Year In Data Analytics Project

Credit Card Fraud Dataset

Ayman El Mahdi(aee24), Shahd Saeed(ss2542), Stella Umasaran(su61), Sagari Muraliegaran(sm2288), Jenna Robson(jr741)

“Can we reliably detect fraudulent transactions using the dataset and what models are the most accurate?”

Contents:

1. Introduction

Credit card fraud is the unauthorised use of someone else’s credit card to make purchases or withdraw funds. In the first half of 2024, there was over £570 million stolen in the UK by fraudsters due to unauthorised access to accounts (UK Finance, 2024). This statistic encapsulates how big and prominent this issue truly is and how important it is to stop these transactions from happening.

This type of fraud can also cause many other issues such as the increase in banking and business costs; this can be due to banks having to reimburse lost money to their members which was lost due to fraud which may then lead to increased interest rates, increased overdraft fees etc. This could also happen with businesses if they have to reimburse the lost money as they may need to get their money back by increasing their prices for their consumers.

Machine learning is a great way to combat this and offers some great advantages over traditional methods of detecting fraud which are: its ability to detect patterns, real-time analysis, and adaptive learning (which means the model can improve with the more data it gets). Meanwhile, traditional methods suffer from high false positive rates and limited scalability (Martin, 2022).

One challenge we expect to come across when making and testing our models is the fact that fraudulent transactions are very rare in the dataset, we are using which is called a class imbalance, this could lead to the models being biased towards non fraudulent transactions. We will try and mitigate this using under sampling where we select the same amount of fraudulent and non-fraudulent transactions to balance the data, and we will see if this leads to a more accurate model, and this will be verified using confusion matrices. The models we will be comparing with both dataset as it is and under sampling will be logistic regression as a baseline, a classification tree, and a neural network.

The dataset is also only 2 days’ worth of transactions, meaning that the models we create may not be very replicable in the real world. We will be treating this as proof of concept instead of an actual model that businesses and banks can use.

Banks and businesses will still hugely benefit from this report as it will provide some of indication as to which models perform best and what the best way to train these models is. This could lead to them attempting to create models that are trained on larger datasets which may lead to more powerful models.

2. Literature

Introduction -

This essay has already highlighted the serious consequences of credit card fraud, but this section will focus on why it's important to explore the existing literature on the topic. Gaining insight into current trends, debates, and gaps in the research helps to build a stronger understanding of what has already been achieved in the field, where the limitations lie, and how this project can contribute something meaningful and new.

By reviewing a range of academic sources, we can better understand how different machine learning techniques have been applied to the problem of fraud detection and evaluate which approaches have been most effective — particularly when working with imbalanced datasets. This foundation is essential for shaping the direction of our own study and for justifying the choice of models and evaluation strategies we employ.

Background -

With the digital world becoming more prevalent, credit card fraud becomes increasingly more threatening to the financial sector, with criminals driving pressure as they consistently developing more sophisticated ways to bypass any barriers that have been put in place. This has forced a response, with researchers along with several institutions, turning to machine learning models to fight back; improving fraud accuracy. These models have the potential to identify subtle patterns in large datasets and adapt to new types of fraudulent behaviour more effectively than traditional rule-based methods (Dal Pozzolo et al., 2017).

However, the fight back against detecting fraud is not that simple. A significant issue we came across during the literature review was class imbalance, where fraudulent transactions make up only a tiny fraction of the dataset, leading to biased models that tend to favour the majority (non-fraudulent) class and potentially overlook actual fraudulent activity. 'Branco' et al. (2015) highlights how this class imbalance can hugely impact model performance when it comes to predictability. He states, “Learning from imbalanced data is a challenging task since most standard learning algorithms assume a relatively balanced class distribution and equal misclassification costs. This often leads to poor predictive performance for the minority class" (Branco, 2015). Consequently, researchers must specifically target this issue by developing various techniques, for example, under-sampling the majority class and over-sampling the minority class. These techniques attempt to improve the model's predictability performance without significantly compromising overall accuracy. Finding the balance remains a central challenge in the development of reliable fraud detection systems.

In addition, another key challenge found in the literature was the lack of transparency and interpretability in many models. Differing models experience different levels of accuracy, notably, neural networks achieving high accuracy. However, the high accuracy comes at a cost, as neural networks are often criticised as “black boxes,” since they tend to be extremely complex. 'Baesens' et al. (2020) talk of possible solutions by introducing the technique robROSE, a robust sampling method that balances the dataset, ensuring that important properties, model transparency, and robustness are preserved.

Lastly, another huge issue complicating the investigation into credit card fraud is the anonymisation of the data. Since many public datasets— such as the one in this essay— use

PCA-transformed features (V1-V28). ‘Whitrow et al.’ (2009) state these features can make understanding and consequently interpreting the underlying patterns, behaviors, or real-world meanings behind fraudulent transactions extremely difficult, limiting the ability to draw actionable insights. However, Bahnsen’ counters this by demonstrating that well-designed feature engineering can boost a model’s performance, even when the data is limited.

These challenges form the foundation for the modelling strategies explored in this project and justify the decisions made later in the methodology.

Handling Class Imbalance in Fraud Detection

A recurring challenge in the literature on credit card fraud detection is class imbalance; as previously mentioned in the introduction, it can be a critical issue. Our dataset, alongside a plethora of other datasets, has a disproportionate representation. Consequently, this imbalance skews the learning process by impacting model performance, causing many models to default toward predicting the majority class, being non-fraudulent transactions. Branco et al. (2015) highlights, “Learning from imbalanced data is a challenging task since most standard learning algorithms assume a relatively balanced class distribution and equal misclassification costs.”

This quote essentially means that standard algorithms struggle to generalise results from fraud detection for minority cases, which will result in poor fraud detection. But the literature provided resolutions to this problem: techniques like undersampling, oversampling, and ensemble methods. Dal Pozzolo et al. (2017), for example, note that while under-sampling helps mitigate bias, it risks discarding valuable non-fraudulent data. Although some of the literature contradicts this, for example, Baesens et al. (2020) introduce *robROSE*, a synthetic sampling method designed to maintain dataset structure while improving class representation.

The literature brings to light some useful sampling techniques to effectively counteract fraud detection and improve model fairness, including under-sampling the majority class, over-sampling the minority class, and combining both approaches within ensemble frameworks to achieve better balance and performance in predictive modelling.

Feature Engineering and Model Design

The existing literature revealed that feature engineering plays a crucial role in the performance of fraud detection models. As previously mentioned, the anonymisation of real-world data (like the Kaggle credit card data used in our study) can cause detrimental consequences. A large portion of the dataset’s features have been transformed to protect the sensitive information they represent, but this only makes meaningful interpretation more difficult. The lack of transparency hinders not only the model's ability to explain but also its ability in automated decision-making.

Bahnsen et al. (2016) stresses the importance of domain knowledge when improving feature representation. Bahnsen states, “It is very important how to extract the right features from the transactional data [which is done by] aggregating the transactions in order to observe the spending behavioral patterns of the customers." By incorporating such domain-specific insights—like transaction timing, frequency, and merchant type— models can better distinguish legitimate and fraudulent behavior. However, this is clearly not an easy task when working with anonymised data. Dal Pozzolo addresses this challenge by advocating for realistic modelling strategies that compensate for the lack of interpretability. 'Dal Pozzolo' recommends robust data processing techniques and temporal validation to ensure models remain generalisable and effective. As they explain, "the design of evaluation strategies must take into account the temporal nature of the data in order to avoid information leakage and overly optimistic results."

Evaluating Model Performance in Imbalanced Contexts

In imbalanced datasets like those used in fraud detection, accuracy can often be low, but there are some studies that highlight the strengths of models. Precision, recall**,** F1-score, and AUC better reflect a model's ability to detect rare fraud cases without producing excessive false alarms.

Whitrow et al. (2009) stresses the importance of "matching the measure to the true objectives," noting that algorithms like neural networks may achieve higher recall but lower interpretability. Logistic regression, while simpler, may miss subtle patterns.

In addition, Dal Pozzolo' et al. emphasizes the importance of needing to minimise false positives while still capturing fraud. He states "that 'precision, recall, F-measure and AUC should be considered simultaneously to capture the trade-off between the number of correctly detected frauds and the number of false alarms.'"

3. Research Question

The main goal of the project is to use a real-world, imbalanced dataset to assess whether machine learning algorithms can accurately identify fraudulent credit card transactions. Additionally, we aim to assess and compare different models to identify which best fits for fraud detection while minimising false positives and negatives.

This investigation follows two key research questions:

- Can predictive models accurately distinguish fraudulent from legitimate transactions in the presence of severe class imbalance?
- Which model performs best in terms of detecting fraud while maintaining a low rate of false positives and false negatives?

These questions are necessary to address the real-world demands of fraud detection systems. The first question investigates how well machine learning can detect rare fraudulent activity in a large dataset of legitimate transactions. A key component of this work is detecting if predictive models remain capable of identifying significant patterns in the dataset given their imbalance and anonymous features.

For the second question, the focus is on comparing different machine learning models. Not every algorithm works the same way or yields the same kinds of information. Based on a linear combination of input characteristics, the straightforward and understandable model known as logistic regression calculates the likelihood of fraud. Although it is helpful for finding general, linear links in the data and produces results that are simple to understand, it can find it difficult to uncover intricate or non-linear patterns that are frequently linked to fraudulent activity.

Classification trees, on the other hand, divide the dataset into branches according to certain feature thresholds, which facilitates the identification of variable interactions and decision rules. Although these trees are easy to understand and helpful in detecting non-linear interactions, improper adjustment can cause them to become unstable or overfit.

Conversely, neural networks are incredibly adaptable models that can discover complex, non-linear patterns in vast amounts of data. They are particularly effective at spotting tiny signs of fraud that more basic algorithms can overlook. They are less transparent and more difficult to authenticate in regulated businesses like banking, though, because of their complexity, which comes at the expense of interpretability and increased processing needs.

The dataset that is being used in this study, the Kaggle Credit Card Fraud Detection dataset, has an anonymous feature which presents extra difficulties. For the protection of confidential information, it has 28 numerical characteristics (V1-V28) produced using Principal Component Analysis (PCA). Two other features are included with these: ‘Time’, which shows how many seconds have passed since the initial transaction and ‘Amount’, of each transaction. The target variable, ‘Class’ indicates whether a transaction is legitimate (0) or fraudulent (1). The dataset still has enough structure to look at predicted trends, even though the anonymisation of characteristics limits interpretability.

By answering these questions, we will be able to provide valuable insight into the strengths and weaknesses of different predictive modelling techniques for managing actual financial data. This study seeks to further the continuous development of effective and useful fraud detection systems. The knowledge gathered from this study can help financial institutions choose appropriate machine learning methods, direct future research aimed at enhancing fraud detection in anonymised or class-imbalanced datasets and advise software developers.

4. Exploratory Data Analysis

The first step to a good exploratory data analysis is to prepare the data, the first step in this case was to check if there were any missing values and remove them. When we first loaded the data into R studio, we had 284807 observations and once we performed “na.omit(df)“ we still had the same number of observations indicating that there were no missing values in the dataset.

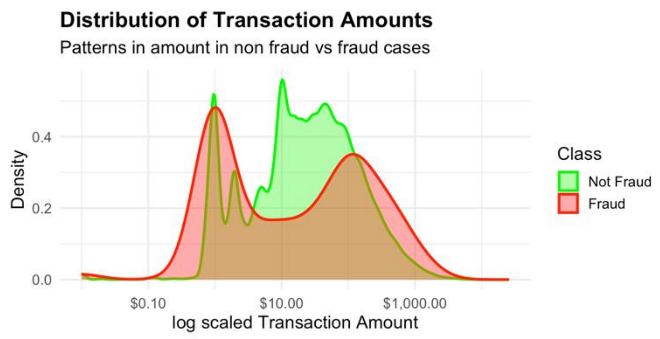

Figure 1: A Density Plot. This density plot displays the distribution of transaction amounts for fraud and non – fraud cases, with a log scale applied to the x – axis. The use of log scaling addresses the highly skewed nature of the transaction data, allowing for a clearer visual comparison across all amounts. The green curve represents non – fraudulent transactions, which are more spread across mid to higher amounts, while the red curve represents fraudulent transactions, which are more concentrated at lower amounts. This suggests that lower – value transactions may be more commonly associated with fraud in this dataset.

Two – Sample T – Test:

A two-sample t – test was conducted to determine whether there is a statistically significant difference in the mean transaction amounts for non – fraud and fraud cases.

Null Hypothesis (H_0): There is no statistical difference in the mean transaction amounts for non – fraud and fraud cases.

Table 1: Results for the two – sample t – test

T	DF	P – Value	95% CI	Mean (NF)	Mean (F)
-2.93	492.61	0.003561	-56.68, -11.16	88.30	122.21

The table above displays the results of the t – test. Since the p – value = 0.003561 <0.05, we can reject the null hypothesis. There is a statistical difference in the mean transaction amounts for the non-fraud and fraud cases. Fraudulent transactions have significantly larger amounts on average (122.21) than non – fraudulent transactions (88.30).

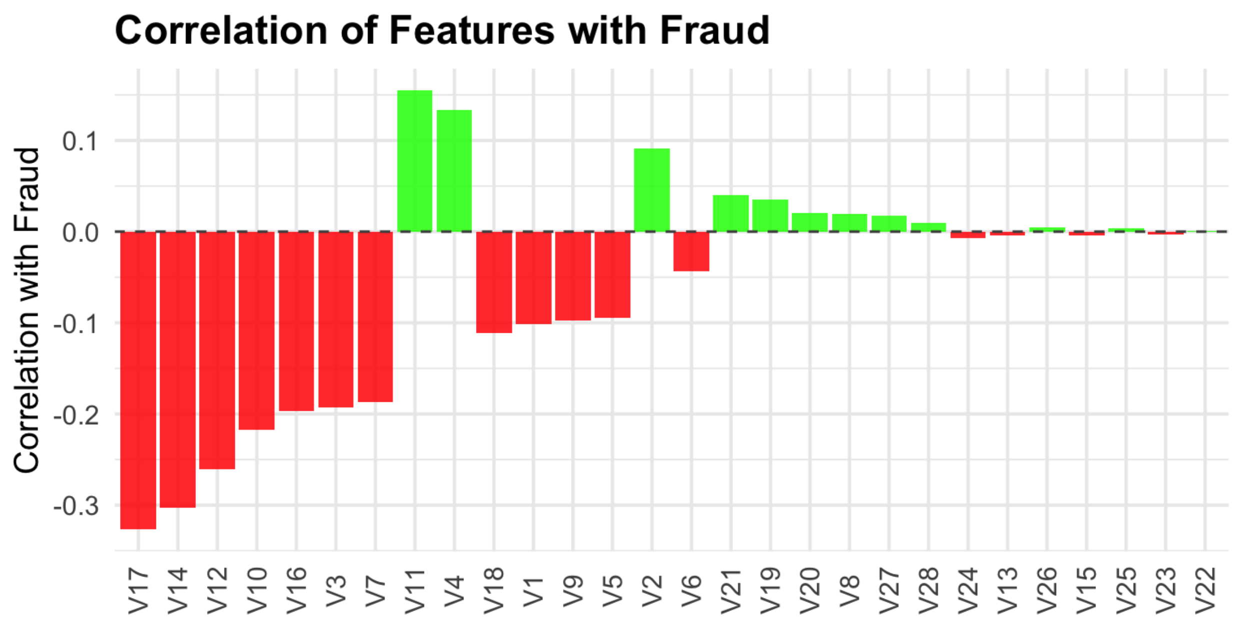

Figure 2: *This bar plot shows the correlation between various features. Features like V17, V14, V12, V10, V16, V3 and others (represented by the red bars) have a strong negative correlation with fraud. This suggests that higher values of these variables are associated with a decreased likelihood of fraud. Additionally, features like V11, V4, V2, V21 and others (represented by the green bars) show a positive correlation with fraud. This suggests higher values of these variables are associated with an increased likelihood of fraud. To conclude, variables with strong or negative correlations could be valuable predictors for fraud detection models.*

5. Method Overview

In this section I will be explaining how we will be making, analysing, and comparing the models. The first step would be splitting the data. We split the data into train data and test data with an 80/20 split. This was because the dataset was quite large so we could afford to have a smaller testing sample and it still be good enough to draw conclusions. After this initial split we used the train data to create a new set of data which is the under sampled train data, this was a much smaller set of data as it was made up from all the fraud cases in the train data and for each one of these cases just one non fraud set and this created a set of data with an even split of non-fraud and fraudulent datasets. The reason this dataset was used was to see if under sampling would lead to more accurate results since it won't overly rely on the fact that fraud is rare so the model wouldn't predict fraud as much. This is important to us since we aren't just looking for the most accurate model, but we are also looking for models that overpredict fraud rather than under predict it. The reason for this is to minimise the amount of fraud slipping past our model.

The first models we will be creating we will be making are logistic regression models both with our normal train data and our under sampled train data. We will then make reduced versions of both these models which only include the significant features, this is important as we do want models that don't predict fraud based on insignificant features. We will not be going into using interactions since most of the features are anonymised which means we won't be able to use any of our own knowledge to decide what features to check interactions between.

The next classification tree model and neural network are much simpler, since they pick which features and interactions to use automatically, so all we need to do for these models is train them with our train data and under sampled train data.

After the models are made, we will do some exploratory plots including Residual plots, Precision Recall Curve plots for our models trained with the normal train data since it is imbalanced, ROC curves for the models that were trained with the under sampled data and feature importance plots for our classification tree and neural network models. These plots will be used to make initial impressions of how well each model performs. The final type of plot we will use is a confusion matrix for each one of these models to see exactly how it performs against our test data and then we will be selecting which models we think are best at preventing fraud.

6. Modelling

In this section of the report, the creation of six predictive models will be explained and analysed.

Model 1: Logistic Regression Model based on Train Data

The first model created was the logistic regression model. A logistic regression estimates the probability of an event happening, such as fraud or not fraud, based on a set of independent explanatory variables (IBM, 2021). To build the logistic regression model, the dataset was firstly split into training and testing sets using an 80/20 ratio. It was fitted using all available variables, resulting in an AIC of 1771.3. A final refined model was then created by using only statistically significant predictors. The statistically significant predictors were found to be V4, V8, V13, V14, V20, V21, V22, V27 and V28. Although the refined final model's AIC slightly increased to 1791.2, this trade – off was considered acceptable, as the final model incorporates solely meaningful predictors, enhancing its interpretability and avoiding potential overfitting from insignificant variables.

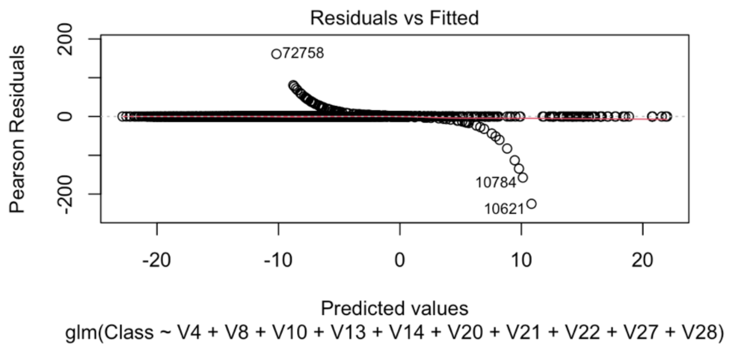

Figure 3: *This residual plot displays the Pearson residuals versus fitted values for the logistic regression model. Most residuals cluster tightly around zero, indicating a reasonable model fit for most observations. However, there is a curve and some high leverage outliers, such as the points 72758, 10784, and 10621, which means the model is making errors on some points that don't follow the pattern.*

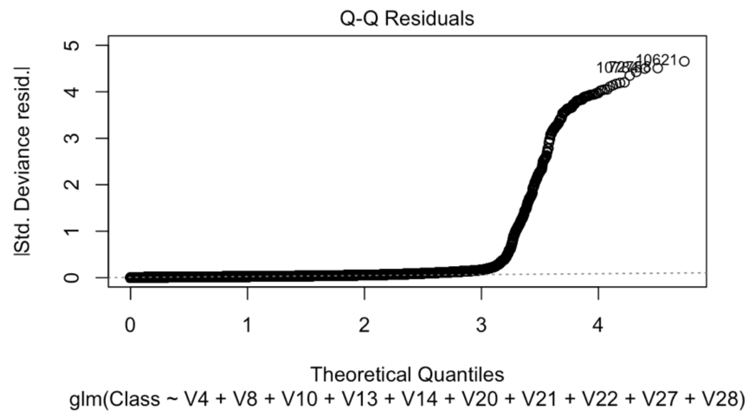

Figure 4: *This Q-Q plot shows that most residuals are close to zero, causing the line to stay flat until around quantile 3. After that, it rises, which means that the model's errors were much larger than expected.*

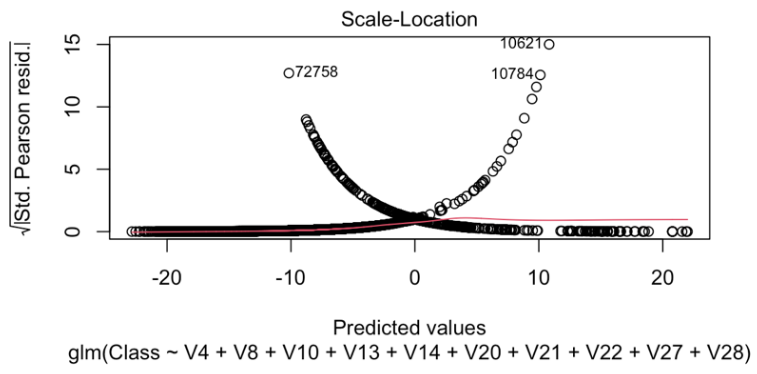

Figure 5: *This scale-location plot is meant to show us if the errors are spread out evenly. The ideal would error points plotted flat and randomly. This plot shows us that they curve a lot and there are a few outliers as well. This indicates that the model may suffer from outliers affecting the predictions and a changing error size.*

To conclude, based on the residual diagnostics, the logistic regression model demonstrates a reasonable fit for the majority of observations, as seen by the clustering of residuals near zero. However, the curvature in the residual plot, uneven spread in the scale – location plot, and deviations from the Q – Q line suggest potential issues with model assumptions. Particularly, the model seems to struggle with high – leverage outliers and shows that the size of the prediction errors changes across observations. These results suggest the model performs averagely overall due to its predictive accuracy being compromised for certain observations.

Model 2: Logistic Regression Model based on Under Sampled Train Data

Additionally, an under sampled test set was created, ensuring an equal number of fraudulent and non – fraudulent transactions to address the class imbalance in the dataset during analysis (780 instances total). This smaller dataset was then used to build the second logistic regression model. Like the first model, it was first fit with all available features, resulting in an AIC of

262.18. A final refined version of the model was then created using only the most statistically significant predictors. These statistically significant predictors were found to be V4, V5, V8, V10, V11 and V20. Although the refined final model's AIC slightly increased to 262.18, this trade – off was considered acceptable, as the final model incorporates solely meaningful predictors, enhancing its interpretability and avoiding potential overfitting from insignificant variables.

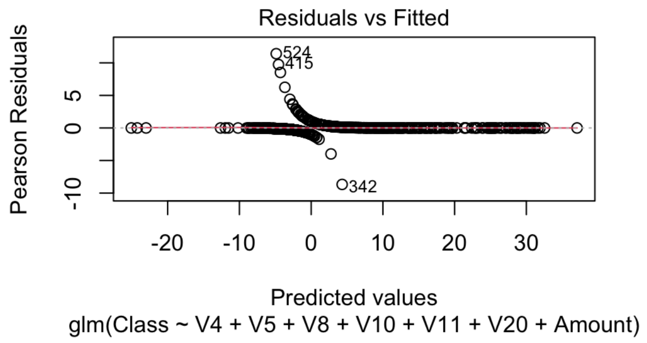

Figure 6. *Residuals vs Fitted Plot for Model 2 (Under Sampled Logistic Regression). This plot displays the Pearson residuals against the fitted values for the logistic regression model trained on the under sampled dataset. Most residuals are clustered close to zero, suggesting reasonable model fit overall. However, the curvature and a few high – residual outliers (e.g., observations 524, 415, and 342) suggests potential problems with non – linearity and points that the model struggles to predict accurately.*

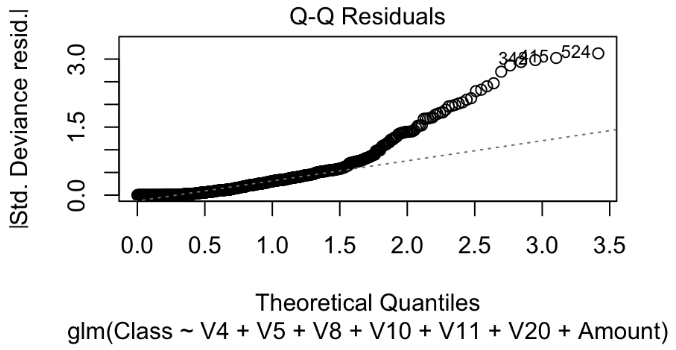

Figure 7. *Q – Q Plot of Residuals for Model 2 (Under Sampled Logistic Regression). This Q – Q plot compares the standarised deviance residuals of the model to a theoretical normal distribution. Although most points lie close to the reference line, suggesting normality of residuals, there is an upward deviation in the upper tail. This implies the model slightly underestimates higher residuals and may struggle to fully capture the distribution of all values, specifically extreme values.*

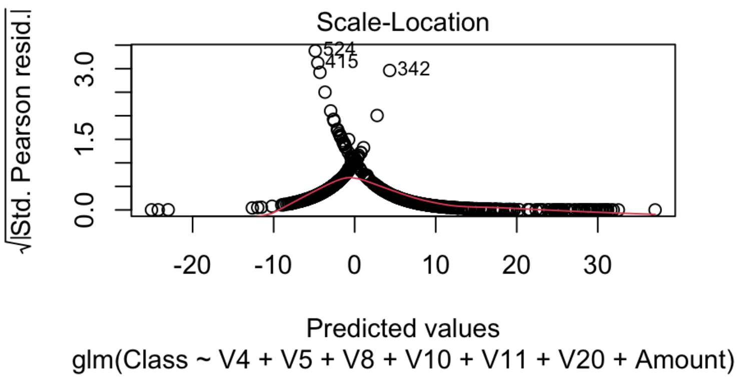

Figure 8. *Scale – Location Plot for Model 2 (Under Sampled Logistic Regression). This plot evaluates the spread of residuals across fitted values. Preferably, the points should be randomly dispersed with a horizontal trend, indicating constant variance. However, this plot reveals a distinct curve and a concentration of high values at the center, suggesting non – constant variance and the presence of outliers (e.g., observations 524, 415, 342). This implies the model's prediction error varies across observations, which could affect the model's accuracy and reliability.*

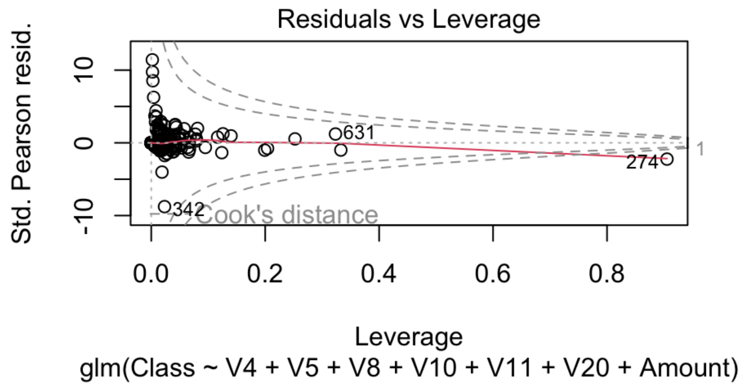

Figure 9. *Residuals vs Leverage Plot for Model 2 (Under Sampled Logistic Regression). This plot highlights the influence of individual observations on the logistic regression model. Most points have low leverage and small residuals, which is preferred. However, a few points (e.g., 274 and 631) have high leverage and fall near or outside Cook's distance lines, suggesting they have a strong influence on the model's estimates. This implies the presence of potentially problematic observations that may disproportionately affect model performance.*

To conclude, while model 2 performs averagely well on the under sampled data, and addresses class imbalance, the residual diagnostics imply that prediction errors are not equally distributed, and the model is sensitive to outliers.

Model 3: Classification Tree Model based on Train Data

Model 3 was built using a classification tree trained on the original, imbalanced training dataset. A classification tree is a type of learning model that splits the data into branches based on decision rules learned from the features. At each split, the algorithm selects the variable and values that best separate the data into fraud and non – fraud cases and the continuation of this process results in a tree structure (IBM, 2021). An advantage of classification trees over logistic regression is they're better suited to dealing with complex and non – linear relationships. Furthermore, another advantage of classification trees is they can better handle outliers because the data is split based on feature values. Additionally, another advantage of classification trees is they can better handle large and high – dimensional data. This is due to the partitioning which decreases the dimensionality of the data, making it easier to establish patterns and make predictions based on those patterns. It's ability to do this is further explained by the fact that classification trees select only the most important variables to split the data – they automatically do this whilst this is manually done in logistic regression (Willig, 2023).

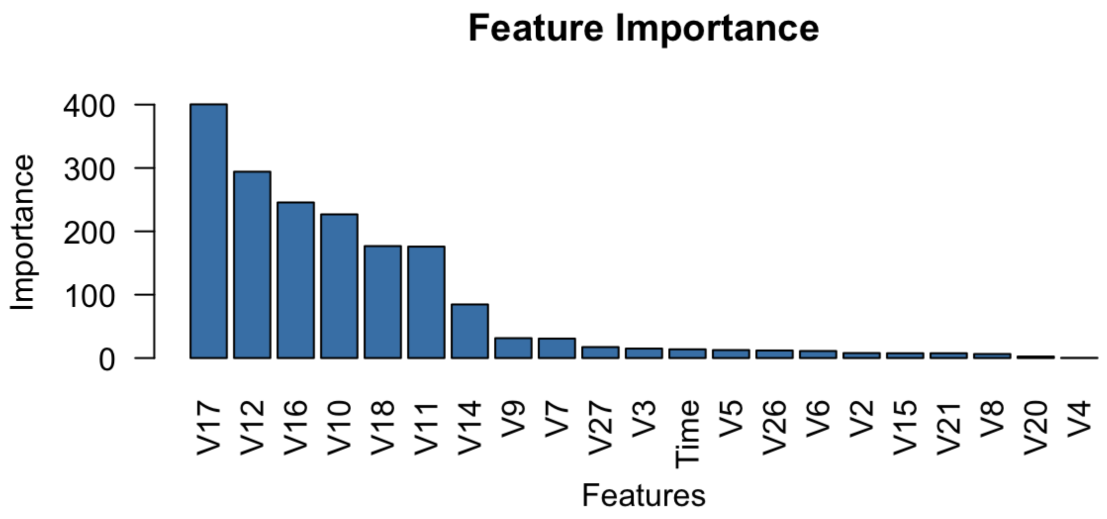

Figure 10. *Feature Importance Plot. This feature importance plot shows V17 is the most influential variable, followed by V12, V16, and V10, suggesting these features played a significant role in classifying transactions as fraudulent or not. The more we go towards the right, we find much less influential features such as V4, V20 and V8. The classification tree's automatic feature selection aligns with the correlation analysis discussed earlier on in the report. Both highlight that some of the most influential features in detecting fraud are V17, V12 and V10.*

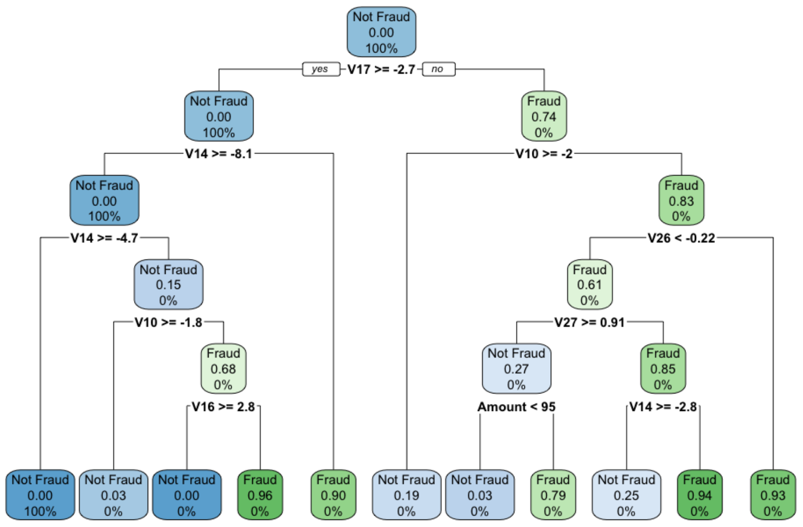

Figure 11. *Classification Tree Model Plot. This classification tree visually represents how the model makes fraud and non – fraud predictions based on feature thresholds. It further confirms V17 is the most significant feature in establishing fraudulent transactions, as suggested by the feature importance plot and correlation analysis. It also shows the conditions under which fraudulent and non – fraudulent transactions are classified.*

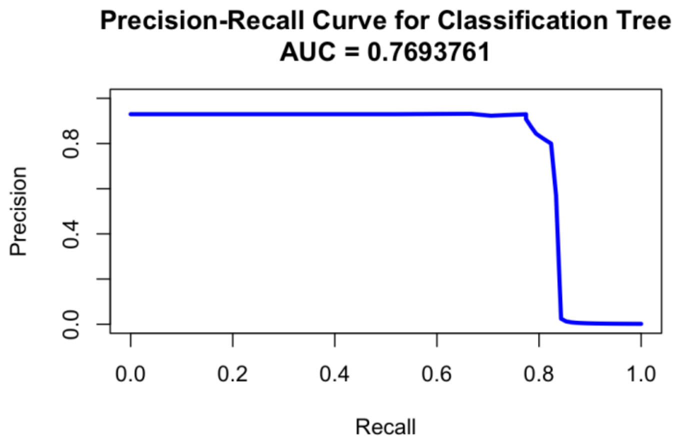

Figure 12. *Precision Recall Plot. This precision recall plot is used to evaluate the performance of models that are trained on imbalanced datasets. The AUC (area under the curve) is 0.769 which is quite good since a random model would get a value close to the percentage of fraudulent payments in the dataset, which in our case is 0.17% of the dataset. This suggests a random model would get an AUC of around 0.0017. This model does significantly better than this, suggests that our model is not random. The plot itself shows how the model performs; at lower recall values the model identifies few payments as fraud and does it with high precision, meaning its very confident that these are not fraud. Past 80% recall the precision drops to 0 meaning that there are many false positives in this model.*

To conclude, based on the feature importance, classification tree structure and precision – recall performance, the classification tree model seems to be an averagely effective model for fraud detection. It identifies many key influential variables like V17 and V14 and achieves a good AUC score of 0.769. However, the drop in precision at high recall levels suggests the model produces many false positives when trying to identify fraud causes. Overall, it performs decently, especially given the class imbalance in the dataset, but it can't be relied upon in high recall scenarios.

Model 4: Classification Tree Model based on Under Sampled Train Data

For the fourth model, we used the under sampled train data to build the second classification tree model. The use of the under sampled dataset rectifies the class imbalance allowing the tree to learn specific patterns to fraudulent transactions.

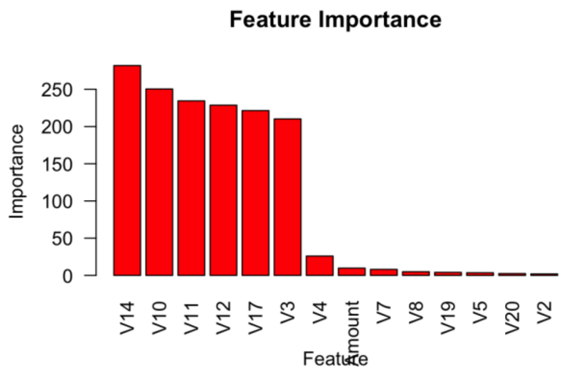

Figure 13. *Feature Importance Plot. This feature importance plot shows V14 is the most influential variable followed by V10, V11, V12 and V17. In contrast, V17 was the most influential variable in the first model. However, V10, V12 and V14 are top contributors in both models suggesting strong predictive power.*

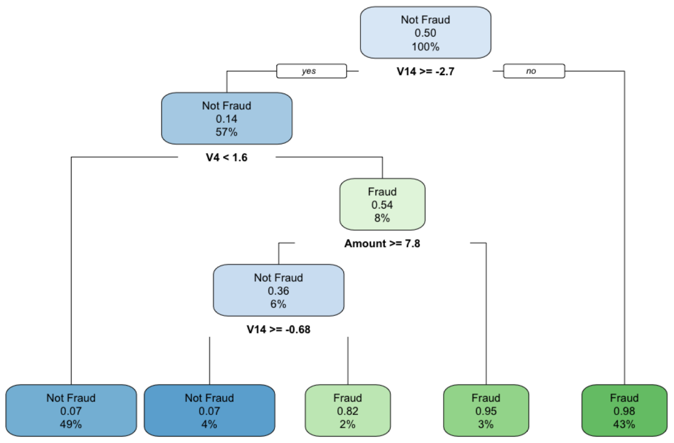

Figure 14. *Classification Tree for Under Sampled Logistic Classification Model. As suggested by the feature importance plot, the classification tree further confirms V14 is the most influential variable distinguishing fraudulent transactions. In contrast, as briefly discussed, the first feature importance plot and the first classification tree model suggested V17 as the most influential variable. Furthermore, the first tree heavily used V17, V12, V10 etc. whereas this tree uses V14, V4 and Amount. Additionally, the under sampled classification tree shows more balanced fraud detection paths, suggesting a more advanced ability to establish fraud due to the class balance. To summarise, the second tree is finetuned and is better at detecting fraud accurately due to fixing the class imbalance.*

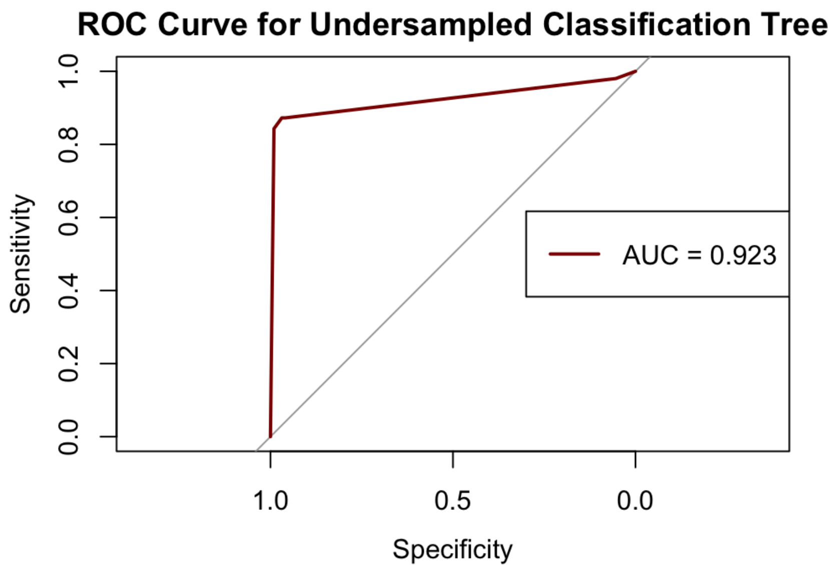

Figure 15. *This is an ROC plot for the under sampled classification tree model, the sensitivity represents the proportion of actual fraud cases that were correctly identified, and the specificity represents the proportion of non-fraud cases of false positives. The area under the curve is 0.923 which means that the model has a 92.3% chance of saying that a fraud case is more likely to be fraud than not fraud.*

To conclude, the second classification tree model based on the under sampled train data performs better than the other classification tree model. It achieves a high AUC of 0.923, suggesting it excellently distinguishes between fraud and non – fraud cases. The improvement in AUC is a benefit of the under – sampling, which helped address the class imbalance present in the previous model. Additionally, although key variables differ slightly from the first model, consistent top variables like V10, V12, and V14 reaffirm its reliability. Overall, this model strongly performs in identifying fraudulent transactions.

Model 5: Neural Network using Train Data

For the fifth model, a neural network model was created using the nnet package in R, trained on the original training dataset. Neural networks are inspired by the human brain's structure and are powerful machine learning models (IBM, 2021). An advantage of neural networks is less statistical training is needed. Another advantage is they have the ability to detect complicated nonlinear relationships and all possible interactions between different variables (Tu, 1996). This makes them extremely effective when it comes to detecting fraud patterns. Neural networks do this by processing all input features through weighted connections – higher weights suggest a stronger influencer on the final prediction and their selection reduces prediction error. This process inevitably minimises the influence of irrelevant features and focuses on the more significant features that affect the model's predictions (IBM, 2021).

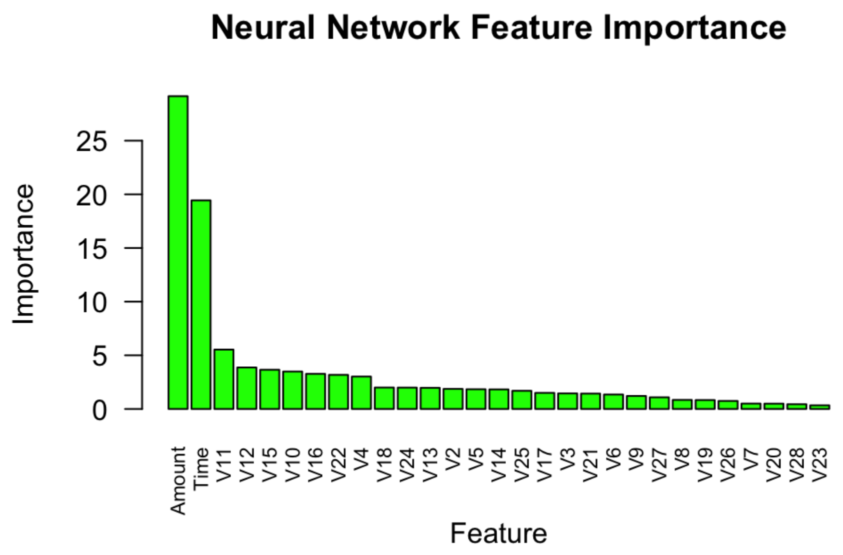

Figure 16. *Neural Network Importance Plot. In comparison to the previous feature importance plots, this feature importance plot places a larger importance on "Amount" and suggests it is the most influential variable. This result aligns with the results of the two – sample t – test, which found a statistically significant difference in the mean transaction amounts between fraudulent and non – fraudulent payments. This supports the idea that transaction amount is an important factor in fraud detection.*

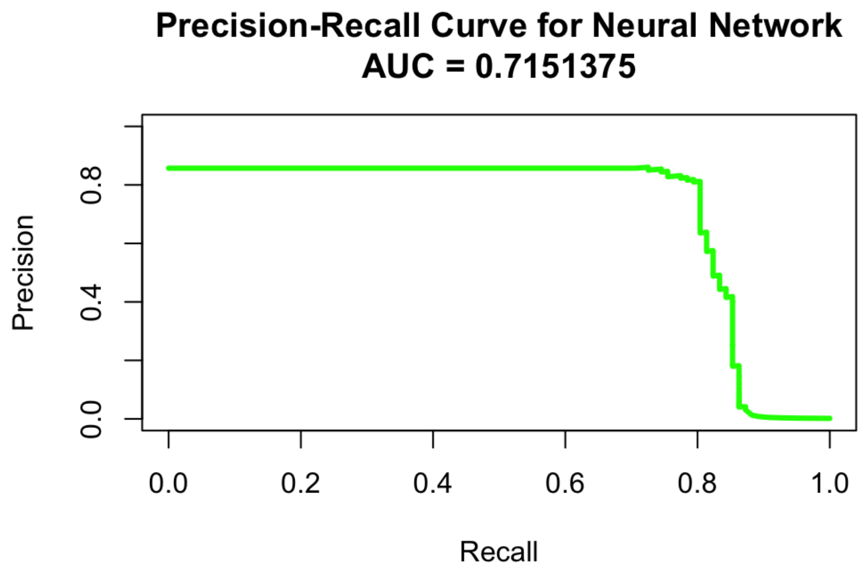

Figure 17. *This is another precision recall curve plot that was chosen for this model since it is good at examining imbalanced datasets and since this neural network is trained on the normal train data, it is a good fit. This is very similar to the classification tree precision recall plot. It has a smaller AUC (0.72) which may indicate that the model isn't as accurate as the classification tree (AUC of 0.77.) The plot shows how similar it is to the classification tree because like the classification tree we see that at lower recall values the model doesn't predict many values as fraud, and it does it confidently. Past 80% recall the precision drops to 0 meaning that there are once again many false positives in this model.*

To conclude, the neural network model performs averagely when detecting fraud. Although it effectively identifies transaction amount as a key predictor – supporting the t – test findings – it performs slightly worse than the previous model, with a lower AUC of 0.72. Furthermore, the precision – recall plot suggests the model performs well with low recall but struggles with false positives as recall increases. Overall, it is a reasonable model but slightly less accurate than the classification tree.

Model 6: Neural Network using Under Sampled Data

For the sixth model, a neural network model was created using the nnet package in R, trained on the under – sampled dataset.

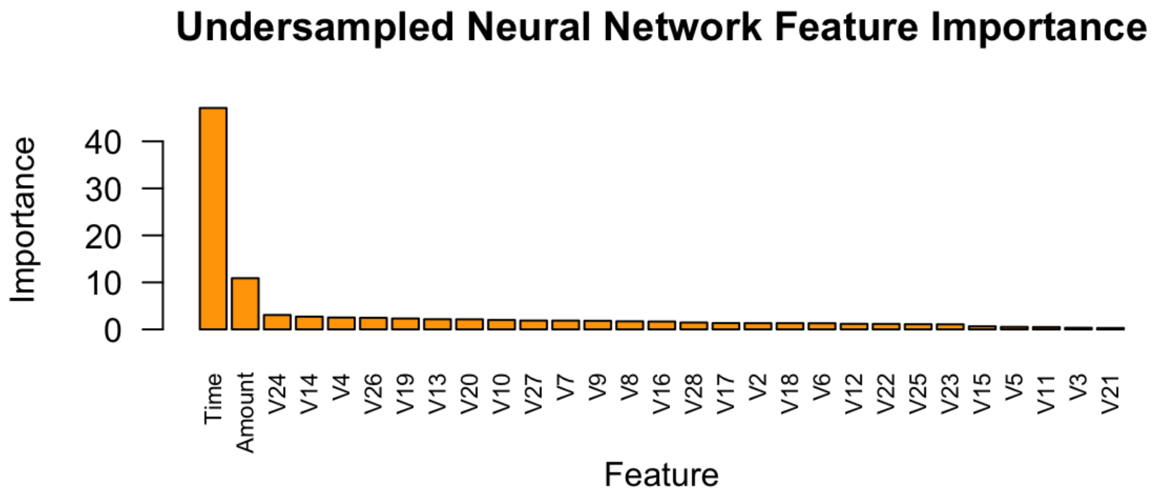

Figure 18: Neural Network Importance Plot for the Under Sampled Train Data. In contrast to the previous neural network feature importance plot, this one places an overwhelmingly large importance on "Time" which is distantly followed by "Amount". All other features (like V17, V14, V12 and V10) have very low importance despite being highlighted as significant predictors in earlier models. This is perhaps due to the under sampling as a smaller balanced dataset was trained on highlighting different patterns.

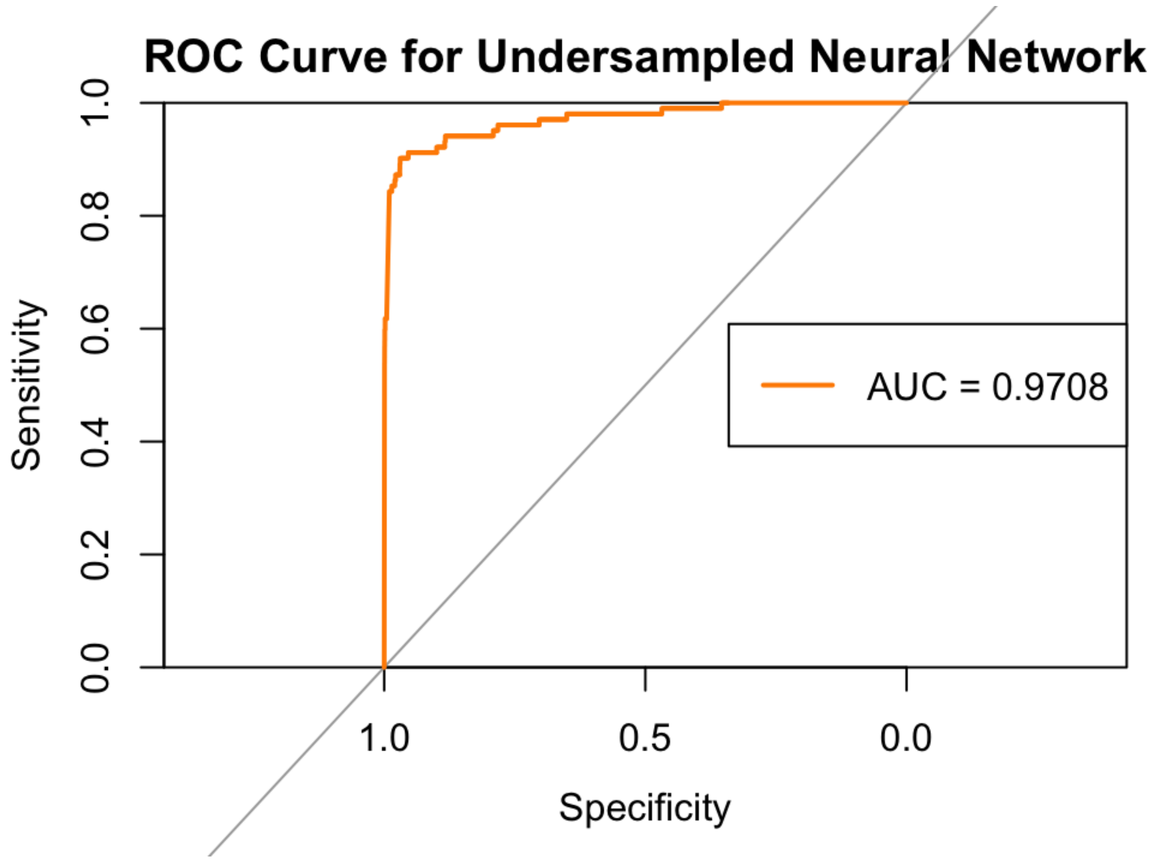

Figure 19: The roc curve was used here because this model has been trained on the under sampled train data. This means that the ROC plot is a good indicator of performance. The area under the curve is 0.9708 which means that the model has a 97.08 chance of ranking a fraudulent case more likely to be fraud than a non-fraud case. The curve is at the top left corner which may indicate that the model has high sensitivity. This means that the model is likely to deliver a lot of true positives whilst having low false positives.

To conclude, the under – sampled neural network model demonstrates strong predictive performance, with a high AUC of 0.97 suggesting an excellent ability to distinguish between fraud and non – fraud cases. Although it relies heavily on “Time”, the model’s low false positive rate suggests it is a highly effective model, specifically for detecting fraud in a balanced dataset.

7. Results:

This study developed six machine learning models to predict fraudulent credit card transactions. The performance of the six models created were assessed using residual diagnostics (for regression models), feature importance plots, ROC curves, precision-recall curves, and confusion matrices (shown in **Figure 4)**. The confusion matrices show how many transactions each model correctly classified as either non-fraudulent or fraudulent, and the number of misclassified transactions. Area Under the Curve (AUC) values were used to quantitatively compare the model performances across all approaches.

Logistic Regression (Model 1: Imbalanced)

The first model shown on top left of **Figure 4** is the Logistic Regression model; this model was trained on the full imbalanced dataset, and showed an AUC of 0.77. The residual plots indicated a reasonable model fit overall, however there were signs of high-leverage outliers and non-constant variance. The confusion matrices confirmed the poor fraud detection, it correctly identified 56,848 non-fraudulent transactions but only 12 fraudulent ones, the model misclassified 62 fraud cases and incorrectly flagged 40 genuine transactions as fraud. The model had low false positive rate but the results showcase the models poor sensitivity as it significantly underestimated fraudulent transactions, making this model unsuitable for real-world fraud detection.

Logistic Regression (Model 2: Under-sampled)

The top right of **figure 4** is the Under-sampled Logistic Regression model which has much better performance with an AUC of 0.88. The residual diagnostics suggested better fit however the outlier still impacted the model's performance. This model identified 91 out of 102 fraud cases, with 11 false negatives. However, this model misclassified 61,673 non-fraud cases as fraudulent increasing the false positives 1,673 false positives. This caused the model to be less precise, but this is more favorable for fraud detection, where missing fraud cases is far riskier than flagging legitimate ones.

Classification Tree (Model 3: Imbalanced)

The middle left of **figure 4** is the Classification Tree model this model was trained on imbalanced data and produced an AUC of 0.769. It showed clear decision rules and identified influential feature like V17 and V12. The model detected only 6 fraud cases and missed 79, while correctly classifying 56,854 non-fraud cases. This model has poor sensitivity like the logistic model on imbalanced data, it was biased towards the majority class.

Classification Tree (Model 4: Under-sampled)

The second Classification Tree trained on balanced data on the middle right of **figure 4** performed much better with an AUC of 0.923. The model identifying 89 fraudulent cases missing only 13 and producing 2,286 false positive. The tree used different splits than its predecessor, with V14 emerging as the top predictor. The confusion matrix showed that the model has balanced performance, lower false negatives indicating better learning from fraud patterns.

Neural Network (Model 5: Imbalanced)

The Neural Network model was trained on the original dataset shown on the bottom left of F**igure 4** had an AUC value of **0.72**, it identified “Amount” as an important feature aligning with the t-test findings. The performance was limited detecting only 25 frauds while missing 80, the model struggled at higher recall levels with a rise of false positives.

Neural Network (Model 6: Under-sampled)

This model has the highest AUC value of 0.9708, indicating excellent differentiation between fraud and non-fraud cases. The model identified 94 out of 102 fraudulent cases with 8 false positives and 3,624 false positives. Although “Time” dominated the feature importance plot suggesting a shift in what the model learns from under-sampled data. The model had the highest AUC value of 0.9708 and presented the best precision-recall balance among all six models it has high precision, sensitivity, and ROC results.

Model	**AUC**
Logistic Regression (Imbalanced)	0.77
Logistic Regression (Under-sampled)	0.88
Classification Tree (Imbalanced)	0.769
Classification Tree (Under-sampled)	0.923
Neural Network (Imbalanced)	0.72
Neural Network (Under-sampled)	0.9708

Table 2*. The table shows a summary of Area Under the Curve values.*

The AUC values show a pattern, the models trained on under-sampled data consistently outperformed the models trained on imbalanced data. The under-sampled neural network achieved the best result in both AUC and confusion matrix performance, making it the strongest model overall.

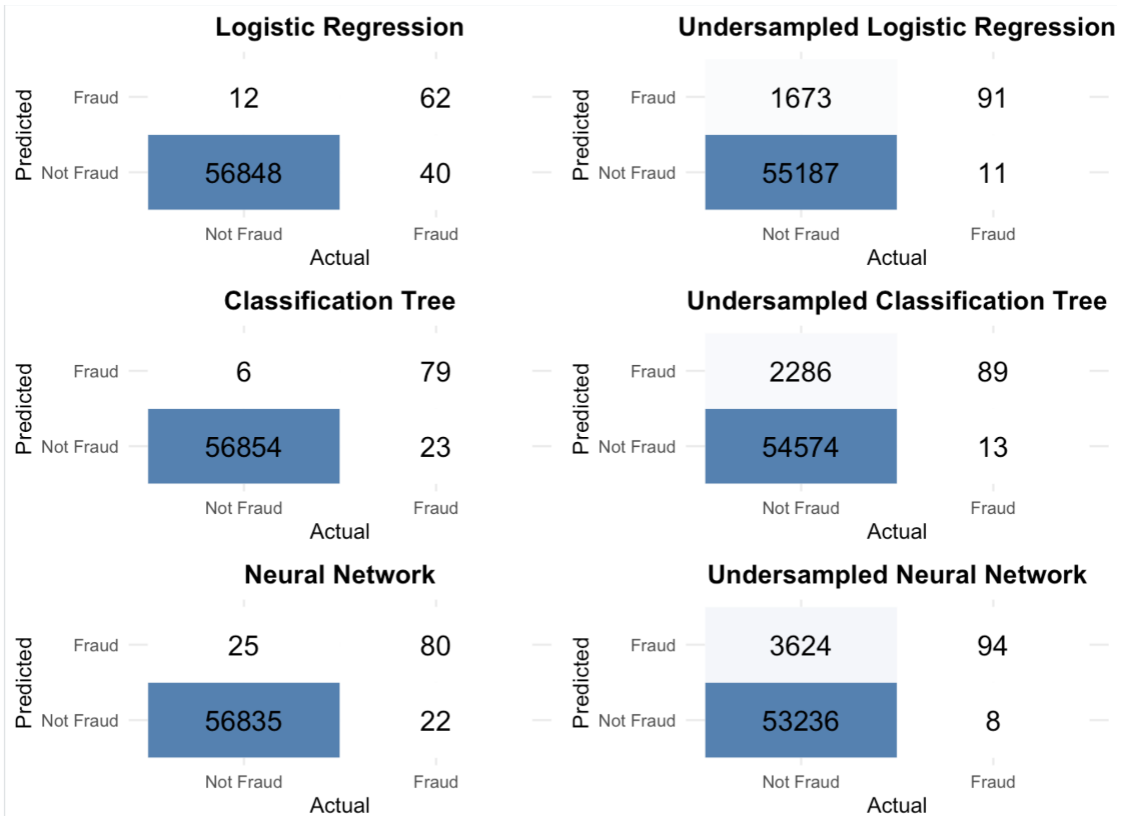

Figure 20. *Confusion matrices for each model (Logistic Regression, Classification Tree, and Neural Network) trained on both imbalanced and under-sampled datasets.*

8. Conclusion

This project set out to evaluate the effectiveness of machine learning models in detecting fraudulent credit card transactions. The results highlight the critical importance of addressing class imbalance. Across all models, those trained on under-sampled data outperformed their imbalanced counterparts, confirming that balancing the dataset significantly increased model sensitivity and reduces false negatives.

The under sampled neural network emerged as the best-performing model, with an AUC of 0.9708, correctly detecting 94 out of 102 fraud cases. It effectively balanced sensitivity and specificity, providing a practical solution for fraud detection systems. The under-sampled classification tree followed closely, offering a slightly more interpretable model with strong performance (AUC = 0.923). It was also the most accurate at avoiding false negatives which is evident as it only predicted not fraud when it was fraud only 8 times which was the least out of all the models.

The logistic regression models were useful baselines, their linear structure and sensitivity to outliers limited their effectiveness, especially on imbalanced data. The under-sampled logistic regression showed improvement but still the flexible tree-based and neural network approaches.

The finding reinforced the principle that in fraud detection it is better to overestimate fraud / accepting some false positives than to miss it. The under-sampled tree and neural network aligned with this as they both priortised sensitivity.

For future work, it would be beneficial to explore techniques like SMOTE for synthetic oversampling, ensemble models like Random Forest. Also, working with less anonymized and more recently collected data to improve the interpretability and relevance of the study. This study shows that with appropriate preprocessing and model selection, machine learning can be a powerful effective tool for combating financial fraud.

9. References

- Martin, G. (2022). *Fraud Detection using Machine Learning and AI*. [online] Experian UK. Available at: https://www.experian.co.uk/blogs/latest-thinking/guide/machine-learning-ai-fraud-detection/.

- UK Finance (2024). *Over £570 million stolen by fraudsters in the first half of 2024*. [online] UK Finance. Available at: https://www.ukfinance.org.uk/news-and-insight/press-release/over-ps570-million-stolen-fraudsters-in-first-half-2024.

- Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., & Bontempi, G. (2017). Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy. *IEEE Transactions on Neural Networks and Learning Systems*, 29(8), 3784–3797.

- Bahnsen, A. C., Aouada, D., Stojanovic, A., & Ottersten, B. (2016). Feature engineering strategies for credit card fraud detection. *Expert Systems with Applications*, 51, 134–142

- Branco, P., Torgo, L., & Ribeiro, R. P. (2015). A Survey of Predictive Modelling under Imbalanced Distributions. *ACM Computing Surveys*, 49(2), Article 31.

- Baesens, B., Höppner, S., Ortner, I., & Verdonck, T. (2020). robROSE: A robust approach for dealing with imbalanced data in fraud detection. *Statistical Methods & Applications*, 30(3), 841–861

- Comparative Evaluation of Anomaly Detection Methods for Fraud Detection in Online Credit Card Payments

- Fraud Detection Using Optimized Machine Learning Tools Under Imbalance Classes [Google scholar to find]

- IBM (2021a). *Logistic Regression*. [online] Ibm.com. Available at: https://www.ibm.com/think/topics/logistic-regression [Accessed 20 Apr. 2025].
- IBM (2021b). *What Is a Decision Tree?* [online] Ibm.com. Available at: https://www.ibm.com/think/topics/decision-trees [Accessed 20 Apr. 2025].
- IBM (2021c). *What is a Neural Network?* [online] IBM. Available at: https://www.ibm.com/think/topics/neural-networks [Accessed 20 Apr. 2025].
- Tu, J.V. (1996). Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of Clinical Epidemiology*, [online] 49(11), pp.1225–1231. doi:https://doi.org/10.1016/s0895-4356(96)00002-9.
- Willig, G. (2023). *Decision Tree vs Logistic Regression*. [online] Medium. Available at: https://gustavwillig.medium.com/decision-tree-vs-logistic-regression-1a40c58307d0 [Accessed 20 Apr. 2020].

10. Appendix (R Code)

```
library(ggplot2)
library(scales)
library(dplyr)
library(tidyr)
library(rpart)
library(rpart.plot)
library(nnet)
library(PRROC)
library(pROC)
library(caret)
library(NeuralNetTools)

#Loading Data and Cleaning
df <- read.csv("creditcard.csv")
df <- na.omit(df)
df$Class <- factor(df$Class, labels = c("Not Fraud", "Fraud"))

#Distribution of Transaction Amounts Fraud vs No Fraud
ggplot(df, aes(x = Amount, fill = Class, color = Class)) +
  geom_density(alpha = 0.4, size = 1) +
  scale_x_continuous(labels = dollar_format(prefix = "$"), trans = "log10") +
  scale_fill_manual(values = c("Not Fraud" = "green", "Fraud" = "red")) +
  scale_color_manual(values = c("Not Fraud" = "green", "Fraud" = "red")) +
  labs(
    title = "Distribution of Transaction Amounts",
    subtitle = "Patterns in amount in non fraud vs fraud cases",
    x = "log scaled Transaction Amount",
    y = "Density",
    fill = "Class",
    color = "Class"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(face = "bold"),
```

```r
    plot.subtitle = element_text(margin = margin(b = 10))
  )

#Hypothesis Test
t.test(Amount ~ Class, data = df)
#used a welch's test due to the class imablance
#p value is less than 0.05 which means that we there is a statistically significant difference in the mean transaction
amount
#conclusion is that there is a higher mean in the fraud group compared ot non fraud.


#Correlation Plot
df$Class_num <- ifelse(df$Class == "Fraud", 1, 0)

correlations <- df %>%
  select(starts_with("V")) %>%
  summarise(across(everything(), ~ cor(.x, df$Class_num))) %>%  # First pipe ends here
  pivot_longer(cols = everything(), names_to = "Feature", values_to = "Correlation") %>%  # Removed extra ) here
  arrange(desc(abs(Correlation))) %>%
  mutate(Feature = factor(Feature, levels = Feature))

ggplot(correlations, aes(x = Feature, y = Correlation, fill = Correlation > 0)) +
  geom_col(show.legend = FALSE, alpha = 0.85) +
  scale_fill_manual(values = c("TRUE" = "green", "FALSE" = "red")) +
  geom_hline(yintercept = 0, color = "gray30", linetype = "dashed") +
  labs(
    title = "Correlation of Features with Fraud",
    x = NULL,
    y = "Correlation with Fraud"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    axis.text.x = element_text(angle = 90, vjust = 0.5),
    plot.title = element_text(face = "bold"),
    plot.subtitle = element_text(margin = margin(b = 10))
  )

#Splitting the Data
set.seed(42)

# 80% train 20% test
train_indices <- sample(1:nrow(df), 0.8 * nrow(df))
train_data <- df[train_indices, ]
test_data <- df[-train_indices, ]

#undersampling
fraud_cases <- sum(train_data$Class == "Fraud")
undersampled_train_data <- train_data %>%
  group_by(Class) %>%
  sample_n(size = fraud_cases) %>%
  ungroup()

#Logistic Regression
logistic_regression <- glm(Class ~ Time + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10 + V11 + V12 + V13 + V14 +
V15 + V16 + V17 + V18 + V19 + V20 + V21 + V22 + V23 + V24 + V25 + V26 + V27 + V28 + Amount, family =
"binomial", data = train_data)
summary(logistic_regression)
#AIC of 1771.3

reduced_log_regression <- glm(Class ~ V4 + V8 + V10 + V13 + V14 + V20 + V21 + V22 + V27 + V28, family =
"binomial", data = train_data)
summary(reduced_log_regression)
#1791.2 but this model only uses significant features meaning it is more useful
plot(reduced_log_regression)
```

```
#Undersampled Logistic Regression
undersampled_logistic_regression <- glm(Class ~ Time + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10 + V11 +
V12 + V13 + V14 + V15 + V16 + V17 + V18 + V19 + V20 + V21 + V22 + V23 + V24 + V25 + V26 + V27 + V28 +
Amount, family = "binomial", data = undersampled_train_data)
summary(undersampled_logistic_regression)

reduced_undersampled_logistic_regression <- glm(Class ~ V4 + V5 + V8 + V10 + V11 + V20 + Amount, family =
"binomial", data = undersampled_train_data)
summary(reduced_undersampled_logistic_regression)
# AIC 265.57
plot(reduced_undersampled_logistic_regression)

#Classification Tree

classification_tree <- rpart(Class ~ Time + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10 +
                               V11 + V12 + V13 + V14 + V15 + V16 + V17 + V18 + V19 + V20 +
                               V21 + V22 + V23 + V24 + V25 + V26 + V27 + V28 + Amount,
                             data = train_data,
                             method = "class",  # For classification
                             control = rpart.control(minsplit = 20, cp = 0.01))
#Plot for Feature importance
importance <- classification_tree$variable.importance

# Barplot
barplot(importance,
        main = "Feature Importance",
        xlab = "Features",
        ylab = "Importance",
        col = "steelblue",
        las = 2)

rpart.plot(classification_tree)

tree_probs <- predict(classification_tree, newdata = test_data, type = "prob")[, "Fraud"]

test_labels <- ifelse(test_data$Class == "Fraud", 1, 0)

pr <- pr.curve(scores.class0 = tree_probs[test_labels == 1],
               scores.class1 = tree_probs[test_labels == 0],
               curve = TRUE)

plot(pr,
     main = "Precision-Recall Curve for Classification Tree",
     auc.main = TRUE,
     color = "blue",
     legend = FALSE)




#Undersampled Classification Tree
undersampled_classification_tree <- rpart(Class ~ Time + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10 +
                                            V11 + V12 + V13 + V14 + V15 + V16 + V17 + V18 + V19 + V20 +
                                            V21 + V22 + V23 + V24 + V25 + V26 + V27 + V28 + Amount,
                                          data = undersampled_train_data,
                                          method = "class",
                                          control = rpart.control(minsplit = 20, cp = 0.01))
#Plot for Feature importance
importance2 <- undersampled_classification_tree$variable.importance

# Barplot
barplot(importance2,
        main = "Feature Importance",
```

```r
        xlab = "Feature",
        ylab = "Importance",
        col = "red",
        las = 2)

#tree diagram
rpart.plot(undersampled_classification_tree)

#ROC plot
undersampled_probs <- predict(undersampled_classification_tree, newdata = test_data, type = "prob")[, "Fraud"]

true_labels <- ifelse(test_data$Class == "Fraud", 1, 0)

roc_obj <- roc(true_labels, undersampled_probs)

plot(roc_obj,
     col = "darkred",
     lwd = 2,
     main = "ROC Curve for Undersampled Classification Tree")

auc_value <- auc(roc_obj)
legend("right", legend = paste("AUC =", round(auc_value, 4)), col = "darkred", lwd = 2)



#Nueral Network
nueral_network <- nnet(Class ~ Time + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10 +
                         V11 + V12 + V13 + V14 + V15 + V16 + V17 + V18 + V19 + V20 +
                         V21 + V22 + V23 + V24 + V25 + V26 + V27 + V28 + Amount,
                       data = train_data,
                       size = 5,
                       maxit = 200,
                       decay = 0.01,
                       trace = TRUE)

#feature importance
nnet_importance <- varImp(nueral_network)
importance_df <- data.frame(
  Feature = rownames(nnet_importance),
  Importance = nnet_importance$Overall
)
importance_df <- importance_df[order(-importance_df$Importance), ]


barplot(importance_df$Importance,
        names.arg = importance_df$Feature,
        main = "Neural Network Feature Importance",
        xlab = "Feature",
        ylab = "Importance",
        col = "green",
        las = 2,
        cex.names = 0.7)



#prescision recall curve

nnet_probs <- predict(nueral_network, newdata = test_data, type = "raw")

# Convert to binary
true_labels <- ifelse(test_data$Class == "Fraud", 1, 0)

pr_curve <- pr.curve(scores.class0 = nnet_probs[true_labels == 1],
                     scores.class1 = nnet_probs[true_labels == 0],
                    curve = TRUE)
# Plot the curve
plot(pr_curve,
     main = "Precision-Recall Curve for Neural Network",
     color = "green",
     auc.main = TRUE)


#Undersampled Nueral Network
undersampled_nueral_network <- nnet(Class ~ Time + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10 +
                       V11 + V12 + V13 + V14 + V15 + V16 + V17 + V18 + V19 + V20 +
                       V21 + V22 + V23 + V24 + V25 + V26 + V27 + V28 + Amount,
                     data = undersampled_train_data,
                     size = 5,
                     maxit = 200,
                     decay = 0.01,
                     trace = TRUE)

undersampled_nnet_importance <- varImp(undersampled_nueral_network)
undersampled_importance_df <- data.frame(
  Feature = rownames(undersampled_nnet_importance),
  Importance = undersampled_nnet_importance$Overall
)
undersampled_importance_df <- undersampled_importance_df[order(-undersampled_importance_df$Importance), ]


barplot(undersampled_importance_df$Importance,
        names.arg = undersampled_importance_df$Feature,
        main = "Undersampled Neural Network Feature Importance",
        xlab = "Feature",
        ylab = "Importance",
        col = "orange",
        las = 2,
        cex.names = 0.7)


undersampled_nnet_probs <- predict(undersampled_nueral_network, newdata = test_data, type = "raw")

# Convert to binary
true_labels <- ifelse(test_data$Class == "Fraud", 1, 0)

roc_obj_undersampled <- roc(true_labels, undersampled_nnet_probs)

plot(roc_obj_undersampled,
     col = "orange",
     lwd = 2,
     main = "ROC Curve for Undersampled Neural Network")

# Calculate and display AUC
auc_value_undersampled <- auc(roc_obj_undersampled)
legend("right",
       legend = paste("AUC =", round(auc_value_undersampled, 4)),
       col = "orange",
       lwd = 2)

#confusion plots

#function to make confusion plots
plot_confusion_matrix <- function(predictions, reference, title) {
  cm <- confusionMatrix(predictions, reference, positive = "Fraud")
  plt <- as.data.frame(cm$table) %>%
    ggplot(aes(x = Reference, y = Prediction, fill = Freq)) +
    geom_tile(color = "white") +
    geom_text(aes(label = Freq), color = "black", size = 5) +
    scale_fill_gradient(low = "white", high = "steelblue") +
    labs(title = title,
         x = "Actual",
         y = "Predicted") +
    theme_minimal() +
    theme(legend.position = "none",
          plot.title = element_text(face = "bold", hjust = 0.5))
  return(plt)
}



#Logistic Regression
reduced_log_reg_pred <- predict(reduced_log_regression, newdata = test_data, type = "response")
reduced_log_reg_pred_class <- factor(ifelse(reduced_log_reg_pred > 0.5, "Fraud", "Not Fraud"),
                                     levels = c("Not Fraud", "Fraud"))
p2 <- plot_confusion_matrix(reduced_log_reg_pred_class, test_data$Class,
                            "Logistic Regression")

#Undersampled Logistic Regression
undersampled_log_reg_pred <- predict(reduced_undersampled_logistic_regression,
                                     newdata = test_data, type = "response")
undersampled_log_reg_pred_class <- factor(ifelse(undersampled_log_reg_pred > 0.5, "Fraud", "Not Fraud"),
                                          levels = c("Not Fraud", "Fraud"))
p3 <- plot_confusion_matrix(undersampled_log_reg_pred_class, test_data$Class,
                            "Undersampled Logistic Regression")

#Classification Tree
tree_pred_class <- predict(classification_tree, newdata = test_data, type = "class")
p4 <- plot_confusion_matrix(tree_pred_class, test_data$Class,
                            "Classification Tree")

#Undersampled Classification Tree
undersampled_tree_pred_class <- predict(undersampled_classification_tree,
                                        newdata = test_data, type = "class")
p5 <- plot_confusion_matrix(undersampled_tree_pred_class, test_data$Class,
                            "Undersampled Classification Tree")

#Neural Network
nnet_pred <- predict(nueral_network, newdata = test_data, type = "class")
nnet_pred_class <- factor(nnet_pred, levels = c("Not Fraud", "Fraud"))
p6 <- plot_confusion_matrix(nnet_pred_class, test_data$Class,
                            "Neural Network")

#Undersampled Neural Network
undersampled_nnet_pred <- predict(undersampled_nueral_network,
                                  newdata = test_data, type = "class")
undersampled_nnet_pred_class <- factor(undersampled_nnet_pred,
                                       levels = c("Not Fraud", "Fraud"))
p7 <- plot_confusion_matrix(undersampled_nnet_pred_class, test_data$Class,
                            "Undersampled Neural Network")
p2
p3
p4
p5
p6
p7

library(gridExtra)
grid.arrange(p2, p3, p4, p5, p6, p7, ncol = 2)
```

12. Contribution

Intro – Ayman

Lit Review – Jenna

Research Question – Sagari

EDA – Shahd

Method Overview – Ayman

Model Process – Shahd

Results – Stella

Conclusion – Stella

R Code – Ayman