ENHANCING HEALTHCARE INTEGRITY THROUGH ADVANCED BIG DATA ANALYTICS FOR MEDICARE FRAUD DETECTION

Leon Correia
017410965

Arjun Rai
016980899

Esha Aggarwal
017435756

Akshay Sodha
017427475

***Abstract*—Medicare fraud is a persistent challenge in the U.S. healthcare system, resulting in significant financial losses and eroding public trust. To address this issue, we propose a robust big data pipeline for real-time Medicare fraud detection. Our pipeline leverages advanced technologies, including Apache Kafka, Apache Spark, and Docker, to ingest and process diverse healthcare data sources. Using sophisticated algorithms and machine learning models, the pipeline identifies fraudulent patterns and predicts potential instances of fraud. Through rigorous evaluation, we demonstrate the pipeline's performance in terms of accuracy, scalability, and real-time processing capabilities. Our approach offers a proactive and scalable solution to combat Medicare fraud, contributing to the integrity and sustainability of the healthcare system.**

I. Introduction

The prevalence of Medicare fraud presents a significant challenge to the U.S. healthcare system, with billions of dollars lost annually to fraudulent activities. Beyond the financial implications, Medicare fraud undermines the trust of citizens in the healthcare system and compromises the quality of care received by beneficiaries. Addressing this issue is not only crucial for preserving the integrity of healthcare institutions but also for ensuring the efficient allocation of resources and the delivery of high-quality care to those who need it most. By leveraging advanced technologies like big data analytics and sophisticated data pipelines, we can enhance our ability to detect and prevent Medicare fraud, thereby safeguarding taxpayer dollars, restoring public trust, and improving healthcare outcomes for millions of Americans.

The objective is to build a robust big data pipeline for real-time Medicare fraud detection, leveraging advanced technologies to:

1) Ingest diverse data sources, including Medicare claims and provider data.
2) Process and transform data to uncover fraudulent patterns.
3) Deploy machine learning models for anomaly detection and fraud prediction.
4) Evaluate performance in terms of accuracy, scalability, and real-time processing, iterating for continual improvement

II. Literature Review

width=0.52024-04-28 011206.png In the study, [1] by Bineet Kumar Jha, Sivasankari G, and Venugopal K R from the Department of Information Science and Engineering, CMR Institute of Technology, Bangalore, India, the authors explore the application of big data analytics in fraud detection and prevention within the retail sector. Their research emphasizes the challenges posed by fraudulent activities such as shoplifting, skimming, and counterfeiting, highlighting the need for advanced analytics techniques to effectively identify and mitigate fraud. The study underscores the significance of big data analytics in analyzing vast transaction data to detect patterns indicative of fraudulent behavior, offering innovative approaches to enhance security measures and reduce losses due to fraud.

In the study, [2] conducted by Sara Makki, Rafiqul Haque, Yehia Taher, Zainab Assaghir, Gregory Ditzler, Mohand-Saïd Hacid, and Hassan Zeineddine from Laboratoire LIRIS, Universite de Lyon, Villeurbanne, France, the authors provide a comprehensive review of fraud analysis approaches in the age of big data. The paper covers various techniques from data mining, machine learning, and statistics that have been proposed to address the challenges posed by fraudulent activities in diverse sectors such as banking, insurance, and public services. The review traces the evolution of fraud detection techniques from traditional methods to advanced big data analytics, highlighting the importance of handling large and heterogeneous datasets to identify complex fraud patterns. The study discusses the integration challenges and advanced analytics techniques utilized for fraud detection, offering valuable insights into the application of big data analytics in combating fraud across various industries.

In the study, [3] by Anusmita Poddar, Prasanna Kulkarni, and N.A. Natraj from Symbiosis Institute of Digital and Telecom Management, Pune, India, the authors explore the application of big data analytics in decision management within the banking and financial sector. Their research delves into various aspects such as customer segmentation, spending patterns, product cross-selling, risk management, sentiment analysis, feedback analysis, and fraud management. The study highlights how big data analytics enables banks to extract insights from vast datasets, thereby enhancing decision-making processes and improving overall operational efficiency. By leveraging big data analytics, banks can efficiently redesign and redefine their functioning across various sectors, from product cross-selling to regulatory compliance management. The study emphasizes the role of big data analytics as a

transformative tool in the finance and banking sector, capable of creating, managing, and analyzing large volumes of data characterized by volume, velocity, and variety.

III. Methodology

The methodology for the project encompasses a comprehensive set of procedures designed to handle and analyze large datasets efficiently. The approach outlines our efforts to create a streaming data pipeline that can automate data analytics and machine learning for medicare fraud detection.

1) **Data Collection** : The **CMS Part D Prescriber** Dataset contains prescription drug data for Medicare beneficiaries under the Part D Prescription Drug Program. It includes prescriber identifiers (NPI), drug details, usage statistics, and payment information, with over 25 million rows and 21 columns. The **LEIE Database** lists individuals and entities barred from federally funded healthcare programs due to fraud or misconduct. It includes exclusion type, effective dates, and identifying information, facilitating cross-referencing for fraud detection. **The Payments Received by Physicians Dataset** tracks payments from pharmaceutical companies to physicians, detailing total payments, transaction specifics, and physician demographics, with over 11 million rows and 75 columns. These datasets provide crucial insights for detecting anomalies and potential fraudulent activities.
2) **Big Data Stream Processing Pipeline with Spark and Kafka**: To facilitate the real-time processing of streaming data within our Medicare Fraud Detection project, we utilize Docker, a platform that supports containerization. This allows us to deploy Apache Spark and Apache Kafka in isolated environments, ensuring consistent, reproducible configurations and scalable deployments. We configure and launch images for Kafka and spark respectively. We set up the required images: kafkainit, kafka, kafdrop, zookeeper and spark. Apache

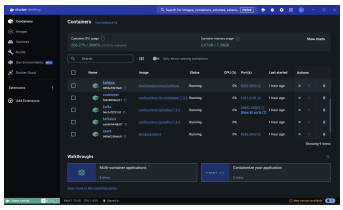

Fig. 1. Setting up Docker Images for Spark and Kafka

Kafka is a distributed streaming platform that enables the building of real-time data pipelines. It's particularly useful in scenarios where high throughput and fault tolerance are required. For our project, we publish the three data streams into three distinct Kafka topics, each representing a specific data source.

Apache Spark is employed for its exceptional ability to process large datasets quickly through its advanced analytics capabilities and in-memory computing power. Spark ingests the streaming data from Kafka

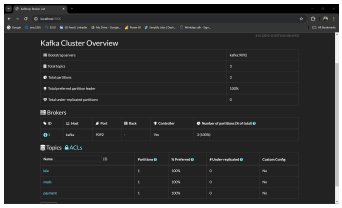

Fig. 2. Kafka Topics Creation

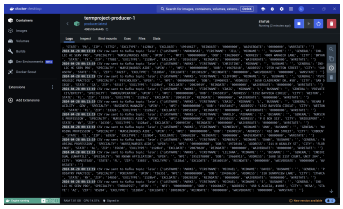

Fig. 3. Kafka producer logs (publishing to the topic)

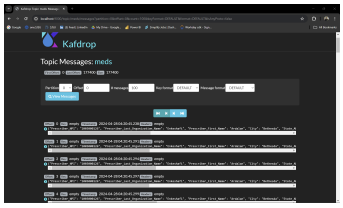

Fig. 4. Verifying the consumer using Kafdrop at localhost:9000

and performs a series of transformations and actions to convert raw data streams into structured dataframes. These dataframes are specifically aligned with the three predefined topics. The spark is subscribed to the kafka topics for processing the data streams

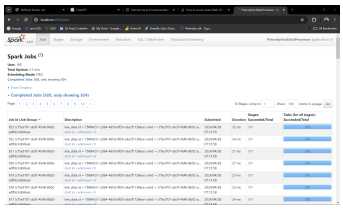

Fig. 5. Data Processing in Spark

3) **Data Cleaning**: This step involves removing duplicates, handling missing values, and correcting inconsistencies across the datasets. Given the large scale of data, automated scripts are developed to ensure efficiency and consistency in the cleaning process. Further through exploratory data analysis (EDA), we gain insights into the distributions and relationships of the data. Tools like Tableau and Power BI are used for visual exploration, helping to identify patterns or anomalies that merit deeper investigation.
4) **Locality-Sensitive Hashing** : LSH is a technique used for approximate nearest neighbor search in high-dimensional spaces. It's beneficial when dealing with

large datasets and high-dimensional feature spaces, where traditional methods like brute-force search become computationally expensive. The 'EXCLTYPE' column in our dataset contains categorical data representing different types of exclusions from Medicare. These exclusions could indicate various reasons, including fraudulent activities, billing errors, or compliance violations. LSH allows us to transform categorical data into numerical representations suitable for similarity analysis. In our context, LSH is applied to the 'EXCLTYPE' column to identify similar types of exclusions based on their categorical values. LSH generates hash codes for each unique 'EXCLTYPE' value. Similar 'EXCLTYPE' values are likely to be hashed to the same or nearby buckets with high probability. Within the LSH framework, we use techniques like MinHashing to create compact representations (signatures) of each 'EXCLTYPE' value. These signatures are then partitioned into hash tables, where each hash table indexes the 'EXCLTYPE' values based on their signatures. During query time, when searching for similar exclusions, LSH allows for efficient retrieval by searching the hash tables for similar MinHash signatures. This enables fast and approximate similarity search, even in datasets with a large number of unique exclusion types. By applying LSH to the 'EXCLTYPE' column, we can efficiently identify clusters or groups of similar exclusion types. This helps in analyzing patterns of exclusion and understanding commonalities among excluded individuals, potentially indicating trends related to fraudulent activities.

5) **Machine Learning**: Training the Logistic Regression Model: We employed a Logistic Regression model, a statistical approach that predicts the probability of a binary outcome based on one or more predictor variables. Logistic Regression is well-suited for binary classification tasks, such as identifying potential Medicare fraud cases (represented as 'Exceptions') in our dataset.
Model Fitting: The LogisticRegression class from the scikit-learn library was utilized to instantiate the model. The training process, called "fitting," involves adjusting the model's parameters to minimize a cost function, which, in this case, is the log-loss or binary cross-entropy. This optimization process is conducted using the fit method with the following inputs: x_train_scaled: The scaled feature set used for training, ensuring that all numerical features contribute equally to the model by bringing them onto the same scale. y_train: The target variable for training, is the encoded label indicating whether each record is an 'Exception' or 'Non-Exception.'
Making Predictions: After the model has been trained, it's applied to the test data to predict the probabilities that each case is an 'Exception'. This is achieved with the predict_proba method, which outputs probabilities for both classes. We are interested in the probability of being an 'Exception' (hence the index [1]), as it serves as our decision metric for potentially fraudulent cases.
6) **Results Explained**: In the context of Medicare fraud detection, the heatmap provides valuable insights into how closely related our numerical features are to the potential for fraud (denoted by EXCLTYPE). Features with a high correlation might be particularly indicative of fraud and warrant closer examination or could be weighted more heavily in our predictive models. The lack of strong correlations among the features themselves suggests that they may provide unique information to the model, which can be beneficial in developing a robust fraud detection algorithm.

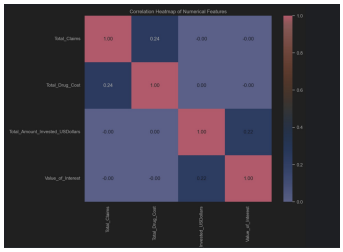

Fig. 6. Correlation Heatmap

The pie chart illustrates the significant class imbalance within our dataset: a vast majority are 'Non-Exceptions', with only a small fraction classified as 'Exceptions'. This reflects the real-world scenario where fraudulent activities are relatively rare compared to legitimate transactions.

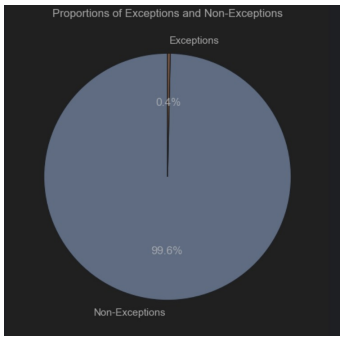

Fig. 7. PieChart of Exceptions

IV. Evaluating Model Performance

The model's performance is evaluated using the Area Under the Receiver Operating Characteristic Curve (AUC-ROC), a crucial metric for binary classification models. The ROC curve is a plot of the true positive rate against the false positive rate at various threshold settings. The AUC summarizes the ROC curve into a single value, measuring the model's ability to discriminate between the two classes.

AUC values vary from zero to one. A model with a score of 0.5 is no better than random guessing, but a score of 1 reflects an ideal model that distinguishes between 'Exceptions' and 'Non-Exceptions'. Higher AUC values indicate improved

model performance, with a greater possibility of properly recognizing both true 'Exceptions' and true 'Non-Exceptions'.

The Logistic Regression model received an AUC score of 0.92, indicating its ability to identify between fraudulent and non-fraudulent Medicare instances. This metric will be used as a benchmark to assess the model's utility in a real-world scenario and to drive future enhancements to the Medicare fraud detection system.

V. Use of Privacy Techniques

While k-anonymity could not be implemented in this project, it remains a valuable technique for enhancing privacy in data analysis. K-anonymity ensures that individuals cannot be identified from released data by ensuring that each data record is indistinguishable from at least k-1 other records. To implement k-anonymity, sensitive attributes such as personal identifiers are generalized or suppressed to achieve anonymity while preserving the utility of the data for analysis purposes. For example, in healthcare datasets, patient identifiers like names and social security numbers can be generalized to age ranges or geographic regions. However, implementing k-anonymity requires careful consideration of the trade-off between privacy and data utility, as excessive generalization can lead to the loss of valuable information for analysis. Additionally, techniques such as differential privacy can be combined with k-anonymity to provide stronger privacy guarantees while allowing for more accurate data analysis. Integrating k-anonymity into future iterations of this project could enhance privacy protection for sensitive healthcare data, ensuring compliance with privacy regulations and mitigating the risk of data re-identification.

VI. Innovation

Innovation in healthcare fraud detection occurs when advanced data analysis methods and vigilant monitoring intersect. By combining powerful technologies like Apache Kafka and Apache Spark with complex machine learning algorithms, we can transform how we identify and stop fraudulent activities in healthcare. Our approach involves bringing together various datasets, including the CMS Part D Prescriber Dataset, the LEIE Database, and Payments Received by Physicians Dataset, to get a full picture of how prescriptions are made, which providers are excluded, and financial interactions. By analyzing data in real-time, we can quickly spot unusual patterns and signs of potential fraud, allowing us to take action promptly. Moreover, by continually refining our detection methods based on ongoing assessment and feedback, we develop proactive strategies to prevent fraud that can adapt to new tactics. This innovative approach not only improves how efficiently and effectively we detect fraud but also helps safeguard the honesty and stability of the healthcare system for everyone involved.

VII. Pair Programming

In our project, we adopted a pair programming approach to enhance collaboration and code quality. Pair programming involves two developers working together on the same code, with one writing the code (the "driver") and the other providing feedback and suggestions (the "navigator"). This collaborative effort allowed us to catch errors early, make better design decisions, and produce more robust code. We leveraged platforms like GitHub for version control and collaboration, enabling seamless tracking, reviewing, and merging of code changes. Additionally, Google Colab provided a collaborative coding environment where team members could work together in real-time, enhancing our productivity and efficiency. Overall, pair programming proved to be a valuable technique in our project, fostering teamwork and accelerating our development process.

VIII. Agile/Scrum methodology

In this project, we adopted an Agile Scrum methodology to ensure effective project management and collaboration. We utilized Trello as our project management tool, facilitating seamless communication and task tracking among team members. Through iterative sprints, we prioritized and tackled project tasks, breaking down our objectives into manageable increments. Daily stand-up meetings provided a forum for team members to discuss progress, share insights, and address any challenges. The Agile approach allowed us to adapt to evolving requirements and feedback, ensuring that we delivered a high-quality big data pipeline for Medicare fraud detection in a timely and efficient manner. The detailed dashboard can be accessed on this link Trello Dashboard by Atlassian

IX. Team Work

In our project, working together as a team played a crucial role in our achievements. By communicating effectively and collaborating closely, we combined our individual strengths and knowledge to reach our goals. We held regular meetings where we shared ideas and made decisions together, creating an environment where everyone's input was valued. Each team member took responsibility for their tasks and also offered support to others when necessary. Using tools like Trello helped us stay organized and focused on our shared objectives. Ultimately, our teamwork and dedication to excellence were key factors in the success of our project.

X. Technical difficulty

This project presents technical challenges in integrating and coordinating various advanced technologies to build a smooth big data pipeline. Configuring Apache Kafka for real-time data ingestion requires precise setup and optimization to ensure reliable data streaming. Similarly, using Apache Spark for distributed data processing demands expertise in parallel computing and optimizing algorithms for handling large datasets efficiently. Containerizing pipeline components with Docker adds complexity, requiring a solid understanding of containerization principles for scalability and consistency across deployment environments. Additionally, integrating machine learning algorithms for fraud detection entails challenges

in model training, evaluation, and deployment, calling for expertise in data science and machine learning engineering. Overall, managing these technical hurdles requires careful planning, execution, and ongoing maintenance to achieve a seamless and effective pipeline.

XI. Roles and Responsiblities

Team members and roles:
1) Leon Correia - Containerization and Kafka
2) Arjun Rai- Data Collection and LSH
3) Esha Aggarwal- Documentation and Spark
4) Akshay Sodha- Machine learning and documentation

XII. Conclusion

In this project, we developed an extensive big data pipeline aimed at detecting Medicare fraud, seamlessly combining Apache Kafka, Apache Spark, and Docker. Our pipeline commenced with Apache Kafka, acting as a robust and distributed data ingestion platform, where we established distinct Kafka topics for each CSV file, ensuring smooth data streaming into our pipeline. Utilizing the distributed computing capabilities of Apache Spark, we orchestrated complex data processing operations, including transformations, aggregations, and feature engineering, on the incoming data streams. We also applied machine learning for fraud detection, training models on historical data and employing methods like logistic regression to forecast fraudulent activities with exceptional precision. Docker played a crucial role by containerizing our Kafka and Spark environments, guaranteeing uniformity and adaptability across various deployment scenarios. With Docker, we achieved reproducibility and isolation, encapsulating Kafka brokers, Spark workers, and other pipeline elements within self-contained containers. The potential for automation and scalability inherent in our pipeline can facilitate enhanced real-time fraud detection, with the flexibility to scale horizontally to accommodate large data volumes and computational requirements. The integration of pre-trained machine learning models into the pipeline represents an exciting opportunity to enhance the effectiveness of fraud detection while reducing the time and resources required for model development and deployment. Overall, our project exemplifies the potential of big data technologies in combating Medicare fraud and contributes to the integrity and resilience of the healthcare landscape.

Future Scope

In our current project, we have developed a robust big data pipeline for Medicare fraud detection, integrating Apache Kafka and Apache Spark to ingest, process, and analyze healthcare data. Looking ahead, there is significant potential for enhancing the pipeline's capabilities by integrating pre-trained machine learning models for fraud detection. By incorporating pre-trained models into the pipeline after the data has been processed by Spark, we can streamline the deployment of sophisticated fraud detection algorithms and improve the accuracy and efficiency of fraud detection. Additionally, future iterations of the pipeline could explore the integration of advanced techniques such as deep learning models or ensemble methods to further enhance fraud detection performance. Furthermore, we can implement a continuous integration and continuous deployment (CI/CD) pipeline to automate the process of deploying updated or newly trained models into production, ensuring that our fraud detection system remains up-to-date and effective in identifying evolving fraud patterns.

XIII. Project Artifact links

Term Project Elevator Pitch Video Github Repository

Acknowledgment

This work was completed by Our project team. The authors would like to thank the data providers for generously sharing their valuable data. The authors are also grateful for the use of various software and tools, including Kafka, Spark, Docker, QuillBot, Grammarly, Trello, and LucidCharts.

References

1) Rizani, M. Nazhif, and Hiroyuki Iida. "Analysis of counter-strike: Global offensive." 2018
International Conference on Electrical Engineering and Computer Science (ICECOS). IEEE, 2018.
2) Berner, Christopher, et al. "Dota 2 with large scale deep reinforcement learning." arXiv preprint arXiv:1912.06680 (2019).
3) Xenopoulos, Peter, Harish Doraiswamy, and Claudio Silva. "Valuing player actions in counter-strike: Global offensive." 2020 IEEE international conference on big data (big data). IEEE, 2020.
4) Centers for Medicare Medicaid Services. (n.d.). Medicare Part D Prescribers. [Data set]. Retrieved from https://data.cms.gov/provider-summary-by-type-of-service/medicare-part-d-prescribers.
5) U.S. Department of Health Human Services, Office of Inspector General. (n.d.). List of Excluded Individuals and Entities (LEIE). Retrieved from https://oig.hhs.gov/exclusions/exclusions_list.asp.
6) U.S. Food Drug Administration. (n.d.). Drugs@FDA: FDA Approved Drug Products. Retrieved from https://www.fda.gov/Drugs/InformationOnDrugs/ucm079750.htmcollapseOne.

Sr. No.	Criteria	How it is met
1.	Presentation Skills - Includes time management	Made a PPT and will be presenting in class
2.	Code Walkthrough	Connected MySQL, SnowFlake with Python
3.	Discussion / QnA	In Class
4.	Demo	In Class
5.	Version Control	GitHub
6.	Use of Git / GitHub or equivalent; must be publicly accessible	Used Git to push the project into GitHub repository
7.	Lessons learned Included in the report and presentation? How substantial and unique are they?	Yes
8.	Innovation	Unique way of representing Information through Discord.
9.	Teamwork	Collaborative work using google meet
10.	Technical difficulty	Included in Report
11.	Practiced pair programming?	yes
12.	Practiced agile / scrum (1-week sprints)?	Yes, Spreadsheets
13.	Used Grammarly / other tools for language?	Grammarly and Quillbot
14.	Slides	Yes,PPT Submitted
15.	Report Format, completeness, language, plagiarism, whether turnItIn could process it (no unnecessary screenshots), etc	Submitted
16.	Used unique tools	MongoDB, Discord, Airflow
17.	Performed substantial analysis using database techniques	Yes Using Pandas,Matplotlib and Seaborn
18.	Used a new database or data warehouse tool not covered in the HW or class	no
19.	Used appropriate data modeling techniques	Yes
20.	Used ETL tool	MongoDB and Python
21.	Demonstrated how Analytics support business decisions	Explained in Report
22.	Used RDBMS	No
23.	Used Datawarehouse	MongoDB
24.	Includes DB Connectivity / API calls	PyMongo
25.	Used NOSQL	MongoDB