

**A Post-graduate Dissertation**

on

**Real-Time Fraud Detection in Financial Transactions with Streaming Data Analytics**

by

**Sushma Reena Paul Raj**

**23011869**

from

**MSc Data Science**

**School of Mathematics, Computer Science & Engineering**

**Liverpool Hope University**

submitted on

**4th October 2024**

Supervised by

**Dr Mark Baxendale & Dr Neil Buckley**

[**barretm@hope.ac.uk**](mailto:barretm@hope.ac.uk)

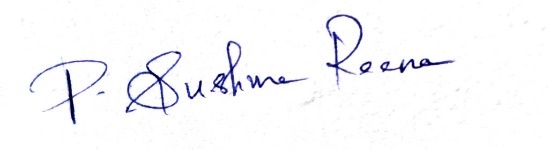
[**bucklen@hope.ac.uk**](mailto:bucklen@hope.ac.uk)

**DECLARATION**

This dissertation is submitted in partial fulfilment of the requirements for the MSc degree offered by the School of Mathematics, Computing and Engineering, Liverpool Hope University.

I confirm that this dissertation is my own work and wherever required I have acknowledged the work of others.

I also permit Liverpool Hope University to store a copy of this dissertation in a public archive.



Signature:

Date: 04/10/2024

**ABSTRACT**

This work aims at presenting real-time fraud detection using streaming data analytics with machine learning approaches. Given the popularity of digital transactions, the threat is even higher, and hence, there needs to be more robust and quicker means of identifying the same. In many cases, the traditional batch-processing approaches are not very useful to detect and prevent frauds as they are time-consuming This project seeks to adopt the application of streaming data analytics to track and analyse transactions in real-time fashion hence the ability to quickly identify a suspicious transaction.

This is done in such a way that it will incorporate machine learning models into a streaming analytics context where we will be using frameworks like Apache Kafka for the data ingestion process and Apache Spark for data processing on a real-time basis. This is because through training the models on the historical transaction data and giving them feedback in terms of transactions that were fraudulent, the system is in a position to pick out a new fraud that is emerging or even fraud that has occurred in the past. The project also pays attention to improvements related to data preprocessing, feature selection and model appraisal to achieve improved system performance.

By using simulated as well as real transactions, the efficacy of the system in terms of latency, speed, and usefulness shall be demonstrated with comparative analysis made with traditional batch processing techniques. The expected result is to enhance the efficiency of the fraud detection system by providing an ability to learn from the environment and adapt to dynamic risks that are not currently available in the existing approaches. In addition to the contribution to the field of fraud detection, this project provides valuable improvement to the use of machine learning with streaming analytics in financial systems.

***Keywords:*** *Fraud detection, Apache Kafka, Apache Spark, Python, Real-time, Machine learning, streaming Data Analysis*

**ACKNOWLEDGEMENTS**

On a personal level, I would like to thank my mother and the rest of my family for their support and encouragement throughout my childhood and during my studies. Their belief in me has helped me work extra hard and to the best of my ability and emerge a winner from each endeavour.

I want to express my deep appreciation for the education in India that has given me the base for that I began; to the teachers and tutors during the years of my diploma. These two people appreciated my desire for education and the hard work that put into my passion for learning as well as embracing the use of technology in society.

My special thanks to my university in Liverpool, UK for offering me the chance to expand my learning experience and meaningful research. I would also like to thank the professors and academic staff of my university who have contributed to this wonderful experience. Their knowledge, support and encouragement were very important towards achieving my success.

Another word of appreciation goes to the project supervisors for their guidance during the course of this project. They were very supportive of me at both academic and personal levels, which encouraged me to strive for the best.

Last but not least my final thanks to my friends in India and my friends in the UK, I dedicate your friendship to being a source of comfort, a helping hand and encouragement. You have been there in moments of triumph as well as in moments of failure and for that I am most grateful.

**CONTENTS**

Chapter I: Introduction …………………………………………………. 5

Chapter II: Literature Review …………………………………………... 10

Chapter III: Methodology ………………………………………………. 19

Chapter IV: Results and Discussion …………………………………….. 32

Chapter V: Conclusion ………………………………………………….. 34

Reference ……………………………………………………………….. 36

**CHAPTER I: INTRODUCTION**

The new face of financial systems has also brought about the conduct of most transactions online, which has the advantage of convenience, but also has many risks. Ever since the use of digital payments as the medium of transactions has gained much popularity among consumers and businesses, the rate of fraud has also increased and has presented significant risks to financial institutions, popular shopping platforms like e-commerce and to users. These fraudulent activities lead to financial losses in terms of monetary charges, and loss of customer's confidence and reputation. Therefore, the necessity of the perfect, efficient, and timely detection of fraud has never been so high.

Fraud, prevention is the ability to recognize situations where various persons or companies try to obtain money or property by unlawful means. While a transition to digital forms of transactions is taking place, fraudulent activities have been on the rise cutting across almost all sectors such as the banking and insurance industries as well as e-commerce industries. Activities like payment fraud, account takeover fraud, return fraud, ACH fraud, and chargeback fraud are more complex in nature and increasing rapidly which are difficult for business organizations as well as for consumers (Cameron, S. (2023)) (Kanade, V. (2021)).

Traditional approaches that have been applied in the detection of fraud involve the use of statistical data analytical tools such as regression analysis, probability distribution and data matching. These methods have proved somewhat effective but are generally inadequate for real-time operations as a result of the dynamics involved in transactions. Over time, the use of more sophisticated techniques of data analysis has been used in recent years such as machine learning, neural networks and artificial intelligence techniques. These approaches improve the knowledge of probable and potential fraud cases as well as the patterns that imply fraud from former data. Yet, most of these processes are still based on the batch approach that hampers the timeliness of detection and response.

As financial systems undergo evolution and improve in terms of sophistication and interconnectivity, conventional fraud detection techniques that mostly comprise of batch processing and rules of thumb do not suffice. These methods include batch processing of transactions and using known rules to look for suspicious activities. Although these approaches have proven useful in some ways, their implementation in the current world has failed. Batch processing results in time lags that enable fraudsters to exploit the accounting systems before instances of fraud are identified. Furthermore, the rule-based systems that are based on the pre-ordained sets of rules cannot be effective in tracking the new strategies of fraudsters. As these schemes become more proactive, traditional methods are no longer efficient, hence the need to get more active and real-time solutions.

Streaming data analysis presents a more effective solution by processing the data as it is produced in real time. While batch processing works on the principle of handling the data in batches and at fixed time intervals, streaming analytics processes data as soon as it is generated allowing quicker decision-making. Concerning fraud, real-time or streaming data analysis is most useful, since it supports immediate identification of fraud and addresses the issue before it surges even more. Further, streaming analytics can resolve the issue of a high volume of data, which is characteristic of high-transaction environment financial systems (Anderson, S. (2022)).

One of the central components of contemporary fraud detection systems is machine learning, which is a part of artificial intelligence that allows computers to discover and implement patterns from data on their own without being taught. Through the use of machine learning algorithms, a large amount of historical data of the transactions can be analysed in order to recognize the pattern of fraudulent activities. After the training process, the proposed model’s integration can be used for recognizing new transactions as fraudulent in a streaming data pipeline. The high degree of variability in the development of machine learning models makes it ideal for the fight against fraud since it does not remain frozen when new strategies are developed by fraudsters (Neural Technologies. (n.d.)).

This project will bring together real-time streaming data analysis with machine learning techniques to create a real-time fraud detection system. These technologies will be interlinked whereby the system will always be running the transactions, flagging any suspicious behaviour, as well as recognizing any new fraud trends in the process. By this approach, the project aims to improve the intensity and effectiveness of fraud identification itself, making the financial systems safer and stronger for the contemporary world.

Through data processing protocols like Apache Kafka in ingestion, and Apache Spark for real-time processing, this project aims to develop a scalable and strong system that is fit for the high volume and velocity of data common in financial systems. The objective here is to increase the accuracy and efficiency of fraud detection by a considerable margin while also making the online experience more secure for citizens. But beyond this, it provides insight into how these advanced technological tools can be used in strengthening fraud defence in the financial sector.

**Problem Statement**

With the advancement of technology that comes along with a new way of transacting financially, society has become vulnerable to fraud. The increase in e-commerce, mobile money, and banking activities has extended new windows for fraudsters to penetrate the loopholes in the financial systems. The continuously rising levels of digital business transactions call for sophisticated fraud prevention mechanisms capable of detecting and preventing fraud rapidly. There are often delays in the transaction data analysis, which follows traditional batch processing methodologies and may not be flexible enough to respond to the changing fraud strategies. These limitations can reduce the efficiency of fraudulent activity detection and increase the chances of losing significant amounts of money.

On the other hand, real-time fraud detection using streaming data analytics has been identified to possess the potential of solving all these by processing transaction data in real-time to identify and counter frauds as they happen. This study seeks to compare the efficiency of real-time detection and prevention of fraud using a streaming data analytical approach with a static batch processing approach. The performance measures such as the latency, the accuracy, the ability to adapt the model and so on will be explored in the study to decide whether real-time systems have better performance in the identification of fraud. The findings will show whether it is feasible to apply streaming analytics as a better approach to identifying fraud in complex and highly active areas.

**Research Objectives**

The following are the objectives included in building the system which is expected to give a perfect solution to the problem faced by the financial sector.

* **Evaluate Streaming Data Processing Frameworks:** Discuss many streaming data processing methodologies which include Apache Kafka and Apache Spark Streaming as well as deploy them to create a reliable real-time fraud detection system.
* **Develop and Integrate Machine Learning Models:** Develop machine learning models for streaming applications with the purpose of fraud detection and include them in the pipeline for further usage. Such models should be able to learn from past actual transaction data and its further training should be able to adapt to new and unknown types of fraud.
* **Compare Real-Time and Batch Processing Approaches:** Compare real-time fraud detection systems with batch processing methods. That is why the comparison will be carried out based on such characteristic parameters as latency, accuracy, as well as precision, recall, F-measure, and AUC-ROC.
* **Simulate and Test the System:** Conduct experiments of transaction streams that are proportional to real-time with synthetic and actual data. To evaluate how well the system can capture fraud, one must analyse the system’s response time and thoroughness in a simulation setting.
* **Monitor and Handle Model Drift:** Identify ways within the real-time fraud detection system to constantly track the performance of subsequently deployed machine learning models. Controlling concept drift: put measures that would help in updating the models to adapt to the changes hence continuing to detect fraud effectively.
* **Assess System Scalability and Adaptability:** Examine the applicability of a real-time fraud detection system to stream data flow in terms of a large amount of data. Assess the organisation’s ability to detect new fraud schemes and the ability of the system to cope with the fluctuations in the number of transactions.
* **Implement Real-World Deployment and Testing:** Use the developed fraud detection system in a live environment; it periodically evaluates the accuracy and latency of the system and modifies it accordingly to ensure maximum performance.

**Research Questions**

The following are the questions considered throughout the project and expected to be solved and answered at the end.

* In what ways, do streaming data analytics in real-time fraud detection systems fare in comparison with the conventional batch processing systems?
* Which of the machine learning algorithms can be effectively applied for real-time fraud detection and how it’s possible to incorporate it into Stream Data Processing?
* How does utilising streaming data analytic frameworks, like Apache Kafka and Apache Spark increase the effectiveness of the process for identifying and mitigating fraud in real time?
* How effective is the real-time fraud detection as compared to the batch methods in terms of the false positives and the false negatives of the process?
* What are some of the problems that are likely to be encountered when implementing real-time fraud detection mechanisms and how can they be prevented?

**Importance of the Study**

* **Revolutionizing Fraud Detection**: The goal of this study is to shift the paradigm of fraud detection from its conventional method of identifying and preventing fraud in high-transaction organizations such as banking, finance and e-commerce where there is usually rampant fraud activities and fraudsters’ activities are evolving and improving.
* **Need for Real-Time Detection**: An increase in online sales is putting more pressure on the necessity for real-time fraud detection as batch processing takes too long and results in significant losses.
* **Speed and Accuracy of Fraud Detection**: Through the application of streaming data analysis, this research will explore how real-time fraud detection will improve the rate and efficiency of identifying fraud.
* **Utilizing Advanced Streaming Frameworks:** The study will examine how efficient streaming technologies like Apache Kafka, Apache Spark and so on are for handling real-time transactional data.
* **Continuous Learning with Machine Learning:** Streamed data makes models update opioids automatically, enabling them to discover increasing fraud trends in high-speed and high-volume settings.
* **Minimizing Financial Losses:** The use of real-time fraud detection allows cutting the losses since the organizations respond to the scam right at the moment its operators try to cause other harm.
* **Improving Customer Trust:** Fraud detection in a shorter time is useful in improving customer confidence as the management of the business skirts fraud before the latter affects the customer in one way or another.
* **Comparing Real-Time vs. Batch Processing:** The study will also—as a result—give detailed insights into the comparison between real-time and batch-processing-based methods, thus revealing the specific advantages and the specific disadvantages when it comes to fraud detection using these two kinds of methods.
* **Contribution to Scalable Fraud Detection Systems:** The series of findings that the study presented will be useful for designing an improved and more scalable fraud detection system in order to meet the increased threat of digital fraud.

Further, this report will speak about the past work, the methods used to solve the above-mentioned questions, its results and future work.

**CHAPTER II: LITERATURE REVIEW**

As there is an increasing trend in digital transactions, the risks of fraud have also evolved and intensified. Real-time fraud has therefore become an essential tool of risk management for financial institutions, and businesses such as e-commerce that are prone to fraudulent activities. The incorporation of real-time streaming data and analytics, as well as machine learning models, is a strong way to address such issues to identify fraudulent transactions in real-time. This paper aims to review the most recent literature regarding real-time fraud detection and especially, how the use of streaming data analysis and machine learning techniques can improve the detection’s speed, accuracy and ability to handle large numbers of transactions. By comparing and contrasting a number of trends such as unsupervised learning, deep learning, or clustering-based approaches, this literature review aims to identify the best and more promising directions concerning real-time fraud detection and to follow the further development of the field.

**Challenges in Real-Time Fraud Detection**

**Data Imbalance and Complexity:**

One of the major challenges in fraud detection is data imbalance, where fraudulent transactions are far fewer than legitimate ones. Because of this imbalance, we get skewed models with a high False Negative Rate; that is, models which have low accuracy in detecting fraud. Zareapoor and Shamsolmoali (2015) deal with the common problem of data imbalance in fraud detection by using a Bagging ensemble classifier. Impudent transactions are much fewer compared to genuine ones – leading to biased datasets that may worsen models. The Bagging technique minimizes this by dividing the data into several sets, developing different decision trees on these sets and then getting their results. This approach aids in making the model more stable as well as in decreasing false positives and false negatives, making it possible to overcome the difficulties and the imbalance that characterise fraud detection.

Likewise, Prasad and Srikanth (2024) concentrate largely on addressing the data imbalance issue in real-time fraud detection with FROST (Fraud Oversampling Technique) boosting. It is also usually the case that fraudulent transactions make up a small share in the totality of the transactions which is why it is difficult for conventional models to detect them. FROST increases the detection efficiency due to an increased number of minority class samples, which results to a balanced sample data set for learning. This technique substantially cuts down the number of false negatives and guarantees that the model remains functional in real-time environments which addresses the two issues of data skewness and the multiplexity in real-time fraud detection systems.

**Real-Time Processing Requirements:**

In addition to data imbalance, real-time processing requirements is seen as another significant challenge in fraud detection. According to Thennakoon et al. (2019), real-time processing is very crucial in fraud detection especially when it comes to credit card transactions. This at the same time exposes that the failure to timely identify fraudulent activities may result into extensive losses. To this end, the authors deploy the application of machine learning models that enable rapid decision-making. Their approach is aimed at making sure that any fraudulent transaction is detected right at the moment it is being made so that the damage control is instant. It would be more suited for the current financial environment since it addresses risks on a real-time basis, while its system can be adapted seamlessly to today’s financial systems, which require quick decision-making.

Ahmad et al. (2017) showed that detecting an anomaly as and when it occurs is very effective in mitigating fraudulent practices especially when analysing streaming data. The study undertones that early detection of these irregularities is very important to avert fraud since delays can lead to great losses financially as well as an organization’s reputation. Based on the literature, the authors present an unsupervised anomaly detection system in which data is analysed in real-time as it is received as a means of employing effective counteractions against various suspicious actions. This focus on real-time processing helps make fraud possible to contain when it occurs, which is very important for modern anti-fraud detection systems. Altogether, these works show that the data imbalance and the necessity of real-time solutions are two major issues in fraud detection. Even if techniques such as Bagging and FROST can minimize the problems of data skewness, real-time machine learning models that are crucial in detecting fraud are important in an exception environment.

**Machine Learning Techniques in Fraud Detection**

**Ensemble Methods:**

Bagging ensemble classifiers are used by Zareapoor and Shamsolmoali (2015) to improve the credit card, fraud detection system. The Bagging method is an ensemble technique that combines multiple decision trees that were trained on different subsets of data resulting in higher classification stability and reduced number of false positives. This approach is more useful especially when it comes to solving the issues with the vast number of look-alike fraud and non-fraud incidents. such practices as the Ensemble method proved advantageous when models are hard to predict complex transaction patterns, thus the technique can be suited best for the fraud detection domain.

**Deep Learning Approaches:**

Chen and Lai (2021) used Deep Convolutional Neural Networks (DCNN) to improve the authenticity of modern credit card fraud estimations. Their work reported a 99% detection rate and is an example of how deep learning can capture fine-grained structures in transactions, which are otherwise invisible to other approaches DCNN is particularly suitable for real-time monitoring of a large number of transactions. In the same way, Sanober et al., (2021) used deep learning in wireless communications, it advancing a secure deep learning algorithm for fraud detection. This way, we show that deep learning applies to fraud in virtually any domain, ranging from finance to telecommunication.

**Clustering and Anomaly Detection:**

More recently, Habeeb et al. (2019) categorised and discussed clustering mechanisms for real-time anomaly detection in big data. Their approach is centred on something called anomaly detection where one looks for unusual patterns in sets of huge data, and that is useful in detecting fraud. Since the data points are grouped in clusters based on similarity and since the model can recognize one or more outliers, it is highly accurate in detecting fraudulent cases, even in situations that change frequently over time. It suits the circumstances of big data and accelerates as well as improves the identification of fraud-related issues.

Likewise, Huang et al. (2018) proposed the CoDetect framework that integrates network information and anomaly feature detection mechanisms. CoDetect also enhances the detection rates and eliminates the probability of false positives because it manages several data sources. This research proves that integrating clustering procedures with another approach to network anomaly detection can design effective and accurate fraud identification systems, P=0.0028; which underlines the need for the application of an interdisciplinary model to fight against fraud in different settings. Altogether, these machine-learning techniques show how the development of sophisticated models can enhance the prevention of fraud systems. Whereas the Bagging kind of ensemble methods help to stabilize the models in complex transactional environments, deep learning and clustering help to identify hidden patterns or real-time anomalous behaviour. This presents effective approaches for dealing with the upsurge of fraud in present-day computerized and financial environments.

**Big Data Analytics in Fraud Detection**

**Frameworks for Real-Time Data Processing:**

Wide-accepted big data frameworks are effective in providing support to real-time fraud detection solutions since it is possible to manage the transactional data set within a short time. In their systematic literature review, Mohamed et al. (2019) categorized and discussed a series of big data analytics frameworks and their significance in managing large and complicated data. This work forms the groundwork for explaining how such frameworks could be used to improve the existing capability of detecting fraud. Building on this, Armbrust et al. (2018) present Structured Streaming in Apache Spark as a giant leap in real-time data processing. Structured Streaming makes the construction of streaming applications lightweight, but at the same time provides high efficiency and low latency, which are critical for real-time fraud detection. Isah et al. (2019) follow this up by reviewing a range of distributed data stream processing systems such as Apache Spark and Kafka. Their work is focused on how those frameworks can be also scalable, fault-tolerant and efficient for constructing highly effective systems for fraud detection. By implementing these frameworks, organizations can acquire real-time analytics, so that fraudulent activities can be identified and dealt in much better way.

**Streaming Data Analytics:**

Real-time data analytical frameworks play a critical role mainly due to the fact that the data feeds are continuous and this makes it easy to detect fraud on a real-time basis. This Cao et al. (2019) illustrate in their description of TitAnt system that can detect real-time transaction fraud in online shopping platforms with equally amazing speed and accuracy. As this system uses streaming data analytics it gives its predictions in millisecond which gives a testification of the importance of such frameworks to address financial fraud effectively.

Thus, the use of deep learning techniques with the help of Apache Spark is described by Ileberi et al. (2022) to improve the detection of fraud in real time. There are two most important properties in Spark that they leveraged, were stream processing capability and the deep learning model; thereby getting an accuracy rate of more than 96% when it comes to detecting fraud transactions. This integration appreciates the feasibility of big data analytics in the processing of large and fast streams of data as used in the detection of fraudulent activities in the system. The two papers demonstrate the increasing importance of real-time data analysis in today’s world of fraud detection that is in some cases, in real-time.

**Real-Time Event and Anomaly Detection**

**Unsupervised Learning for Anomaly Detection:**

There is nothing as important as the unsupervised learning methods in anomaly detection if at all one has little or no access to labelled data. Real-time anomaly detection or UDAD as defined by Ahmad et al. (2017) is important in acknowledging or flagging new stream data that may harbor fraudulent behaviors. Their method promotes better abilities to fight fraud as it occurs since their method identifies shifts in real-time, a strength particularly important in environments where the patterns change frequently.

Continuing with the extension of the concept of real-time anomaly detection the study done by Hasan et al. (2018) uses event detection on Twitter data streams to provide insights. While their work isn’t aimed at financial fraud detection, the authors claim their approach could be used in other fields as well, focusing on the potential of the unsupervised learning method. In this paper, by exemplifying how they apply anomaly detection for real-time analytics of data from social media, their study can be useful to other researchers and practitioners to exploit unsupervised learning in such contexts, that require the continuous analysis of streams of data, and in several domains of interest. This connection show that the unsupervised learning methods can actually be employed not only in the sphere of financial fraud detection but also in the real-time problems where quick identification of the anomalies is required.

**Sliding Window and Clustering Algorithms:**

Ed-daoudy and Maalmi (2019) derive a sliding window anamorphosis algorithm most suitable for IoT and big data real-time anomaly detection environments. This method also pinpoints the proper handling of streaming data by using the sliding window technique for continuous computations, monitoring and updating the data in case any anomaly detection is done. Indeed, the clustering approach is used to group the data points so that one can easily be in a position to identify the outliers which may refer to some errant activities such as fraud. This method has been developed with the view of attaining efficiency in memory and speed of execution which is handy for real-time systems.

This approach echoes the clustering-based approach mentioned by Habeeb et al. (2019) in the conditions that clustering techniques can be used in real-time anomaly detection in big data environments. In both cases, clustering is demonstrated to be useful in handling large volumes of data and in identifying anomalies as and when they arise. As for Ed-daoudy and Maalmi, they concentrate on sliding windows to deal with streaming data, and Habeeb et al. consider a greater amount of techniques where scalability and effectiveness in searching for outliers in big data sets are discussed. Together, these papers demonstrate how the application of clustering algorithms can deliver effective real-time anomaly detection approaches across a range of sectors, but most notably where speed and high accuracy are supreme.

**Real-Time Fraud Detection Systems**

**Multi-Entity Fraud Detection:**

Prasad and Srikanth (2024) enhance real-time fraud detection with the Multi-Entity Fraud Detection System and where the datasets are imbalanced the Fraud Oversampling Technique (FROST). As FROST oversamples minority class (fraudulent transaction), it increases the accuracy of detection per unit of time, especially, true negative rates. The system addresses transactions between numerous entities and retains actual time processing when dealing with substantial networks of financial circuits. It emphasises the use of oversampling approaches in improving the efficiency and accuracy of the anti-fraud methods in real life.

**Predictive Modeling and Real-Time Detection:**

This paper considers Patil et al. (2018) stress on predictive modelling as an important step in credit card fraud detection. Thus, their approach uses the data from prior transactions and analyzes relationships between them to carry out a diagnosis and make a prognosis of fraudulent actions. The model can also, in effect, predict fraud by focusing on past activities and detecting it in real-time processes. This way of predictive modelling is essential when it comes to quickly flagging and denying fraudulent transactions before substantial companies lose money; this underlines the role of data analytics in improving the reliability and speed of fraud detection systems.

The TitAnt system was introduced by Cao et al. (2019) who affirm that it is highly effective for real-time fraud detection by predictive analysis. The system analyses transactional data in milliseconds and achieves an accuracy level of 99% as a probability of prediction. This demonstrates the efficiency of the system in exposing fraud within a very short time and reducing losses enormously. Drawing on the TitAnt system, it becomes possible to observe the benefits of using high-level computational methods for real-time analysis and prediction of economic fraud.

**Security Considerations in Fraud Detection**

**Security Threats in Fraud Detection Systems:**

Sanober et al. (2021) worked on improving the security of fraud detection systems in wireless communication networks by proposing a novel secure deep learning algorithm. On the same note, their approach provides a solution to the weaknesses normal systems have in identifying and mitigating fraud-related activities to organisations that are easily targeted through cyber threats. Using deep learning approach, the proposed model also enhances the fraud detection accuracy but enhances the system security against hacker intrusion. This research emphasises the importance of implementing enhanced security measures in fraud detection systems due to risks and threats.

**AI and IoT Environments:**

In their study, Choi and Lee (2018) suggest the application of AI methods in tackling financial fraud in the context of the modern and constantly developing IoT. More connectivity leads to the creation of sophisticated and more insecure systems within financial transactions. The authors accentuate how a whole range of AI, especially machine learning and deep learning algorithms can be used to analyse and detect fraudulent behaviours in real-time. Their work focuses on how AI can be incorporated into IoT to improve the capacity to detect fraud, so as the IoT environment grows, it is protected against fraud in the monetary sector.

**Integration of Machine Learning with Streaming Analytics**

**Case Studies of Successful Integrations:**

The use of machine learning (ML) in streaming analytics is a strong defence in the fight against fraud, primarily in real-time transactions. Zhu et al. (2020) described a well-structured machine-learning approach combined with real-time stream processing in credit card fraud detection. They proposed the use of a big data processing framework known as Spark and the integration of deep learning methodologies in an attempt to enhance the efficiency of Weighted Extreme Learning Machines (WELM) when addressing imbalanced classification problems. The integration obtained competitive accuracy levels higher than 96% in the training dataset, as well as in the testing dataset, which proves this hybrid model. This case study can be used to show how real-time streaming analytics can be integrated with high-powered machine learning to provide a strong solution against fraud that operates in real-time in complex data-driven environments. The effectiveness of merging has been observed in some specific cases, alongside future perspectives of the given integration, pointing out its relevance for the fight against fraud in the context of contemporary data-oriented conditions.

**The Future of Integration:**

It is also possible to conclude that the further development of integrating machine learning with streaming analytics for fraud detection will continue because of such factors as the further growth of new technologies and frameworks to improve the effectiveness of real-time data processing. Armbrust et al. (2018) described Structured Streaming in Apache Spark where it is possible to implement continuous computations utilizing a low-latency stream processing engine a critical component in real-time fraud detection. Likewise, Isah et al. (2019) provided an understanding of distributed data stream processing to which Apache Flink and Kafka belong, focusing on the fact that they constitute intensely scalable, failure-robust systems. Such enhancements indicate that subsequent models of fraud detection will be more versatile, extendable and capable of functioning in large-scale milieus, and dynamic data environments, possibly rendering them more efficient in detecting and combating fraudulent strategies.

**Summary**

We can infer from the literature review that the role of machine learning and real-time data processing is considered to be the key to fraud detection. These approaches solve problems of imbalanced datasets which is one of the main issues associated with fraud detection techniques such as Bagging and then overcoming the problem of creating a model for such imbalanced datasets while deep learning detects small patterns of transactional data and has high accuracy. There is also significant applicability of clustering methods in anomaly detection especially when applied in complex data settings for outlier identification. Therefore, there is a need for real-time analysis through streaming data analytics to allow for efficient identification of fraud and subsequent action. Specific integrations with Apache Spark, Flink, and Kafka show maximum advancement in the integration of ML with such frameworks enhancing the flexibility and accuracy of the use cases where the detection of fraud is required especially with growing data complexity. These models are important because fraud strategies change dynamically in any organization.

**Future Direction**

In future developments, there will also be a need for machine learning (ML) and real-time streaming analytics to address the problem of fraud. The future work will therefore be aimed at enhancing the efficiency, precision and flexibility to work on large, volatile datasets. Hence, an increased blending of multiple ML approaches along with distributed data stream frameworks such as Apache Spark and Flink has been observed to be a promising approach for the development of scalable and dependable real-time fraud detection solutions. Research will also be conducted on development of methods solving the problem of fraud detection in the growing volume and diversity of financial processes in the multi-channel environment. In cybersecurity, implementing principles used in data science as well as AI will be crucial in integrating elements of novelties that will help build new systems that are adaptive to new forms of fraud in real time. The future of fraud detection relies upon high-speed processing while at the same time, minimising false positives and negatives to prevent tremendous losses and generate customer trust. It is here that the ability to connect advanced ML with sizable streaming analytics platforms will become critical to this transition.

Using the insights gathered from the literature review, in this project, Apache Kafka and Apache Spark will be used together in the streaming pipeline for real-time fraud detection as a new approach. Several studies underline the constraints of batch processing in dynamic situations and stress the significance of real-time data processing to quickly detect fraudulent activity. Because Kafka and Spark can manage massive, real-time data streams, they are perfect for fraud detection systems that need to react quickly.

**CHAPTER III: METHODOLOGY**

This section contains the outline of the steps involved in the building of a website, which is represented in the following Flow chart 1. To achieve the goal of the project in an efficient way, Python is chosen to be used as the programming language with the help of Google Colab, which is an IDE for multiple programming languages and suitable for using the various frameworks directly online and gives easy access to them. The entire methodology will explain clearly their use in the project.

Exploratory Data Analysis

Data Pre-processing

Data Separation

Batch Processing

Sending the dataset to kafka

Algorithm Selection

Model building and saving for streaming pipeline

Loading kafka dataset processing using spark and prediction

Performance Evaluation

Flow Chart 1: Methodology

**Data Description**

The chosen data for this project is the Financial Fraud Detection dataset taken from Kaggle datasets. This dataset is an artificial construct of mobile money transactions which has been specifically derived to mimic real-life financial transactions with integrated fraudulent behaviours. The data set is obtained from real log files of a mobile money service operating in a selected African country, but most of the experiments reported here are generated from the PaySim simulator. Reducing dimension by a quarter so that the data is smaller for easy handling especially when analysing in Kaggle platforms. In it, there are different kinds of transactions and it covers a simulated time of 30 days. The dataset consists of 6362620 entries and 11 variables. This dataset consists of 10 input variables and 1 output variable and the target variable is isFraud. The following are the features in the dataset.

|  |  |  |
| --- | --- | --- |
| S.No | Features | Description |
| 1. | step | Represents a unit of time in the real world, with 1 step equating to 1 hour. The total simulation spans 744 steps, equivalent to 30 days. |
| 2. | type | Transaction types include CASH-IN, CASH-OUT, DEBIT, PAYMENT, and TRANSFER. |
| 3. | amount | The transaction amount in the local currency. |
| 4. | nameOrig | The customer initiating the transaction. |
| 5. | oldbalanceOrg | The initial balance before the transaction. |
| 6. | newbalanceOrig | The new balance after the transaction. |
| 7. | nameDest | The transaction's recipient customer. |
| 8. | oldbalanceDest | The initial recipient's balance before the transaction. Not applicable for customers identified by 'M' (Merchants). |
| 9. | newbalanceDest | The new recipient's balance after the transaction. Not applicable for 'M' (Merchants). |
| 10. | isFraud | Identifies transactions conducted by fraudulent agents aiming to deplete customer accounts through transfers and cash-outs. |
| 11. | isFlaggedFraud | Flags large-scale, unauthorized transfers between accounts, with any single transaction exceeding 200,000 being considered illegal. |

**Special Considerations:**

**Privacy Handling:** For securing the financial details the columns oldbalanceOrg, newbalanceOrig, oldbalanceDest and newbalanceDest are set to zero which benefits fraud detection tasks to utilise other attributes except the balance-related ones.

This dataset is beneficial to researchers in the area of financial fraud especially in designing mobile money fraud detection algorithms since they get a real-life scenario when designing the algorithms (Eedala, S.H. (2024)).

**Data Separation**

As the project deals with real-time streaming analysis, a separate stream of data is needed to be used in emulating and evaluating the performance of the streaming algorithms. However, it cannot use an online database because of financial issues that limit the number of publications we can access. Fortunately, we can work with a vast amount of data in this case as even after partitioning a part of the data for the purpose of streaming it still remains significant in size. To address this, we developed a custom function, train\_data\_stream\_split, to effectively divide our existing dataset into two components: a training dataset and a simulated data stream. The training dataset is used for training the model and the data stream dataset is used as the streaming dataset in real time.

By using the fixed random seed with the value 42, it is made sure that the same split is produced whenever the function runs on the same dataset. To eliminate the bias, the data indices are randomly permutated before data analysis is conducted on them. This helps in sampling our training data as well as the streaming data to tackle the entire dataset. We define a split ratio usually 80% for training and 20% for streaming though this can be adjusted. This ratio enables us to set aside sufficient records to train while having an adequate amount for simulating streaming mode. Then, after shuffling the data, the split point has been defined.

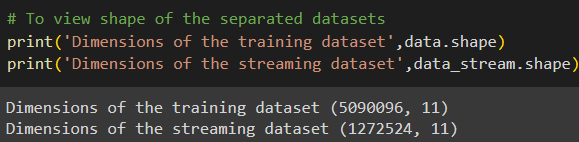


Figure 1: Showing the separated dataset

**Data Pre-Processing and Exploratory Data Analysis**

Data Pre-processing is the process of preparing the raw data in a way that can be used by machine learning algorithms to build the best model that gives the optimal solution for the problem that has been chosen to solve. It is most important for the machine learning algorithm because it reduces its complexities. It involves several steps like handling null values, removing outliers etc., that can be chosen based on the nature of the dataset (Simplilearn. (2023)). The following are the processing steps used to modify the data to be suitable for model building.

As a first step in making the data comfortable, renaming the column has taken place which is not so necessary for the model building. However, it will help the processor by giving a little more clarity to the data. Removing duplicate data is the next step which has been used. By removing the duplicates, some amount of storage can be saved and can reduce the size of the dataset, which will be more helpful when dealing with a large dataset like the project does. It can also improve the performance of the machine learning model, especially with large datasets (Sagacity Solutions. (n.d)). The dataset used in the project does not contain duplicate entries. As most of the machine learning algorithms do not support the dataset with missing values, null values need to be detected. Fortunately, this dataset does not contain null values.

The presence of outliers in the dataset may cause overfitting during model building and affect the model performance. So, a boxplot was used to detect the outlier by importing very important data visualisation libraries matplotlib and seaborn. After visualising the plot, it was found that all the numerical features in the dataset contain outliers. It is evident in the figure 2. Due to the realistic nature of the project where the data is translated into real situations that may arise in application, the outliers were kept in a bid to know the performance of the model when extreme events occur such as an increase in usage, a malfunctioning sensor or occurrences related to finance for instance, price shock events among others. Outliers are important in constructing a model that is stronger and more resistant to changes in live data streams that may occur unexpectedly. They were not removed from our model but rather treated by employing deviation decisions such as decision trees and ensemble classification or real-time outlier detection (Anglen, J. (2024)).

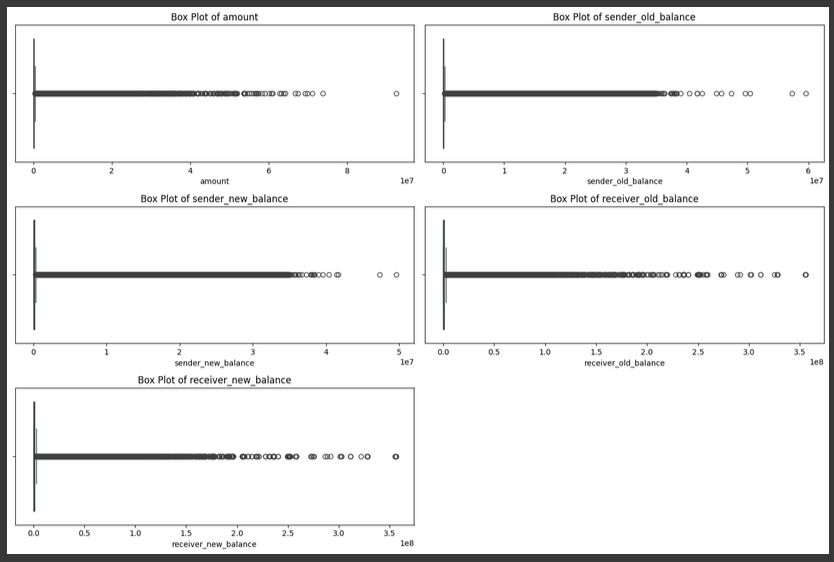


Figure 2: Boxplots of numerical features

When two features ‘isfraud’ and ‘isFlaggedFraud’ were compared, it was inferred that there is a huge difference between the actual fraud and illegal transactions which were flagged as fraud. There are only 15 flagged frauds whereas there are 6610 actual frauds in the training dataset from a total of 5090096 transactions. So, it was decided to remove ‘isFlaggedFraud’, an unnecessary feature with little information. To make the machine learning model understand the relationship between sender and receiver easily, a new column ‘type2’ was created which is filled according to the conditions depending on transaction participants whether ‘origin’ and ‘destination’ are customers (C) or merchants (M). As a result, we have ‘CC’ for customer to customer, ‘CM’ for customer to merchant, ‘MC’ for merchant to customer and ‘MM’ for merchant to merchant. This new feature gives the model the capability to understand the relationship of participants in a transaction and this maybe may show some pattern of fraud. Then, the action of switching the position of the newly created ‘type2’ column to the second position for better dataset management and comprehension. Then those two features were removed. Instead of dealing with a large number of different customer and merchant IDs in two different columns, this new column will be helpful with understandable patterns and useful in enhancing the chances of obtaining fraudulent transactions.

From the data description, it is clear that the feature ‘step’ corresponds to the order of the hour, which is when the step equals 1 means it is the first hour of the first day of simulation which is 1 am and step equals 744 is the last hour of the simulation. For further convenience, this column is processed as two separate columns ‘hour’ by grouping the same hour and ‘day\_of\_week’ with entries of day groups. However, we won’t get an exact hour and day, they are still useful with some clear information. To achieve this, the DataFrameMapper from the sklearn\_pandas library is used to preprocess the dataset by transforming each column according to a certain rule. To be more precise, DataFrameMapper uses a list of tuples to map preprocessing jobs to each column in a given dataset. Every tuple in the input list relates to a certain data frame column (Github.io. (2019)).

The custom transformer StepToHour and StepToDay respectively convert the step column to hour and day of the week to enhance the Proposed Attributes and other time-related attributes. The RecategorizeType class recategorises the type column into ‘CASH\_OUT’ and ‘TRANSFER’ or puts all transactions in one group under ‘OTHER’ to make categorisation easier. These transformations can be combined using the DataFrameMapper, where type and type2 are binarized using LabelBinarizer while the columns as amount, sender\_old\_balance and receiver\_new\_balance stay the same. A useful tool for converting categorical labels into binary vectors is Label Binarizer, a utility class offered by the scikit-learn toolkit. To change labels into a format that works with machine learning algorithms, it uses a one-hot encoding approach (DataScience-ProF. (2023)). This flow of data enables all possible data preprocessing procedures to be completed in one cycle where both categorical and numerical data are prepared for other possible processing methods in machine learning including model training. As the dataset is large, it was not able to visualise and get inferences properly. However, some visualisation through the plot and just numbers are used to explore the data. Those analyses were very useful for data pre-processing. So, they explained above along with the pre-processing step and even all along the methodology. Figure 4 shows the transformed dataset after data pre-processing.

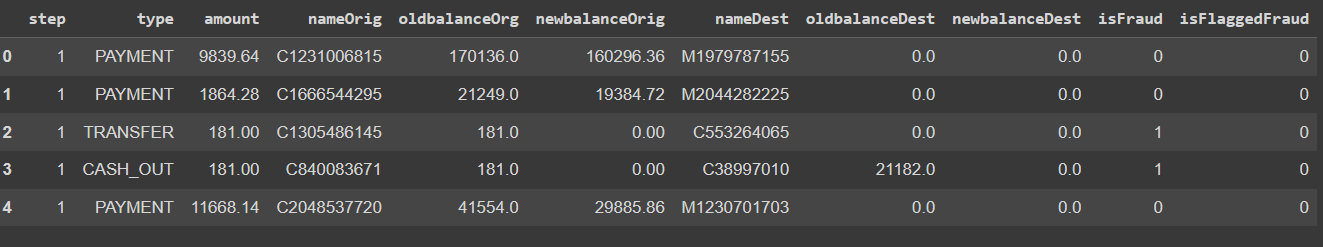


Figure 3: Before Data Pre-processing

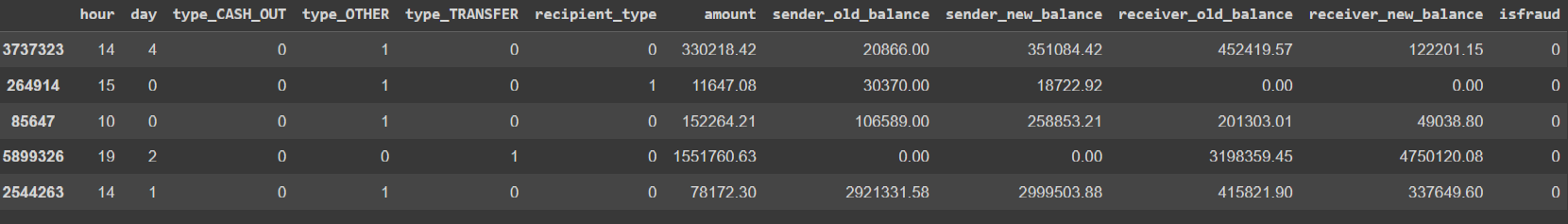


Figure 4: After Data Pre-processing

**Algorithm Selection**

As for this Fraud detection problem of consent, there are some challenges that have to be dealt with. The challenges are as follows,

* **Imbalance Data:** A good example is when fraudulent transactions are comparatively fewer than genuine ones, resulting in highly imbalanced data. A lot of models fail to learn from this imbalance.
* **Non-linear Relationships:** The nature of the fraud schemes might not always be clear and might intersect with other features in various ways.
* **Evolving Fraud Patterns:** The fraud patterns may well behave dynamically, and hence, adaptability and flexibility may be regarded as important characteristics of the model.
* **Scalability:** Due to the large volume of financial data, significant attention is paid to algorithms that consume as little time as possible and minimal volumes of memory.
* **Interpretability:** In fraud detection especially, the mechanic must not only provide an output for fraud prediction but also explain why the specific transaction is flagged to build credibility for the model.

Among Machine learning algorithms, there are a few algorithms for classification problems. Among those classification algorithms, a few important algorithms have been selected that have fraud detection in their application. The selected algorithms along with their usefulness are as follows,

**Logistic Regression:** Logistic Regression is a well-known supervised learning algorithm used for binary classification tasks, including fraud detection. It's a linear model of the probability of fraud happening based on input features, assuming a linear relationship between the features and the log odds of the outcome. However, fraud detection presents challenges like imbalanced data with fewer fraud cases and non-linear relationships, making logistic regression less effective in capturing complex fraud patterns. While interpretability is a key advantage, allowing clear insights into how features influence fraud predictions, logistic regression struggles with imbalanced datasets, often favouring the majority class, which consists of non-fraudulent transactions. It also fails to handle non-linear interactions between features, common in fraudulent behaviour (Hosmer, D.W and Lemeshow, S. (2000)).

**Random Forest:** Random Forest is a powerful ensemble learning method widely used for classification problems, including fraud detection. One of the primary advantages of Random Forest is its ability to handle imbalanced data, which is common in fraud detection scenarios. The algorithm can effectively learn from the minority class which is fraudulent transactions while maintaining accuracy for the majority class. Also, Random Forest can capture non-linear relationships between features, making it adept at modelling the complexities of financial transactions. Its robustness to overfitting allows it to generalize well, especially in high-dimensional data environments. Furthermore, Random Forest provides feature importance scores, enhancing interpretability by identifying which features contribute most to the predictions. However, Random Forest can be computationally intensive, particularly for large datasets, resulting in longer training times. It may also experience slower inference speeds compared to other models, which is more important in real-time streaming data analysis (Chen X, Ishwaran H. (2012)).

**XGBoost:** XGBoost (Extreme Gradient Boosting) is a robust and widely-used machine learning algorithm particularly effective for classification tasks, including fraud detection. One of the main strengths of XGBoost is its efficiency with large datasets, allowing it to process vast amounts of data quickly while maintaining accuracy. The algorithm excels at capturing non-linear relationships between features, making it well-suited for the complexities inherent in financial transactions. Also, XGBoost is designed to handle imbalanced data effectively, which is a common challenge in fraud detection scenarios. Its built-in regularization techniques help mitigate overfitting, enhancing the model's generalization capabilities. Furthermore, XGBoost supports incremental training, allowing models to be updated as new data becomes available without retraining from scratch. However, XGBoost can be slower than LightGBM in certain scenarios, particularly when dealing with extremely large datasets, which may affect performance in real-time applications (Chen, T. and Guestrin, C. (2016)).

**LightGBM:** LightGBM (Light Gradient Boosting Machine) is a highly efficient gradient-boosting framework that is particularly well-suited for large-scale fraud detection problems. LightGBM is specifically designed for large-scale data processing, making it efficient and scalable for fraud detection in environments with vast and complex datasets. It includes built-in features to handle imbalanced datasets, such as the is\_unbalance and scale\_pos\_weight parameters, ensuring that the model can focus on detecting rare fraudulent activities. In addition to its speed and memory efficiency, LightGBM excels at capturing non-linear relationships, which are critical in uncovering complex fraud patterns. Furthermore, it supports online learning and incremental updates, making it ideal for real-time streaming data applications. LightGBM’s lower latency in predictions, compared to models like Random Forest and XGBoost, makes it particularly suitable for real-time fraud detection systems where quick response times are crucial (Ke et al. (2017)).

Among the discussed algorithms, LightGBM stands out as the most suitable model for fraud detection due to its combination of scalability, speed, and adaptability. Its ability to handle imbalanced data and quickly adjust to evolving fraud patterns makes it highly effective in real-time fraud detection scenarios. So LightGBM is selected for the model building this project.

**Batch Processing**

After selecting the algorithm, batch processing has been performed using the selected model. In this batch processing only machine learning is used. No extra tools like Hadoop, Apache Hive etc… are used. Here is the explanation of the implementation of the LightGBM model. First, to make the model work, the necessary libraries are imported. LightGBM is the primary machine learning library used in the code, which is more efficient for large datasets and is very well-suitable for handling imbalanced datasets, a common issue in fraud detection. Other than LightGBM some of the scikit-learn metrics including accuracy\_score function, classification\_report, confusion\_matrix, and roc\_auc\_score functions are imported to measure the model’s performance in terms of classification and the model’s efficiency in handling imbalanced classes (NVIDIA Technical Blog. (2022)). The dataset is split into two components: variables (X) and the dependent variable (y). The features are as described, where each of them describes some detail of the transaction carried out; on the other hand, the target variable isfraud identifies a specific transaction as fraudulent. The scikit-learn's train\_test\_split function is used to divide the data into training and test sets, with 67% of the data allocated for training the model and the remaining 33% reserved for testing purposes. This is quite common in machine learning to ensure that the model has the ability to perform well with unseen data.

To train the LightGBM model it needs its own data structure - lgb.Dataset. required for training and testing datasets to transform into this format where LightGBM can optimise its computation. The test dataset is passed as a reference for validation during training to assess the model's performance at each iteration. Defining the parameters for the LightGBM model is the next step. To control the various aspects involved in training the model needs a step to define parameters which help in controlling. So, the next step is defining those needed parameters. Firstly, the objective is defined and set to binary which helps the model perform a binary classification task which is fraud or not fraud. Next, the evaluation metric is set to binary\_logloss, which helps to measure the error in predicting probabilities. To indicate that the model will use gradient-boosted Decision Trees, the boosting method is set to gbdt. As it uses gradient descent method, the smaller the learning rate, the better the performance. So, setting the learning rate to 0. If there are more leaves in the tree, it leads to more complexity and may increase the risk of overfitting. So, Num\_leaves are set to 31. Next, the Max Depth is set to -1 to make the depth unconstraint. To set the minimum data points in one leaf, min\_data\_in\_leaf is set to 20. To suppress the warnings, the parameter verbose is set to -1.

To fit the LightGBM model, the training phase used the lgb.train function. The model is trained for more than 100 cycles, and the set of training and testing datasets is given below. LightGBM has both ‘valid’ and ‘test’ sets where the model is trained with the ‘valid’ set, but afterwards, the boosting round is evaluated on the ‘test’ set through applicable metrics so that LightGBM can correct its training based on its generalization to unseen data. After the training of the model, the model then estimates the probability of fraud in the test dataset by using lgbm\_model.predict. These probabilities express the probability of being fraudulent in every single transaction. For these probabilities, a 0.5 is chosen as a threshold, and the predictions thus generated will be binary (fraud or not fraud). Any transaction within this dataset with a predicted probability greater than 0.5 is considered fraudulent.

To evaluate the performance of the model, several metrics are used. First, Accuracy determines the overall degree of correctly classified transactions which proves to be unrepresentative in the case of imbalanced datasets such as fraud detection, where a vast majority belong to the non-fraudulent category. In such cases, high accuracy can be attained with the mere prediction of most transactions as non-fraud even when most of the fraudulent ones are not detected The Confusion Matrix supplies a more extensive and definitive outlook of the model in terms of true positive (accurately predicted fraud), true negative (accurately predicted non-fraud), false positive (Fraud suspected incorrectly) and false negative (Fraud not suspected at all). The above matrix plays an important role in analysing the performance of the model based on the fraud detection rate and false alarm rate.

The ROC AUC Score test (Receiver Operating Characteristic – Area Under the Curve) carries out further tests of the efficiency of the model in distinguishing between fraudulent and non-fraudulent transactions. It is a better measure than accuracy because it shows true positive and false positive rates any time a threshold is set regardless of the imbalanced nature of the data. Last but not least, the Classification Report displays the most important measures including precision, recall, and F1-score, which may provide knowledge into how much the model tends to be as effective as possible in detecting actual cases of fraud minimising the number of false positives at the same time, so it is an integrated approach to assessing the effectiveness of the model in the fraud detection domain (Agrawal, S.K. (2021)). Figure 5 shows the output of the above-explained implementation.

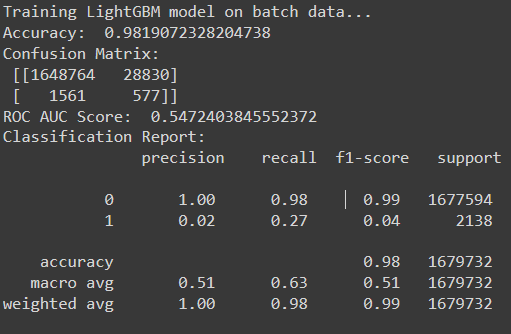


Figure 5: outcome of batch processing

**Real-time Streaming Pipeline**

As it was said already, two frameworks, Apache Kafka and Apache Spark are used here in the streaming pipeline. Apache Kafka is an Apache-licensed, distributed event streaming platform for publish-subscribe messaging to process a large number of events in parallel with fault tolerance and scalability in mind. In this project, it is used for data ingestion (Spiceworks. (n.d.)). Apache Spark is an open-source, distributed computing system designed for big data processing and analytics. It provides a unified platform for processing large datasets quickly, with built-in modules for streaming, machine learning, and graph processing. Spark is known for its speed, ease of use, and ability to handle both batch and real-time data (Databricks (2019)). To make them work and work properly together, suitable versions of both are installed. Here Kafka version 2.7 and pyspark version 3.3 are installed. For initiating and working with Kafka, the Zookeeper server and Kafka server are started. Zookeeper is used to coordinate the Kafka brokers; this involves making sure that Kafka brokers are in sync at any one time, selecting a leader and making sure that there is a fail-safe mechanism. The Kafka server is a broker which is responsible for scheduling a message, reading a message, dealing with partition and scaling as well as providing fault tolerance to producers and consumers in the Kafka cluster.

First, the model building which is suitable for real-time is designed using the LGBMClassifier from LightGBM, which is contained in the Scikit-learn pipeline, and the pipeline combines preprocessing and model building. First, handling the data maintaining a systematic approach the target column is removed and the data is divided into features and labels. The pipeline above takes advantage of StandardScaler to upscale the features this is essential more so when working with algorithms that are sensitive to feature scaling. The model is then trained on the training dataset, the model is then tested based on metrics such as classification report, accuracy, confusion matrix and ROC AUC score. This approach has its benefits since it can be easily saved using the joblib.dump() and permits the use of the trained model in real time and continuing predictions without retraining the model. Due to the pipeline structure, more preprocessing or modelling techniques can be added depending on the different machine learning workflow needs. In total, this approach improves the possibilities for usability as well as expands on the efficiency within the model deployment process. The figure below shows the outcome of the above-discussed approach.

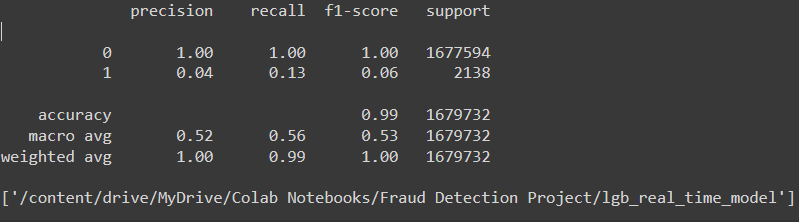


Figure 6: outcome of LightGBM Pipeline

Next, a partition of transactions of a financial nature is streamed to a Kafka topic called ‘fraud\_transactions. The dataset used here is the dataset which is separated from the original as data\_stream. It then starts a Kafka producer which is initialised with a connection to the Kafka server ‘localhost:9092’ Each transaction is serialised into JSON this is followed by pushing the output of the JSON serialiser through the kafka-topic-producer. It pretends a dataset by selecting out certain columns which are no longer needed, ‘isFraud’ and ‘isFlaggedFraud’. The dataset is split into the first and last 10,000 lines and then united. The producer converts each row into a dictionary, and streams these transactions to the Kafka topic in a loop, while also storing the transactions in a list for later use. To mimic real-time streaming, a 10ms delay is added between the consecutive transactions. Last of all, the producer forces the send of any remaining transactions, to ensure that all messages get through. Figure 7 is an evident to make sure that the dataset has been sent to Kafka successfully.

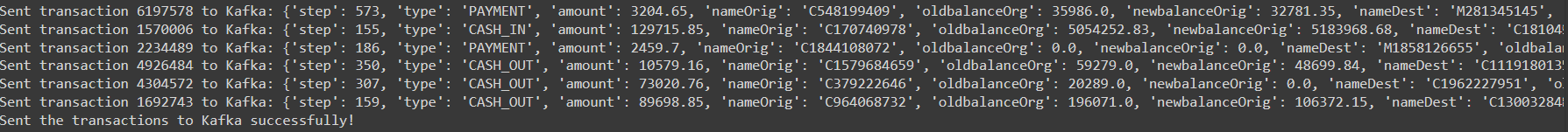


Figure 7: outcome of Data Ingestion

At the same time, a Kafka consumer is created for consuming messages from the same Kafka topic and each transaction message received is unserialized and forwarded to Spark. A session named Spark’s session is set to process the data. The data produced by Kafka will be in unstructured form. So, the data is parsed. The transaction is augmented with new features such as hour and day\_of\_week from the step field. It also uses one hot encoding to the type column or grouping transactions by their kind, like “CASH\_OUT” and “TRANSFER”, creating its own array called recipient\_type, that will be used to differentiate between customer and merchant transactions. These transformations correspond with the format of the data to the feature set desirable by the pre-trained LightGBM. The fraud detection model is loaded from a scikit joblib file to predict the probability of fraud for each customer transaction as a function of its features. The prediction function, UDF in Spark is implemented, where the LightGBM model is used to predict in real time the transaction data with the result of Fraud prediction with 1 if the transaction is fraudulent while 0 for non-fraud. The prediction results are added as new columns to the DataFrame which offers the details of the transaction and the prediction result. This setup represents a use-case of a streaming application where the Kafka producer streams real-time transaction data, the Kafka consumer processes this data by using Spark and transforms this data before feeding it to the LightGBM model which in real-time determines the occurrence of fraud. By constantly streaming and processing transactions, the system is designed to have a continuous fraud detection process. The following figure shows the prediction that happened in real time.

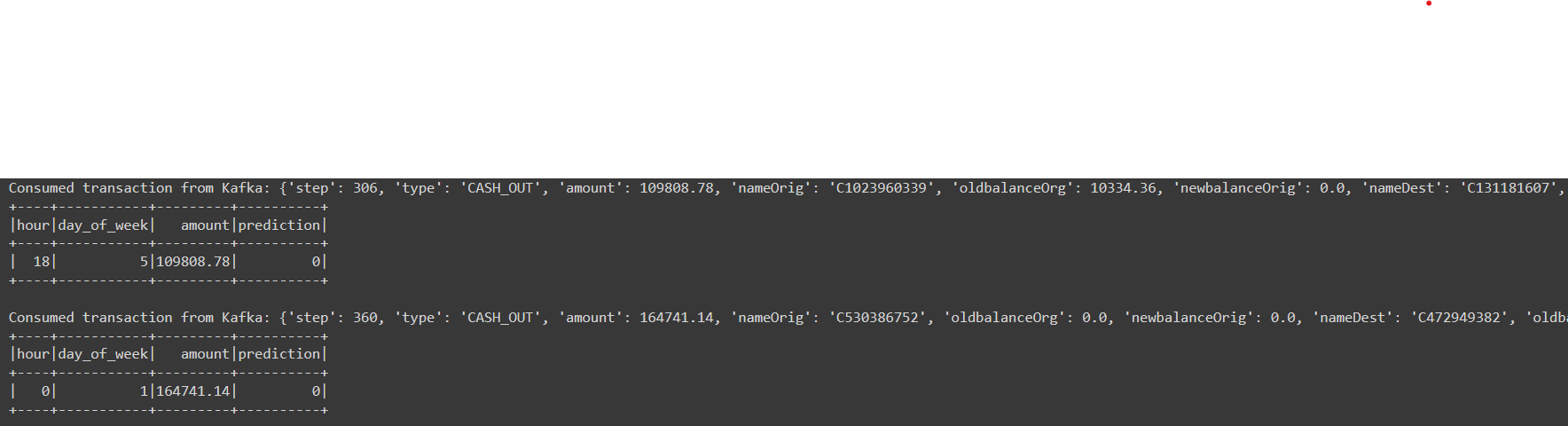


Figure 8: outcome of real-time prediction

**Challenges Faced:**

Initially, it was tiring to do the deployment on the local device using the IDE in it. So, both frameworks were installed and set up. In the beginning, they were running properly later due to insufficient space in the device. It was difficult to uninstall the non-suitable version and install the suitable version and set it up each time. So, Google Colab is used it is very easy to access and allows one to access any version of any framework easily. Even in Google Colab, both were difficult to use as they were not coordinated with each other directly which did allow the data to stream from Kafka to Spark. Many approaches were tried. Finally, the above approach is used which consists of an intermediate stage between both frameworks which is a variable where the streaming data is stored each time and used by Spark.

**CHAPTER IV: RESULTS AND DISCUSSION**

After implementing both batch and real-time fraud detection models, further development and testing proved the chosen technique and compared the results of real-time streaming data analytics for the chosen type of fraud. This section reports on the outcome of experiments, results of the batch processing system as well as the real-time system.

**Comparison of Streaming Data Analytics and Batch Processing System**

Real-time fraud detection systems refer to a process whereby transactions are analysed to allow for the taking of action on activities that may be regarded as suspicious. This is notably important in financial industries because any time that is wasted can cost a lot in terms of fraud being executed. On the other hand, batch processing systems work in time intervals, the time difference makes it possible for fraudulent transactions to occur before the anomalies are detected. The real-time systems enable real-time detection and action, thereby reducing the incurring of major losses which is evident from figure 8. While batch process system take time since the transactions occur in a block implying that a certain amount of time elapses before the transaction results are updated. Therefore, streaming methods are more suitable for fraud detection in environments where time is of the essence, in the sectors, such as banking and finance, where numerous transactions take place at all times.

**Machine Learning Algorithms for Real-Time Fraud Detection**

In this project, LightGBM, a tree-based gradient boosting algorithm for both batch and real-time fraud detection the following algorithms were used. LightGBM also proved quite effective when working with structure data which requires little preprocessing; it has high efficiency in distributed systems, which makes it ideal for real-time issues, such as fraud detection in the stream processing environment. Machine learning models especially those based on deep learning techniques like the LightGBM can be loaded into real-time systems as we saw from integration with Apache Spark. This system retrieves data from Kafka, transforms it and makes predictions of fraud in real-time using the model. Vectors were assembled using the VectorAssembler framework in Spark and a pre-built LightGBM model was used in UDF in Spark to make real-time predictions. Real-time fraud detection using machine learning algorithms in this architecture also confirms that a model such as LightGBM can be deployed in real-time applications with a negligible time lag.

**Effectiveness of Streaming Data Analytics Frameworks**

Both Apache Kafka and Apache Spark improve the efficiency of fraud detection since they allow for high availability and low latency processing of real-time transaction feeds. These structures enable large-scale systems and robust solutions that can accomplish millions of transactions almost concurrently. Further, Kafka serves as a queuing system where transaction data cannot be processed as quickly as it is generated, but it guarantees that data will be given to the fraud detection system as soon as it becomes available, thus reducing the latency of data pipelines; Kafka is a horizontally scalable system where the data input is increasing. Spark always computes data in-memory meaning data transformations and feature extraction do not have significant latency so fraud detection models make predictions almost in real time. The Kafka and Spark-based real-time system is shown to be capable of handling transaction streams from large-scale systems while keeping the response time necessary to quickly identify frauds.

**Accuracy Comparison of Both Systems**

As it was discussed in the methodology, the performance of both the batch processing system and the real-time system was evaluated using various metrics, including accuracy, precision, recall, and F1-score. From figures 5 and 6, it is found that the Recall of the batch model for fraud transactions which is 0.27 is better than the Recall of the real-time model which is 0.13. This may lead to fewer false positives. But both models exhibit strong accuracy. The real-time model shows even slightly higher accuracy which is 99% than the batch-processing model, though with slightly higher false negatives which is missing fraud.

**Challenges of Real-Time Fraud Detection**

Real-time fraud detection can be a very complex process, fraught with potential difficulties and problems. However, when it comes to the implementation of real-time fraud detection, there seem to be the following problems. They are as follows,

**High Data Throughput:** Considering that financial institutions perform millions of transactions daily, no significant latency in processing and management of large amounts of data to the system is achieved by consideration of the following factors.

**Latency Optimization:** The real-time system has to work in milliseconds because any second delay will cause fraud detection to become irrelevant. This can be resolved by properly configuring Kafka and more specifically Spark to overcome the problem of managing the system’s scalability and performance. For example: separating Kafka topics and optimizing micro-batch in Spark.

**Model Drift:** At one point in time, the pre-trained model can be a problem since its learning is solely based on historical forms of fraud. The real-time system needs to update models more often or include adaptive components for new fraud features or more elaborate models of fraud. This can be fixed by learning online algorithms or updating models every now and then with fresh data to minimize the rate of fraud activities.

**CHAPTER V: CONCLUSION**

**Summary**

To summarize, as part of this project, a real-time fraud detection system shows that it is possible to employ new data streaming frameworks such as Apache Kafka, and Apache Spark to detect fraudulent transactions while they are being processed. The real-time model was more accurate than historical batch-based systems and the analysis reflected near real-time response times when considering transaction data. However, the model tends to make false positives and false negatives, hence their mitigation is a problem as depicted below in the performance indicators. Given that precision and recall can still be improved, the system underlines the idea of improving the models while taking into account the changes in fraudulent activities.

One important realization is that the real-time system could work even more efficiently if the data feeds are taken from a live database or cloud storage as well as using IDE in a local device in which the entire system’s storage and power can be harnessed. In this project, the system applied a pre-specified data set that is not quite as near to the reality of fast real-time transactional datasets. A more accurate and insightful result could be obtained by critically streaming real-time transactions from actual financial databases in the effort by which the model will attempt to work on real-world and changing data patterns.

Therefore, real-time fraud detection using streaming data is an effective way of approaching or preventing and controlling fraud incidences in financial systems, but it should be enhanced constantly and be rather flexible. The system is still a reactive one resulting from historical data; however, by interfacing with real-time data, fine-tuning performance and using adaptive machine learning techniques, it can become a better proactive solution for combating fraud within varying processes.

**Future Works**

* **Integration with Financial Institutions:** Implementing this system in actual real-life banking structures that track and analyse transactions with the aim of detecting fraud instantly to avert the occurrences of incidents that may affect customers.
* **Cloud-Based Fraud Detection:** The idea is to expand the project to cloud platforms such as AWS or Google Cloud for more scalability and computing resources. This would afford scalability and achieve real-time fraud detection without the use of internal IT infrastructure.
* **Anomaly Detection Expansion:** To make the system perform not only detecting fraud transactions but also money laundering or any abnormal activities in the accounts. It would expand its applicability to other financial domains as well.
* **Cross-Platform Integration:** Expanding the ability of the solution to be implemented in multiple payment methods, online stores, and mobile payments, in turn giving the solution across the different payment systems.
* **Collaborative Fraud Detection:** Another aspect of the system is the ability to link with other counter-fraud internal and external databases to enhance the detection procedure. The structure of the model would be also improved if it was based on the analysis of global trends in fraud.
* **Real-Time Model Adaptation:** Finding ways within the limitations of adaptive learning so that the model progresses with the new types of fraud to avoid producing false positives or false negatives.

**REFERENCES**

[1] Agrawal, S.K. (2021*). Evaluation Metrics For Classification Model*, Classification Model Metrics. Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/

[2] Ahmad, S., Lavin, A., Purdy, S. and Agha, Z. (2017). *Unsupervised real-time anomaly detection for streaming data*, Neurocomputing, 262, pp.134–147. doi:https://doi.org/10.1016/j.neucom.2017.04.070.

[3] Anderson, S. (2022). *What is Streaming Data Analytics? Use Cases, Examples, and Architecture | StreamSets*. [online]Software AG. Available at: <https://www.softwareag.com/en_corporate/blog/streamsets/what-is-streaming-analytics.html>.

[4] Anglen, J. (2024). *Outlier Detection: What You Need to Know?*, Rapidinnovation.io. Available at: https://www.rapidinnovation.io/post/outlier-detection-what-you-need-to-know

[5] Ariyaluran Habeeb, R.A., Nasaruddin, F., Gani, A., Amanullah, M.A., Abaker Targio Hashem, I., Ahmed, E. and Imran, M. (2019*). Clustering‐based real‐time anomaly detection—A breakthrough in big data technologies*, Transactions on Emerging Telecommunications Technologies, p.e3647. doi:https://doi.org/10.1002/ett.3647.

[6] Armbrust, M., Das, T., Torres, J., Yavuz, B., Zhu, S., Xin, R., Ghodsi, A., Stoica, I. and Zaharia, M. (2018). *Structured Streaming*, Proceedings of the 2018 International Conference on Management of Data. doi:https://doi.org/10.1145/3183713.3190664.

[7] Cameron, S. (2023). *What Is Fraud Detection and Why Is It Important?* ComplyAdvantage. Available at: <https://complyadvantage.com/insights/what-is-fraud-detection/>.

[8] Cao, S., Yang, X., Chen, C., Zhou, J., Li, X. and Qi, Y. (2019). *TitAnt*. Proceedings of the VLDB Endowment, 12(12), pp.2082–2093. doi:https://doi.org/10.14778/3352063.3352126.

[9] Chen, J.I.-Z. and Lai, K.-L. (2021). *Deep Convolution Neural Network Model for Credit-Card Fraud Detection and Alert*. June 2021, 3(2), pp.101–112. doi:https://doi.org/10.36548/jaicn.2021.2.003.

[10] Chen, T. and Guestrin, C. (2016). *XGBoost: a Scalable Tree Boosting System,* Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining *- KDD ’16*, pp.785–794.

[11] Chen X, Ishwaran H. (2012). *Random forests for genomic data analysis*, Genomics. Jun;99(6):323-9. doi: 10.1016/j.ygeno.2012.04.003. Epub 2012 Apr 21. PMID: 22546560; PMCID: PMC3387489.

[12] Choi, D. and Lee, K. (2018). *An Artificial Intelligence Approach to Financial Fraud Detection under IoT Environment: A Survey and Implementation,* Security and Communication Networks, 2018(5483472), pp.1–15. doi:https://doi.org/10.1155/2018/5483472.

[13] Databricks (2019). *Apache Spark*, Databricks. Available at: https://www.databricks.com/glossary/what-is-apache-spark.

[14] DataScience-ProF. (2023). *Understanding Label Binarizer: A Powerful Tool for Categorical Data Transformation*, Medium. Available at: https://medium.com/@TheDataScience-ProF/understanding-label-binarizer-a-powerful-tool-for-categorical-data-transformation-7d1f4a010992

[15] Devicharan Rai M and N, J.S. (2023). *Credit Card Fraud Detection using Machine Learning and Data Mining Techniques - a Literature Survey*, Zenodo (CERN European Organization for Nuclear Research). doi:https://doi.org/10.5281/zenodo.8190094.

[16] Ed-daoudy, A. and Maalmi, K. (2019). *A new Internet of Things architecture for real-time prediction of various diseases using machine learning on big data environment,* Journal of Big Data, 6(1). doi:https://doi.org/10.1186/s40537-019-0271-7.

[17] Eedala, S.H. (2024). *Financial fraud detection dataset*, *Kaggle*. Available at: https://www.kaggle.com/datasets/sriharshaeedala/financial-fraud-detection-dataset

[18] Github.io. (2019). *Dunya’s Blog*. Available at: https://dunyaoguz.github.io/my-blog/dataframemapper.html - dataframemapper

[19] Hasan, M., Orgun, M.A. and Schwitter, R. (2017). *A survey on real-time event detection from the Twitter data stream*, Journal of Information Science, 44(4), pp.443–463. doi:https://doi.org/10.1177/0165551517698564.

[20] Hosmer, D.W and Lemeshow, S. (2000). *Applied logistic regression*, 2nd Edition, John Wiley & Sons, Inc., New York, 1-32 & 69-79 doi: <http://dx.doi.org/10.1002/0471722146>

[21] Huang, D., Mu, D., Yang, L. and Cai, X. (2018). *CoDetect: Financial Fraud Detection With Anomaly Feature Detection,* IEEE Access, 6, pp.19161–19174. doi:https://doi.org/10.1109/access.2018.2816564.

[22] Ileberi, E., Sun, Y. and Wang, Z. (2022*). A machine learning based credit card fraud detection using the GA algorithm for feature selection*, Journal of Big Data, 9(1). doi:https://doi.org/10.1186/s40537-022-00573-8.

[23] Isah, H., Abughofa, T., Mahfuz, S., Ajerla, D., Zulkernine, F. and Khan, S. (2019). *A Survey of Distributed Data Stream Processing Frameworks*, IEEE Access, 7, pp.154300–154316. doi:https://doi.org/10.1109/access.2019.2946884.

[24] Kanade, V. (2021). *What Is Fraud Detection? Definition, Types, Applications, and Best Practices*, Spiceworks. Available at: <https://www.spiceworks.com/it-security/vulnerability-management/articles/what-is-fraud-detection/>.

[25] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.-Y. (2017). *LightGBM: A Highly Efficient Gradient Boosting Decision Tree*. Neural Information Processing Systems. Available at: https://proceedings.neurips.cc/paper\_files/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.

[26] Mayhew, M., Atighetchi, M., Adler, A. and Greenstadt, R. (2015). *Use of machine learning in big data analytics for insider threat detection*, MILCOM 2015 - 2015 IEEE Military Communications Conference. doi:https://doi.org/10.1109/milcom.2015.7357562.

[27] Mohamed, A., Najafabadi, M.K., Wah, Y.B., Zaman, E.A.K. and Maskat, R. (2019). *The state of the art and taxonomy of big data analytics: view from new big data framework,* Artificial Intelligence Review, 53(2), pp.989–1037. doi:https://doi.org/10.1007/s10462-019-09685-9.

[28] Neural Technologies. (n.d.). *How CEP and Streaming Analytics help in Early Fraud Detection*. [online] Neuralt.com. Available at: <https://www.neuralt.com/news-insights/how-cep-and-streaming-analytics-help-in-early-fraud-detection>

[29] NVIDIA Technical Blog. (2022). *Real-time Serving for XGBoost, Scikit-Learn RandomForest, LightGBM, and More | NVIDIA Technical Blog*. Available at: https://developer.nvidia.com/blog/real-time-serving-for-xgboost-scikit-learn-randomforest-lightgbm-and-more/

[30] Patil, S., Nemade, V. and Soni, P.K. (2018). *Predictive Modelling For Credit Card Fraud Detection Using Data Analytics*, Procedia Computer Science, 132, pp.385–395. doi:https://doi.org/10.1016/j.procs.2018.05.199.

[31] Prasad, M.D.V. and Srikanth, T., 2024. *Multi-Entity Real-Time Fraud Detection System using Machine Learning: Improving Fraud Detection Efficiency using FROST-Enhanced Oversampling*, Journal of Electrical Systems, 20(7), pp. 1380-1394.

[32] Sagacity Solutions. (n.d). *What is data deduplication? how to improve data uniqueness*, Sagacitysolutions.co.uk. Available at: https://www.sagacitysolutions.co.uk/about/news-and-blog/data-deduplication/

[33] Sanober, S., Alam, I., Pande, S., Arslan, F., Rane, K.P., Singh, B.K., Khamparia, A. and Shabaz, M. (2021). *An Enhanced Secure Deep Learning Algorithm for Fraud Detection in Wireless Communication,* Wireless Communications and Mobile Computing, 2021, p.e6079582. doi:https://doi.org/10.1155/2021/6079582.

[34] Simplilearn. (2023). *Data preprocessing in Machine Learning: A beginner’s guide*, Simplilearn.com. Available at: <https://www.simplilearn.com/data-preprocessing-in-machine-learning-article>.

[35] Spiceworks. (n.d.). *What Is Kafka? Definition, Working, Architecture, and Uses*. Available at: https://www.spiceworks.com/tech/data-management/articles/what-is-kafka/ - kafka

[36] Thennakoon, A., Bhagyani, C., Premadasa, S., Mihiranga, S., & Kuruwitaarachchi, N. (2019). *Real-time Credit Card Fraud Detection Using Machine Learning*, 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 488-493.

[37] Zareapoor, M. and Shamsolmoali, P. (2015). *Application of Credit Card Fraud Detection: Based on Bagging Ensemble Classifier,* Procedia Computer Science, 48, pp.679–685. doi: https://doi.org/10.1016/j.procs.2015.04.201.

[38] Zhu, H., Liu, G., Zhou, M., Xie, Yu., Abusorrah, A.., & Kang, Q. (2020*). Optimizing Weighted Extreme Learning Machines for imbalanced classification and application to credit card fraud detection*, Neurocomputing, 407, 50-62. http://doi.org/10.1016/j.neucom.2020.04.078