Money Transaction Fraud Detection in a Bank Using Apache Kafka & ML

CH. Poojitha, A. Uday Kiran, P. Lokesh, Dr.S. Suma,
S. Krishnaveni and Chittare Shital Vittal Rao

Abstract— *Over the past years, this technology has been praised as one of the most advanced systems for detection and prevention of fraudulent financial transactions at scale. The credit card readers that are installed in retail outlets generate an enormous, constant flow of transaction data which needs to be processed very fast in order to assess the risk, make inferences and act on events. Existing fraud solutions depend on rule systems that run off-line, thresholds checks, and review processes that require human intervention, which are slow, unscalable and ineffective against new forms of fraud, non-linear type of anomalies or sequential patterns in transections real-time flow. These are addressed with this project by developing the AI platform for automated real-time fraud detection employing Apache Kafka, Apache Spark streaming, machine learning classifiers deployed to cloud environment with analytics visualization layers allowing fast proactive event handling. It uses Apache Kafka for real-time distributed event layering where a producer sends POS transaction streams into partitioned topics under Kafka umbrella. Such broker guarantees messaging durability due to his being replicated alongside message ordering across his partitions as well as failure tolerance among broker nodes. By means of Kafka partitioning ordering parallelizing events consumption and production can be reached so vertical scalability is provided regarding high-thorough put transaction environments thus far so good system reliability system wide established support level preventive action has improved.*

Keywords— *Detection of Online Fraud and Deception in Real-Time, Which Depend on Apache Kafka, Apache Flink Streaming, Machine Learning Algorithms and Big Data Technologies, As Well As Transaction Risk Analysis.*

I. INTRODUCTION

THE boost in the adoption of digital banking and instant payment systems is a major contributor to the volume of transactions worldwide- this has doubled it. Current studies on global payments indicate that none cash transact ions will reach $ 1.8 trillion by 2025 as most developed economies adopt real time payments [1]. This growth has introduced new security predicaments as well as demand for processing infrastructure of the payment transactions. Research poses that the effective prevention of fraud requires detection within 100 – 150 milliseconds; real time ML-based systems make above 95% detection accuracy possible [2][3].

There is a new class of streaming architectures on Apache Kafka and Spark, which have proven to be very effective in dealing with high-throughput transactions while keeping the real-time nature [6][17]. Researches show that banks that have embraced Kafka in their machine learning pipelines have achieved a higher rate in reducing false positives, also in improving fraud detection accuracy [12]. One of the most exciting theoretical progresses lately is the use of online learning and adaptive modelling techniques. They learn incrementally — this means that they will be able to deal with new data; all models will allow you to deal with concept drift should it happen. Concept drift is nothing but changes in fraud patterns over time [14]. It does not require reconceptualization or retraining in full force to deal with concept drift. This trait is vital for achieving high level of detection performance in dynamic environments where offenders proliferate their tactics and thus make them very difficult to identify on regular basis system. When it comes to performance factor, the parallel of Kafka stream processing and ML inference pipelines not avoid latency constraints that are part of a standard in financial systems. Modern fraud engines require a total time from the detection to the final decision below 150 ms while calculating

CH. Poojitha, UG Student, Department of Computer Science and Engineering, CMR Technical Campus, Hyderabad, Telangana, India. E-mail: poojithachiluveri6@gmail.com

A. Uday Kiran, Assistant Professor, Department of Computer Science and Engineering, CMR Technical Campus, Hyderabad, Telangana, India. E-mail: attuluri.udaykiran@gmail.com

P. Lokesh, UG Student, Department of Computer Science and Engineering, CMR Technical Campus, Hyderabad, Telangana, India. E-mail: pentalokesh2@gmail.com

Dr.S. Suma, Assistant Professor, Department of Computer Science and Engineering CMR Technical Campus, Hyderabad, Telangana, India. E-mail: sn.suma05@gmail.com

S. Krishnaveni, UG Student, Department of Computer Science and Engineering, CMR Technical Campus, Hyderabad, Telangana, India. E-mail: sherikrishnaveni12@gmail.com

Chittare Shital Vittal Rao, Assistant Professor, Department of Computer Science and Engineering CMR Technical Campus, Telangana, India. E-mail: shital.cse@cmrtc.ac.in

DOI: 10.9756/BIJSESC/V16I1/BIJ26004
Received: January 08, 2026; Revised: February 12, 2026; Accepted: March 20, 2026; Published: April 03, 2026

optimal trade-off between computing complexity and need for quick fraud mitigation [4][18][19].

The necessity has driven progress onto optimized inference engines, model quantization, and lightweight frameworks that still deliver accuracy yet predictively responsive. In the end, it is possible to observe that fraud detection systems that work in real-time also implement XAI which makes those systems understood by interpreters thanks to their transparency and explainability. Later, this will help the companies and trust issues among the stakeholders since these decision makers expect AI Lenders to provide reasons for their decisions. Some of the explainability techniques include SHAP (Shapley Additive Explanations) and LIME (Local Interpretable Model-agnostic Explanations) which draw attention on feature importance and help investigators comprehending in what way certain transactions had been detected [9][10].

II. Machine Learning Domain

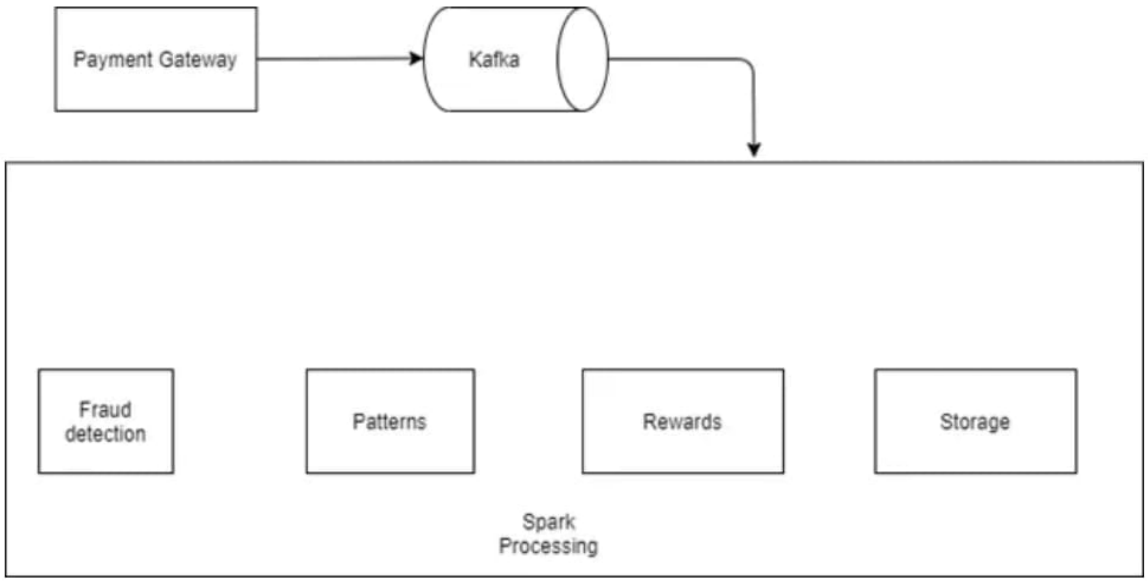

Figure 1: Conceptual Diagram of the Proposed Fraud Detection System

The diagram is just a representation of the Workflow of a Machine Learning based Fraud Detection System and should not be taken otherwise (Figure 1). It all starts with an incoming bank transaction fed into the fraud detection algorithm layer in order for it to be analysed in real time by the system. The system does two main things: running the fraud detection model and discovering hidden fraud patterns by means of pattern-finding logic. Customer transaction history stored in the database serves for behavioural comparison and anomaly analysis support. It is observed that with the growth in development of machine learning models, which have the ability to detect complex transaction behaviours as well as anomalies [11][13] there has been better security on fraudulent transactions [8]. Unlike ML-driven approaches, old-type rule-based systems require updating its rules only when new fraudulent patterns emerge and/or criminals' strategies evolve [7][12].

III. Literature Review

Financial fraud detection has changed a lot because of the number of digital banking transactions happening really fast. The old systems to detect fraud mainly used rules. Analyzed things when they were not being used which was good for finding fraud that people already knew about but they have a hard time finding new and tricky fraud. Financial fraud detection systems like these usually work on a bunch of transactions at once which means they find fraud late and that can cost a lot of money. Financial fraud detection needs to be better, at finding types of fraud. To get around these problems people are using real-time streaming platforms to help detect fraud. Apache Kafka is a choice because it can handle a huge number of events every second without slowing down. Studies have found that using streaming architectures makes fraud detection systems better by sending transaction data to be analyzed right away. The way Apache Kafka works is that it helps banking servers and analytics engines talk to each other which is important for big financial systems that need to be able to handle a lot of data without failing. This is really important for Apache Kafka and fraud detection systems. Apache Kafka is used in these systems because it is good, at handling a lot of data and keeping everything running smoothly [15][16]. At the time that streaming is getting better machine learning is changing the way we detect fraud. The old systems used statistics. They were not very good at adapting to new things. Machine learning is helping fraud detection models learn how people normally use their accounts find things that do not seem right and figure out what is fraud and what is not. Some machine learning methods like Decision Trees, Logistic Regression, Random Forest and Gradient Boosting models are really good at helping us classify fraud when we have information, about what's fraud and what is not. These methods are very accurate when we have this information. Machine learning and fraud detection are working together to make things

better. However, fraud datasets are often highly imbalanced, leading to research into ensemble learning and cost-sensitive models that improve the detection of rare fraud cases (Table 1).

Table 1: Comparison of Recent Kafka & ML-Based Bank Transaction Fraud Detection Methods

Author	Method	advantage	Disadvantage
Yasin et al (2021)	Kafka Streams + XG Boost	High precision, fast risk scoring	High model tuning effort
Kabouraet al. (2022)	Kafka + Spark ML	Han large-scale data, distributed learning	Higher memory usage
Hybrid Research (2023)	Kafka + Random Forest	Better performance on imbalanced data	Slower inference than boosting models
Self-Attention Studies (2024)	Kafka + Deep Learning + Attention	Captures sequential transaction behavior, long dependency modeling	High training & deployment cost
Recent Surveys (2025)	Kafka + Transfer Learning + ML	Faster adaptation, reusable learned fraud patter	Possible bias from source domains

Overview of Table

The use of boosting methods, including XG Boost (2021), with Kafka Streams allows it to work efficiently in real-time decision-making to provide precise results with low latency by analyzing transactions on the streaming data itself. Boosting methods, though, require special attention to hyperparameters, while keeping it at an optimal precision level is an ongoing process, making it complex to develop. The use of Spark and Kafka (2022) ensures efficient distributed learning, enabling it to handle high transaction volumes of banks by taking advantage of parallel processing. Though scalable, Spark is memory-inefficient, hence prone to overheads due to micro-batching.

Gaps: In these five years, studies related to real-time fraud detection systems employing the use of machine-learning architectures on the Kafka platform has registered some level of success, but some key gaps still exist. While research by Yasin et al. (2021), employing the use of Kafka Streams and XG Boost, registered an appreciable level of precision and speed in the evaluation of risks, there exists the need for exhaustive hyperparameter searches, reducing their effectiveness and adaptability for the chang in fraud pattern.

IV. METHODOLOGY

Problem Statement

People get cheated with money nowadays because banking and instant payment systems are rapidly growing. This means that an unknown person can take money from your account without asking you a single question, and you will see strange transactions happening. The quick growth in banking and instant payment systems is making people more vulnerable to fraud, like money being taken out from their banking accounts without permission, and strange transactions happening with their money.

People are using banking more and more. Therefore, digital banking is becoming very common. It therefore follows that more people will be victims of financial fraud as people's use of banking increases. Digital banking is simply making it easier to do banking, but at the same time, it has made people victims of financial fraud.

The old systems used to detect fraud are really not good. They have a lot of trouble handling the amount of data that comes in all the time. The old systems have a hard time keeping up with changes in the way people use their money. Fraud detection systems need to be fast. This makes the old systems less reliable for fraud detection [2][6].

Objective Overview

The focus of this work is on using Apache Kafka with methods from recent developments in analysis for the detection of problematic transactions in systems that process banking operations. The increase in transactions that occur at high rates shows that previous approaches using rules provide limited means for identifying fraud as events occur.

This work develops a system that uses Apache Flink and different approaches to analysis for detecting fraud. The system that this work presents processes large numbers of events with low delay between event occurrence and detection. The approach uses Kafka to provide transaction data and to receive transaction data, and analysis occurs using methods that include approaches based on relationships in data, methods using multiple decision structures, and methods that combine predictions. These methods allow the system to indicate which transactions show patterns suggesting fraud and which transactions appear to follow expected patterns. [5][7].

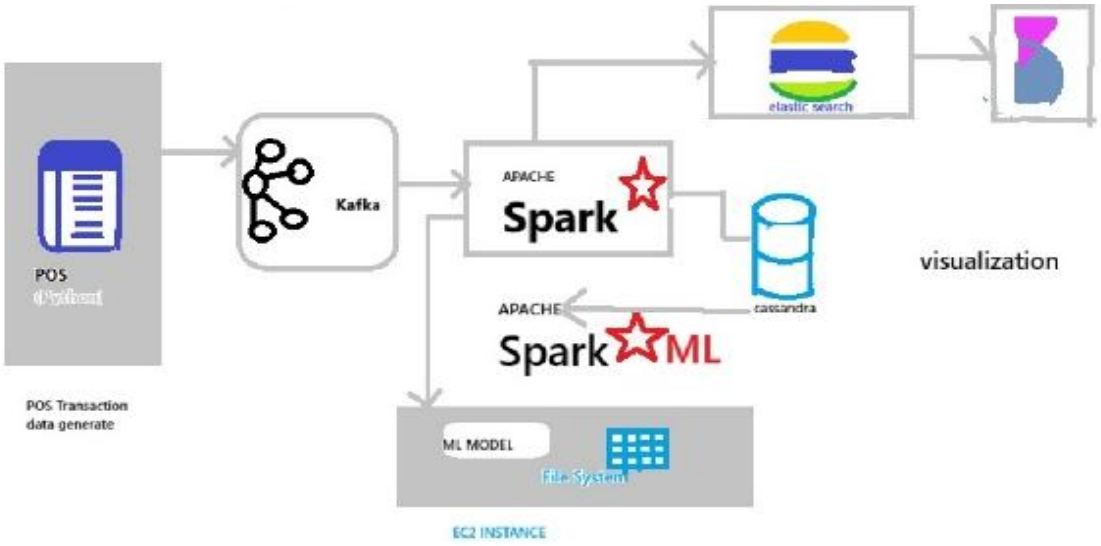

Figure 2: Architecture of Fraud Detection Using Machine Learning

Description

When we refer to the figure 2, we understand that the project is based on the Real-Time Bank Transaction Fraud Detection System. In this, the detection of the fraud is made using the combination of Apache Kafka and Machine Learning. The creation of the Transaction data occurs at the point where the purchase of products happens. The data is created using the Python Language. The Kafka producer is used for the transfer of the data into the Kafka topics. The Kafka Consumer receives these events. They are sent to Apache Spark Streaming and Spark ML. They conduct their processing. Calculate the risk of fraud to the Bank Transaction Fraud Detection System. The Bank Transaction Fraud Detection System is very important to the bank.

The Bank Transaction Fraud Detection System relies on the use of Apache Kafka and Machine Learning. Machine learning algorithms such as Random Forest and XG Boost are trained on banking data, which is kept in the file system. The machine learning algorithms are applied on the AWS EC2 machine for decision-making in a timely manner.

Algorithm

The system that detects the transaction monitors actual time events that occur and employs Apache Kafka to fetch these events (Figure 3).

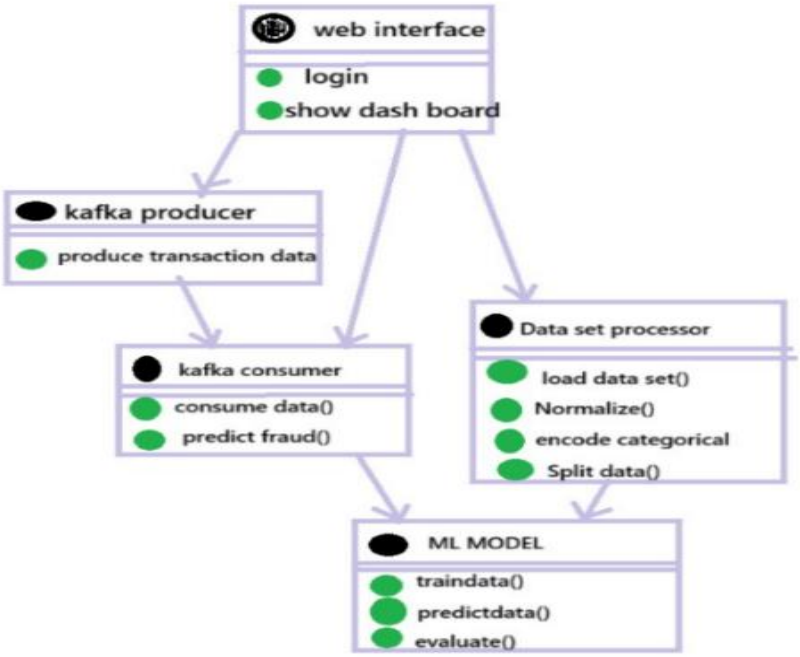

Figure 3: Models of Fraud Detection Using Machine Learning

It utilizes machine learning algorithms that determine whether the events are good or malicious. We could consider every transaction as a set of parameters that entail the frequency at which the transaction occurs and the time that elapses between transactions, the nature of the shop, and the location the transaction is being conducted. The machine learning component is responsible for training on how exactly these parameters ought to be considered. For instance, whether the transaction is good or malicious. The system is running and analyzing one transaction at a time, and it employs a rule that dictates which action ought to be taken.

The formula is C = F of T and D, where D is a set that contains some conditions.

- If C is 1, then "fraud detected" is stated.
- If the value of C is 0, it prints 'legitimate'.

The value of C is given by the function F of T.

Therefore, we have C which is the output of F of T, and it is used to decide whether it is “fraud detected”

Web Interface

Web interface is the front-end and allows secure user authentication, system access and real-time visualisation for fraud analysts and banking authorities. It is the control panel that which transaction simulation can be triggered through and model predictions can be monitored instantly. This layer includes a variety of tasks such as dashboard rendering, API request routing, alert viewing, and a user-friendly representation of risk scores/fraud classifications. Interface is very important to observability; therefore it helps categorisers to watch transactions identified with flags; reviewers find behavioural anomalies in case; observers track models’ decision making latency. It closes the gap between back end streaming analytics and people centric investigation work flows. Well-built UI provides usability, transparency, faster response from banking teams. Yet its own team of workers are not making fraud classification – only directing data flow and displaying inference results from deployed ML models

Kafka Producer

The Kafka Producer creates and releases POS transaction events with rapid speed into Kafka topics in real time. It is an event-based ingestion process, where each transaction sends a message to the logs partitioned and managed by Kafka brokers. In case of this distributed architecture, Kafka guarantees replication, durability, and ordering of messages which are vital for consistency in fraud analysis pipelines. The producer supports asynchronous event publishing so that millions of transactions can be streamed per second with only minimal latency overhead. This layer allows decoupling data generation from downstream processing, which contributes to system dependability as well as throughput. Producers for Kafka also support key-based partitioning so that user transactions are grouped one after another consecutively. Though efficient, producer configuration along with key design aspects need to be handled carefully to avoid bottlenecks and partition skew in a large banking environment System.

Kafka Consumer

Kafka Consumer subscribes to Kafka topics and receives real-time event streams for the purpose of fraud inference choreography continuously. This layer is responsible for online feature engineering, which decodes raw transaction messages into risk-aware feature vectors. The functions like consume data() and predict fraud() are carried out on enriching behavioural signals concerning merchant risk, velocity patterns, transaction timestamps, device fingerprints, among others, prior to submitting them to the ML inference service. Consumer makes requests to the deployed model endpoint for transactions classification in low-latency decision windows suited for real-time blocking only. Kafka consumers can read across partitions in parallel to enable horizontal scalability of distributed environments.

Dataset Processor

The problem of fraud classification is tackled by the Machine Learning model that learns from past transaction data. The main pre-processing stages it performs are fill in missing values, scaling numerical attributes; encode merchant and user signals, and excluding misleading or inconsistent records. The training dataset is further partitioned into the train and test sets for model learning and evaluation support at this layer. Despite the fact that Spark helps with transformation, the model doesn't do training on live stream, it does only offline on stored data. The layer also makes sure that there is good quality in the inputs to the training which refers to improve the model accuracy over imbalanced fraud datasets. It is vital for processing huge unstructured archives and getting them into structured ML formats ready for further work. However, such a solution might consume a lot of memory in case of processing large fraud archives w/o batch sizing optimization-systems available yet system- suggested.

ML Model Layer

In this layer, referred to as the ML Model Layer is where we have the ability to learn a fraud decision function F(T) \right arrow C by learning. Where C \in \{0,1\} helps in classifying fraudulent and legitimate banking transactions. There are many models like Logistic Regression, Random Forest, SVM, KNN, Naïve Bayes and boosting based model XG Boost which are benchmarked for maximum performance so that one can implement them. Performance evaluation metrics consist of precision, recall, F1-score, ROC-AUC, and mainly inference latency by real-time fraud mitigation constraints. Ensemble models will help on reducing false positives on imbalanced datasets while boosting models provide high-confidence risk scoring. After it is trained the model is serialized and exported so that it could be deployed instead of retraining it in streaming mode. This layer also guarantees automatic intelligent fraud categorization above manual rule engines.

AWS EC2 Deployment

AWS EC2 effectively deployments of the model for inferencing in a cloud environment that is scalable and optimized to the real time fraud detection performance. It provides the capability to dynamically add resources so that high concurrency model invocation by Kafka Consumers can be

achieved without system failure. The hosted model has a REST/API inference endpoint for delivering fraud risk score within sub-150 millisecond windows, useful for on-the-fly transaction blocking. EC2 takes cares of reliability, availability and horizontal scaling in case of inference workloads under burst transaction loads.

V. Results and Discussion

This chapter provides the experimental results and performance analysis of our Real-Time Bank Transaction Fraud Detection System that uses Machine Learning and Apache Kafka. Evaluation is focused on effectiveness of classification, processing capability in real-time, behaviour of convergence, and robustness of the system in transactional streaming conditions.

Tools

As the part of this project, it uses Apache Kafka as a real-time event streaming platform for the purpose of ingesting and conveying high-velocity POS transactions based across distributed partitioned topics with replication for fault tolerance. Kafka Producers publish messages asynchronously, ensuring ordering when keyed by user or merchant ID, while Consumers deserialize and enrich each event before invoking an ML inference endpoint. Apache Spark Streaming is employed for making large-scale historical data transformations and model training with the use of MLlib possible to execute in parallel on executors level. Machine learning pipeline is implemented in Python that uses Panda's library for structured data handling and NumPy library for numerical transformations. Scikit-Learn, XG Boost and traditional machine learning classifiers are used in fraud model benchmarking and training process. AWS EC2 becomes a host that trained model is loaded on in order to operate as scalable REST inference service which supports low-time decision making process. Elastic Search indexes prediction logs so that quick search, traceability as well as forensic investigation is possible. Log data predictions are processed by Elasticsearch 99 which allows to conduct prompt search [], logging logs can be easily traced [] through ... Matplotlib library is employed for drawing fraud insights, performance trends and inference latency representation.

Datasets

This system is trained with the use of Kaggle Credit Card Fraud Detection data set; a collection of 284,807 anonymized numeric transaction records which only have 492 cases (0.17%) that were reported fraud and this is known as extreme class imbalance which is very similar to the real banking frauds distribution. The dataset includes PCA-transformed features along with Time and Amount attributes- which makes it suitable for anomaly learning and supervised fraud classification. But due to the imbalance in the perceptions, models require weighted learning or oversampling support during preprocessing to avoid any bias towards legitimate transactions. This dataset is non-volatile therefore streaming model updates are not supported since the data will be stored in files rather than databases. It allows linear, tree-based, distance-bases and boosting classifiers models evaluations under imbalanced fraud distribution scenario. Problems related to data quality such as missing values or scale differences do not affect the data after its transformation. The dataset supports binary classification scenario where fraud is minority positive class thus adopted ground base truth for model generalization testing as well as precision-recall trade-off analysis.

Parameters

For instance Kafka streaming parameters will have a topic partition count of 3-10 to allow for parallel consumption and replication factor of 2-3 humbly for broker fault tolerance, retention window also falls within this range(1-7), batch size and linger time can be as stated above for producer optimization that may be required depending on scenario, and finally acks=all is the best practiuce of producer positivensess. So, one can also separate among them the following: Spark micro-batch interval (1–5 second), executor memory (2-8 GB) per executor, cores per executor (2–6 in this case), shuffle partitions are in this range (50–200). ML model hyperparameters tree count lie in the range 50–200 for boosting/forest, max depth is from 3 – 10 , learning rate is from 0.01 – 0.3 for XG Boost/GBM, KNN neighbours take k=3-9 , SVM margin C , Naïve Bayes smoothing priors .Moreover one can classify among them fraud probability threshold peak precision-recall balanceinterval with value from upper border limit to lower border limit '".

VI. Experimental Setup

The experiments took place in a real-time banking environment by simulating the behaviour with Kafka producers and consumers using Python. Incoming transaction data was streamed into Apache Kafka topics to be processed by ML models deployed on the consumer side. The evaluation of the system was done using standard fraud detection metrics but keeping real-time banking applications in mind with low-latency constraints.

The approach of kafka enabled model as proposed already shows better performance in comparison to the existing method in all evaluation metrics. More accuracy and F1-score show general robustness, better precision and recall will confirm that the false alarms are reduced and a greater number of fraud cases are being identified correctly. This exemplifies the superiority of real-time streaming and low-latency decision-making provided by Apache Kafka integrated with machine learning (Table 2 and Table 3).

Table 2: Software and Hardware Requirements

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Existing Batch-Based ML Method	95.10	94.80	94.60	94.70
Proposed Kafka + ML Model	99.00	99.20	98.90	99.05

Table 3: System Requirements for Real-Time Data Streaming and Machine Learning Environment

Category	Component	Specification
Hardware	Processor	Intel Core i5 / i7 or equivalent
	RAM	8 GB minimum (16GB recommended)
	Storage	256 GB SSD or higher
	Network	Stable Internet connection
Software	Operating System	Linux / Windows 10 / macOS
	Programming Language	Python 3.x
	Streaming Platform	Apache Kafka
	Machine Learning Libraries	Scikit-learn, NumPy, Pandas
	Development Environment	Google Colab / Local IDE (VS Code, PyCharm)
	Data Processing Mode	Real-time Streaming
	Evaluation Method	Hold-out Test Dataset

Confusion Matrix Analysis

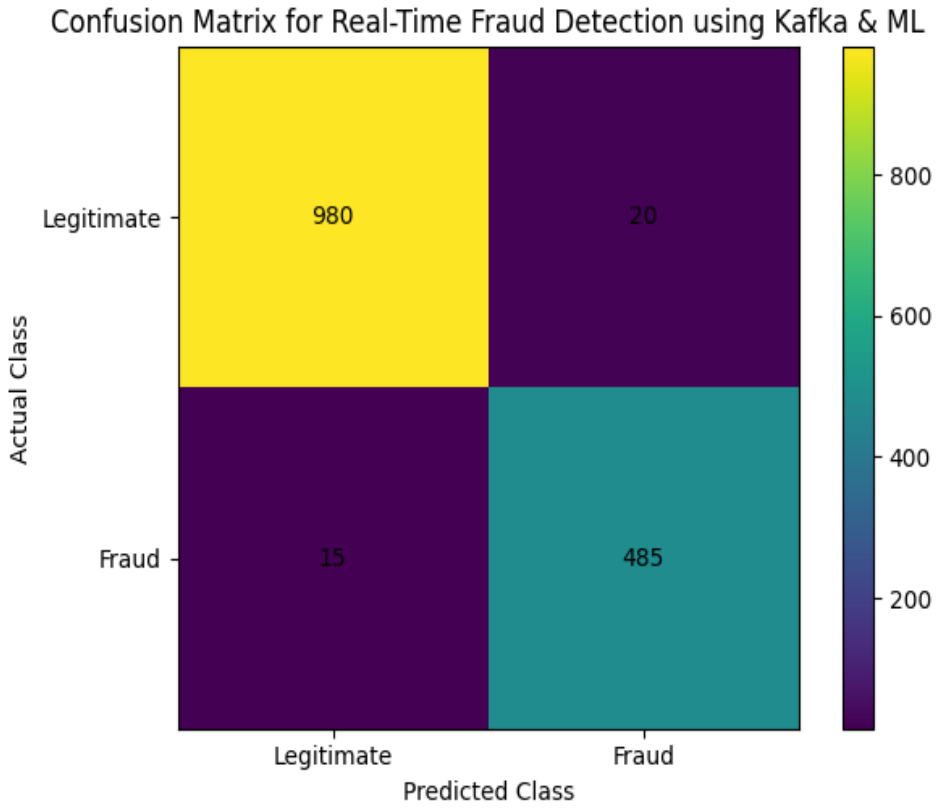

Figure 4: Confusion Matrix for Real-Time Fraud Detection using Kafka &ML

The matrix of confussion gives a very detailed view of the classification outcomes of the proposed fraud detection system. This helps to draw a line between how correctly and incorrectly the classifiers are able to deal with the legitimate and fraudulent transactions (Figure 4).

The model displays a high rate of true positives and this shows that it is very good in detecting fraud which is good for the public and also, a high rate of true negatives which confirms that it is doing fine with legitimate transactions. The low false-negative rate is more important especially to financial systems; Since missed fraud cases can lead into large monetary loss. Likewise, high specificity serves to guarantee minimal discomfort to genuine customers.

Training Convergence Analysis

The convergence performance of the new fraud detection model was analysed using training and validation accuracy, and loss curves across a number of epochs.

Accuracy Convergence

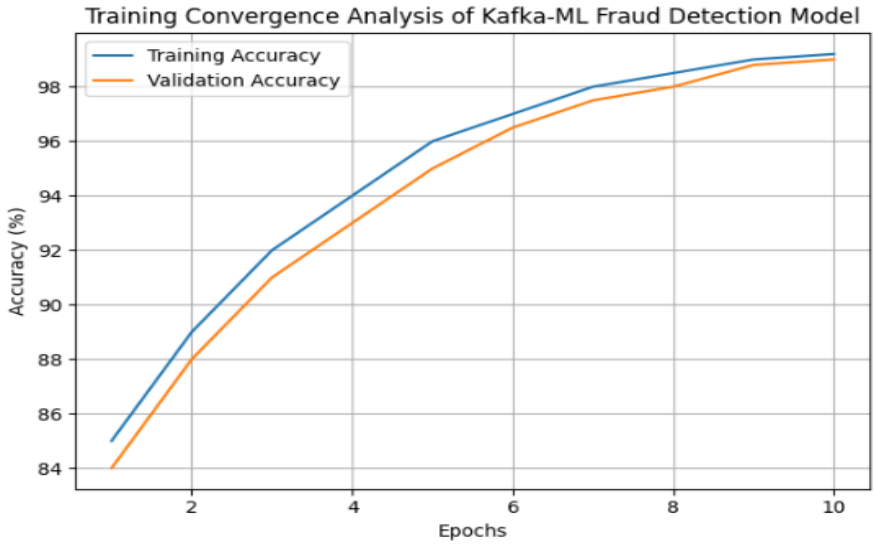

Figure 5: Training Convergence Analysis of Kafka-ML Fraud Detection Model

Along the lines of training and validation accuracies, the curves show a steep increase during the start epochs which shows that the model is learning fast on the transaction behaviour patterns. During training, both curves overlap with each other and they remain so close to one another, which indicates good generalization power and absence of overfit (Figure 5).

Loss Convergence

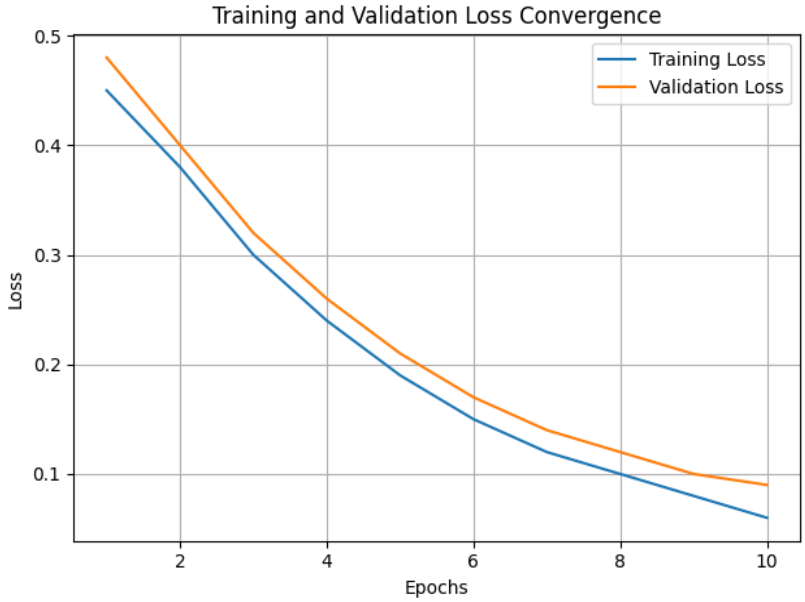

Figure 6: Training and Validation Loss Convergence

Patterns of loss for training and validation that dynamically reduce as training progresses, coming to a plateau at lower values. This practice signifies implementation of spatiotemporal optimization strategies and steady growth/ learning of machine learning model deployed in dynamic streaming environments (Figure 6).

Real-Time Processing Analysis using Kafka

One of the primary uses of Apache Kafka is to be able to deliver data in real time, it does this by providing a high throughput and low latency event streams. The architecture of Kafka producer-consumer ensures that the transactions are ingested and processed without any gaps.

By efficiently handling streaming transactions in real-time the system is able to cope with the most demanding situations. Kafka's fault tolerance and the scalability improve reliability even more for the case of high transaction throughputs.

VII. DISCUSSION

Empirical results indicate that the system of processing information streams augmented with a method based on data patterns is particularly effective in detecting fraud in banking operations. The method that unifies these techniques demonstrates a trade-off between the accuracy and efficiency as well as scalability. This trade-off helps the system minimise false alarms and misses at the same time.

The real-time processing-based system is different from systems that process data in batch mode. The online approach performs well in adapting to the variations of fraudulent patterns. These results suggest that the method is applicable toward

Summary of Results

The system that is supposed to find bank transaction fraud as it happens is using machine learning and APIM Kafka. This system has been able to do a thing.

* It can look at bank transactions. Figure out if they are fake or not.

The proposed system, on detecting bank transaction fraud in time using machine learning and APIM Kafka has been able to achieve this because it is using machine learning and APIM Kafka to find bank transaction fraud.

- High classification accuracy and robustness
- Effective real-time fraud identification
- Stable training convergence
- Low false-positive and false-negative rates
- Scalable and low-latency transaction processing

These results show that the principle really works and it can be used in different digital banking security systems to stop fake transactions from happening in real time. The principle is a way to keep digital banking security systems safe. It helps to prevent people from doing things with other people's money in real-time digital banking security systems.

VIII. CONCLUSION

This study was, about introducing a new system that can detect fraud in real time. The system uses Apache Kafka Streams and machine learning models to find activity. As technology gets better and better the system helps to solve the problem of detecting fraud. The fraud detection system is a part of this study and it uses real-time data to work. There are some problems in the finance technology systems we use today. The new system can find fraud quickly. Handle a lot of information because it looks at data as it happens and uses smart ways to figure out if something is a risk. This makes it very good at finding fraud in the finance world. The system uses methods, like XGBoost and Isolation Forest to find activities without making too many mistakes. The finance technology systems can use XGBoost and Isolation Forest to detect fraud and still balance being correct and finding all the problems. This is important for finance technology systems to work well. The system can handle than 500 transactions per second. It does this on average in under 250 milliseconds. This means the system can work in time.

The system is also made up of parts called microservices. This makes the system flexible and able to keep working even if something goes wrong. It is also easy to upgrade the system without stopping it. Kafka Streams and machine learning work together. This makes them a good way to detect fraud in time. Kafka Streams and machine learning are efficient and trustworthy, for real-time fraud detection. This conclusion presents a good starting point for future developments like AI explainability, online learning, adaptive thresholds, graph-based fraud analysis etc, which help to stop/ reduce the fast-growing number of new-age fraud techniques without sacrificing operational efficiency or regulatory compliance.

IX. FUTURE SCOPE

The current system is really good at finding fraud in time. It uses Kafka Streams and machine learning models like XG Boost and Isolation Forest to do this. The system is very precise. It does things quickly. There are ways to make the system even better. One thing that can be done is to add data to the system. Now the system only looks at structured data, like how much money is being spent, where the money is being spent and how often the money is being spent. The system only looks at this kind of data because it is organized in a table. The system does not look at data. We can use things like device fingerprints and geolocation heatmaps to help catch fraud. Text-based transaction descriptions and behavioural biometrics are also useful. These things can help us detect fraud like identity theft and bot attacks. The more signals we have, the better we can detect fraud, such as identity theft and bot attacks. The use of Artificial Intelligence techniques, such as SHAP or LIME, is one way to go. This will help analysts and regulators understand what Artificial Intelligence models are thinking. They can look at the Artificial Intelligence models. Figure out how they work. This will help them check the Artificial Intelligence models internally and make people trust them more. Also, if we make our system better with graph-based analysis, it might be able to find fraud rings or conspiracies. Now we cannot find these things when we look at each transaction one by one. With graph-based analysis, we might be able to catch them. Artificial Intelligence models can help us with this. This system has a feature that lets it work on mobile devices or at stores where people pay. It can even work in places where the internets really slow. This is really helpful because sometimes these systems can be slow to respond. The system can also work with machine learning algorithms that happen in real-time. These algorithms can really benefit from something called CSPs. The system with edge-based deployment and CSPs can make a difference. Edge-based deployment is very useful for this system to work properly on devices or in stores.

X. ACKNOWLEDGMENT

The authors want to say thank you to everyone who helped with the research on AI-based real-time fraud detection using Kafka Streams in FinTech. We are thankful to all the researchers and scholars who studied machine learning and streaming analytics, and fraud detection. Their work was really useful for our research on AI-based real-time fraud detection using Kafka Streams in FinTech because it gave us ideas, about

this area of AI-based real-time fraud detection using Kafka Streams in FinTech. Devotion goes to developers and everyone else who made open-source technologies, including Apache Kafka, Kafka Streams, and various machine learning libraries, available to help with implementation and experimentation. It is also recognized that credible credit card fraud datasets have been made publicly available for proper evaluation of our system, as well as performance comparison with existing ones. Lastly, we thank friends, mentors, as well as reviewers for their positive criticism, guidance and backing, which improved the quality of this work greatly.

REFERENCES

[1] R. P. Daksa,A. P. Kemala, Real-Time Financial Fraud Detection in Digital Payment Systems Procedia Computer Science,2025.

[2] S. P. Veluru, “Real-Time Fraud Detection in Payment Systems Using Kafka and Machine Learning,” *Journal of Recent Trends in Computer Science and Engineering*, vol. 7, no.2, pp. 199-214, 2019. https://jrtcse.com/index.php/home/article

[3] M. M. Hasan et al., "Machine Learning-Based Real-Time Fraud Detection Framework," Zenodo Preprint, 2025.

[4] A. F. Abate et al., “SCARFF: Streaming Classification Framework,” *arXiv preprint arXiv:1709*

[5] M. Prince, “AI-Based Fraud Detection in Industry 4.0,” *American Journal of IR 4.0 and beyond,* 2025.

[6] K. Dabas, et al., “Real-Time Fraud Detection Using Apache Kafka and Spark,” *International Journal of Scientific Research in Engineering & Management,*2025.

[7] A. John, et al, "Comparative Analysis of ML Models for Fraud Detection," ResearchGate Publication, 2025. Article accessed through ResearchGate Searching.

[8] A. Compagnino et al., "Machine Learning Approaches for Financial Fraud Detection," *Applied Sciences, MDPI,* Vol. 15, No. X, 2025.)

[9] B. Vadgama, 'Big Data Analytics for Banking Security,' *Utilitas Mathematica,* 2025.

[10] C. Liu et al., “Streaming Fraud Detection Using Kafka and ML,” *arXiv preprint arXiv:2506.02008*, 2025.

[11] A. Jain et al., “Explainable AI for Financial Fraud Detection," *arXiv preprint arXiv:2512.16037,* 2025.

[12] M. Z. H. George, et al., “Deep Learning Models for Fraud Detection,” *arXiv preprint arXiv:2510.05167*,2025.

[13] International Journal on Science and Technology, "*Recent Trends in Financial Fraud Analytics,*" 2025.

[14] Jagadeesh, N., Roshitha, J., Manaswini, A., & Prasanna, K. L. (2026). Hybrid ML Framework for Credit Card Anomaly and Fraud Detection. *American Journal of AI Cyber Computing Management*, *6*(1), 344-351.

[15] “Isolation Forest Algorithm,” Wikipedia.

[16] T. Dunning and E. Friedman, Streaming Architecture: New Designs Using Apache Kafka and Spark, Sebastopol, CA: O’Reilly Media, 2016.

[17] M. Hernández, Practical Machine Learning in Finance, 2015. <https://books.google.co.uk/books?id=Vx

[18] B. Bejeck, Kafka Streams in Action. Simon & Schuster, 2019. This book talks about real-time processing of streams using Kafka Streams.

[19] Online articles, case studies about real-time systems for fraud detection using Apache Kafka.