
THE UNIVERSITY OF QUEENSLAND

AUSTRALIA

# INNOVATIVE VISUALIZATION TECHNIQUES FOR GLOBAL CRIME AND PAYMENT DATA

by Jaskeerat Singh (47610039)

School of Information Technology and Electrical Engineering,

The University of Queensland.

Submitted for Master of Data Science Capstone 1

15th September 2023

**Introduction**

The rapid advancement and widespread adoption of digital currency have transformed crypto into an alternative to traditional payment methods (Guych et al., 2018). However, this surge in popularity has also attracted criminals seeking to exploit the system through legal loopholes, leading to market manipulation and Ponzi schemes (Bartoletti et al. 2018). My project aims to unveil the complex network of fraudulent transactions by developing an analytical system that provides a real-time framework for law enforcement agencies to promptly and effectively detect and visualize the sequence of transactions to the origin and, in turn, respond to financial cybercrimes.

In this project, I propose developing a real-time analytics application that employs machine learning and statistical analysis to generate alerts for identifying suspicious transactions. When a transaction is flagged, the system will assist law enforcement agents in conducting more effective target research. Over the years, researchers have explored various algorithms for fraud detection, but no application exists for users to visualize provenance flow (Tae et al., 2019). The proposed solution will be a user-interactive system capable of detecting and visualizing fraudulent exchanges. These transactions will be visually represented within the complex web of bitcoin exchanges and help provide a provenance or money flow.

The entire project, designed as a microservice architecture, will be divided into four sub-parts: connecting to a Real-Time Data API, preprocessing incoming raw data, conducting real-time fraud analysis, and providing user-interactive visualization. The dataset used for this project comprises Bitcoin transactions with four key features: sender address, receiver address, timestamp, and transaction amount (Wood, 2021).

This proposal will begin by examining the current methods employed for detecting fraud in traditional transaction systems. Subsequently, I will address the challenges and reasons behind needing a dedicated system for real-time data processing. Next, I will establish project objectives, and finally, I will propose the initial iteration of a solution for real-time transaction handling and how these different technologies help visualise provenance flows.

**Project composition**

This proposal forms one element of a broader project comprising four distinct components. The first component involves conducting in-depth analysis and preprocessing of cybercrime data, while the second component focuses on feature engineering and feature selection for data analysis. The third component aims to explore AI-powered visualization techniques. In contrast, the fourth and final

project under me aims to integrate all these elements into a real-time analytical support system for performing fraud (anomaly) detection. Collectively, these four components contribute to our primary goal of implementing innovative visualization techniques for global crime and payment data.

**Background and Related works**

In this section, I will discuss the various technologies implemented for managing real-time data in traditional transactional systems, their application in detecting cryptocurrency fraud, and the machine learning algorithms employed in these systems.

The increasing adoption of cryptocurrencies and their growing acceptance by financial institutions as a mode of payment for services highlights the need for real-time fraud detection mechanisms and a robust system capable of managing a high volume of transactions. At the centre of this project lies the need for handling real-time streams of large datasets and implementing an anomaly detection system, which plays a crucial role in identifying fraudulent transactions. The project draws inspiration from analytical systems deployed in other fields that integrated anomaly detection methods with cutting-edge Big Data technologies like PySpark and Kafka to process continuous streams of data effortlessly.

Over the years, researchers have developed analytical systems using the microservices architectural framework. Microservices architecture represents a system design approach that breaks down complex applications into smaller, independent, and modular services. Each service in the application is referred to as a 'microservice' and has a specific function or task within the larger application. Big data systems are designed to handle large volumes of data while meeting performance quality requirements such as scalability and availability. For example, a study investigated the implementation of Microservice-Oriented Big Data Architecture (MOBDA), which incorporates data processing techniques like predictive modelling to manage big data (Asaithambi et al., 2020). This microservices approach allows users to create services that operate independently of one another, enabling iterative changes and improvements to the system without one system affecting other.

In a recent study that focused on processing incoming data streams, researchers found that PySpark and Kafka effectively managed and analysed data in real time (Abakarim et al., 2018). The study aimed to improve the detection of fraudulent activities in traditional transactions, like credit card transactions. A comparative analysis of different machine learning algorithms showed that these big data technologies efficiently handled and processed data streams, making a strong case for implementing them in a cryptocurrency fraud detection system.

For performing fraud detection, various anomaly detection algorithms are employed to classify transactions as normal or fraudulent. These machine-learning algorithms fall into two categories: supervised learning (Tae et al., 2019) and unsupervised learning (Amarasinghe et al., 2018). In supervised learning, a model is trained by using labelled data. Labelled data is crucial in supervised machine learning, where algorithms are trained to learn patterns and make predictions based on the provided labels. For Fraud detection, an application can consist of a machine learning algorithm in combination with sampling techniques. These sampling techniques can help handle the skewness of the data (Ileberi et al., 2021). This skewness can occur when a single class dominates the label feature (high amount of 1 or 0). A trained model is developed from the available labelled data in supervised learning. Here, from a group of available machine learning models, a group of algorithms is first trained. Based on the performance, the best-performing model is chosen for fraud detection of future transactions. Different approaches such as neural networks, classifiers like adaboost and ensemble methods like decision trees can be employed for fault detection.

Unsupervised learning methods for fraud detection, like clustering and statistical analysis, can also be utilised (Mittal et al., 2019). These outlier detection techniques help identify data points that deviate from the rest of the observed data. These outliers fall into three categories: point outliers, when a single transaction appears unusual; contextual outliers, when a group of transactions stands out; or collective outliers, a combination of both.

After identifying fraudulent transactions, we must store these records efficiently. It is essential to choose a storage solution that takes advantage of the relationships among transaction addresses, supports real-time storage, and provides fast query responses. In a recent study, graph databases demonstrate their capability to conduct analytical computations and help provide a window for visualizing transaction exchanges (Prusti et al., 2021). Use of graph databases also allows users to apply graph-related methodologies, such as centrality measures and node embeddings, to generate features for fraud detection.

While various researchers have extensively studied machine learning algorithms, their practical implementation in a real-time interactive system has yet to see much support. A study by proposes a real-time streaming pipeline employing neural networks for classifying data (Abakarim et al., 2018). This system utilises Kafka-streamed datasets to compare the performance of SVM, regression, and deep neural networks on labelled data. Visualising the money flow's provenance is crucial to provide users with an interactive visualisation of fraudulent transactions. A paper proposes a novel model for capturing the flow (Geng et al., 2022). This model suggests using a triplet data structure and a framework for identifying and leveraging transaction behaviour to enhance traceback accuracy.

Although these independent research methods aim to solve problems in different domains, these systems built for traditional transactions are able to function well as an alert mechanism but has a host of requirements for a real time system which might cause a system designed for crypto currency to be changed drastically.

A real-time system for identifying cryptocurrency fraud that works to give a quick snapshot to law agencies holds significant potential. My project aims to develop an analytical system in collaboration with the research done by a team member working on provenance flows. Together, we aim to help law enforcers by providing a way to view fraudulent transactions.

**Problem identification**

In this section, I will discuss the challenges we must address to improve and provide a better system than these related projects. These challenges involve choosing appropriate anomaly detection algorithms, addressing data skewness in labelled datasets, utilizing conventional storage solutions, and handling large data volume for provenance flow.

The current traditional system operates by using a machine learning model to alert for transactions. This trained model identifies and regularly updates itself based on the instances of fraud that it subsequently confirms as fraudulent. It evaluates each transaction against the trained model and marks transactions as suspicious in real time. These systems rely on the availability of labelled data for model training. While traditional systems have the ability to generate alerts on detection, they lack user interactivity for presenting the money flow from the origin. These two issues complicate the working of a real-time application.

Cryptocurrency transactions are typically anonymized, making labelled fraud data scarce. When available, the training data often exhibits high skewness. Supervised learning methods require human-verified or previously encountered data. If the training data is heavily skewed toward legitimate transactions or unavailable, attempting to balance it using techniques like resampling or SMOTE may lead to a higher rate of false alerts for genuine transactions (Sisodia et al., 2017). These false alerts increase the end user's workload and require their identification and removal from the machine learning training pipeline, further increasing the user's workload.

Storing incoming transactions also plays a crucial role in managing real-time data. Unlike traditional databases, which rely on tables and rows to store structured data, graph databases like Neo4j utilize nodes and edges to represent complex relationships and interconnected data. By representing incoming data streams as graphs rather than rows in a table, we gain the ability to leverage graph-

specific algorithms such as node embeddings. When dealing with large volumes of big data characterized by complex relationships, traditional databases struggle to retrieve and navigate data efficiently. Using join operations between static individual tables becomes computationally expensive as data size and complexity increase. Traditional databases are designed for ensuring data consistency rather than supporting low-latency, real-time data ingestion and querying. Due to this, these traditional systems will need to make significant infrastructure investments to handle substantial volumes of incoming streaming data.

| Aspect | Graph Database | Relational Database |
|---|---|---|
| Data Structure | Uses a graph data model with nodes and edges to represent relationships. | Utilizes tables with rows and columns to store structured data. |
| Schema Flexibility | Offers schema flexibility, allowing for dynamic changes to the data structure | Requires a predefined and rigid schema that enforces data consistency and structure. |
| Relationship Handling | Excellently suited for managing and querying complex relationships between data points. | Capable of handling relationships, but joins and foreign keys can make complex relationships more challenging to query. |
| Performance | Performs exceptionally well for highly interconnected data, such as social networks and recommendation engines. | Typically excels at handling structured data and performing traditional SQL queries efficiently. |
| Use Cases | Ideal for scenarios with data that has complex, many-to-many relationships, such as social networks and fraud detection | Well-suited for transactional systems, reporting, and applications with well-defined schemas, like financial systems and inventory management. |

Table 1. Comparison between Graph and traditional databases

These existing analytical systems work at generating alerts but do not effectively display the flow of money. This flow of funds is necessary for users to trace the origin of fraudulent transactions. For instance, when a law enforcement user examines recently alerted transactions, with this system, they can look into the accused's historical transactions. Hence, this system allows them to understand how accomplices collaborated and establish communication with the original scammer.

When building a real-time system, the system must also adapt to fluctuations in application user demand. In response to these fluctuations, inadequate resource allocation can negatively impact user satisfaction and increase operational costs. When using visualizations, users can select nodes to access database data for building a provenance flow. When building such a functionality, traditional systems may require costly computations in the backend for data joining, filtering, and extraction to present the correct user output.

To solve these issues, I propose some objectives for the project which will act as a measure of success in the implementation of the project.

**Objectives**

To tackle the abovementioned challenges, I propose development of a real-time user-interactive analytical system to support cybercrime investigations. The system aims to achieve the following objectives:

- Implement a Real-Time System by developing a real-time analytical system utilizing MLops methodologies. This approach ensures that the system adapts dynamically to user requirements, enhancing its effectiveness
- Establish an MLops Pipeline to facilitate the smooth development, connections between components and deployment of code. This streamlined process will improve efficiency and enable faster updates.
- Setup connection to Real Time Streaming Api that generates data in batches, serving as a reliable data source for the system's operations.
- Perform Real-Time Data Cleaning and Processing, culminating in the storage of cleaned raw data within a graph database.
- Implement graph centrality techniques to produce features for alert mechanism by using graph orientated databases
- Select Optimal Unsupervised Learning Algorithm through experimentation, thus identifying the most suitable unsupervised learning algorithm and statistical methods for fraud detection. Leveraging advanced analytics will enable the unravelling of intricate connection in transaction networks.
- Validate System Performance by rigorously assessing the system's performance under real-world conditions to ensure its reliability and efficiency. This validation process guarantees that the system delivers timely and effective responses.

Effective handling of large data streams is a vital performance indicator for our system. Our proposed system will handle incoming records, perform analytics on the transaction, and update graph-based databases in the background. This process will have a significant influence on the user experience. Furthermore, I suggest enabling rapid changes in visualisation as users interact with the visualisation system. This objective implies performing fast computations in the backend to filter data and extract suitable information from the graph database, especially when the user-defined scope of the visualisations changes.

Based on the defined objectives, I propose a solution to address all the issues and provide an application for the end user.

**Proposed solution**

I will implement the data science process, which includes five stages: formulating problem, data preparation, modelling and optimisation, evaluating results and designing application, to achieve the defined objectives and improve upon the issues identified in traditional systems.

**Step 1: Problem formulation**

The proposed application comprises of four distinct programs, each dedicated to a specific task. These programs include the Kafka producer, cleaning and preprocessing module, machine learning and statistical analysis for fraud detection and querying the graph database for visualization purposes (See Figure 1). Each of these developed components will follow the MLops framework. Together, these components will establish a structured sequence of data manipulation and storage operations.

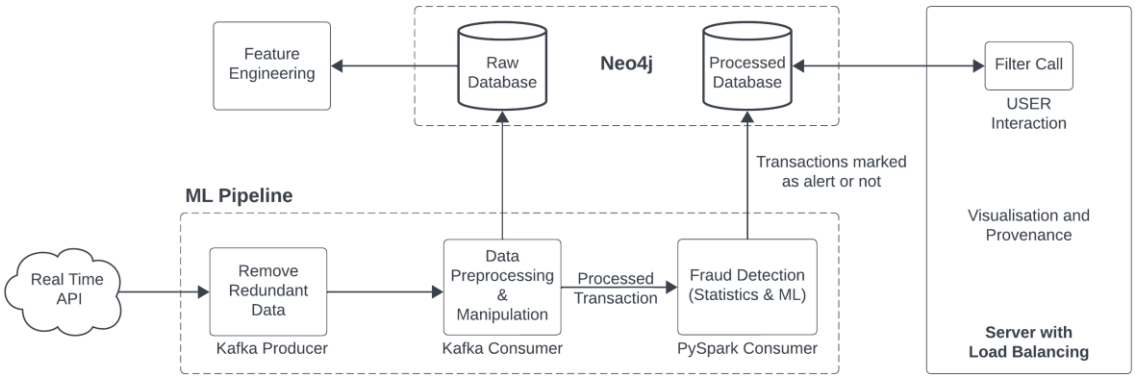


Figure 1. Proposed solutions system architecture

These individual components are as listed below –

Kafka Producer - In this phase, we will receive data stream in real-time from an API. The API will deliver data in the form of records, where each record represents a transaction. Initially, we will clean the incoming data by removing redundant rows with null values in specific columns. Next, we will divide the cleaned data into batches based on the timestamps provided in the transactional log. These batches will be crucial in identifying potential fraudulent data points and treating them as outliers.

Kafka Consumer Preprocessing – After the producer sends the data, it will call a function that performs defined preprocessing and data cleaning methods. This step involves handling missing data, eliminating duplicate rows, and standardizing the data. We will use the Satoshi to USD API converter to obtain the value in US dollars. This conversion will help provide information in a consistent, user-understandable form. Additionally, I will convert incoming timestamps into milliseconds, enabling us to group transactions based on timestamps, such as those occurring within the past 10 minutes, for predictive statistical analysis. Finally, I will store this cleaned raw data in a graph database.

PySpark Predictive statistics and machine learning - After processing the incoming transactional data, I will use various unsupervised machine-learning techniques and statistical analysis to check and generate alerts. Each processed transaction will go through an algorithm that will label it as either genuine or fraudulent. Since I do not have labelled data for training the model, it will be challenging to evaluate the performance of each algorithm. Therefore, I will implement different statistical techniques to detect anomalies based on different parameters. We will use a binary value of 0/1 to mark these processed records, indicating whether they represent an alert. Afterwards, we will store these updated records in a graph database.

Visualisation - After marking the transactions as genuine or fraudulent, we will shift our focus to creating a user-interactive visualization. In this phase, we will plot a real-time updating graph with incoming transactions. Each fraudulent transaction will be interactive, displaying additional information upon interaction. When users click on a transaction, the graph will activate a function connected to the database to retrieve the most updated data after applying user-defined filters. These visualization techniques can effectively manage the complexity of transactional data. The system aims to present provenance flows through these visualizations, unveiling intricate patterns.

Real-Time System - In the project's final phase, we will fully integrate the application, creating a real-time analytical system that automatically scales to meet changing demand. This system ensures users have a consistent visualization experience and is built upon MLops methodologies, simplifying development, version control, and deployment.

**Step 2. Data Preparation**

For this project, I plan to utilize a publicly accessible deanonymized dataset from Harvard Dataverse (Wood, 2021). The dataset consists of multiple files, totalling 52GB of data, and includes four distinct features (See Figure 2). Notably, the dataset lacks any labelled data. It contains information about the sender, receiver, date-time, and the transferred amount of bitcoin present in Satoshi format. I will transmit data row-by-row from this deanonymized dataset to emulate a real-time API. Each transaction will encompass sender details, receiver details, timestamp, and the purchased Bitcoin amount.

| | timestamp | source_address | destination_address | satoshi |
|---|---|---|---|---|
| 2 | 1585500382 | bc1qs0pkt4j5e7e8xacd6839z37zcn5uzlexxwwwd8 | bc1qg8tdup76h468tpcy4auek9uqlxgqdh2d6sneqr | 693000 |
| 3 | 1585500382 | bc1qs0pkt4j5e7e8xacd6839z37zcn5uzlexxwwwd8 | bc1qeugyjar80nkfppdnwx9dzz4dc28xqp4qxpdguk | 285013 |
| 4 | 1585500382 | bc1qs0pkt4j5e7e8xacd6839z37zcn5uzlexxwwwd8 | bc1qsyg3dmjcl38c23t5r3pcaxcheclz38upsc6p5p | 1696868 |
| 5 | 1585500382 | bc1qs0pkt4j5e7e8xacd6839z37zcn5uzlexxwwwd8 | bc1qs0pkt4j5e7e8xacd6839z37zcn5uzlexxwwwd8 | 6.91E+08 |
| 6 | 1585500382 | 1MRR3UJQfNhE9TsvXzbXnifd85qiG3wzpV | 3CEaLdfcACUvBahJgugM5L93d1DTaPvFt4 | 362384 |
| 7 | 1585500382 | 1MRR3UJQfNhE9TsvXzbXnifd85qiG3wzpV | 17M2X6tjgriwDhvfPGbuBn5a22LhiCvT1S | 7652 |
| 8 | 1585500382 | 1JsGpekKgf74bjHrtNFivoFd74NxFzELeD | 3CEaLdfcACUvBahJgugM5L93d1DTaPvFt4 | 362384 |
| 9 | 1585500382 | 1JsGpekKgf74bjHrtNFivoFd74NxFzELeD | 17M2X6tjgriwDhvfPGbuBn5a22LhiCvT1S | 7652 |
| 10 | 1585500382 | 39EKjvDK6UEkpaBcLVbBen1M4dhh2iE9Lq | 3CTUJE6tuHy8KFykCUx53EwYG59h1Rm8Er | 107745 |
| 11 | 1585500382 | 39EKjvDK6UEkpaBcLVbBen1M4dhh2iE9Lq | 34NZzbi6qDsJ9yuwtgUBpPBMRcMFdzU3VF | 320711 |
| 12 | 1585500382 | 39EKjvDK6UEkpaBcLVbBen1M4dhh2iE9Lq | 3G5hBoSmASwMR1aK7SfVaR7hXr1nZPgUBe | 316412 |

Figure 2. Raw data

Given the dataset's intended use for streaming to the application and the absence of labelled data, we will employ unsupervised learning techniques. In the data consumption program, I will verify the availability of values in the columns mentioned above, such as addresses. If unavailable, redundant rows will be removed and excluded from the graph database. For acquiring a consistent, user-friendly currency view for end users, we will rely on a third-party Bitcoin API to obtain conversion rates from Satoshi (the smallest denomination of the cryptocurrency Bitcoin) to USD (See Figure 3). To store this processed data, I will use a graph database like Neo4j to take advantage of storing complex relationships between the addresses mentioned in the processed transactions. Each row of the data will symbolize a transactional relationship between the source and target addresses. In the graph database, each node will signify addresses (source and target), while the directed edge between them will represent 'to-from' transactions with attributes like datetime and value.

```
Data Inserted: {'timestamp': Timestamp('2020-03-29 16:46:22'), 'source_address': '1MRR3UJQfNhE9TsvXzbXnifd85qiG3wzpV', 'destination_address':
 '17M2X6tjgriwDhvfPGbuBn5a22LhiCvT1S', 'satoshi': 7652.0, 'usd': 0.47863263050803695}
Data Inserted: {'timestamp': Timestamp('2020-03-29 16:46:22'), 'source_address': '1JsGpekKgf74bjHrtNFivoFd74NxFzELeD', 'destination_address':
 '3CEaLdfcACUvBahJgugM5L93d1DTaPvFt4', 'satoshi': 362384.0, 'usd': 22.66712064480194}
Data Inserted: {'timestamp': Timestamp('2020-03-29 16:46:22'), 'source_address': '1JsGpekKgf74bjHrtNFivoFd74NxFzELeD', 'destination_address':
 '17M2X6tjgriwDhvfPGbuBn5a22LhiCvT1S', 'satoshi': 7652.0, 'usd': 0.47863263050803695}
Data Inserted: {'timestamp': Timestamp('2020-03-29 16:46:22'), 'source_address': '39EKjvDK6UEkpaBcLVbBen1M4dhh2iE9Lq', 'destination_address':
 '3CTUJE6tuHy8KFykCUx53EwYG59h1Rm8Er', 'satoshi': 107745.0, 'usd': 6.739450179572456}
Data Inserted: {'timestamp': Timestamp('2020-03-29 16:46:22'), 'source_address': '39EKjvDK6UEkpaBcLVbBen1M4dhh2iE9Lq', 'destination_address':
 '34NZzbi6qDsJ9yuwtgUBpPBMRcMFdzU3VF', 'satoshi': 320711.0, 'usd': 20.06047432865434}
Data Inserted: {'timestamp': Timestamp('2020-03-29 16:46:22'), 'source_address': '39EKjvDK6UEkpaBcLVbBen1M4dhh2iE9Lq', 'destination_address':
 '3G5hBoSmASwMR1aK7SfVaR7hXr1nZPgUBe', 'satoshi': 316412.0, 'usd': 19.7915718615145}
Data Inserted: {'timestamp': Timestamp('2020-03-29 16:46:22'), 'source_address': '39EKjvDK6UEkpaBcLVbBen1M4dhh2iE9Lq', 'destination_address':
 '339GMZk1RUZUSZi93jW3wnSQZ8VUZgtQCi', 'satoshi': 253035.0, 'usd': 15.827340258834436}
```

Figure 3. Cleaned data

**Step 3: Modelling and optimization**

In the anomaly detection stage, after processing the raw data, I will use a combination of Spark streaming window functions and various anomaly detection algorithms such as k-means clustering and graph-based community detection for developing machine learning models.

First, I will employ built-in PySpark window functions to facilitate point and contextual anomaly detection. Point detection methods consider past records' availability and calculate a moving average of the bitcoin transaction value that adjusts based on the window size. For example, the window size of 10 minutes calculates an average bitcoin value of all the records received in the last 10 minutes. Any new value associated with an incoming transaction that significantly deviates from the upper limit of an IQR range gets flagged as an outlier. In contextual detection, the number of transactions performed by a single address within a specific time period gets compared against a predefined threshold. Therefore, this method helps identify addresses engaging in numerous low-value transactions, flagging them as outliers for further review and alerting.

Secondly, I will apply unsupervised machine learning algorithms, such as k-means clustering and community detection methods for graph databases. These algorithms will utilize various graph network properties, including degree, clustering coefficients, and community detection, to partition graph nodes into distinct communities. Finally, nodes exhibiting minor affiliations with any community get classified as suspicious and trigger alerts.

I propose to use algorithms that can help find anomalies using techniques like:

- **Point outliers** represent data points that significantly deviate from the majority of the dataset (See Figure 4). These points are individual data observations that stand out as unusual or exceptional within a specified time range when compared to the rest of the data.

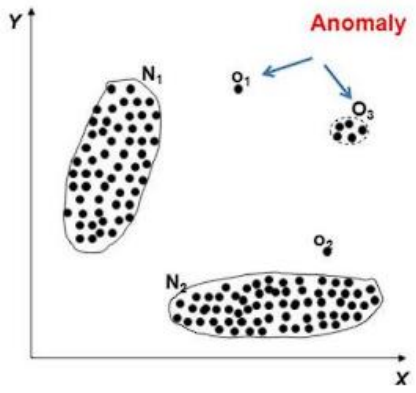


Figure 4. Point Anomaly detection (Amarasinghe et al., 2018)

- **Multivariate collective outliers** include ordered and unordered anomalies frequently encountered in transaction records. These anomalies can result from the combined effects of variables. For instance, an address performing low-value transactions but a large number of transactions might collectively constitute an anomaly, even if each variable falls within the normal range when considered individually.
- **K-means clustering** is a technique employed in transactional fraud detection to group similar transactions based on their features. Through K-means, transactions are divided into distinct clusters, and then anomalies are detected that notably diverge from the expected patterns within clusters (See Figure 5). Detecting these outlier clusters can help pinpoint potential instances of fraud within large transaction datasets, providing a valuable tool for fraud detection and prevention.

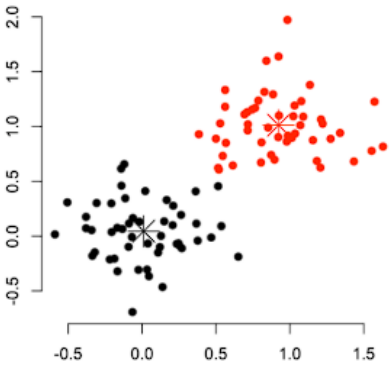


Figure 5. Points away from clusters to be marked as fraudulent (Amarasinghe et al., 2018)

- Graph databases are a powerful tool for detecting transactional fraud. They represent transactions as nodes and their relationships as edges, enabling advanced analysis (See Figure 6). These databases incorporate various techniques, including centrality measures like degree centrality, which aid in identifying potential hubs of fraudulent activity.

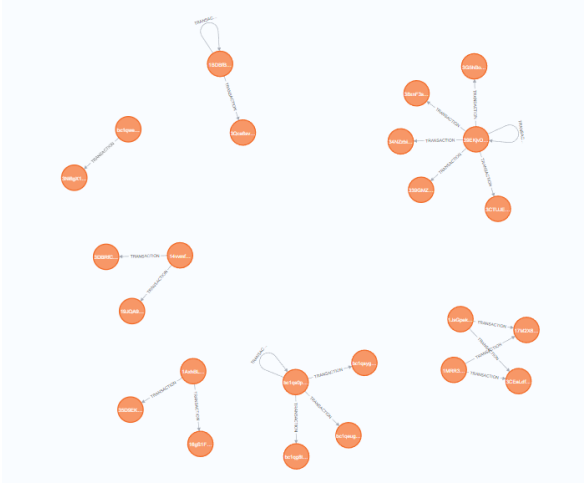

Figure 6. Neo4j Graph Database

- These graphs also employ community detection algorithms to uncover clusters of transactions that might suggest coordinated fraud rings or networks (See Figure 7). Additionally, graph databases employ anomaly detection methods such as **Local Outlier Factor (LOF) and Isolation Forest** to uncover nodes that diverge from the surrounding graph structure.

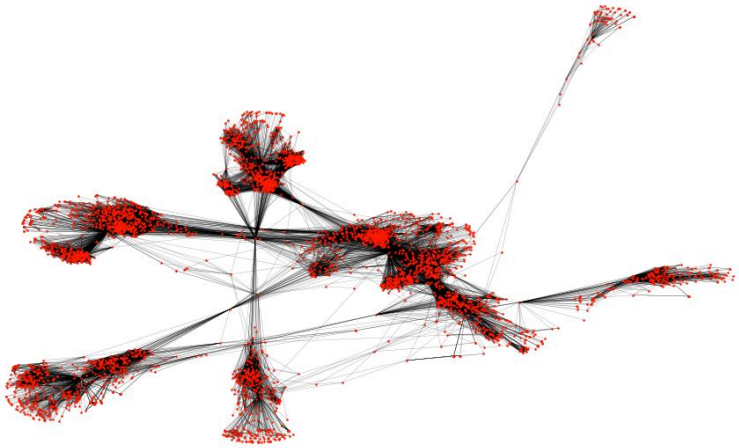

Figure 7. Directed connections in a graph

**Step 4: Results evaluation**

Based on the defined project objectives, we will evaluate system performance by assessing its ability to handle incoming streamed transactions.

In the real-time data consumption stage, a particular focus is on creating an application that maintains a steady flow of incoming records from the source to the database without significant delay. The database, as defined, must efficiently handle complex relations in real time and should be instantly accessible to users for reading.

During the anomaly detection stage, we cannot assess performance using conventional metrics like accuracy, F1 score, confusion matrix, or AUC score without access to labelled data. Instead, the system's capability to continuously maintain an accurate real-time view of the system with up-to-date visual information is the main priority for the application.

To ensure an efficient response time, I estimate the processing time required to retrieve the provenance flow for a transaction at approximately 10 seconds. This estimated duration encompasses the user's transaction selection, the processing of the request to obtain the provenance flow, and the subsequent presentation of this flow to the user through visualization.

**Step 5: Application design**

We will develop an interactive application after testing the program and data flow. This application comprises both a frontend and a backend system. The front end will display a clustered graph, enabling users to select a transaction and view its provenance flow. Meanwhile, the backend will manage incoming streamed data. To create this analytical system, I intend to leverage open-source resources.

**Resources**

This section will discuss the tools, programming languages, and storage options I plan to use in this project. I will employ -

Kafka serves as both a publisher and consumer model for transmitting data to stream the data from the source to the backend system.

PySpark, a combination of Python and Spark, will be employed to implement the anomaly detection algorithm and establish the provenance flow model.

Neo4j database will store the incoming records as a graph, illustrating the relationships between transaction addresses.

Plotly Dash, a Python web framework based on Flask, to develop an interactive user website.

**Proposed schedule**

The project's proposed development is visually presented as a timeline.

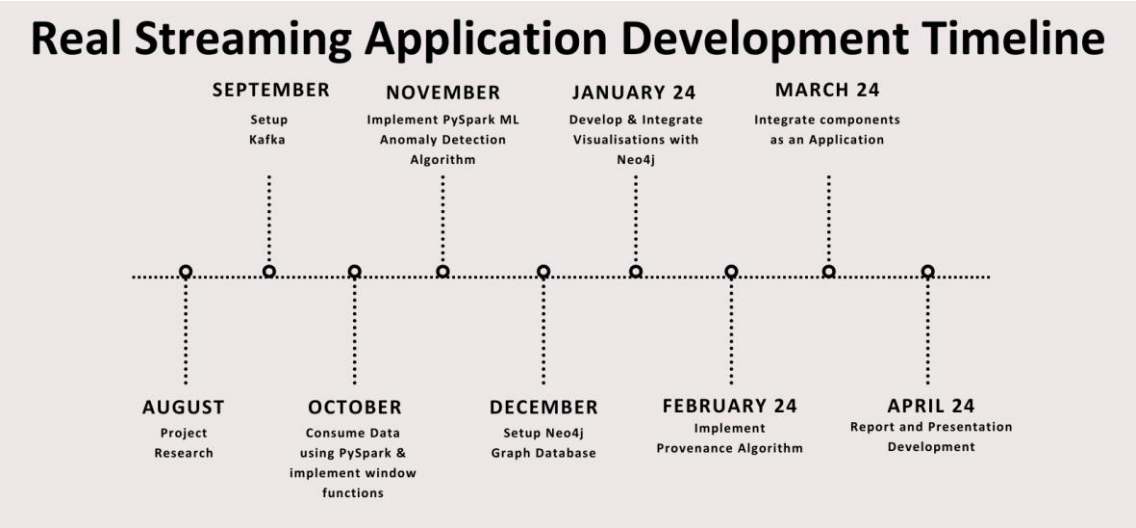


Figure 8. Project schedule as a timeline

Each team member will focus on a specific component of the project. We will individually develop these components in 2023, and in 2024, we will integrate them into a cohesive application(See Figure 8).

I aim to select and implement algorithms for the chosen dataset. Collaborating with teammates, we will use Dash to implement the provenance algorithm and create a sample visualization within a web app. We will define and test various parameters for the PySpark code after developing the Neo4j graph database.

**Costs**

This project relies on open-source software and applications. I developed it following the guidelines of MLops and have used no licensed software, which means there are no financial costs at this stage. Most of the time dedicated to the project involves research and collaboration with team members.

Up to this point, the team has dedicated most of our time to clarifying project objectives, researching architecture components and their usage and finding machine learning algorithms for implementation. I spent two weeks understanding project objectives and teamwork division. Three weeks were devoted to discussing various components' pros and cons and capabilities. It took an additional week to install and establish connectivity between components.

In the next phase, I will invest two months in developing and implementing different scenarios for fraud detection. After developing each component, most of the time will go towards system integration and testing. The testing process will include stress testing to evaluate the system's ability to handle incoming data and observe how the user-interactive web application performs under heavy queries.

**Conclusion**

In summary, the project is still in development, and various components may change. We have applied data science techniques to address existing system challenges and have defined the project's objectives. Through this iterative process, the proposed system aims to provide users with a real-time, interactive tool for tracking fraudulent transactions. While the project necessitates collaboration among team members and significant effort due to the complexity of handling streaming data, a successful implementation will offer law enforcement users an interactive application for monitoring financial flows. A well-executed system will enable future researchers to integrate or modify components, creating a more robust solution.

References

Abakarim, Y., Lahby, M., & Attioui, A. (2018). An efficient real-time model for credit card fraud detection based on deep learning. In Proceedings of the 12th International Conference on Intelligent Systems: Theories and Applications (pp. n/a). Rabat, Morocco. doi:10.1145/3289402.3289530

Amarasinghe, T., Aponso, A., & Krishnarajah, N. (2018). Critical analysis of machine learning based approaches for fraud detection in financial transactions. In Proceedings of the 2018 International Conference on Machine Learning Technologies (pp. 12–17). Jinan, China. doi:10.1145/3231884.3231894

Asaithambi, S. P. R., Venkatraman, R., & Venkatraman, S. (2020). Mobda: microservice-oriented big data architecture for smart city transport systems. Big Data and Cognitive Computing, 4(3), 17. https://doi.org/10.3390/bdcc4030017

Bartoletti, M., Pes, B., & Serusi, S. (2018). Data mining for detecting bitcoin ponzi schemes. 2018 Crypto Valley Conference on Blockchain Technology (CVCBT), Zug, Switzerland, (pp. 75-84). doi:10.1109/CVCBT.2018.00014.

Geng, Z., Cao, Y., Li, J., & Han, Y. (2022). Novel blockchain transaction provenance model with graph attention mechanism. Expert Systems with Applications, 209, 118411. doi:10.1016/j.eswa.2022.118411

Godase, A., & Attar, V. (2012). Classifier ensemble for imbalanced data stream classification. In Proceedings of the CUBE International Information Technology Conference (pp. 284–289). Pune, India. doi:10.1145/2381716.2381769

Guych, N., Nuryyev, S., Anastasia, S., Spyridou, A., Simon, Y., & Jennet, A. (2018). Factors influencing the intention to use cryptocurrency payments: An examination of the blockchain economy. In TOURMAN 2018 Conference Proceedings, Rhodes: Greece (28 October 2018) (pp. 303-310).

Ileberi, E., Sun, Y., & Wang, Z. (2021). Performance evaluation of machine learning methods for credit card fraud detection using smote and adaboost. IEEE Access, 9, 165286-165294. doi:10.1109/ACCESS.2021.3134330.

Li, Q., & Xie, Y. (2019). A behavior-cluster based imbalanced classification method for credit card fraud detection. In Proceedings of the 2019 2nd International Conference on Data Science and Information Technology (pp. 134–139). Seoul, Republic of Korea. doi:10.1145/3352411.3352433

Liu, Y., Ao, X., Qin, Z., Chi, J., Feng, J., Yang, H., & He, Q. (2021). Pick and choose: a gnn-based imbalanced learning approach for fraud detection. In Proceedings of the Web Conference 2021 (pp. 3168–3177). Ljubljana, Slovenia. doi:10.1145/3442381.3449989

Mao, R., Li, Z., & Fu, J. (2015). Fraud transaction recognition: a money flow network approach. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (pp. 1871–1874). Melbourne, Australia. doi:10.1145/2806416.2806647

Mittal, S., & Tyagi, S. (2019). Performance evaluation of machine learning algorithms for credit card fraud detection. In 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 320-324). Noida, India. doi:10.1109/CONFLUENCE.2019.8776925.

Prusti, D., Das, D., & Rath, S. K. (2021). Credit card fraud detection technique by applying a graph database model. Arabian Journal for Science and Engineering, 46(9), 1-20.

Sisodia, D. S., Reddy, N. K., & Bhandari, S. (2017). Performance evaluation of class balancing techniques for credit card fraud detection. In 2017 IEEE International Conference on Power, Control, Signals, and Instrumentation Engineering (ICPCSI) (pp. 2747-2752). Chennai, India. doi:10.1109/ICPCSI.2017.8392219.

Tae, C. M., & Hung, P. D. (2019). Comparing ml algorithms on financial fraud detection. In Proceedings of the 2019 2nd International Conference on Data Science and Information Technology (pp. 25–29). Seoul, Republic of Korea. doi:10.1145/3352411.3352416

Tran, P. H., Tran, K. P., Huong, T. T., Heuchenne, C., HienTran, P., & Le, T. M. H. (2018). Real-time data-driven approaches for credit card fraud detection. In Proceedings of the 2018 International Conference on E-Business and Applications (pp. 6–9). Da Nang, Vietnam. doi:10.1145/3194188.3194196

Wang, Y., Li, R., & Niu, Y. (2022). A deep neural network based financial statement fraud detection model: evidence from china. In Proceedings of the 2021 4th Artificial Intelligence and Cloud Computing Conference (pp. 145–149). Kyoto, Japan. doi:10.1145/3508259.3508280

Wood, E. (2021). Bitcoin transactions from 2009 to 2020 [Data set]. Harvard Dataverse. https://doi.org/10.7910/DVN/ZLBYTZ

Zhou, W., Xue, X., & Xu, Y. (2022). Credit card fraud detection based on self-paced ensemble neural network. In Proceedings of the 4th International Conference on Information Technology and Computer Communications (pp. 92–98). Guangzhou, China. doi:10.1145/3548636.3548650