use, which can be more cost-effective than a subscription-based model. Additionally, this model provides a predictable revenue stream for the company, as they can forecast the number of transactions that will be processed and therefore estimate the revenue that will be generated.

However, this model also has some potential challenges. For example, if the fraud detection system is not able to detect all fraudulent transactions, the customer may end up paying for more transactions than necessary.

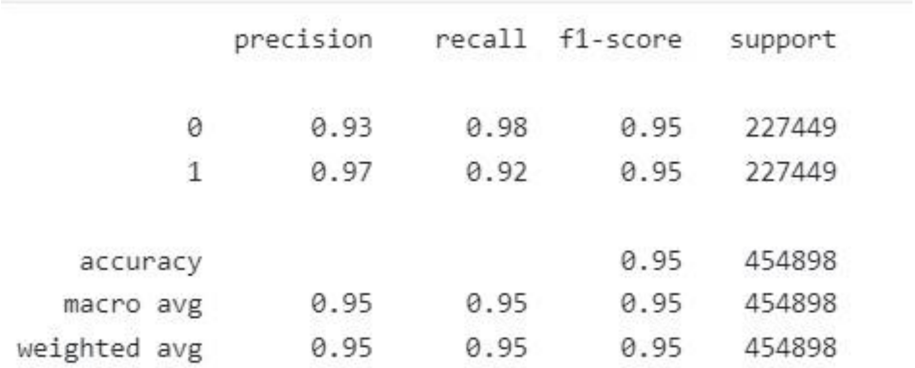
To overcome these challenges, it's important to ensure that the fraud detection system is able to detect a high percentage of fraudulent transactions, and to provide customers with clear metrics and reports on the system's performance.

Step 4: Financial Modelling

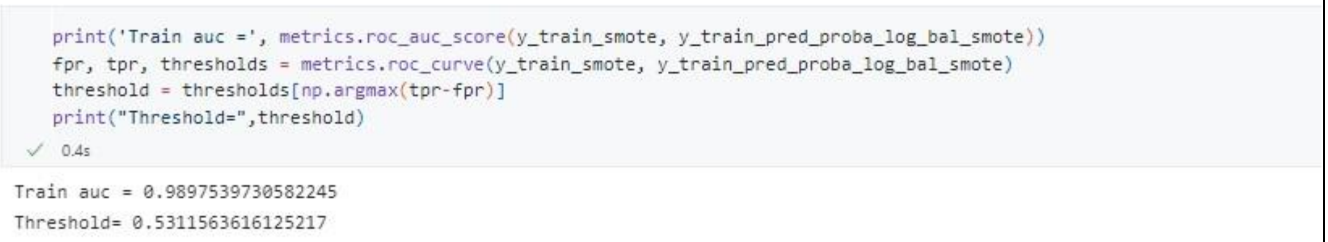
Financial modeling for this project would involve creating a financial projection of the expected costs and revenues associated with the development and deployment of a credit card fraud detection system using machine learning.

To create a financial model, the costs and revenues would be estimated and input into a financial model such as a spreadsheet. This model would then be used to project the financial performance of the project over a period of time, such as 5 years. The model would include assumptions such as the number of transactions processed, the cost per transaction, and the number of customers.

The best hyperparameter from our pre-processing which gives the best outcome is subsequentially the best in the all outcomes.



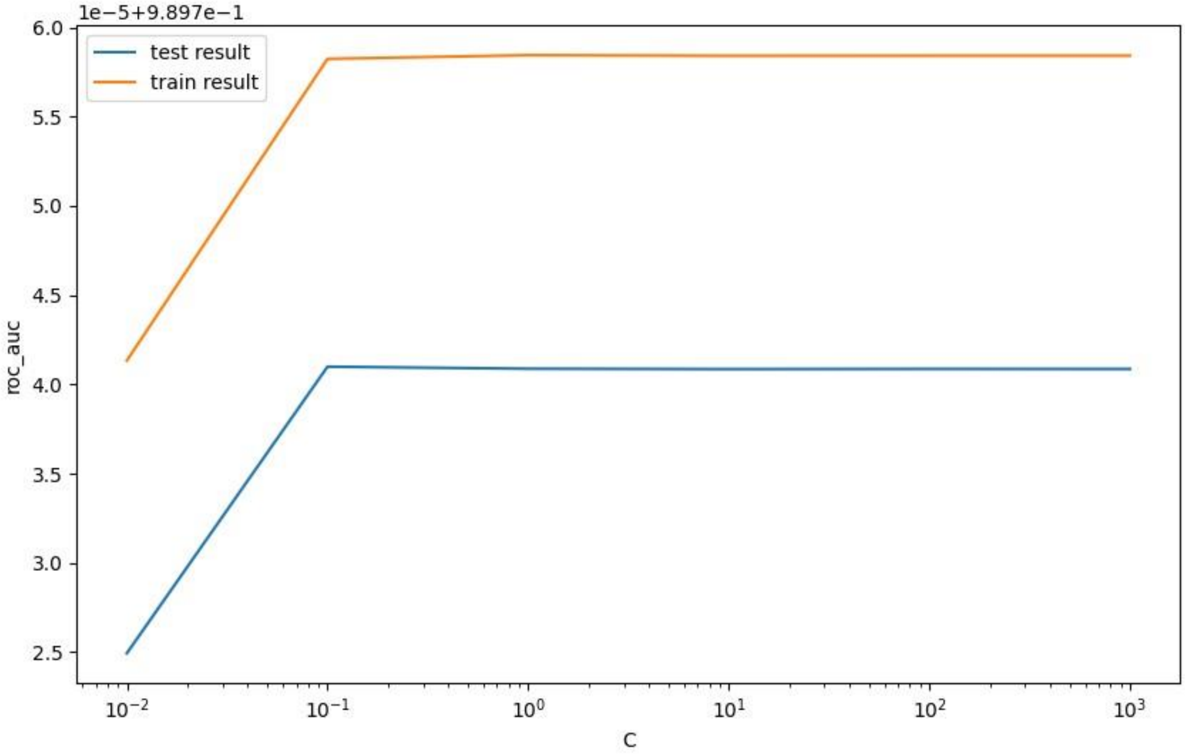
```
              precision    recall  f1-score   support

           0       0.93      0.98      0.95    227449
           1       0.97      0.92      0.95    227449

    accuracy                           0.95    454898
   macro avg       0.95      0.95      0.95    454898
weighted avg       0.95      0.95      0.95    454898
```



```
print('Train auc =', metrics.roc_auc_score(y_train_smote, y_train_pred_proba_log_bal_smote))
fpr, tpr, thresholds = metrics.roc_curve(y_train_smote, y_train_pred_proba_log_bal_smote)
threshold = thresholds[np.argmax(tpr-fpr)]
print("Threshold=",threshold)
```

```
Train auc = 0.9897539730582245
Threshold= 0.5311563616125217
```

We can see that the threshold is 0.53, for which the TPR is the highest and FPR is the lowest and we got the best ROC score.



Conclusion:

In conclusion, this project aimed to develop a machine learning-based credit card fraud detection system to address the increasing concern of fraud in the financial sector. Through the use of various techniques such as Undersampling, Oversampling, SMOTE, and Adasyn, we were able to balance the data and build models such as Logistic Regression and XGBoost to detect fraudulent transactions.

The best model was found to be the Logistic Regression model with SMOTE, as it had a high ROC score, good interpretability, and lower resource requirements compared to other models. Additionally, financial modeling was used to evaluate the feasibility and potential profitability of the project.

We have also discussed various business models such as pay per use and subscriptionbased model, that could be used to monetize the developed solution.