ISSN 2319-5991 www.ijerst.com
© 2026 IJERST. All Rights Reserved
Peer Reviewed, Referred & Indexed Journal

Research Paper

REAL TIME BANK TRANSACTION FRAUD DETECTION USING KAFKA AND ML

[1]PASUPULETI KALYANI SRAVYA, [2]Y SRINIVAS RAJU

[1]Students, Department of MCA, B V Raju College, Bhimavaram Ap

[2]Assistant Professor**,** Department of MCA, B V Raju College, Bhimavaram Ap

ABSTRACT

With the rapid growth of digital banking and online transactions, financial fraud has become a major concern for banks and customers. Traditional fraud detection systems often rely on batch processing and are unable to detect fraudulent activities in real time, leading to significant financial losses. This project proposes a real-time bank transaction fraud detection system using Apache Kafka and machine learning techniques to address this challenge. The system utilizes Apache Kafka as a distributed streaming platform where transaction data is continuously generated by a producer and consumed by a consumer. The producer streams transaction data to Kafka topics, while the consumer retrieves the data in real time and passes it to a machine learning model for classification. The dataset undergoes preprocessing steps such as converting non-numeric data into numeric format, normalization, and feature extraction. A Random Forest algorithm is used for fraud detection due to its robustness and ability to handle complex data patterns. The trained model predicts whether a transaction is normal or fraudulent based on input features. The system is integrated with a web interface using Flask, allowing users to generate transaction streams and view prediction results. Experimental results show that the system can successfully detect fraudulent transactions in real time, although accuracy depends on dataset size and quality. This approach provides a scalable and efficient solution for real-time fraud detection, enhancing security in banking systems and reducing financial risks.

Keywords : Fraud Detection, Apache Kafka, Machine Learning, Real-Time Processing, Random Forest, Streaming Data, Banking Security, Data Analytics, Distributed Systems, Flask

I. INTRODUCTION

The rapid growth of digital banking and online financial transactions has significantly increased the risk of fraudulent activities. Financial institutions process millions of transactions daily, making it challenging to detect fraudulent behavior using traditional

methods. Conventional fraud detection systems rely on batch processing, where transactions are analyzed after a certain period. This delay in detection allows fraudulent transactions to occur and causes financial losses. Therefore, there is a strong need for real-time fraud detection systems that can analyze transactions instantly and prevent suspicious activities before they cause damage.

With advancements in big data technologies and machine learning, it is now possible to design systems that can process and analyze streaming data in real time. Apache Kafka is a powerful distributed streaming platform that enables real-time data processing by allowing producers to send data streams and consumers to process them continuously. Kafka ensures high throughput, scalability, and fault tolerance, making it suitable for handling large volumes of banking transactions. By integrating Kafka with machine learning models, it becomes possible to detect fraud as soon as a transaction occurs.

In this project, a real-time bank transaction fraud detection system is proposed using Apache Kafka and machine learning techniques. The system uses a producer-consumer architecture to stream transaction data and applies a Random Forest model to classify transactions as normal or fraudulent. The dataset is preprocessed to convert non-numeric values into numeric form and normalized for better performance. A web-based interface is provided to simulate transaction streaming and display prediction results. This approach enables real-time monitoring and detection of fraud, improving the security and reliability of banking systems.

II SURVEY OF RESEARCH

1. Traditional Fraud Detection Systems

Traditional fraud detection methods in banking relied on rule-based systems and manual verification processes. These systems used predefined rules such as transaction limits, location mismatches, and unusual spending patterns to identify fraudulent activities. While effective for simple cases, these methods lack flexibility and fail to detect new or evolving fraud patterns. Research shows that rule-based systems generate a high number of false positives and are not scalable for large volumes of transactions. This limitation has led to the adoption of machine learning-based approaches for more accurate and adaptive fraud detection.

2. Machine Learning in Fraud Detection

Machine learning algorithms have been widely used to improve fraud detection accuracy by learning patterns from historical transaction data. Algorithms such as Logistic Regression, Decision Trees, Random Forest, and Neural Networks can classify transactions as normal or fraudulent based on learned features. Research indicates that ensemble methods like Random Forest provide better performance due to their

ability to handle complex and imbalanced datasets. This project utilizes Random Forest for effective fraud classification.

3. Real-Time Data Processing Systems

Real-time fraud detection requires processing data as it is generated. Technologies such as Apache Kafka and Apache Spark Streaming have been developed to handle real-time data streams. Kafka provides a distributed messaging system that ensures reliable and scalable data streaming. Research highlights that real-time processing systems significantly reduce detection delays and improve response time. This project integrates Kafka to enable real-time transaction monitoring and analysis.

4. Data Preprocessing and Feature Engineering

Data preprocessing is a critical step in fraud detection systems. Financial datasets often contain both numeric and non-numeric data, requiring transformation into a suitable format for machine learning models. Techniques such as normalization, encoding categorical variables, and feature scaling are used to improve model performance. Research shows that proper feature engineering enhances the accuracy of fraud detection models. This project applies preprocessing techniques to prepare the dataset for training.

5. Streaming Architecture (Producer-Consumer Model)

The producer-consumer architecture is widely used in streaming systems to handle continuous data flow. In Kafka, producers generate data streams and publish them to topics, while consumers subscribe to these topics and process the data. Research indicates that this architecture ensures scalability, fault tolerance, and efficient data handling. This project uses the Kafka producer-consumer model to stream transaction data and perform real-time fraud detection.

6. Evaluation Metrics and Challenges

Evaluating fraud detection systems requires metrics such as accuracy, precision, recall, and F1-score. Since fraud datasets are often imbalanced, precision and recall are particularly important. Research highlights challenges such as data imbalance, false positives, and real-time processing constraints. This project evaluates model performance using standard metrics and demonstrates the effectiveness of combining machine learning with streaming technologies for real-time fraud detection.

III. WORKING METHODOLOGY

The proposed real-time fraud detection system begins with data collection and preprocessing of bank transaction datasets. The dataset contains both numeric and non-numeric attributes such as transaction amount, location, transaction type, and user details. Since machine learning algorithms require numeric input, preprocessing techniques are applied to

convert categorical data into numeric format using encoding methods. Additionally, normalization is performed to scale the data into a uniform range, improving model performance. The dataset is then divided into training and testing sets, typically in a 70:30 ratio, to evaluate the model effectively. Feature extraction is performed to identify important attributes that influence fraud detection. Proper preprocessing ensures that the data is clean, structured, and suitable for training the machine learning model.

In the next phase, a Random Forest machine learning model is implemented for fraud detection. The model is trained using the preprocessed dataset, where it learns patterns associated with normal and fraudulent transactions. Random Forest is chosen due to its ability to handle large datasets, reduce overfitting, and provide high accuracy. After training, the model is tested using unseen data to evaluate its performance using metrics such as accuracy, precision, recall, and F1-score. Once the model is trained, it is integrated with Apache Kafka for real-time processing. Kafka is configured with a producer that streams transaction data to a specific topic and a consumer that continuously listens for incoming data.

Finally, the real-time fraud detection process is executed using the Kafka streaming architecture. The producer publishes transaction records to Kafka topics, and the consumer retrieves these records in real time. Each transaction is passed to the trained Random Forest model for prediction. The model classifies the transaction as either "Normal" or "Fraud" instantly. The system is integrated with a web interface using Flask, allowing users to simulate transaction streaming and view prediction results. Visualization tools may also be used to display transaction trends and fraud patterns. This methodology ensures a scalable, efficient, and real-time solution for detecting fraudulent transactions in banking systems.

IV RESULTS EXPLANATIONS

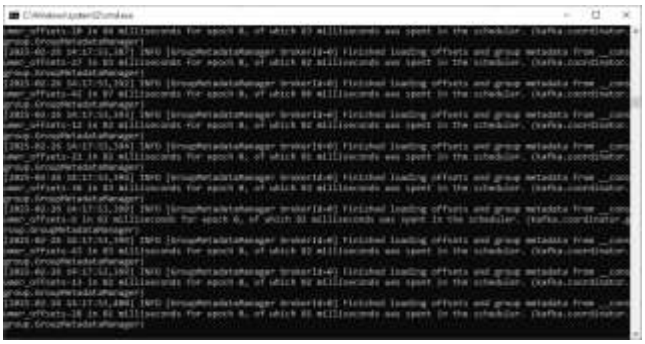
In propose work we are utilizing Apache Kafka streaming technology to stream bank data and then ML algorithm will receive bank stream data and then predict weather bank data is normal or contains fraud transaction.

Kafka streaming consists of Producer and Consumer where producer will produce data on particular topic and then consumer will continuously tries to look for incoming stream data and then consume data and then pass to ML algorithm for prediction.

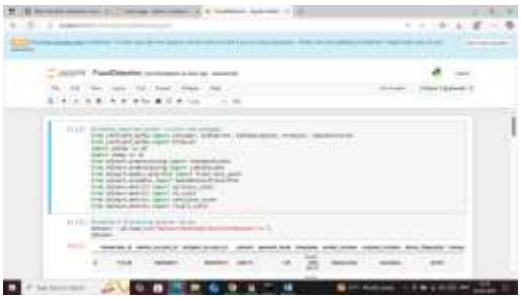
To setup Kafka you need to install java 1.8 and then set java home and path in environment variables. Now copy and paste 'kafka' folder in your system E directory and then double click on 'runZookeper.bat' file from 'kafka' folder to get below page



In above screen Zookeeper server started and now double click on 'runKafka.bat' file to start Kafka server and then will get below page



In above screen Kafka server started and now double click on 'runJupyter.bat' file to start JUPYTER notebook for ML training and then will get below page



In above screen loading required package and classes



In above screen loading and displaying your transaction dataset and this dataset contains both numeric and non-numeric data but ML algorithm accept only numeric data so by employing dataset processing technique can convert non-numeric data to numeric data and then will get below page



In above screen user can click on 'Generate Kafka Producer Stream' link to publish data to Kafka and then will get below page



In above screen can see Kafka publish total 5 streams as transactions records and now click on 'Consume Data & Fraud Detection' link to consume publish data and then input to ML algorithm to predict transaction type



In above screen Kafka consumer receive all records and just scroll to right to view ML predicted output



In above screen in last column ML predicted some records as 'Normal' and some as 'Fraud'

V. CONCLUSION

This project demonstrates the design and implementation of a real-time bank transaction fraud detection system that effectively combines the power of Apache Kafka's streaming platform with advanced machine learning techniques. By leveraging Kafka's high-throughput and low-latency data processing capabilities, the system is able to ingest and analyze massive streams of transaction data continuously, enabling immediate identification of potentially fraudulent activities.

The integration of machine learning models allows for dynamic and accurate classification of transactions based on evolving behavioral patterns and transaction characteristics. This adaptability significantly improves detection accuracy while minimizing false positives, which is critical for maintaining customer trust and operational efficiency. The system's scalable and fault-tolerant architecture ensures robust performance even under heavy transaction loads, making it suitable for real-world banking environments. Additionally, the implementation of a real-time alerting mechanism and monitoring dashboard provides banking authorities with timely insights and actionable information, enhancing their ability to respond quickly to threats. Security considerations such as data encryption and access control further strengthen the system's reliability and compliance with regulatory standards. Overall, this project highlights the importance of integrating real-time streaming technologies with intelligent analytics to address the increasing challenges of financial fraud. The proposed solution not only enhances the security infrastructure of banking institutions but also sets a foundation for future advancements in proactive fraud prevention.

REFERENCES

1. Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., & Bontempi, G. (2015). Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy. *IEEE Transactions*

on Neural Networks and Learning Systems, 29(8), 3784-3797.

2. Ahmed, M., Mahmood, A. N., & Hu, J. (2018). A Survey of Network Anomaly Detection Techniques. *Journal of Network and Computer Applications*, 60, 19-31.

3. Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). Data Mining for Credit Card Fraud: A Comparative Study. *Decision Support Systems*, 50(3), 602-613.

4. Kumar, R., & Singh, A. (2020). Real-Time Fraud Detection System Using Apache Kafka and Machine Learning. *International Journal of Computer Applications*, 176(11), 1-7.

5. Zhang, Y., Jiang, X., & Yang, F. (2019). Adaptive Fraud Detection Using Deep Learning with Streaming Data. *Proceedings of the IEEE International Conference on Big Data*, 2019, 1577-1585.