Integrated Fraud Detection Using Machine Learning, Deep Learning, and Data Feature Analysis: A Comparative Study

By

Isaac Morton

A DISSERTATION

Submitted to

The University of Liverpool

in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

20/09/2024

STUDENT DECLARATION

I hereby certify that this dissertation constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writings of another.

I confirm that I have not copied material from another source nor committed plagiarism nor commissioned all or part of the work (including unacceptable proof-reading) nor fabricated, falsified or embellished data when completing the attached piece of work.

I declare that the dissertation describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

Isaac Morton

I. ACKNOWLEDGEMENTS

I would like to dedicate this dissertation to my parents, it is only because of their unwavering support and encouragement that I was able to complete this. Their belief in me has been a constant source of motivation and for that I am deeply grateful.

I also received some guidance from my Uncle, who spoke with me and helped me to understand the real-world applications of this research.

I would also like to extend my sincere gratitude to my supervisor, Josh, for providing me with guidance and encouragement throughout not just the course of this project but my extended time at this university. I am very appreciative of the support he's provided.

Integrated Fraud Detection Using Machine Learning, Deep Learning, and Data Feature Analysis: A Comparative Study

II. ABSTRACT

Due to the increasing rise of digital transactions, fraud risk has significantly increased, making advanced detection methods critical for financial safety. In this dissertation, the comparative effectiveness of traditional Machine Learning and Deep Learning techniques for fraud detection will be explored. Using three distinct datasets, this study will evaluate Logistic Regression, Random Forest and XGBoost models alongside a Convolutional Neural Network Deep Learning approach. This research will focus on how these models handle imbalanced datasets, their scalability and their adaptability to evolving fraud patterns.

XGBoost emerged as the best-performing traditional Machine Learning model, performing well in both precision and recall across structured datasets. CNNs demonstrated potential to detect subtle patterns in datasets but computational demands and the need for extensive hyperparameter tuning limited the performance achieved. The ensemble model which combined CNN and XGBoost produced the most robust results, achieving a high precision and recall especially on dataset 3, which featured well-structured data. This ensemble approach also required a significant amount of computational resources, potentially causing challenges in implementing it.

The project also addresses challenges that are typical in financial fraud datasets- imbalance, through the use of Synthetic Minority Over-sampling Technique (SMOTE) to improve model sensitivity toward fraudulent cases. The research performed in this comparative study highlights that although traditional Machine Learning models like XGBoost can perform well, Deep Learning models offer greater potential, especially through the use of combined, ensemble methods.

The findings of this study contribute to the understanding of how Machine Learning and Deep Learning models can be optimally integrated to enhance fraud detection and provide a framework for further exploration, including the potential for alternative Deep Learning models and real-time deployment challenges.

III. Statement of Ethical Compliance

Statement of Ethical Compliance
This project will use publicly available datasets (A0) and involves no human participants. All data used in this project is from datasets containing already anonymised data and will be processed following ethical guidelines. The datasets include:

- Credit Card Fraud Detection Dataset
- IEEE-CIS Fraud Detection Dataset
- Financial Fraud Detection Dataset

-All sourced from Kaggle.
This data will be handled to ensure adherence to the ethical standards, ensuring confidentiality and anonymity are maintained.

Table of Contents

Chapter 1: Introduction

1.1 Scope

This dissertation investigates the evolving landscape of fraud detection in financial transactions by leveraging both traditional machine learning models and advanced deep learning techniques. As digital transactions continue to grow, fraud poses significant risks to financial institutions and customers, necessitating sophisticated, real-time detection strategies.

This study provides a comparative analysis of traditional machine learning models—Logistic Regression, XGBoost, and Random Forest—alongside a deep learning model, Convolutional Neural Networks (CNNs), to determine the most effective approach for fraud detection. Additionally, the research explores feature importance in dataset preparation, utilising three distinct datasets, each presenting unique challenges and insights into the efficacy of different fraud detection methodologies.

While traditional models have long been favoured for their interpretability and robustness, they often falter with the increasing complexity and volume of data. This study addresses their limitations and evaluates the potential of CNNs, typically used for image data, to be adapted for structured transaction data, thus extending their application to achieve what traditional methods may not.

The study's scope includes a focus on model performance and comparative analysis, with some exploration of preprocessing, feature engineering, and data manipulation strategies. Ultimately, this research aims to contribute to the field by identifying the most effective model for fraud detection across various types of financial datasets.

1.2 Background of Fraud Detection

Fraud detection is massively important in the financial sector, where the stakes are high for businesses and customers alike. As digital transactions become increasingly prevalent, so too does the potential for fraudulent activity, posing a substantial risk to financial institutions, payment processors, and end-users. According to the Association of Certified Fraud

Examiners, businesses worldwide lose an estimated 5% of their annual revenues to fraud, which amounts to trillions of dollars annually. These consequences aren't just limited to financial losses, with brand reputation, customer trust and operational efficiency also being affected. (Akash Gandhar et al. 2024)

Traditionally, fraud detection has been heavily reliant on manual checks and rule-based systems. These methods involve predefined rules and patterns that flag suspicious transactions. For example, rule-based systems might identify unusual transaction sizes, locations, or frequencies as potential fraud indicators. While this approach may be effective to a degree, they are inherently limited. Manual checks are labour-intensive and impractical for large-scale operations, leading to delays and inefficiencies. Rule-based systems are static and depend on human intuition, and can often fail to adapt to new or particularly sophisticated fraud patterns. As fraudsters continually evolve their techniques, these methods start to falter, struggling to keep up and resulting in a high rate of false positives or negatives.

The inclusion of machine learning and deep learning into the field of fraud detection has brought significant improvements, offering dynamic, scalable and more accurate solutions. Unlike traditional manual and rule-based methods, machine learning models can analyse vast datasets to identify complex patterns that may not be apparent to a human analyst. These models continue to improve their predictive accuracy over time, adapting to new data and fraud trends. For example, logistic regression and random forests are capable of handling large-scale data efficiently, whilst the more advanced deep learning techniques, such as CNNs show promise in detecting subtle, non-linear patterns within transactional data. These machine learning and deep learning techniques enhance detection rates and reduce false alarms, providing a more robust defence against fraud.
By employing these machine learning and deep learning techniques, financial institutions are enabled to detect fraudulent behaviour in real-time, which protects their customers and their assets. By leveraging the vast amounts of transactional data available to them, these models offer a significant increase in protection against fraud, especially compared to the traditional manual and rule-based counterparts. As a result, the use of Artificial Intelligence will play a critical role in the fight against fraud, a critical evolution institutions must use to respond effectively to an ever-growing threat.

1.3 Problem Statement

The primary problem that this dissertation attempts to address is the challenge of detecting fraudulent transactions with high accuracy using both advanced machine learning and deep learning techniques. As the volume of digital transactions continues to grow year on year, so too does the frequency and severity of fraudulent activity. In the year 2022 alone, fraudulent financial activity caused a global loss of just over 41 billion dollars, with this figure expected to grow to a cumulative 362 billion dollars between the years 2022-2027. (Juniper Research, 2022). This surge is driven by the rapid growth of e-commerce, mobile and online banking, and digital payments. These areas might provide convenience, but they also provide a large target, for fraudsters, giving them even more of an opportunity to exploit any potential system vulnerabilities.

The traditional methods of manual reviews and rule-based systems as previously described have proven to be inadequate in the continuously evolving landscape of financial fraud (Nicholls et al., 2021). Generating a lot of false positives increases the workload for risk or fraud analysts. (fraud.com 2022)This gives rise to the need for a more adaptive, scalable and accurate fraud detection system, which can be achieved through the use of machine and deep learning. However, despite the potential of these techniques, they also face challenges of their own, such as class imbalance, interpretability, and the computational cost of deploying deep learning algorithms in real-time environments.

With these challenges in mind, this dissertation aims to explore and evaluate the effectiveness of different machine learning and deep learning approaches- Logistic Regression, Random Forest, XGBoost and Convolutional Neural Networks (CNNs), and their effectiveness in accurately identifying fraudulent transactions across multiple datasets. The significance of this research is founded on the potential to contribute to the development of more robust and adaptable fraud detection models, offering insights into the most effective machine learning and deep learning techniques and dataset features for identifying fraudulent behaviour in various types of financial data.

1.4 Approach

To address the challenges associated with detecting fraudulent transactions with high accuracy, this project employs a multi-faceted approach that involves both traditional machine learning models and deep learning techniques, all utilising multiple datasets, providing a diverse testing ground for the evaluation of different fraud detection techniques.

The study begins by preprocessing the data and performing some feature engineering to ensure that the datasets are all suitable for training a machine/deep learning model. This includes handling missing values, transforming categorical variables into numerical formats, normalizing continuous features to improve model performance, and in some cases creating new features within the dataset.
Given the inherent class imbalance in fraud detection datasets, where the minority class (which is the class with fewer instances) is the class identified as being fraudulent due to being significantly rare than non-fraudulent cases, this has to be tackled. This is done through the use of the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE generates samples for the minority class, ensuring that the models have enough data to learn from to reduce the risk of bias towards the majority class.

The approach of this study involves a comparative analysis of traditional machine learning models: Logistic Regression, XGBoost and Random Forest with a deep learning model: Convolutional Neural Networks (CNNs). Traditional models are chosen for their interpretability, computational efficiency, and widespread application in fraud detection, giving a baseline comparison. CNNs, whilst typically used for image data, are explored for their ability to analyse complex and non-linear patterns in structured data, which is suitable for transaction data.

The performance of the models is evaluated using metrics such as accuracy, precision and F1-score. The analysis in this study aims to determine the most effective model for fraud detection considering both traditional and deep learning approaches. The findings are expected to provide valuable insights into how to use machine learning and deep learning techniques optimally when it comes to detecting fraudulent activities in various types of financial data.

1.5 Objectives of the study

For this dissertation, the objectives set out to achieve were shaped by the goal of enhancing the detection of fraudulent transactions using traditional and advanced machine learning techniques. This study aims to explore and analyse the performance achieved by the different models and, from this, determine the most effective strategies for identifying fraudulent activity in various types of financial data.
The objectives of this study are as follows:

1. To develop Robust Fraud detection Models: This is the first objective that must be achieved from this study, to perform a comparison of the models, they need to all be worked on equally and fairly, producing robust models using each technique.
2. To evaluate and Compare the Effectiveness of these models: Once the models are fully developed, this study aims to perform an in-depth analysis of the performance that these models can achieve. Through the use of key performance metrics, this comparison will provide insights into the relative strengths and weaknesses of both the traditional and deep learning approaches used.
3. To optimise models for Practical Implementation: The models must be optimised and developed for practical use, this will be done by adjusting decision thresholds and finding the right balance between precision and recall, fine-tuning the hyperparameters to maximise the performance. Also, tackling the class imbalance by employing SMOTE.
4. To identify the most significant features for Fraud Detection: This study also aims to give insights into what features in a dataset allow for the most effective and efficient models, through the use of three datasets and analysing the features within these datasets, this research will provide guidelines for selecting and engineering features that make a significant impact on model performance.

By achieving these objectives, the study aims to contribute to the development of robust and realistically implementable fraud detection systems.

1.6 Research Questions

This study is guided by core, key research questions, designed to explore the comparative effectiveness of different machine learning and deep learning techniques for fraud detection in financial datasets.
These research questions are:

1. How does the performance of traditional machine learning models compare to deep learning models for detecting fraudulent transactions?
   This question aims to determine the effectiveness of each approach for fraud detection, by comparing traditional models which are known for their interpretability and lower computational cost with deep learning models that are capable of

analysing the more subtle and complex patterns in the data, this study seeks to understand the trade-offs and advantages of each approach.

2. What are the most significant features within financial datasets that contribute to the effectiveness of fraud detection?
   Through the use of multiple datasets, identifying which data features are the most influential in detecting fraud can help pinpoint how to refine and optimise fraud detection models.
3. How does handling class imbalance affect the performance of machine learning and deep learning models in fraud detection?
   Typically, in transactional datasets, the transactions that are fraudulent account for a very small proportion of all transactions, which leads to a very significant class imbalance. This question investigates the impact of using SMOTE to create a balanced dataset and ensure the models do not have a heavy bias towards the majority class.
4. What are the practical considerations for deploying these models in real-world financial environments?

Beyond the pure performance metrics for each model, there are other concerns in the real world for these models, such as computational cost, scalability and adaptability to evolving fraud patterns. This study will examine these real-world metrics to identify how applicable these models are.

These research questions are fundamental to understanding how various machine learning and deep learning techniques can be applied to enhance fraud detection capabilities, providing a comprehensive analysis of these models to guide developments in this field.

1.7 Outcome

For this project, there are multiple expected outcomes, aiming to provide insights into how machine and deep learning models can be effectively utilised for fraud detection in financial transactions, allowing for a comprehensive understanding of how they work and could be applied in the real world. This study expects to demonstrate that Convolutional Neural Networks (CNNs) offer superior accuracy and adaptability when compared to traditional machine learning methods. Another outcome of this study will be to give an understanding of how optimising decision thresholds, to balance precision and recall to minimise false negatives and false positives produced by these models. Through experimenting with different thresholds, this study seeks to identify the most effective configurations for fraud detection models, ensuring a robust outcome, capable of handling real-world data.

Additionally, another expected outcome of this study is to provide a comprehensive grasp of the significance of feature selection and engineering in the development of machine learning and deep learning models, pinpointing the key attributes that most effectively distinguish between fraudulent and legitimate transactions.

The findings of this dissertation are expected to have practical applications, contributing to the field of fraud detection by providing a clear comparison of traditional and deep learning approaches, outlining the key strengths and weaknesses of each model, and talking through the best strategies to implement them. The expected outcome of this study is to be able to

support the development of more accurate, efficient, and adaptable fraud detection systems that won't fall behind the rapidly evolving modern era of financial crime.

Chapter 2: Background and Literature Review

2.1 Related Work

2.1.1 Overview of Fraud Detection Techniques

The use of AI in fraud detection is a critical area that has seen a significant increase in attention due in part to the increasing amounts of fraudulent activities. The use of Machine learning and Artificial Intelligence techniques has shown promise in identifying and mitigating these fraudulent activities.
A surge in the use of digital payment technologies like e-commerce and online banking, whilst convenient has also led to new challenges, particularly with regards to the heightened risks related to fraudulent activity. This increasing shift has been accelerated by the covid-19 pandemic, which has significantly boosted the use of non-cash transactions. (Khando, Islam, and Gao, 2023).
Traditional methods, such as manual reviews and rule-based systems have become less effective in this context due to their inability to scale and adapt to the increasing amount of fraudulent activity, and their growing sophistication (Nicholls et al., 2021).

Due to this inadequacy, in the fight against fraud, there has been a need to improve their defences, and this has been done by moving away from rule-based and manual approaches, and towards employing machine learning and deep learning techniques, as they offer the ability to analyse vast datasets and detect complex patterns that humans may miss, improving detection rates and reducing false positives. (Gandhar et al., 2024)

However, there are limitations faced by these models, particularly struggling to deal with highly imbalanced datasets, and adapting to evolving fraud tactics. This means that to shut down as much fraudulent activity as possible, moving away from traditional machine learning and towards deep learning could be the answer, with Convolutional Neural Networks (CNNs) showing promise in overcoming these limitations, capable of extracting complex features from transaction data and improving detection rates further. (Jurgovsky et al. 2018)

Overall, through the integration of both Machine Learning and Deep Learning techniques, a significant advancement in the field of fraud detection can be achieved, providing a solution that, compared to rule-based and manual approaches, is more dynamic, scalable, and accurate. Financial institutions will face fraud schemes that continue to grow more complex, necessitating the use of these solutions. (Le Borgne et al., 2015).

2.1.2 Evolution of Fraud Detection Approaches

Early Methods: Manual checks and rule-based systems

In order to analyse how Fraud Detection must evolve, understanding the history of the area is important. The initial, traditional approach to fraud detection was reliant on manual checks that were performed entirely by humans who would review transactions based on their personal experience and intuition. This had limited effect, due to being labour-intensive and prone to human error. The first automated systems were developed as transaction volumes increased, built to use predefined rules like transaction limits or restrictions on locations in order to identify any activity that could be potentially fraudulent. Again, these systems were effective but only to an extent, as their rigidity made them less effective as fraud tactics continued to evolve.

Shift to Machine Learning Techniques

Rule-based systems and the limitations they had meant financial institutions still needed to develop better methods of fighting fraud. This led to the adoption of machine learning techniques. These ML models, such as random forest and XGBoost take advantage of the vast amounts of data that these institutions have available to them, learning from historical data to identify patterns and any anomalies that could indicate fraud. These models are capable of adapting to new patterns over time meaning that they provide a more dynamic, robust approach to fraud detection.

Emergence of Deep Learning Techniques

The newest development in the evolution of fraud detection is the use of deep learning techniques. Deep learning(DL) can revolutionise the field of fraud detection as they are able to analyse vast and complex datasets. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) are two techniques currently being pioneered in this field since they allow for the detection of subtle and non-linear patterns that a simpler, traditional model might not recognise. These models excel particularly with high-dimensional data (such as transaction features, and user behaviour) and recognising sophisticated fraud schemes. The scalability and adaptability of DL approaches make them well-suited for real-time fraud detection.

Current Trends and Future Directions

Today, the best approaches combine multiple ML and DL techniques as an ensemble model, improving accuracy as much as currently possible, and reducing the risk of undetected fraud, or false positives.

Future directions in fraud detection could involve integrating technologies like natural language processing (NLP) and blockchain technology to enhance transparency and security. Another potential direction that shows promise is using federating learning, which allows models to learn collaboratively from decentralised sources without sharing raw data, preserving privacy while maintaining detection capabilities. These advancements could offer robust solutions for the ever-evolving fraud schemes by enhancing data security and increasing detection efficiency in real-world applications.

2.1.3 Importance of Automation in Fraud Detection

In the modern era, online transactions are increasingly high-volume, so the shift to automation in fraud detection is essential. Automated methods enable the real-time analysis of large datasets, reduce response times, minimise human error, and enhance scalability in

high-volume environments, making it essential for modern financial systems. A study by Gemalto study highlights the importance of automation, stating as much as 70% of consumers are less likely to do business with an organisation that has experienced a data breach. “the implementation of Automated Fraud Detection is not merely a security measure; it is a strategic investment in maintaining trust and credibility in the eyes of both customers and stakeholders.” (Tookitaki, 2023; Gemalto, 2023).

2.2 Literature Review

2.2.1 Traditional Machine Learning in Fraud Detection

Dal Pozzolo et al. (2018) – “Credit Card Fraud Detection: A realistic modelling and a novel learning strategy”

Dal Pozzolo et al. (2018) present an in-depth look at credit card fraud detection via the use of traditional machine learning techniques. The study highlights the importance of realistic modelling and the challenges that come with having imbalanced datasets, which is a very common problem in this area. The authors identify undersampling as a potential way to address the class imbalance problem and emphasise the need for evaluation strategies to ensure the reliability of models in real-world applications. The way the authors address the problem of class imbalances is very relevant to this study, as that is a key area overcome in order to maximise the performance of each model. Whilst this study presents some good findings, the focus is limited to a single dataset, which may not fully capture the variability and diversity found in real-world financial datasets.

Bhattacharyya et al. (2011) - "Data Mining for Credit Card Fraud: A Comparative Study"

This study compares various traditional ML techniques, Logistic Regression, Decision Trees, and Support Vector Machines (SVMs) for the use of detecting credit card fraud. The authors of the study evaluate each of these methods using real-world datasets and highlight the fact that no single technique outperforms all others across every metric. They also discuss the importance of the trade-offs between accuracy, interpretability and computational cost when improving model performance. The relevance of this study comes from the comparative analysis of multiple ML models it provides.

Whitrow et al. (2009) - "Transaction Aggregation as a Strategy for Credit Card Fraud Detection"

Whitrow et al. explore the use of transaction aggregation methods to enhance the performance of traditional machine learning models. Through grouping transactions to generate additional features that capture customer behaviour the model's performance improves, demonstrating that enhancing features increases the accuracy of ML approaches such as Logistic Regression. This study is relevant due to the overlapping focus on feature engineering to improve model performance, specifically with the use of Logistic Regression.

Zareapoor, M., & Shamsolmoali, P. (2015) - "Application of Credit Card Fraud Detection: Based on Bagging Ensemble Classifier"

This comprehensive study provided by Zareapoor and Shamsolmoali investigates the use of ensemble methods, specifically the Bagging Ensemble classifier for use in credit card fraud

detection. The authors present the argument that individual classifiers such as decision trees or naïve Bayes may not perform well consistently across datasets due to their respective inherent limitations. Through the use of a bagging approach which combines multiple classifiers, they demonstrate how this gives improved detection rates and robustness against overfitting. This study emphasises the benefits of ensemble learning methods. This study is relevant for this dissertation as it explores the use of multiple methods to overcome the limitations of traditional ML models.

D. Shah & Lokesh Kumar Sharma (2023) "Credit Card Fraud Detection using Decision Tree and Random Forest"

This paper looks at the advancements made and the widespread use of online shopping, and how the substantial increase of these technologies leads to a larger risk of credit card fraud. To combat this, they suggest the use of various machine learning algorithms, such as Naïve Bayes, Logistic Regression, SVM, Decision Trees, Random Forest and Genetic Algorithms. They conclude that machine learning plays a crucial role in the analysis of transaction datasets and in identifying fraudulent transactions. This is relevant to the areas of research covered in this dissertation due to the wide range of traditional ML techniques covered.

The studies reviewed here highlight the diverse approaches and methods used relating to traditional machine learning for fraud detection, each giving unique insights into the strengths and limitations of these models. Dal Pozzolo et al. (2018) emphasise the critical importance of realistic modelling and handling imbalanced datasets, whilst Bhattacharyya et al. provide a comparative analysis of the different ML models, highlighting that no technique on its own is universally better performing than another. Whitrow et al. (2009) and Zareapoor & Shamsolmoali (2015) further develop these foundations by exploring further strategies including feature engineering and ensemble learning methods in order to enhance model performance and robustness.
Shah and Sharma (2023) illustrate the application of a wide range of traditional ML techniques which shows how the area of fraud detection continues to grow and evolve in response to the increasing complexity and volume of online transactions. Together, these studies reveal that machine learning models are effective tools for fraud detection but require careful tuning and optimisation, and hybrid approaches are the best way to combat the inherent limitations that come with these methods.

2.2.2 Deep Learning in Fraud Detection

Carcillo et al. (2018) – "Financial Fraud Detection Using Deep Learning Techniques"

Carcillo et al. (2018) present a study into the applications of deep learning models, particularly Recurrent Neural Networks (RNNs), in detecting financial fraud. The study shows that RNNs can find complex patterns in transactional data, which in turn leads to an improved detection rate for fraudulent cases. The research presented highlights the potential that deep learning models have to outperform traditional ML algorithms in complex scenarios. This study's relevance lies in how they describe the potential DL techniques have in the field of fraud detection, which is part of the implementation of this project. Although the results are promising, the study's reliance on extensive computational resources and large labelled datasets may limit its applicability for smaller institutions with fewer resources.

Zhang et al (2018) "A Model Based on Convolutional Neural Network for Online Transaction Fraud Detection"

Zhang et al. (2018) propose a novel convolutional neural network (CNN) model specifically designed for online transaction fraud detection. Contrary to traditional models which require extensive feature engineering, this approach uses a feature sequencing layer to reorganise raw transaction data into various convolutional patterns, allowing the CNN to learn the derivative features from the data. The study demonstrates the model's capability to achieve high accuracy (around 91%) and recall (roughly 94%) compared to some existing CNN models, primarily due to its ability to handle low-dimensional and non-derivative data effectively. However, the study also notes that optimising the model's performance requires careful arrangement of input features which is very computationally intensive. This research is highly relevant to this study as it highlights the adaptability and effectiveness of CNNs in detecting complex fraud patterns in online transactions.

Vijaya Lakshmi et al. (2024) - "Building a CNN Model for Credit Card Fraud Detection Using TensorFlow and Keras"

Lakshmi et al. (2024) present an approach to credit card fraud detection that uses CNNs implemented using TensorFlow and Keras. This study focuses on detecting fraudulent transactions through the use of a Convolutional Neural Network model that is trained on a dataset of fraudulent and legitimate transactions. The model is capable of capturing complex patterns in the transaction data which would go unnoticed in traditional methods. This research demonstrates that the CNN model significantly improves detection accuracy, precision, recall and f1 score whilst also managing to handle the imbalanced data. It addresses challenges like high false positives and adaptability to new fraud patterns and offers solutions for real-time processing integration into already existing fraud detection systems. The study also emphasises the importance of balancing improved detection with ethical considerations and privacy concerns and gives suggestions as to guidelines for responsible model deployment. This research is incredibly relevant to this study, tackling a lot of similar areas, with the development of the CNN model and highlighting the potential of CNNs to enhance the accuracy and efficiency of fraud detection systems.

Roy et al. (2018) - "Deep Learning for Detecting Fraud in Credit Card Transactions"

Roy et al. (2018) evaluate a variety of deep learning methods for detecting credit card fraud, taking a look at Artificial Neural Networks (ANNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRUs). This study uses a vast dataset which consists of 80 million transactions in order to compare the effectiveness of these models and address the common challenges they face such as class imbalance and scalability by leveraging a high-performance, distributed cloud computing environment. The findings show that GRUs achieved the highest accuracy (91.6%), followed closely by LSTMs (91.2%) and RNNs (90.4%), while ANNs performed the worst (88.9%). The study emphasises both the importance of network size, with larger networks generally performing better, but also the need for further exploration of hyperparameter tuning and model sensitivity to optimise performance.

The studies reviewed here illustrate the wide range of applications and also the vast, promising potential of DL techniques in fraud detection. Carcillo et al. (2018) and Roy et al (2018) both highlight the strengths of RNNs and their variants, such as LSTM networks and GRUs in capturing complex temporal patterns in transaction data which leads to improved performance compared to traditional ML models.

Zhang et al. (2018) and Lakshmi et al. (2024) focus on the capabilities of CNNs to automatically extract relevant features from raw data and showcase their effectiveness in detecting complex fraud patterns and achieving high accuracy in real-world scenarios. These studies do demonstrate the promise of deep learning models in addressing key challenges such as class imbalance, high rate of false positives and being ability to adapt to new fraud patterns, but they also highlight the need for significant computational resources, extensively labelled datasets, and the need for the tuning of hyperparameters to optimise performance. Through the collective analysis of these studies, the findings suggest that while DL has high potential and can be a powerful tool for fraud detection, further research is required to enhance its practicality and reach its full potential, particularly for institutions with limited resources.

2.2.3 Comparative Analysis of ML and DL techniques

Abdallah et al. (2016) – "Comparative Analysis of Machine Learning Algorithms for Credit Card Fraud Detection"

Abdallah et al. (2016) provide a comparative analysis of a range of machine-learning algorithms used for credit card fraud detection. The study emphasises the importance of feature selection and engineering in improving model performance. The authors suggest that the best rate of detection accuracy and robustness can be achieved through the combination of multiple algorithms, giving relevance to this project due to the use of multiple techniques.

Gandhar et al. (2024) - "Fraud Detection Using Machine Learning and Deep Learning"

Gandhar et al. (2024) give an insightful comparative analysis of various ML and DL techniques applied to credit card fraud detection. The study compares traditional ML techniques such as Random Forests, SVMs, Logistic Regression, Decision Trees, and DL techniques including CNNs, RNNs, and LSTMs. The research highlights that traditional ML models are able to perform adequately but fall short of replicating the accuracy and flexibility of the DL models. However, the study also notes that DL models require significantly more computational resources and extensive labelled datasets, which can be a limitation for their practical implementation in smaller organisations, these are similar findings to this study.

Jose et al. (2023) - "Detection of Credit Card Fraud Using Resampling and Boosting Techniques"

Jose et al. (2023) focus on the performance of multiple ML and DL techniques for fraud detection, including boosting algorithms and ensemble learning methods including traditional ML techniques like XGBoost and CNNs and RNNs for DL.
The study provides a comparison of the models based on the standard metrics, using a highly imbalanced dataset of credit card transactions. The findings of the study suggest that DL techniques outperform traditional ML algorithms, due to their ability to learn from complex and sequential data patterns.

The studies included here highlight the comparative strengths and limitations of ML and DL techniques in credit card fraud detection. The studies of Abdallah et al. (2016) demonstrate the effectiveness of combining multiple ML algorithms to enhance model performance, while Gandhar et al. (2024) and Jose et al. (2023) reveal that DL models, particularly CNNs and RNNs, generally outperform the traditional ML techniques discussed, but also noting that these models come with the cost of significantly higher computational costs and the need for

large, labelled datasets, which poses a challenge for the applicability of these models for smaller organisations.
Overall, these findings concur that Deep Learning techniques offer promising advancements in fraud detection, but require optimisation for a broader use.

2.3 Industry Sources

2.3.1 Industry Trends and Reports

Currently, the financial sector is advancing the effectiveness of fraud detection due to the rapid technological advancements being made, and the increasing volume of digital transactions. Reports in the industry are indicative of a growing reliance on ML and DL techniques being used to combat fraud. According to the 2022 Gartner Market Guide for Online Fraud Detection, the industry is emphasising the use of global networks and machine learning models to increase the effectiveness of fraud detection. The insights of this report from the leading research and advisory company highlight the need for dynamic, adaptive fraud detection systems that are capable of handling the complexities of modern financial transactions. The report also recommends businesses employ multiple tools to prevent fraud, showing how the sector is growing into a more integrated, comprehensive approach using ML and DL. (Gartner. 2022)

Reports from the industry, such as materials released by Fujitsu (2019) discuss how deep learning techniques have the potential to be used in dynamic-making tasks. The document discusses the scalability and adaptability of these models, highlighting how companies are able to utilise these methods for a wide range of applications. Whilst the report does not include specifics about fraud detection, it highlights how CNNs and other advancements in ML and DL can handle complex datasets and subtle patterns, which is what makes them so important for fraud detection. As financial institutions seek to improve the effectiveness of their fraud detections, the insights from industry giants such as Fujitsu- a global technology company, highlight the growing role of scalable, adaptive AI models supporting the general trend of the industry as they push towards sophisticated, integrated AI approaches.

2.3.2 Case Studies and Real-World Implementations

Machine Learning in Fraud Detection: PayPal's Implementation

A case study involving the use of Machine Learning for fraud detection can be found when taking a look at PayPal, a leading online payment service. They have effectively employed machine learning models to enhance their fraud detection capabilities. PayPal has evolved its techniques, starting with logistic regression models and moving on to more advanced models such as gradient-boosted trees (GBTs) and neural networks. The advancements PayPal has made have led to significantly improved detection rates in real-time. Recently, PayPal has adopted the use of deep learning models, further improving their accuracy by 10-20% compared to the use of traditional Machine Learning algorithms alone. As a result of their efforts, they boast an impressive fraud loss of 0.28%, meaning that they have a loss of $0.28 for every $100 made. This is a real-world implementation of ML and DL that highlights the

effectiveness of both techniques when used both alone, and their improved strengths when used together. (ONIX, 2023)

Deep Learning in Fraud Detection: JPMorgan Chase & Mastercard's Usage of AI

Taking a look at a real-world implementation of Deep Learning models, JPMorgan Chase, a global financial services firm employs the use of AI-driven fraud detection to monitor and analyse the vast amounts of transaction data they have in real-time. The bank uses deep learning algorithms called DocLLM, a type of Large Language Model (LMM) to detect anomalies, and can "find inconsistencies and warning signs in a matter of seconds, helping the bank prevent fraudulent activities more effectively." (Effective, 2024).
Mastercard also employs a deep learning-based platform called Decision Intelligence. This analyses the way in which their customers spend money and calculates the likelihood of fraud for each transaction as and when it happens, stopping suspicious transactions before they go through. Decision Intelligence allowed Mastercard to "safely approve 143 billion transactions a year." Decision Intelligence is a type of generative AI which has enhanced their fraud detection rates by an average of 20%, and as high as 300%. (Mastercard, 2024).

Convolutional Neural Networks Used for Insurance Claim Fraud Detection

Looking at a study specifically involving the usage of CNNs, a recent paper by Gambo et al. (2022), published by IEEE proposes the use of CNN models to address the challenges posed by highly imbalanced datasets in fraud detection. By generating synthetic samples that address the imbalanced dataset, the model learns effectively and produces results that are higher than that of a lot of other similar studies and models in use in the field, boasting scores of 0.9982, 0.9965, and 0.9999 for accuracy, precision, and recall, respectively. The study shows the ability to automatically learn complex patterns through the use of CNNs.

2.3.3 Regulatory Considerations and Industry Standards

Regulatory requirements and industry standards are vital for developing fraud detection strategies. These act as guidelines, ensuring that detection models comply with legal and ethical standards whilst still remaining effective in identifying fraudulent transactions.
For example, the General Data Protection Regulation (GDPR) in Europe regulates data privacy and security measures, which has a heavy impact on how financial institutions develop and deploy fraud detection models. These regulations include:

- Secure data handling practices
- Limited data retention periods
- Transparency in how data is processed and used in machine learning models (European Commission, 2021)

The Payment Services Directive 2 (PSD2) emphasises Strong Customer Authentication and mandates that financial institutions employ more advanced fraud detection mechanisms, including risk-based authentication and transaction monitoring systems.
In complying with legal requirements, fraud detection systems will remain effective whilst also protecting consumer and data privacy. (European Commission, 2021).

Chapter 3: Design

3.1 Original Design

3.1.1 System Overview

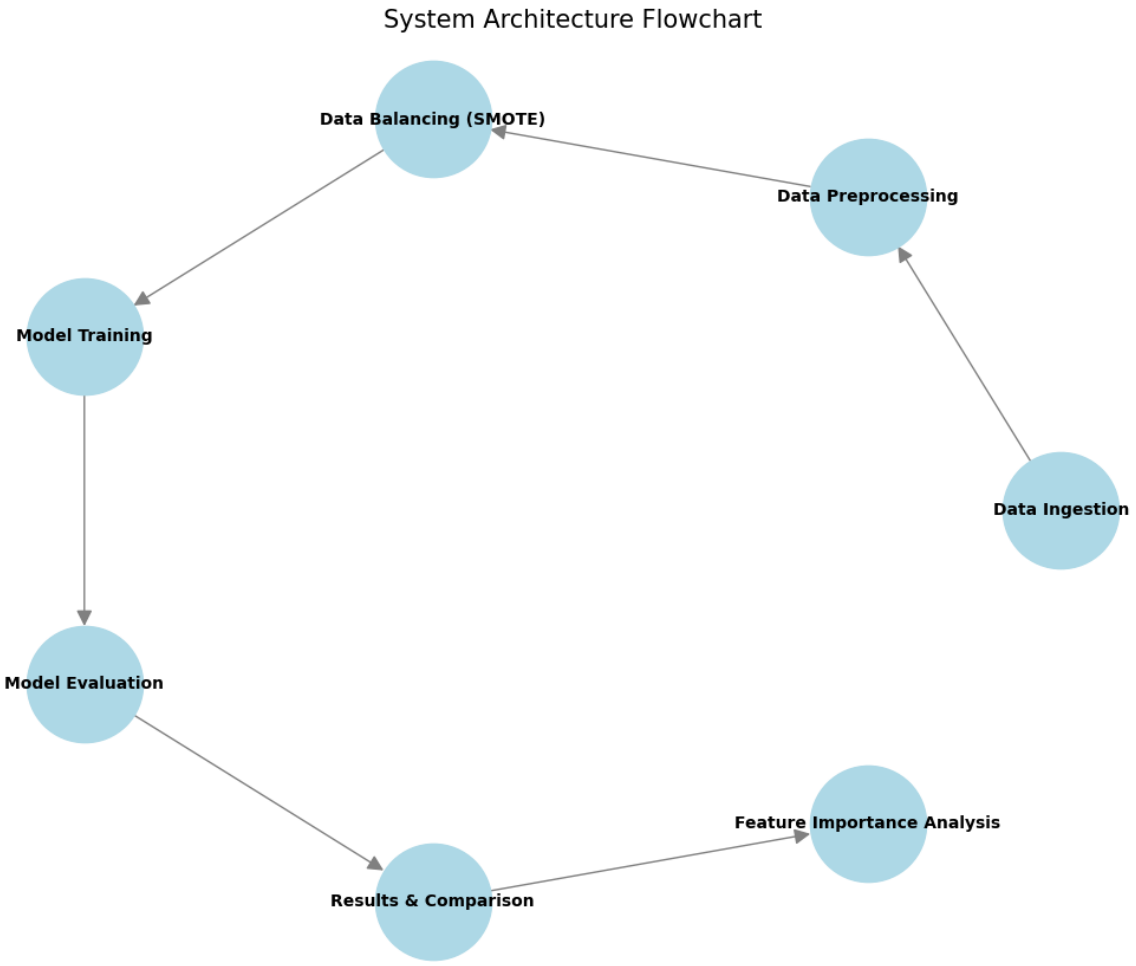
The aim of this project is to create a range of comprehensive fraud detection solutions that leverage both traditional machine learning (ML) models and advanced deep learning (DL) techniques to identify cases of fraud within three different financial transaction datasets. The core components of this system are:
Datasets: Three different datasets were used for training and evaluation: The Credit Card Fraud Detection Dataset, the IEEE-CIS Fraud Detection Dataset, and the Paysim simulation dataset. Each dataset has its own set of characteristics and challenges such as size, feature distribution and class imbalance.

ML and DL models: For this project, three ML models were used, and one DL model. The ML models used- Logistic Regression, Random Forest and XGBoost were chosen for interpretability, computational efficiency, and wide range of use in industry. For DL, a Convolutional Neural Network (CNN) was chosen for its ability to learn complex patterns from high-dimensional data, and the potential for its use to produce highly accurate fraud detection models.

Infrastructure and Tools: The solution is implemented using Python as the primary programming language, using various libraries including Scikit-Learn for ML models and TF/Keras for DL. The model was trained, running on a local machine and evaluated using various performance metrics such as accuracy, precision, recall and F1-Score.

Architecture Overview

1.) Data Ingestion and Preprocessing: Load the data in raw CSV format, and perform the necessary cleaning, normalization, feature engineering
2.) Data balancing: Each dataset used has the same main challenge, the inherent class imbalance typical in datasets used for financial fraud. To overcome this, the technique used in this study was the Synthetic Minority Over-Sampling Technique (SMOTE).
3.) Model Training: After the data is balanced and preprocessed: the data is fed into the ML and DL models for training. Each model is trained separately to identify optimal hyperparameters.
4.) Model Evaluation: After training, the models are evaluated on the test dataset to determine the effectiveness of each model.
5.) Results and Comparison: The results from each ML model and the DL model are compared against each other to identify the most effective approach for fraud detection. Besides model performance, computational efficiency is also taken into account.
6.) Data feature importance: Analysis is performed on the models to identify the most important features in the data for the given models.



The flowchart above is a basic visualisation of the system architecture.

3.1.2 Data Structures and Model Design and Development

Data structures

DataFrames:

Purpose: DataFrames, part of the Pandas library are used as the data structures used for data handling and manipulation.

Usage: Used throughout the project for loading data, handling missing values and storing transformed data.

Advantages: Efficient manipulation with large datasets, compatible with other libraries used in ML/DL.

NumPy Arrays:

Purpose: NumPy arrays are employed for efficient mathematical computations and matrix operations, which are critical when training ML/DL models.

Usage: NumPy arrays are used for storing transformed datasets.

Advantage: NumPy arrays offer high performance and are ideal for operations that require computations such as feature scaling, vectorised operations and batch processing.

Dictionary and List Data Structures:

Purpose: Dictionaries and lists are used for storing metadata, configurations and results.
Usage: Useful for maintaining configuration settings, evaluation metrics, and feature lists which are accessed and updated during different stages of the project.

Model Design and Development

Machine Learning Algorithms:

Logistic Regression:

Purpose: LR is a statistical model used for binary classification tasks. It works by estimating the probability that a given input point belongs to a particular class.
Usage: Chosen due to its simplicity and interpretability. Used to provide a baseline when compared to the more complex models to give insights into the decision boundaries and feature importance.
Structure: A linear model that uses a logistic function for binary classification.
Hyperparameters:

Solver: Optimisation algorithm for model fitting.
Max Iterations: Maximum iterations for solver convergence.
Regularisation parameter (C)- Controls regularisation strength to prevent overfitting,

Strengths: Easy to implement, computationally efficient.
Limitations: Limited performance on complex datasets with non-linear relationships.

Random Forest:

Purpose: RF is an ensemble learning method that builds multiple decision trees during training and outputs the average (mode or mean) prediction of individual trees.
Usage: RF was chosen for its robustness against overfitting and its effectiveness with a wide range of feature types.
Structure: Collection of decision trees leveraging both bagging and feature randomness.
Hyperparameters:

Number of Trees: Number of decision trees in the forest.
Maximum Depth: Controls the maximum depth of each tree.
Minimum Depth: Controls the minimum depth of each tree.
Minimum Samples Split: Minimum number of samples required to split an internal node.
Criterion: Function to measure split quality.

Strengths: High accuracy, robustness to overfitting, ability to handle large datasets and higher dimensionality, good performance with imbalanced datasets.
Limitations: Computationally intensive, less interpretable than simple models and requires tuning of hyperparameters.

XGBoost (Extreme Gradient Boosting)

Purpose: XGBoost is an optimised gradient boosting framework which is designed to be highly efficient and scalable.
Usage: XGBoost was chosen for its superior performance on tabular data and robustness in dealing with imbalanced datasets.

Structure: Builds trees sequentially, with each tree correcting the errors of previous ones.

Hyperparameters:

- Learning Rate: Step size for minimising loss function.
- Number of Boosting Rounds: Number of trees built.
- Max Depth: Maximum tree depth to control model complexity.
- Subsample: Fraction of samples used for each tree to prevent overfitting.
- Gamma: Minimum loss reduction required for further partitioning of a leaf node.

Strengths: Regularisation to prevent overfitting, has parallel processing capabilities, efficient with handling large-scale datasets and integrates well with cross-validation.

Deep Learning Algorithm:

Convolutional Neural Network (CNN)

Purpose: CNNs are deep learning models mainly used for image and video classification, but for this project, they have been adapted due to their ability to analyse data and identify patterns.

Usage: The CNN was chosen to capture complex, non-linear patterns in datasets that might not be detected by traditional ML models. It uses multiple layers (convolutional layers, pooling layers, and fully connected layers) to extract meaningful features and learn the relationships from the data.

Architecture:

- Input Layer: Takes feature vectors representing transaction attributes.
- Convolutional Layers: Multiple filters (kernels) extract spatial features using 1D convolutions suited to transaction data.
- Pooling Layers: Downsample feature maps to reduce dimensionality while preserving important patterns.
- Fully Connected (Dense) Layers: Perform the classification task after feature extraction, with a sigmoid function in the output layer for binary classification.

Activation Functions:

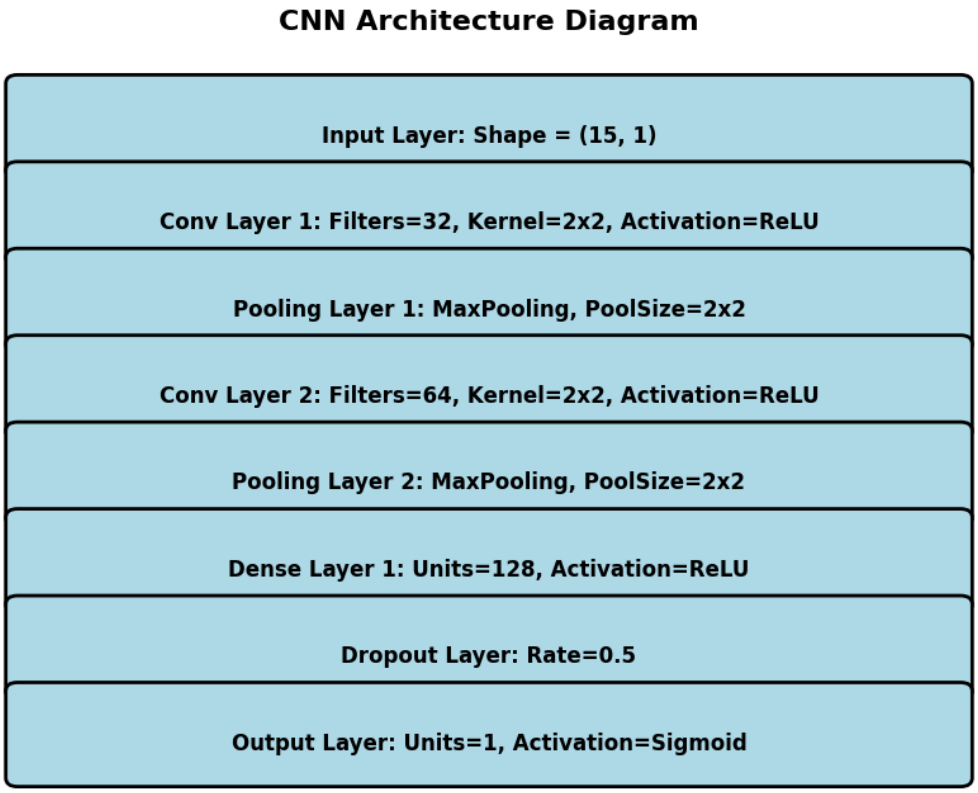
- ReLU (Rectified Linear Unit): Used in convolutional and dense layers for non-linearity and to prevent the vanishing gradient problem.
- Sigmoid: Used in the output layer to generate a probability score for binary classification.

Hyperparameters:

- Learning Rate: Controls step size for minimizing the loss function.
- Batch Size: Number of training samples used to update the model's parameters in each iteration.
- Epochs: Number of complete passes through the training dataset.
- Dropout Rate: Fraction of input units dropped to prevent overfitting.
- Optimiser: Adam optimiser was chosen for its adaptive learning rate.

Strengths: Highly effective for feature extraction, can identify complex patterns, scales well with large datasets.

Limitations: Very high computational expense, requires large amounts of data, is harder to interpret, requires careful tuning of hyperparameters.



Here is a diagram to show the layers of the CNN.

3.1.3 User Interface and User Experience

For this project, no User Interface or User Experience component was needed, as the primary focus is the development and evaluation of fraud detection models using different ML and DL models.
This research is inherently focused on the backend development, involving the manipulation of datasets, implementation of various algorithms and assessment of model performance rather than creating a front-facing application.

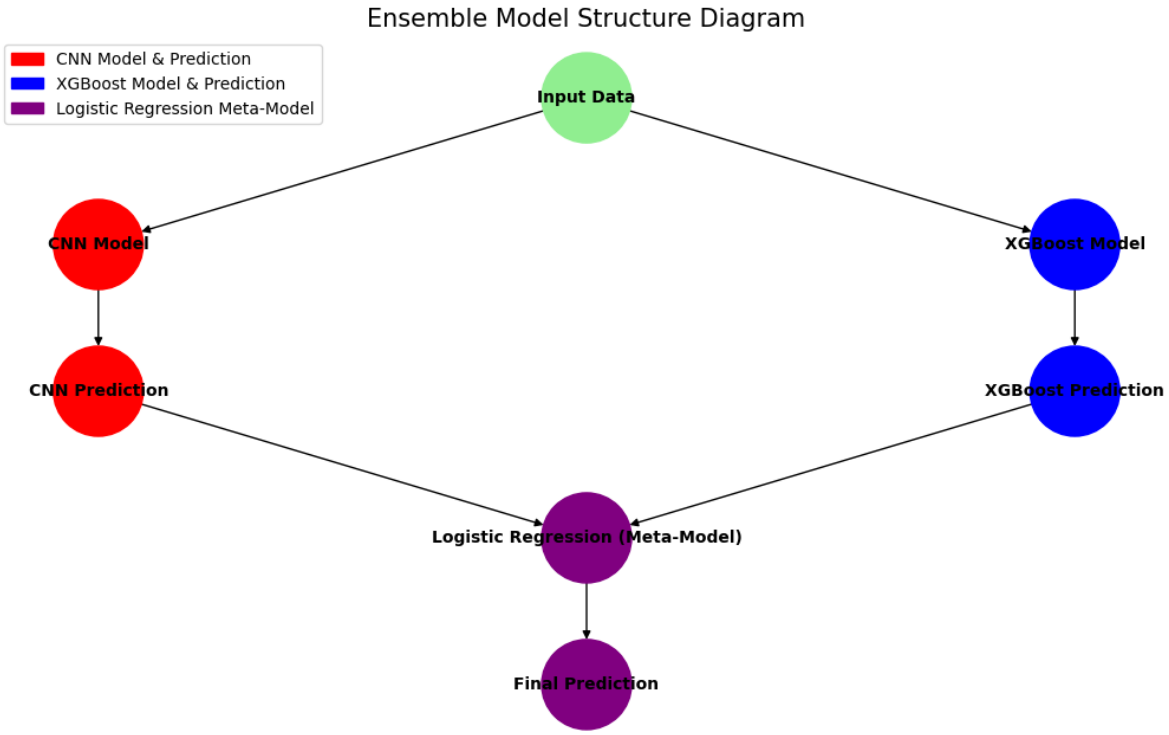
3.2 Changes to Original Design

Throughout the development of the project, some alterations were made to the original design of the project to better the performance, and to adapt with insights learned during the research and evaluation processes.

Inclusion of an ensemble approach: The biggest change made was the addition of an ensemble approach, which combined the DL model with the best-performing ML model, this combination of the CNN and the XGBoost models resulted in the development of the best-performing, most effective fraud detection model.
The decision to implement this ensemble method came from the findings of the research on this topic, with most studies pointing to this combined approach as being the most effective way of detecting fraud. By combining the high accuracy of the XGBoost model with the pattern recognition abilities of the CNN, the aim of implementing this method was to achieve

a balanced, robust fraud detection solution, reducing the rate of false positives and improving performance on the minority class overall.

The inclusion of this ensemble approach had a significant positive impact on the project. The combined model achieved better performance metrics than any individual model across each of the datasets used, demonstrating improved precision, recall and overall accuracy. However, this approach did require additional computational resources and time for training leading to an increase in development time. Overall, the benefits of this approach outweighed the complexity, leading to the most complete, effective solution.



This flowchart is to visualise how the ensemble approach works.

Chapter 4: Implementation

4.1 Development Process

4.1.1 Agile Methodology

For this project, the Agile methodology made sense because of its flexible and iterative nature, as this aligns well with the dynamic nature of developing ML models. The ability to rapidly prototype, test and refine models are crucial steps in a project that includes exploring different algorithms, testing different hyperparameters, and dealing with any unexpected data challenges that arise. Agile's iterative cycles, known as sprints, allow for continuous evaluation and adjustment which ensures that the development of the models responds to any research findings and feedback from the evaluations performed. This adaptability is especially useful when working with multiple datasets and models.

By applying Agile, the project was structured into multiple, manageable sprints, each lasting a week. Before the start of each development day, progress was tracked, potential blockers were identified and any necessary adjustments were made to the development plan. After each sprint was completed, anything that went well or poorly was noted to improve the process for the next sprint. This iterative process allowed for the integration of any new insights learned from the development, results and research conducted throughout the project.

4.1.2 Programming Languages and Libraries

For this project, the primary programming language chosen was Python due to its widespread use in both data science and machine learning. Python is considered the standard in both academic and industry settings for use in machine learning due to its simplicity, readability and extensive support. Python offers numerous libraries for data manipulation, ML and DL, making it the perfect choice for this project, developing fraud detection models.

Key libraries used in this project include Scikit-Learn for traditional ML models, and TensorFlow/Keras for deep learning. Scikit-Learn was used for its robust set of tools for data preprocessing, model training and evaluation making it a good choice for fine-tuning models like LR, RF and XGBoost. TensorFlow and Keras were chosen for the development of the DL model due to their efficiency in handling large-scale data. These libraries are highly optimised for performance and support a wide range of neural network designs making them a great fit for use in developing the CNN.

4.1.3 Tools and Technologies

The development of this project was done using Visual Studio Code (VSCode) for the integrated development environment (IDE). VSCode was chosen due to its versatility, ease of use with libraries, overall support for Python development, and personal familiarity. VSCode allows for efficient coding, debugging and integration with the libraries required for this project.
The project was developed on Windows 11, on a personal PC which had sufficient computational resources to handle the requirements of running the training for the ML and DL models.

For version control and code backup, this project was uploaded weekly to a private GitHub repository at the end of each sprint. This approach ensured that code changes were tracked, and provided insurance in the case of data loss or errors.

4.2 Data Preprocessing

In this project, data preprocessing was a critical step in preparing the datasets for effective fraud detection modelling. The three datasets used- Credit Card Fraud Detection, IEEE-CIS Fraud Detection, and Paysim simulation datasets— all required preprocessing to handle the missing values, normalise the data, perform feature engineering, and balance the datasets to get them ready for use in model training.

Data Cleaning and Handling Missing Values

For this project, the data cleaning involved removing duplicate entries and ensuring numeric consistency within the datasets. The duplicates were removed, which helps combat bias when training the models. All columns were converted to numeric data types where necessary, and for columns that had missing or invalid data, they were set to NaN. Any rows that were missing data in the IsFraud class were removed to maintain data integrity. To handle missing data, a placeholder value of -999 was used, as this helps the models recognize missingness as a potential indicator of fraud without bringing any bias or distorting the dataset.

Normalisation and Feature Engineering

To make the data standardised and have each feature contribute to the learning of the model, numeric columns were scaled using the StandardScaler from Scikit-Learn. This is a crucial step for optimising the performance of LR and the neural networks.
For feature engineering, categorical values were converted into numerical values using one-hot encoding, which creates new binary columns to represent the presence or absence of each category, thereby allowing the ML models to handle these features.

Handling Class Imbalance

Fraud detection datasets inherently suffer from a large class imbalance, where fraudulent cases are the minority class, for example, one of the datasets used in this study had only 3.5% of the cases being fraud. To combat this issue, the Synthetic Minority Over-sampling Technique (SMOTE) was applied to the data, this technique generates synthetic samples for the minority class (in this case, the fraudulent transactions) by interpolating between existing minority instances, helping to balance the dataset without the need to simply duplicate the minority class. For each dataset, a different sampling strategy was chosen, based on performance and evaluation depending on which value gave the best performance.

By implementing SMOTE, each model was trained on a much more balanced dataset which greatly reduces the bias towards the majority class (in this case, the non-fraudulent cases) and improves their ability to detect fraud accurately. This approach, along with the data pre-processing allowed for effective model development.

```
# Load the dataset
dataset3 = pd.read_csv('dataset3- Paysim/PS_20174392719_1491204439457_log.csv')
dataset3.drop_duplicates(inplace=True)

# Ensure numerical columns are of numeric type
numeric_columns = ['amount', 'oldbalanceOrg', 'newbalanceOrig', 'oldbalanceDest', 'newbalanceDest']
for col in numeric_columns:
    dataset3[col] = pd.to_numeric(dataset3[col], errors='coerce')

# Drop rows where 'isFraud' is NaN
dataset3 = dataset3.dropna(subset=['isFraud'])

# Convert categorical column 'type' into dummy/indicator variables
if 'type' in dataset3.columns:
    dataset3 = pd.get_dummies(dataset3, columns=['type'], drop_first=True)

# Define features (X) and target (y)
X = dataset3.drop(columns=['isFraud'])
y = dataset3['isFraud']

# Drop non-numeric columns
non_numeric_columns = ['nameOrig', 'nameDest']
X = X.drop(columns=non_numeric_columns)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42, stratify=y)
```

```
# Scale numeric features
scaler = StandardScaler()
X_train[numeric_columns] = scaler.fit_transform(X_train[numeric_columns])
X_test[numeric_columns] = scaler.transform(X_test[numeric_columns])

# Apply SMOTE to handle class imbalance
smote = SMOTE(sampling_strategy=0.4, random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)
```

Here is an example of the data preprocessing, showing dataset 3 here.

4.3 Model Development

4.3.1 Traditional Machine Learning Models

To develop the traditional machine learning models used in this project, the development included choosing suitable algorithms, optimising their hyperparameters, and evaluating the performance of using appropriate validation techniques. The models which were chosen for this project were selected for various reasons.

Model Selection and Training:

Logistic Regression(LR): Initially selected as a baseline model because of its simplicity, interpretability and computational efficiency. It is a linear model most suited to binary classification tasks which makes it ideal as a starting point from which to gauge the initial effectiveness of the preprocessing steps and understand feature importance. The training process resulted in giving a baseline performance score.

```
### Train the Logistic Regression Model
model_log_reg = LogisticRegression(max_iter=10000, random_state=42)
model_log_reg.fit(X_train_resampled, y_train_resampled)
```

Random Forest(RF): This is an ensemble learning technique, meaning it aggregates two or more learners in order to produce better results, and was chosen due to its robustness against overfitting. The model constructs multiple trees and combines their predictions, leading to increased accuracy and generalisation. This model was trained using a combination of default parameters and some hyperparameter tuning, involving adjusting the number of trees (n_estimators), maximum depth of trees (max_depth), minimum samples per split (min_samples_per_split) and the criteria for measuring split quality.

```
### Train the Random Forest Model
rf_model = RandomForestClassifier(
    n_estimators=100,
    max_depth=15,
    class_weight='balanced',
    random_state=42,
    n_jobs=-1
)
rf_model.fit(X_train_resampled, y_train_resampled)
```

XGBoost(Extreme Gradient Boosting): XGBoost is an optimised gradient boosting algorithm designed for high performance and scalability, making it an ideal choice for this task, as it also performs well when dealing with imbalanced datasets and overfitting. The model was trained with various hyperparameters such as learning rate (eta), number of boosting rounds(n_estimators), maximum tree depth (max_depth), subsampling ratio (subsample) and minimum loss reduction required to make a further partition on a node (gamma).

```
### Train the XGBoost Model
clf_xgb = XGBClassifier(
    n_estimators=100,
    max_depth=10,
    learning_rate=0.2,
    subsample=0.75,
    colsample_bytree=0.75,
    random_state=42
)
clf_xgb.fit(X_train_resampled, y_train_resampled)
```

Hyperparameter tuning and Validation Techniques:

The hyperparameters were tuned with a variety of methods, using a grid search, cross-validation, and some manual testing to identify the optimal parameters for each model.

Grid search: A grid of potential hyperparameter values was defined, which is then systematically evaluated using each possible combination of hyperparameter values to determine which settings gave the best final result.
Cross-Validation: To ensure the robustness of the model and prevent overfitting, k-fold cross-validation was used during hyperparameter tuning, this involves splitting the dataset into *k* subsets, having each model trained on *k*-1 subsets and then testing on the remaining subset, iterating *k* times. This provides a much more reliable estimate of how the model performs on unseen data.
Manual tuning: This was the final method used, fine-tuning the hyperparameters based on model performance, following the initial automated methods. Slight adjustments were made to hyperparameters such as learning rates or tree depths to enhance the model. This process allowed for targeted improvement, optimising the model's performance while balancing computational cost, minimising overfitting and prioritising performance for the minority class. Additionally, more manual tuning was performed on values such as the SMOTE percentage and train-test split percentage to get the highest performance from each model.

4.3.2 Deep Learning Model (CNN)

After developing the ML models, the next focus was building the Convolutional Neural Network (CNN) model for fraud detection. To do this, a series of steps were taken: Network design, data preparation, hyperparameter optimisation, and model testing.

Network design:

The CNN architecture was made to handle the tabular data for fraud detection. First, the model has an input layer designed to accommodate the features of the datasets, for example, dataset 3 with 15 features has a shape of 3x5x1. This is followed by a convolutional layer (Conv2D) with 32 filters and a kernel size of 2x2 to capture spatial patterns in the data. Next is the max-pooling layer (MaxPooling2D) to reduce dimensionality and prevent overfitting. After, dropout layers were added after the convolutional and dense layers to further prevent overfitting. The final layer was a dense output layer with a sigmoid activation function for binary classification.

Hyperparameter Optimization:
Several hyperparameters were optimized through experimentation:

- **Learning Rate:** Controlled by the Adam optimiser to adapt the step size during model training.
- **Batch Size:** Set to 32 to balance memory use and convergence speed.
- **Epochs:** Set to 11 to ensure adequate training while avoiding overfitting.
- **Dropout Rates:** Adjusted to 0.25 and 0.5 after convolutional and dense layers to reduce overfitting.

Model Training:

The model was trained on the resampled and scaled dataset, which included the synthetic samples which were generated from SMOTE to address class imbalance. The training process is used by binary cross-entropy as the loss function. A custom callback function was used to assess the model's performance using a threshold of 0.9 (found to be the optimal threshold from multiple tests), allowing us to easily evaluate the model's performance, and see the accuracy and recall for each epoch. This iterative training process ensured the CNN model was as optimised as possible for identifying fraudulent transactions.

```
# Define the CNN model
model = Sequential([
    Conv2D(32, (2, 2), activation='relu', input_shape=new_shape),
    MaxPooling2D((2, 2)),
    Dropout(0.25),
    Flatten(),
    Dense(64, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')  # For binary classification
])
```

4.3.3 Integration of Models

Learning from the results and research conducted during the course of this project, to make the most effective fraud detection system an ensemble learning approach was implemented, combining the strengths of traditional ML models and the DL model. The integration process involved selecting the best-performing ML model and combining this with the CNN model, enhancing overall performance.

Ensemble Learning Technique:

The ensemble model combined the CNN, known for its ability to capture complex, non-linear patterns in the data, with the XGBoost model, which performed the best out of all ML models based on the evaluation metrics.
The development of the ensemble method was as follows:
A five-fold stratified cross-validation process was employed. During each fold, the CNN and XGBoost models were trained separately, and their predictions on the validation data were recorded as meta-features, which were used to train a final meta-model, a logistic regression classifier which combined the outputs of both the base models to make the final prediction. This model was further fine-tuned using hyperparameter optimisation techniques. Each model's performance was used to adjust the weights assigned accordingly. This stacking

ensemble approach takes advantage of the strengths of both the DL and ML methods, leading to the most robust fraud detection system developed in this project.

```
    # Train base models
    # CNN model
    model_cnn_fold = tf.keras.models.clone_model(model_cnn)
    model_cnn_fold.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
    model_cnn_fold.fit(
        X_tr_cnn, y_tr,
        epochs=10,
        batch_size=32,
        class_weight=class_weights_fold,
        verbose=0
    )

    # XGBoost model
    clf_xgb_fold = XGBClassifier(
        n_estimators=100,
        max_depth=10,
        learning_rate=0.2,
        subsample=0.75,
        colsample_bytree=0.75,
        random_state=42,
        scale_pos_weight=scale_pos_weight_fold
    )
    clf_xgb_fold.fit(X_tr, y_tr)

    # Predict on validation fold
    cnn_val_preds = model_cnn_fold.predict(X_val_cnn).flatten()
    xgb_val_preds = clf_xgb_fold.predict_proba(X_val)[:, 1]

    # Store predictions as meta-features
    meta_train[val_idx, 0] = cnn_val_preds
    meta_train[val_idx, 1] = xgb_val_preds

    # Predict on the test set and average
    cnn_test_preds = model_cnn_fold.predict(X_test_cnn).flatten()
    xgb_test_preds = clf_xgb_fold.predict_proba(X_test_scaled)[:, 1]
    meta_test[:, 0] += cnn_test_preds / skf.n_splits
    meta_test[:, 1] += xgb_test_preds / skf.n_splits

# Ensure meta_train and meta_test are properly populated
if np.any(np.isnan(meta_train)) or np.any(np.isnan(meta_test)):
    print("NaN values found in meta features. Please check the cross-validation loop.")
else:
    # Train meta-model
    meta_model = LogisticRegression(max_iter=10000, random_state=42)
    meta_model.fit(meta_train, y_train_resampled)

    # Evaluate the ensemble model
    ensemble_proba = meta_model.predict_proba(meta_test)[:, 1]
```

4.4 Experimental Setup

For this project, there was a systematic process of training, validating and testing the ML and DL models to analyse their effectiveness in detecting fraud.

Training, Validation and Testing: The dataset was split into training and testing sets to develop and assess the models. The exact split was determined based on manual testing to identify the optimal performance, and there were two different splits used, for datasets one and three, a 75% training, 25% test split was used, and for dataset two an 80% training, 20% test was used. The training of the models involved hyperparameter optimisation and cross-validation, with the CNN model utilising iterative training through numerous epochs. The performance of the models was evaluated using the test set, to assess their ability to detect fraud on unseen data.

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42, stratify=y)
```

4.5 Challenges and Solutions

This project had many technical challenges during the implementation, particularly with managing the large datasets used, addressing the computational intensiveness of running the model training and integrating different models into a unified fraud detection system. Here's a summary of some of the challenges faced, and the way they were overcome.

Large Dataset Management:
Challenge: The datasets used in this project were vast, posing a significant challenge regarding memory usage and processing time.
Solution: To manage large datasets effectively, data cleaning was performed optimally, removing unnecessary entries early on.

Computationally expensive, time-consuming model training:
Challenge: Training each of these models and running the extensive hyperparameter optimisation was computationally intensive, and time-consuming.
Solution: Early stopping and checkpointing were used to stop unnecessary training epochs and save the best model weights. Simpler versions of each of the models were trained first to identify which parameters had the most potential before committing to the more complex versions. Also, parallel processing was implemented where possible.

```
# Implement early stopping
early_stopping = tf.keras.callbacks.EarlyStopping(
    monitor='val_loss',
    patience=5,
    restore_best_weights=True
)
```

Data Quality issues:
Challenge: The datasets contained missing values, duplicate entries and imbalanced datasets, which needed to be solved before use in training the models.
Solution: Data cleaning steps were implemented, handling the missing values and duplicate data and imbalance was addressed with SMOTE to combat potential bias issues.

Chapter 5: Testing and Evaluation

5.1 Testing Strategy

5.1.1 Unit Testing

Unit Testing was carried out to ensure that each component functioned as expected. Performing these tests helped detect errors early on in the development process, allowed for a smooth integration of each of the components and prevented potential issues from cropping up in the final model.

Components Tested:
Model training scripts- for each model
Evaluation functions- to compute performance metrics.

Testing Approach:
Example unit tests included checking that missing values were appropriately handled, and that the samples from SMOTE were correctly generated.
Unit tests were carried out after each significant code change, and all tests passed without failures, confirming the reliability of each component.

5.1.2 Integration Testing

Integration testing was also performed to make sure each component worked together as expected. This was an important step to find out if there were any issues in the interactions between the individual components that might affect the system's performance.

Components integrated:
Data Processing pipeline- data cleaning, Smote balancing.
Model training scripts- for each model tested.
Ensemble model integration- combining outputs from CNN and XGBoost.

Testing Approach:
End-to-end workflow tests were carried out to ensure that the whole system ran without errors and produced the expected results.
Component interaction tests to verify that outputs from one stage correctly flowed into the next stage.

5.1.3 Validation Testing

Validation testing was also conducted to assess the overall functionality and the performance of the final fraud detection system, to make sure that it performed as expected. This was important to verify the model is capable of adapting to unseen data and is effective in detecting fraudulent transactions.

Components Validated:
Cross-validation processes- for each model to evaluate model robustness.
Performance metrics evaluation- comprehensively assess model effectiveness.

Testing approach:
K-fold cross-validation was applied to ensure model robustness and prevent overfitting.
Performance metrics were monitored to guide any adjustments as needed to optimise model performance.

5.2 Evaluation Metrics

5.2.1 Metrics Used

When assessing the effectiveness of a model, there are several ways of measuring success. Before discussing the metrics, understanding True positives, True negatives, False positives and False negatives is necessary.
True positives (TP) are when the model correctly predicted the positive class, false positives (FP) are where the model incorrectly predicted the positive case, true negative (TN) is where the model correctly predicted the negative case, and false negative (FN) are the instances where the model incorrectly predicted the negative classes.
The scores are given in a confusion matrix, indicating the amount of each type.

Performance Metrics

- Accuracy: Measures the ratio of true results (both true positives and true negatives) among the total number of cases examined. Calculated as

$$\text{Accuracy} = \frac{\text{True Positives (TP)} + \text{True Negatives (TN)}}{\text{TP} + \text{False Positives (FP)} + \text{TN} + \text{False Negatives (FN)}}$$

  Accuracy shows how often the model correctly predicts the target variable

- Precision: The ratio of true positive results among the total number of positive results (true positives and false positives) predicted by the model. It is a measure of the accuracy of positive predictions. Calculated as

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

  High precision is a good indicator that when a model predicts a transaction as fraudulent, it is likely to be correct.

- Recall (Sensitivity): Measures the ratio of true positives to the sum of true positives and false negatives (relevant items that are incorrectly identified). It is a measure of the model's ability to identify all relevant instances. Calculated as

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

  High recall indicates that the model successfully captures most of the fraudulent cases, though, without the added context of the other metrics, it's hard to see how many false positives are included.

- F1-Score: The harmonic mean of precision and recall, providing a way to gauge the balance of the two. Calculated as

$$F1 - score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- ROC-AUC (Receiver Operating Characteristic - Area Under Curve): Measures the model's ability to distinguish between fraudulent and non-fraudulent transactions, providing insight into its overall classification performance. Calculated as

$$\text{AUC-ROC} = \int_0^1 \text{TPR}(t)\ d\big(\text{FPR}(t)\big)$$

  Meaning it plots the true positive rate (recall) against the false positive rate. AUC represents the probability that the model will rank a randomly chosen positive instance higher than a randomly chosen negative instance, with a higher AUC score being better.

- Support: The number of instances in each class. It is important to provide context and understanding for the distribution of instances, especially in the context of imbalanced datasets like fraud detection.

Example confusion matrix:

	Precision	Recall	F1-Score	Support
0	0.89	0.95	0.92	300
1	0.68	0.49	0.57	69
Accuracy			0.86	369
Macro avg	0.79	0.72	0.74	369
Weighted avg	0.85	0.86	0.85	369

Looking at this example matrix in the context of fraud detection, where non-fraudulent cases are the 0 class, and fraudulent transactions are represented by 1, this table shows a high overall accuracy (0.86) but has a lower recall for fraudulent transactions, which would mean the model this matrix is representing struggled at identifying fraudulent cases.

5.2.2 Evaluation Results

In this section, the results from the models from this study will be explored.

Logistic Regression

Dataset 1

```
Classification Report with Adjusted Threshold:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     70814
           1       0.87      0.76      0.81       118

    accuracy                           1.00     70932
   macro avg       0.93      0.88      0.91     70932
weighted avg       1.00      1.00      1.00     70932

ROC-AUC Score with Adjusted Threshold: 0.9695
```

The high accuracy here is due to the sheer number of non-fraudulent transactions (case 0)
Precision for the fraudulent case is fairly high, but the recall is worse.
The resulting F1-score is 0.81, which is quite high and shows that this model performs decently well on the first dataset, and could be suited for datasets like this in the real world.

Dataset 2

```
Logistic Regression Accuracy: 0.9387933078199614
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.96      0.97    113975
           1       0.28      0.49      0.36      4133

    accuracy                           0.94    118108
   macro avg       0.63      0.72      0.66    118108
weighted avg       0.96      0.94      0.95    118108

ROC-AUC Score: 0.8357715203949911
```

Again, the accuracy here is high, but that is due to the performance of fraudulent cases, which is by far the majority class.
The precision and recall are both very low, coming to a poor F1-score of 0.36. This indicates that the model captured just under half of the fraudulent cases, but has a very high rate of false positives.
These results show the disparity between the classes and how the model struggles to correctly identify fraudulent transactions.

Dataset 3

```
Logistic Regression Evaluation with Adjusted Threshold:
Optimal Threshold: 0.9978
Adjusted Accuracy: 0.9991
Classification Report with Adjusted Threshold:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00   1588602
           1       0.71      0.52      0.60      2053

    accuracy                           1.00   1590655
   macro avg       0.86      0.76      0.80   1590655
weighted avg       1.00      1.00      1.00   1590655
```

Once again, the model produces a high accuracy, this time rounding to 1 in the confusion matrix, however, this is still due to performing well with the majority class.
For fraudulent transactions, the precision is fair, but recall is still poor, with the F1-score of 0.60 showing that the model does a reasonable job in identifying fraudulent transactions, but there is still a lot of room for improvement.

Random Forest

Dataset 1

```
Random Forest Evaluation with Adjusted Threshold:
Optimal Threshold: 0.7199
Adjusted Accuracy: 0.9994
Classification Report with Adjusted Threshold:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     70814
           1       0.90      0.74      0.81       118

    accuracy                           1.00     70932
   macro avg       0.95      0.87      0.90     70932
weighted avg       1.00      1.00      1.00     70932
```

The RF model performs quite well for dataset 1, the precision for class 1 is 0.9, which is fairly high and indicates a strong ability to correctly identify fraudulent transactions. The recall, however, is lower, meaning the model is still missing some cases.
The F1-Score is reasonable, showing a better balance between precision and recall.

Dataset 2

```
Random Forest Evaluation with Best Threshold:
Optimal Threshold: 0.6391
Adjusted Accuracy: 0.9733
Classification Report with Best Threshold:
              precision    recall  f1-score   support

           0       0.98      0.99      0.99    113975
           1       0.68      0.45      0.54      4133

    accuracy                           0.97    118108
   macro avg       0.83      0.72      0.76    118108
weighted avg       0.97      0.97      0.97    118108
```

Again, dataset 2 is causing some issues, with the precision and recall being notably worse than dataset 1, but still better than the LR model on the same dataset, with the results of this dataset, show that RF struggled to perform for both precision and recall.

Dataset 3

```
Random Forest Evaluation with Adjusted Threshold:
Optimal Threshold: 0.9518
Adjusted Accuracy: 0.9997
Classification Report with Adjusted Threshold:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00   1588602
           1       0.95      0.85      0.90      2053

    accuracy                           1.00   1590655
   macro avg       0.97      0.93      0.95   1590655
weighted avg       1.00      1.00      1.00   1590655
```

With dataset 3, RF performs very well, with high precision, high recall, and a good balance between the two. These results show that the model has a good ability to correctly identify fraudulent cases, and doesn't miss out on too many.

XGBoost

Dataset 1

```
XGBoost Evaluation with Adjusted Threshold:
Optimal Threshold: 0.8810
Adjusted Accuracy: 0.9995
Classification Report with Adjusted Threshold:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     70814
           1       0.95      0.77      0.85       118

    accuracy                           1.00     70932
   macro avg       0.97      0.89      0.93     70932
weighted avg       1.00      1.00      1.00     70932

ROC-AUC Score with Adjusted Threshold: 0.9777
```

XGBoost performs well on the first dataset, with great precision, but doesn't perform quite as well with recall. The F1-Score however is still quite high, which indicates that it shows a good, fairly balanced ability to detect fraudulent transactions whilst limiting false positives to a reasonable extent.

Dataset 2

```
XGBoost Evaluation with Best Threshold:
Optimal Threshold: 0.3125
Adjusted Accuracy: 0.9801
Classification Report with Best Threshold:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99    113975
           1       0.77      0.62      0.69      4133

    accuracy                           0.98    118108
   macro avg       0.88      0.81      0.84    118108
weighted avg       0.98      0.98      0.98    118108
```

Dataset 2 is still proving to be the dataset the models struggle on the most, with worse scores than the XGBoost model on dataset 1, however, it does outperform the other models on dataset 2. The precision score indicates a reasonable proportion of correctly identified fraudulent transactions, while the recall suggests that it is missing some fraudulent transactions.

Dataset 3

```
XGBoost Evaluation with Adjusted Threshold:
Optimal Threshold: 0.9733
Adjusted Accuracy: 0.9997
Classification Report with Adjusted Threshold:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00   1588602
           1       0.93      0.79      0.86      2053

    accuracy                           1.00   1590655
   macro avg       0.97      0.90      0.93   1590655
weighted avg       1.00      1.00      1.00   1590655

ROC-AUC Score with Adjusted Threshold: 0.9994
```

Again, the XGBoost model performs reasonably well. The precision is fairly high, and the recall is strong too, with the resulting F1 score reflecting a well-balanced, decently performing model.

Convolution Neural Network

Dataset 1

```
Epoch 10 - Classification Report at Threshold 0.9:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     70814
           1       0.71      0.80      0.75       118

    accuracy                           1.00     70932
   macro avg       0.85      0.90      0.87     70932
weighted avg       1.00      1.00      1.00     70932
```

Again, we can see the high accuracy due to the perfect performance on the majority class of non-fraudulent transactions.
For fraudulent transactions, precision is reasonably high, and recall is strong, indicating it captures the majority of fraudulent transactions, and the F1-score shows a good balance.

Dataset 2

```
Epoch 10 - Classification Report at Threshold 0.9:
              precision    recall  f1-score   support

           0       0.97      1.00      0.99    142469
           1       0.81      0.22      0.34      5166

    accuracy                           0.97    147635
   macro avg       0.89      0.61      0.66    147635
weighted avg       0.97      0.97      0.96    147635
```

Even with the use of a CNN, dataset 2 continues to have the worst performance, despite having a high precision, the recall is very low, indicating that the model struggled to detect fraudulent cases effectively in this dataset, and the F1-score shows how imbalanced precision and recall is for class 1.

Dataset 3

```
Epoch 11 - Classification Report at Threshold 0.9:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00   1588602
           1       0.73      0.75      0.74      2053

    accuracy                           1.00   1590655
   macro avg       0.86      0.87      0.87   1590655
weighted avg       1.00      1.00      1.00   1590655
```

On dataset 3, the CNN's performance is very balanced, and shows good, similar performance for precision and recall, meaning the model has a fair performance, but still with room for improvement, and performing worse than the XGBoost method.

5.2.3 Summary

Logistic Regression

- Strengths:
  - Computationally inexpensive
- Weaknesses:
  - Struggles across the board with the minority class

Random Forest

- Strengths:
  - Achieves a moderate to high precision score on all datasets, notably struggling with dataset 2 mainly.
  - Better balance between precision and recall than LR.
- Weaknesses:
  - Slightly computationally expensive
  - Recall scores are still low

XGBoost

- Strengths:
  - Best performing ML model overall
  - Strong precision
- Weaknesses:
  - Requires fine-tuning and is fairly computationally expensive
  - Still performs worse with recall, and struggles with dataset 2

CNN:

- Strengths:
  - Strong balance between precision and recall
- Weaknesses:
  - Very computationally expensive
  - Struggles with dataset 2

From these results, it is evident that the ML models perform admirably, particularly the XGBoost and Random Forest models, with the XGBoost being the best performer overall. These models give a good result whilst also being less computationally efficient than the CNN. Despite struggling with recall, the performance is acceptable.

The CNN model, despite showing potential with the strong balance of precision and recall, is outperformed by the XGBoost and RF models, the main reason for this being the computational expense and longer training times, leading to the model being much less fine-tuned than the ML counterparts. The large number of hyperparameters and epochs needed for tuning made grid search and optimisation impractical to perform on the local system. So, whilst this study doesn't show the benefits of the applications of DL models in fraud detection when used on their own, it does show the main drawback: the application of deep learning is a big challenge in scenarios where computational resources and time are limited.

However, this result was used as a basis to take and improve the CNN performance to get the most out of the research and all models developed.

5.3 Feedback and Iterations

Using the results from the ML and DL models, and performing research to understand how to take the findings and models developed and create the most effective, robust fraud detection model, the conclusion was to use an ensemble approach. This would involve combining the CNN model with the XGBoost model, the results of this were as follows.

Ensemble Approach

Dataset 1

```
Ensemble Model - Best Threshold: 0.9988
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     70814
           1       0.96      0.77      0.85       118

    accuracy                           1.00     70932
   macro avg       0.98      0.89      0.93     70932
weighted avg       1.00      1.00      1.00     70932

Average Precision-Recall Score: 0.8166
```

From the combination of XGBoost and the CNN model, there is a slight increase in the performance of dataset 1, with the precision being improved.

Dataset 2

```
Ensemble Model - Best Threshold: 0.1841
              precision    recall  f1-score   support

           0       0.99      0.99      0.99    142469
           1       0.79      0.65      0.71      5166

    accuracy                           0.98    147635
   macro avg       0.89      0.82      0.85    147635
weighted avg       0.98      0.98      0.98    147635

Average Precision-Recall Score: 0.7332
```

In this dataset, we can see the benefits are more pronounced, with both the precision and recall improving on the dataset that every model struggled with when used on their own.

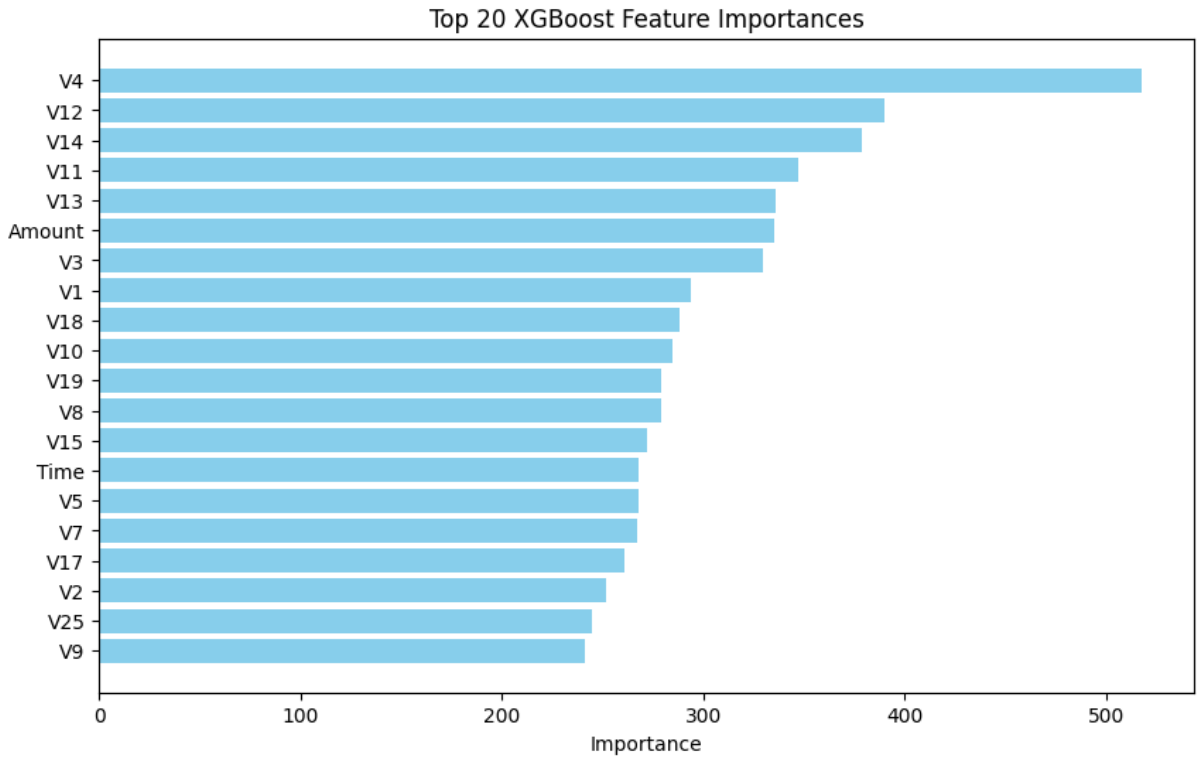
Dataset 3

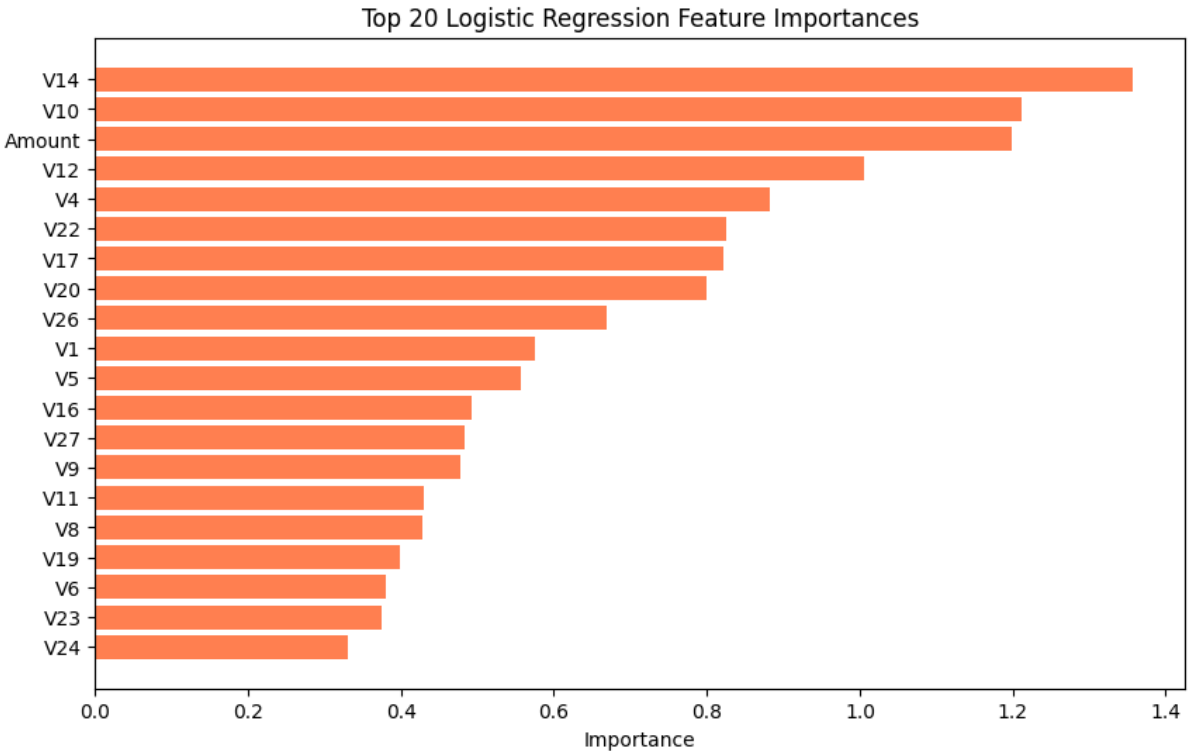
```
Ensemble Model - Best Threshold: 0.9971
              precision    recall  f1-score   support

           0       1.00      1.00      1.00   1588602
           1       0.94      0.90      0.92      2053

    accuracy                           1.00   1590655
   macro avg       0.97      0.95      0.96   1590655
weighted avg       1.00      1.00      1.00   1590655

Average Precision-Recall Score: 0.9666
```

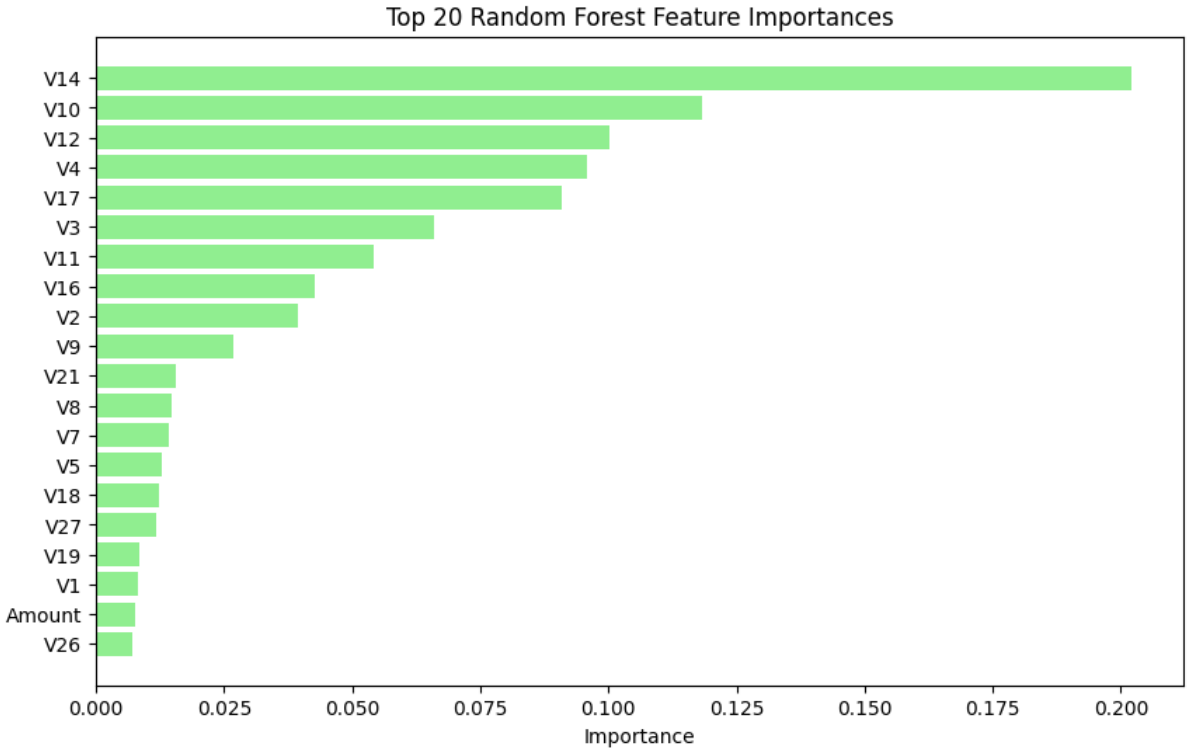
Dataset 3 gives the most improvement, with class – maintaining its perfect accuracy with class 0, and class 1 giving exceptionally high precision, and recall. High scores and a strong balance between accuracy and recall show that the best approach to fraud detection is achieved through this ensemble approach.

5.4 Data Analysis

Dataset 1

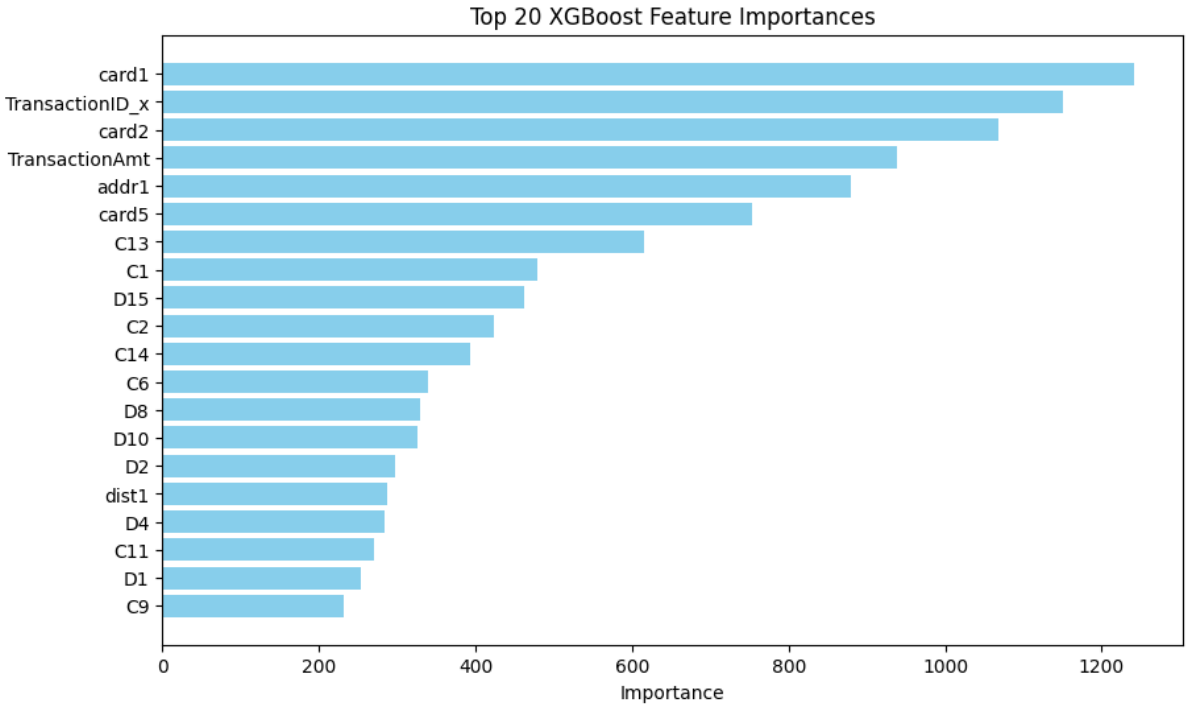


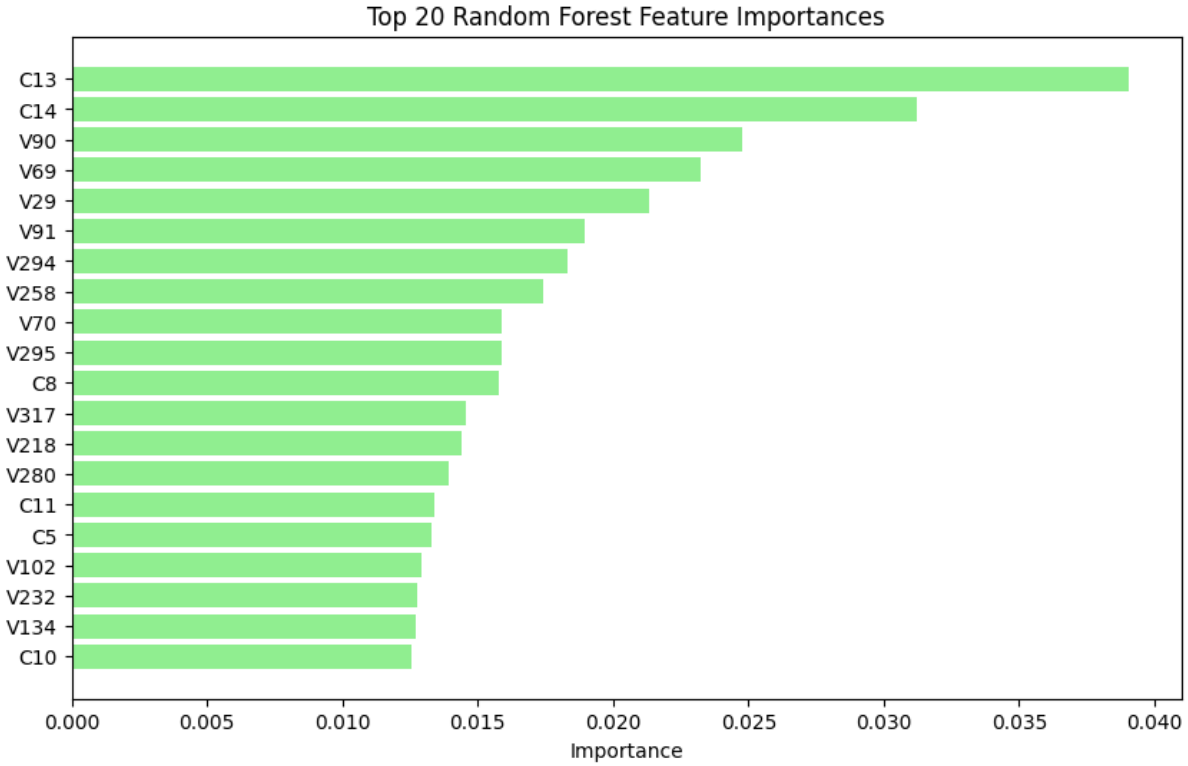


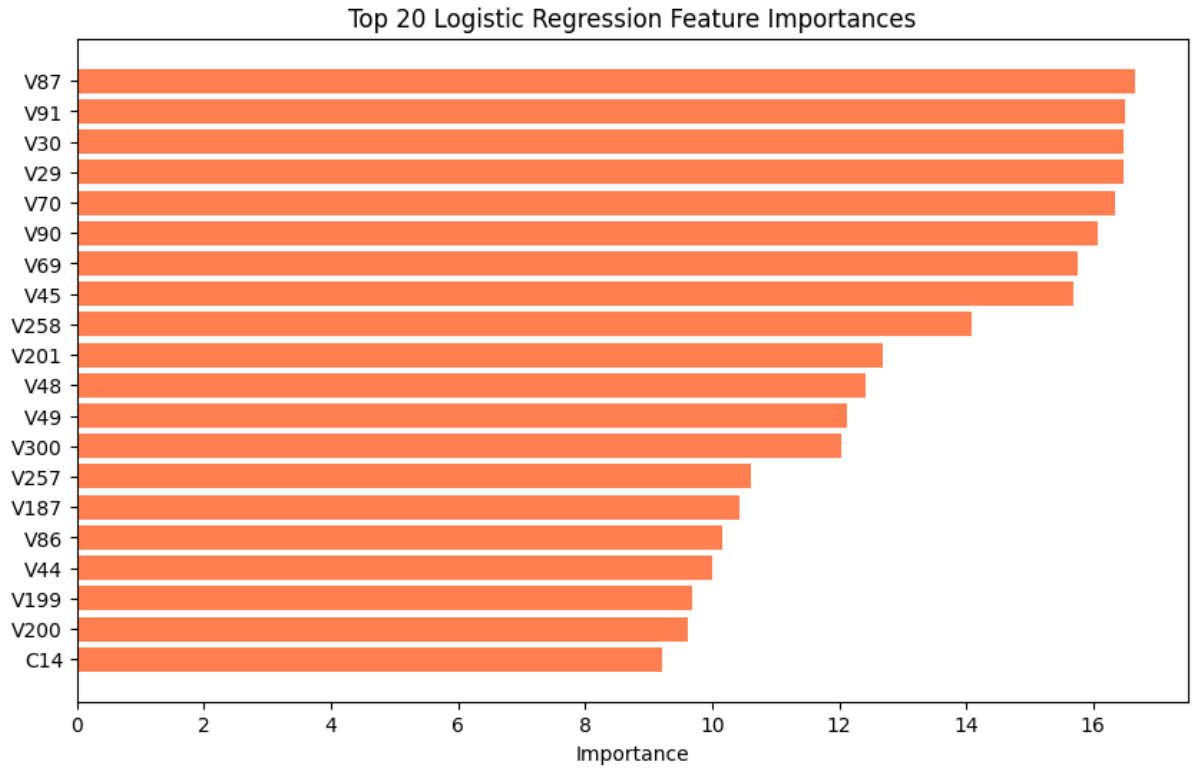


For this dataset, the most important features across each of the models were related to anonymised variables. The amount feature was also important to LR and XGBoost, suggesting that transaction size is an important factor in these models. Since the remaining variables are anonymised, it's hard to go deeper into the meaning of each individual meaning.

Dataset 2

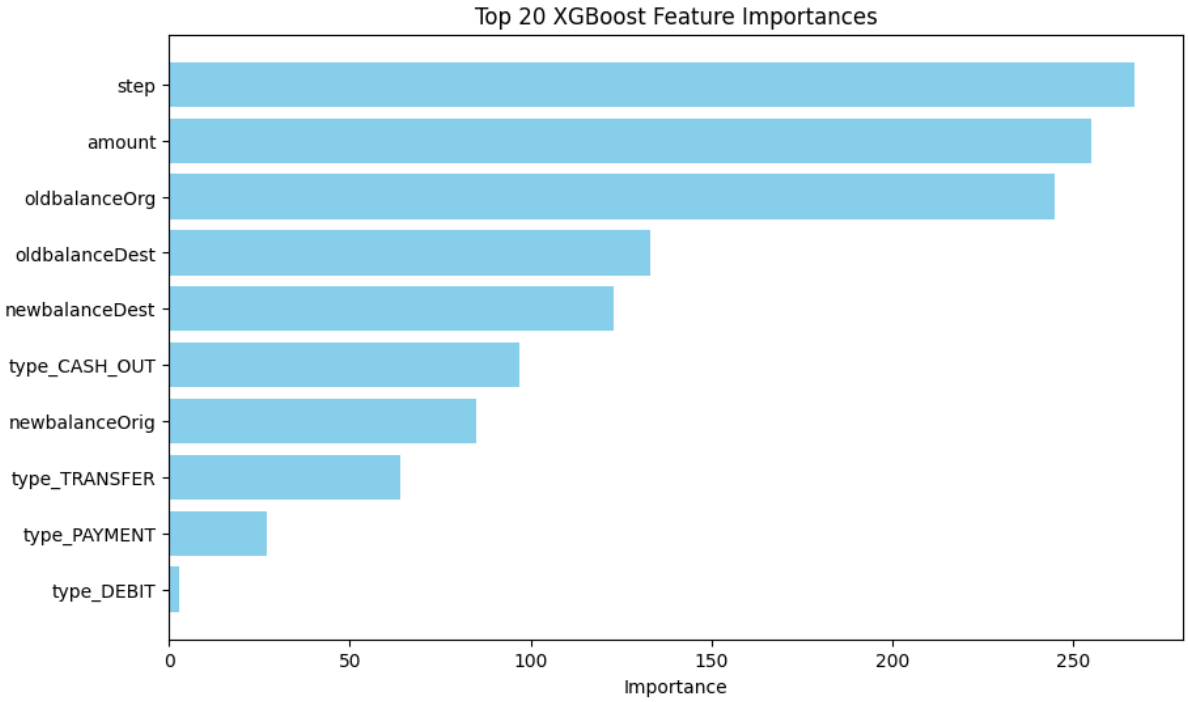

Top 20 XGBoost Feature Importances
card1
TransactionID_x
card2
TransactionAmt
addr1
card5
C13
C1
D15
C2
C14
C6
D8
D10
D2
dist1
D4
C11
D1
C9
0
200
400
600
800
1000
1200
Importance

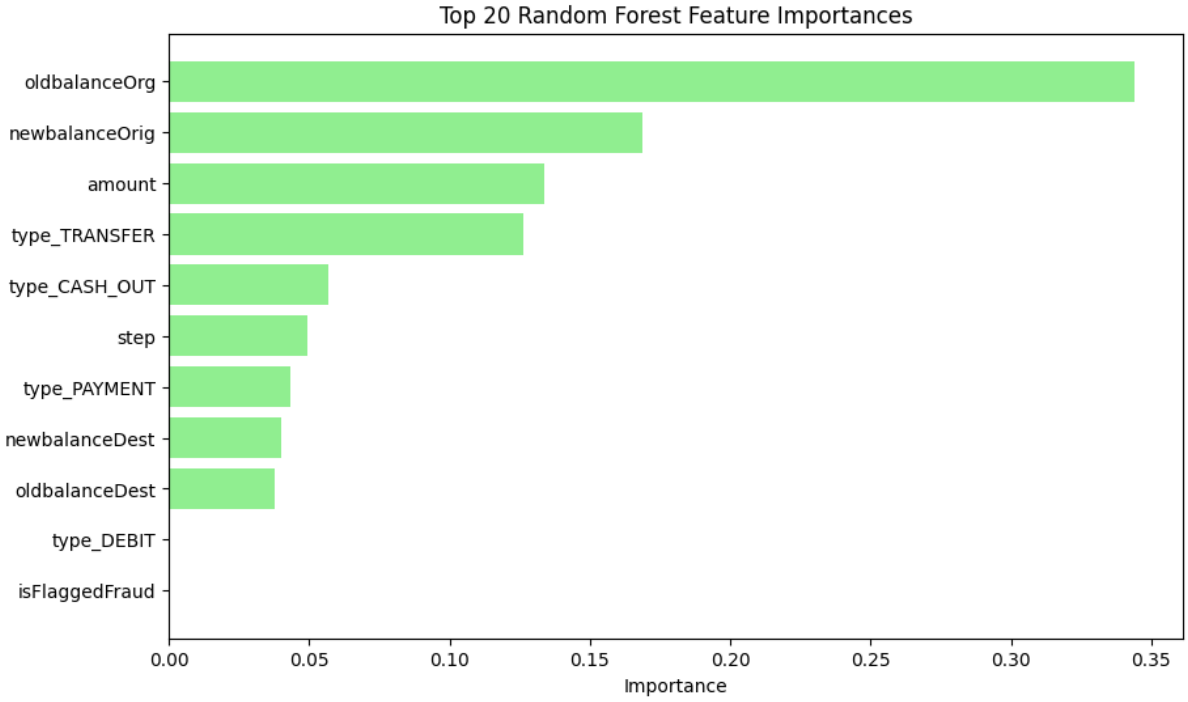

Top 20 Random Forest Feature Importances
C13
C14
V90
V69
V29
V91
V294
V258
V70
V295
C8
V317
V218
V280
C11
C5
V102
V232
V134
C10
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040
Importance

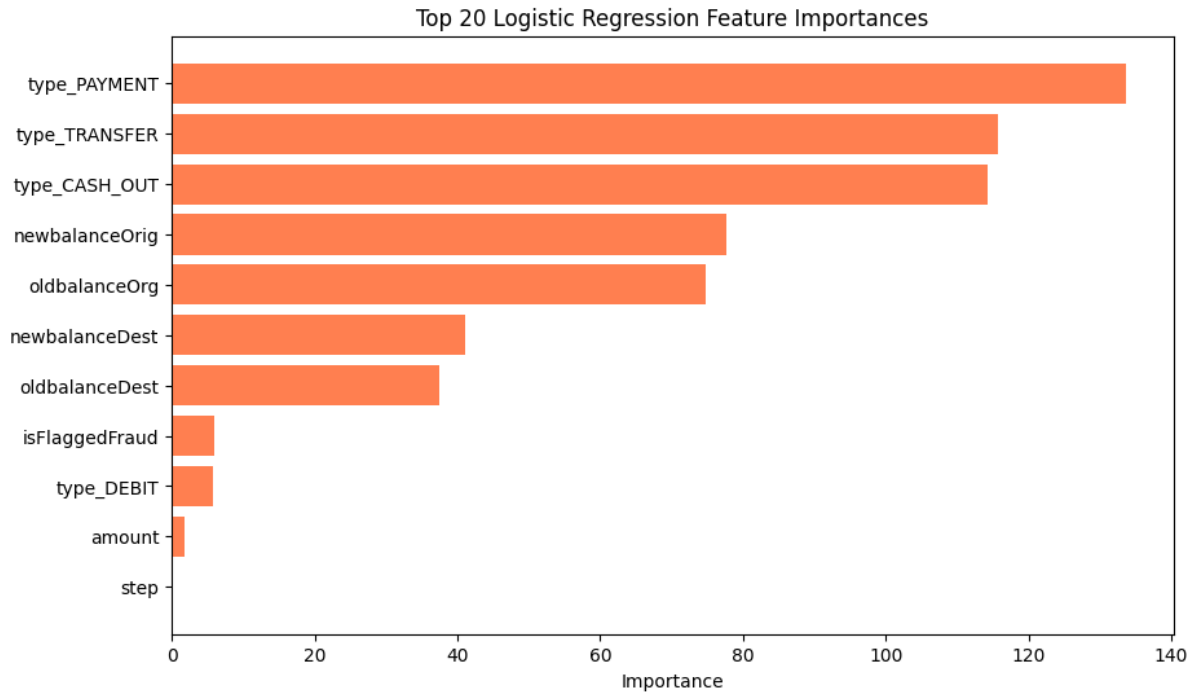


Dataset 2 had by far the most features (over 400) which contributed to each model performing significantly worse compared to the other datasets used. As is evident from these graphs, the models struggled to identify a clear, useful set of features, possibly as a result of the size of features in this dataset, having too much noise for these models. Additionally, the size and complexity of this dataset resulted in computational challenges, making it harder to fine-tune each model.

Dataset 3


Top 20 XGBoost Feature Importances
step
amount
oldbalanceOrg
oldbalanceDest
newbalanceDest
type_CASH_OUT
newbalanceOrig
type_TRANSFER
type_PAYMENT
type_DEBIT
0
50
100
150
200
250
Importance


Top 20 Random Forest Feature Importances
oldbalanceOrg
newbalanceOrig
amount
type_TRANSFER
type_CASH_OUT
step
type_PAYMENT
newbalanceDest
oldbalanceDest
type_DEBIT
isFlaggedFraud
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
Importance



The models performed the best on dataset 3. Features such as oldbalanceOrg, newbalanceOrg were identified as significant across each model, highlighting the use of knowing balance and transaction amounts in identifying fraudulent activity. The dataset had the least, but also the most informative features. The models were able to identify relationships within these features more easily than the other datasets.

Comparative performance across datasets:

Dataset 3 consistently had the best performance for every model, due to its clearer structure, with the transactional features being picked up by the models easier. Having fewer total features allowed the models to focus on the relevant patterns without being overwhelmed by noise.

Dataset 1 performed quite well, but worse than dataset 3, struggling with recall in the minority class.

Dataset 2 provided by far the most challenges. The large number of features introduced noise and the models struggled to distinguish fraudulent patterns.

Conclusion

Analysing the graphs and the insights they give help explain the differences in the performance of each model. Dataset 3 provides the clearest, most interpretable features that allow for good performance. Dataset's 2 overloads of features and complexity in the data overly hindered the performance, with Dataset 1 falling somewhere between, benefiting from strong features but occasionally missing some fraudulent cases where the models trained on dataset 3 didn't. The analysis of these datasets highlights the importance of data being well-structured for models to achieve the highest performance in detecting fraud.

5.5 Final Evaluation

The Machine Learning models XGBoost and Random Forest demonstrated they were able to perform well in both classes after undergoing sufficient hyperparameter training, particularly on datasets 1 and 3 which were more structured and allowed the models to identify patterns in the data more effectively. XGBoost performed better of the two making it the standout model for precision, recall and balance. Logistic Regression, the simplest and most computationally efficient underperformed at dealing with the minority class.

The Convolutional Neural Network provided mixed results. While it was able to perform decently in datasets 3 and 1, it was still worse than the XGBoost and RF models. The longer training times and the need for extensive hyperparameter tuning significantly impacted the results and feasibility of using CNNs for this study, given the computational and time demands.

The results of the ensemble approach however show that despite the challenges in applying CNNs with limited resources, it is achievable. This model achieved the best results for each dataset tested, with good precision and recall.

Overall, the systems developed in this project met the original objectives: building a robust fraud detection system model using both traditional ML and DL methods, with the ensemble approach being the biggest success. The use of the different models demonstrated a range of effectiveness in detecting fraudulent activities across the different datasets, with each model contributing differently to the project outcomes. However, the results did highlight that there are specific areas for improvement, especially with optimising the CNN performance on its own and handling complex datasets like dataset 2. Other potential improvements could include more robust feature selection, focusing on model interpretability and refining the process of hyperparameter tuning to enhance performance while reducing computational overhead.

Chapter 6: Discussion

6.1 Comparative Analysis

Model Performance Across Datasets: XGBoost was the best-performing model consistently over each dataset, performing particularly well in datasets 1 and 3, where it managed a high precision, recall and a good balance between the two. It showed a good ability to handle structured data which made it especially effective with these two datasets. XGBoost however still performed poorly with dataset 2, where too many features brought noise into the model.

Random Forest also performed well, just marginally being outperformed by XGBoost, lagging behind slightly and struggling even more with dataset 2. Despite this, the performances amongst the well-structured datasets, make it a good alternative to XGBoost when

accounting for its slightly lesser computational intensiveness and ability to natively use parallel processing.

Logistic Regression, though much less computationally expensive, underperformed significantly compared to the other models. Its notable weakness in identifying the minority class across all datasets shows the limitations of linear models when detecting complex patterns, such as those found in fraud detection.

Feature Importance Insights gained from this project provide deeper insight into how the different models interpret data, for example, key transactional variables like balance differences and transaction types were highlighted for dataset 3. These features were useful for identifying the fraudulent transactions, leading to dataset 3 having the highest performance, and helping to explain why the noise from excessive features in dataset 2 overwhelmed the models resulting in poor performance.

Trade-offs: The CNN model, despite its potential for capturing the complex and more subtle patterns in non-linear relationships, was outperformed by the XGBoost and RF models, and due to its higher computational cost and longer training time the CNN's performance was hindered by the challenges of fine-tuning the hyperparameters and running a high number of epochs in a resource-limited environment. This is what made XGBoost more practical in the context of this study.

The ensemble method combing the XGBoost and the CNN models returned the best overall performance in this study across all datasets. By using the precision of XGBoost in handling the structured data and the CNN's ability to capture the more subtle patterns in the data, the result was an improved detection rate, this approach proved particularly capable with dataset 3, achieving an almost perfect precision and an exceptionally high recall. Combining these two approaches and addressing some of their individual weaknesses was effective, but it was also the most computationally intensive model, requiring significant processing time and resources to train making it less feasible when resources are limited. This does, however, mean that if more time and resources were spent, this model could potentially improve on results that are already very good.

6.2 Key Findings:

The findings of this project show that XGBoost was the best ML model, outperforming the others across structured datasets, especially dataset 3, where scores for precision, recall and F1-score were high. Random Forest was not far behind, also performing quite well but ultimately wasn't able to match the high performance of XGBoost, especially struggling with dataset 2, due to this dataset's noisiness from too many features. Logistic Regression, while being computationally inexpensive struggled to effectively identify the minority class in all the datasets, highlighting how limited this model is for use in fraud detection.

The Convolutional Neural Network model, while potentially powerful for non-linear relationships, wasn't able to perform as well as was hoped. The main reason for this is the computational expense, and the difficulty and the time required in tuning the hyperparameters. The ensemble approach which combined XGBoost and CNN yielded superior results to every other model individually, particularly excelling in dataset 3, where

the balance was strong and results were exceptional. The ensemble method is not a flawless model however, still unable to achieve particularly high scores in dataset 2 (though it still outperformed the other models by a fair margin) and it also has high computational demands.

6.3 Limitations

This project had several limitations that caused issues and impacted the performance of the models, namely the CNN model's computational expense and the lengthy time required to tune the hyperparameters optimally, performing a grid search was not feasible and as a result, achieving the full potential of this promising model was not fully explored. Additionally, the limited computational resources affected the feasibility of training the models with a higher number of epochs, hindering the ability to maximise the performance of each DL model's performance and evaluate larger or more complex models effectively.

Dataset 2, with a vast number of features introduced noise that the models struggled to cope with, causing difficulties in distinguishing between fraudulent and non-fraudulent transactions.

Lastly, the imbalance in the datasets, despite being addressed in the data processing to train each model through the use of SMOTE, posed significant challenges in training the models to be as effective as possible.

Chapter 7: Conclusion & Future Work

7.1 Conclusion:

The goals this project set out to achieve were to develop, evaluate and compare traditional learning and deep learning techniques for use in fraud detection, and to provide insights into what makes a dataset good for the training of these models. Through the use of three distinct datasets, the study showed that models like XGBoost and Random Forest outperformed the simpler Logistic Regression model. XGBoost in particular was the best overall ML model in the study, effectively dealing with the data and feature importance. Although this model was strong, it still struggled with dataset 2, which was too noisy with an excess of features.

The Convolutional Neural Network model, while theoretically powerful, was hindered by the computational resources and longer training times it demands. Without the ability to fully optimise the hyperparameters and run a high number of epochs as necessary, the CNN model did not outperform the XGBoost and RF models as initially expected. Despite this setback, the study still showed the CNN's use and potential to capture subtle patterns by combining this model with the XGBoost model in an ensemble approach and giving the best overall results of any model in this study, especially in dataset 3, where a strong score for both precision and recall was achieved.

For data preprocessing, techniques like SMOTE proved to be critical in mitigating class imbalance, which was a major challenge in each dataset used. Additionally, the feature

analysis highlighted how the variables play a role in the performance of the models, especially in dataset 3 where fewer, more informative features contributed to the high performance.

Overall, this project has demonstrated the effectiveness of using ML models for fraud detection when trained on well-structured datasets. The ensemble approach, combining the strengths of both the CNN and XGBoost provided the most robust solution, though the computational cost remains a key challenge for both DL models. This project met the goals of developing a robust, effective fraud detection system, but there is still room for improvements and optimisation, particularly for the DL models.

7.2 Future Work

To further expand upon this study, possible areas for further work include:

Deep Learning Model Optimisation: Optimising the CNN would enable more thorough and complete hyperparameter tuning and more training epochs. This would help to unlock the full potential of CNNs for fraud detection.

Feature Engineering and Dimensionality Reduction: After using the datasets as they were, without dropping the features to obtain insights, future work could explore feature engineering and dimensionality reduction from dataset 2, mitigating noise and enhancing model performance.

Alternative Deep Learning Models: Besides using CNNs, future work for this study could include the use of other DL models such as RNNs, particularly LSTMs. This would explore the potential DL has for use in fraud detection to a further extent.

Exploring other ensemble methods: The XGBoost-CNN ensemble approach showed very strong performance, future work could examine other ensemble approaches, such as stacking multiple diverse models or even exploring more complex ensemble architectures such as bagging or boosting and combining that with deep learning models.

Deployment in Real-World environments: Good next steps for the findings of this project would be focusing on deploying these models in a real-time environment, and exploring the challenges this would bring such as model latency, scalability and adaptability, learning more about how continuous learning and model updating would work.

Chapter 8: Project Ethics

This project does not involve any human participants, and as such, ethical considerations related to human subjects were not relevant.
The data used throughout the project was sourced from publicly available datasets, all of which were fully anonymised, ensuring no personal or sensitive information was included. As such, there was no risk of compromising individual privacy.

Additionally, all data was handled in compliance with legal and ethical standards, ensuring that the project met the required ethical guidelines for data usage in academic research. No additional ethical concerns arose during the project as it focused solely on the development and testing of machine learning and deep learning models for fraud detection.

Chapter 9: BCS Project Criteria & Self-Reflection

9.1 BCS Criteria

1. Practical and Analytical Skills: Applying deep learning and traditional machine learning techniques to a real-world problem.
    - The knowledge gained from the module Machine Learning and Bioinspired Optimisation (COMP532) was applied to implement and fine-tune the models.
    - Practical skills in data preprocessing and model evaluation were key to developing an effective fraud detection system.
2. Innovation and Creativity: Developing and comparing different models for fraud detection.
    - Innovation was displayed by comparing traditional machine learning models with advanced deep learning techniques.
    - The ensemble approach demonstrated creativity in addressing the limitations of each individual model.
3. Synthesis of Information: Using multiple datasets to provide a comprehensive solution.
    - The project integrated multiple datasets in order to develop a robust fraud detection model.
    - This provided insights into how models handle different datasets, structured and unstructured.
4. Real Need: Addressing the significant issue of financial fraud.
    - Financial Fraud is a critical issue with significant economic impacts
    - The models developed especially the ensemble approach, offer practical solutions for detecting fraudulent transactions in real-time systems.
5. Self-Management: Organising and executing the project independently.
    - The project plan, including the timelines and milestones helped with self-management.
    - Regular progress reviews ensured the project schedule was adhered to, demonstrating the ability to manage the project independently.
6. Critical Self-Evaluation: Continuously evaluating model performance and making necessary adjustments.
    - Model performance was critically assessed through metrics such as accuracy, precision, and recall, with adjustments made to optimize results.
    - The project involved regular evaluation points, allowing for reflection on successes, challenges, and opportunities for improvement.

9.2 Self-Reflection

This project gave practical experience in applying machine learning and deep learning models to fraud detection. Building on the foundational understanding of these techniques, this project allowed for the application of this knowledge, particularly in the challenges of balancing datasets and managing computationally expensive models like CNNs. Working with large and imbalanced datasets required these skills to improve to manage the class imbalance.

This project also allowed for bettering organisational and problem-solving skills. Setting milestones and evaluating model performance regularly allowed for the iteration and refinement of the models. The development of an ensemble approach, combining the CNN and XGBoost models demonstrated creative thinking to adapt to the challenges faced in this project.

Overall this project has led to significantly enhanced practical knowledge of ML/DL techniques and strengthened project management skills.

10: References

1. Fraud.com (2022) https://www.fraud.com/post/advanced-fraud-detection#Traditional_fraud_detection_methods
2. Juniper Research (2022) https://www.juniperresearch.com/press/losses-online-payment-fraud-exceed-362-billion/
3. Akash Gandhar et al. (2024) Fraud Detection Using Machine Learning and Deep Learning
4. Abdallah A, et al. (2016). Comparative Analysis of Machine Learning Algorithms for Credit Card Fraud Detection.
5. Bhattacharyya, S., et al. (2011). Data Mining for Credit Card Fraud: A Comparative Study.
6. Carcillo, F., et al. (2018). Financial Fraud Detection Using Deep Learning Techniques.
7. Dal Pozzolo, A., et al. (2018). Credit Card Fraud Detection: A Realistic Modelling and a Novel Learning Strategy.
8. Effectiv (2024). Fraud Detection Using AI in Banking. Available at: https://effectiv.ai/resources/fraud-detection-using-ai-in-banking/
9. Fujitsu (2019). Fujitsu AI Solutions for Business and Finance.
10. Gambo, D., et al. (2022). Convolutional Neural Networks Used for Insurance Claim Fraud Detection.
11. Gandhar, R., et al. (2024). Fraud Detection Using Machine Learning and Deep Learning. Available at:
12. Gartner (2022). 2022 Gartner Market Guide for Online Fraud Detection. Available at: https://sift.com/blog/key-takeaways-from-the-2022-gartner-market-guide-for-online-fraud-detection
13. Jurgovsky, F., et al. (2018). Sequence Classification for Credit-Card Fraud Detection.
14. Jose, J., et al. (2023). Detection of Credit Card Fraud Using Resampling and Boosting Techniques.
15. Khando, A., Islam, M., and Gao, H. (2023). Trends in Fraud Detection Using Artificial Intelligence and Machine Learning. Available at: https://www.mdpi.com/1999-5903/15/1/21
16. Le Borgne, Y., et al. (2015). A Survey on Credit Card Fraud Detection Using Machine Learning.
17. Mastercard (2024). Mastercard Supercharges Consumer Protection with Gen AI. Available at: https://www.mastercard.com/news/press/2024/february/mastercard-supercharges-consumer-protection-with-gen-ai/
18. Nicholls, M., et al. (2021). The Evolution of Fraud Detection Techniques in a Digital Era.
19. Onix (2023). Machine Learning in Fraud Detection: PayPal's Implementation. Available at: https://onix-systems.com/blog/machine-learning-in-fraud-detection
20. Roy, A., et al. (2018). Deep Learning for Detecting Fraud in Credit Card Transactions.
21. Shah, D., and Sharma, L.K. (2023). Credit Card Fraud Detection Using Decision Tree and Random Forest.

22. Tookitaki (2023). The Power of Automated Fraud Detection Systems. Available at: https://www.tookitaki.com/compliance-hub/the-power-of-automated-fraud-detection-systems