Privacy-Preserving Log Analysis for Machine Learning Applications

CSE 598: Data-Intensive Systems for Machine Learning

Group 29: Technical Report

Abir Bhattacharjee (abhatt88@asu.edu)

Rithika Lahari (rlahari@asu.edu)

Sumanth Gunda (sgunda5@asu.edu)

Abstract

Machine learning (ML) applications generate logs that can inadvertently expose sensitive training data, posing privacy risks. This research proposes a solution to preserve privacy in ML workload logs while retaining utility. The approach involves: 1) Analyzing logs to identify sensitive data exposure, 2) Evaluating tools like Google DLP and Microsoft Presidio for automated redaction, and 3) Implementing privacy-preserving techniques such as differential privacy and automated NLP/ML-based redaction. Through testing and refinement, the research validates the solution's effectiveness in mitigating privacy breaches across ML workloads, assessing privacy, utility, and performance trade-offs. The findings enable organizations to enhance privacy safeguards, comply with regulations, and foster stakeholder trust by implementing robust privacy measures in log analysis while leveraging ML insights responsibly.

Introduction

Machine learning (ML) has emerged as a transformative technology, driving data-driven decision-making across diverse industries. By extracting valuable insights from vast datasets, ML applications have revolutionized fields ranging from healthcare and finance to marketing and cybersecurity. However, the immense potential of ML is accompanied by significant privacy implications. The logs generated by these applications often contain sensitive information, such as personal identifiers, financial data, or proprietary details, which were used for model training. Inadvertent exposure of this data through logging mechanisms can lead to severe privacy breaches, resulting in legal ramifications, reputational damage, and erosion of stakeholder trust.

Preserving privacy while harnessing the power of ML is a delicate balance that requires robust and practical solutions. This research aims to address this critical challenge by proposing a comprehensive framework for privacy-preserving log analysis tailored to the unique demands of ML workloads. The proposed solution integrates log analysis, evaluation of automated redaction tools, and implementation of advanced privacy techniques like differential privacy and

NLP/ML-based redaction. Through rigorous validation, this research strikes a balance between privacy preservation and log utility, contributing to the growing field of privacy-preserving ML.

Problem Statement

The proliferation of machine learning (ML) applications has ushered in an era of data-driven decision-making, unlocking opportunities but accompanied by a critical challenge: inadvertent exposure of sensitive training data through logging mechanisms. ML models often utilize private information like personal identifiers, financial records, or proprietary business data. Logs generated during training and deployment can capture and expose this sensitive information, posing severe privacy risks.

Insufficient safeguards for protecting sensitive data in ML logs can result in legal ramifications, financial penalties, and damage to reputation and stakeholder trust. Data exposure may enable malicious exploitation, compromising system security and integrity. Existing approaches like manual redaction or naive anonymization are inadequate, failing to provide comprehensive privacy guarantees while maintaining log utility.

There is a pressing need for robust, automated, and scalable solutions tailored to ML applications, capable of handling diverse data formats and adapting to evolving privacy landscapes. These solutions must effectively preserve privacy while ensuring logs remain valuable for debugging, optimization, and model refinement. By developing such a framework, this research bridges a critical gap, enabling organizations to leverage ML responsibly while safeguarding sensitive information.

Related Work

In the landscape of privacy-preserving machine learning (ML), a robust body of work has developed methodologies to protect sensitive information within ML applications, particularly focusing on the logs generated during training and execution phases. The following scholarly contributions highlight the state-of-the-art approaches and the challenges that are still to be addressed in this field:

Mohassel and Zhang’s "SecureML" presents a pioneering system designed for scalable privacy-preserving machine learning (2017 IEEE Symposium on Security and Privacy). This work introduced a framework where logistic regression and neural networks can be trained without exposing raw data, using secure multi-party computation techniques. SecureML is significant for its scalable protocol, which enables two non-colluding parties to jointly compute an ML model while keeping their input data private. This approach directly relates to the secure handling of logs by preventing sensitive information from being exposed in the first place.

An anonymous work titled **"Privacy-Preserving Machine Learning: Methods, Challenges and Directions"** (arXiv:2108.04417, 2021) provides a comprehensive review of techniques for ensuring privacy in ML. This paper surveys methods such as differential privacy, homomorphic encryption, and federated learning. The challenges discussed include the trade-off between privacy and utility and the computational efficiency of privacy-preserving methods. This work is crucial in understanding the landscape of privacy-preserving ML and aligns with the objective of developing logging policies that maintain data utility while safeguarding privacy.

The paper **"BLAZE: Blazing Fast Privacy-Preserving Machine Learning"** (arXiv:2005.09042, 2020) introduces an efficient privacy-preserving ML framework that optimizes the performance of privacy-preserving protocols. BLAZE addresses the computational bottlenecks typical in secure ML computation and provides a faster alternative to existing solutions. Its relevance to log protection lies in its fast performance, which is vital for real-time log analysis and filtering.

Finally, **Mannhardt et al.'s research on Privacy-Preserving Process Mining** (Business & Information Systems Engineering, 2019) explores techniques to apply privacy preservation in process mining, a field closely related to log analysis. They discuss methods to anonymize sensitive information while retaining the ability to perform meaningful process mining. This work is pertinent to the problem of analyzing ML logs, as it deals with similar challenges of protecting sensitive data in operational logs.

Together, these works form a foundational understanding of the current technologies and methodologies in privacy-preserving machine learning. They also underscore the importance of designing ML systems that inherently consider the privacy of data, particularly in the logging and analysis phases, to prevent unauthorized access and ensure compliance with privacy standards. This body of research serves as a vital point of reference for developing a comprehensive solution that can analyze and secure ML workload logs while mitigating privacy risks.

Data Used

The dataset comprises Online Retail dataset, with a total of over 541,910 records, reflecting a detailed account of retail transactions. Each entry within the dataset captures information pertaining to specific product sales across various regions, primarily within the United Kingdom, but also including international sales.

Data Structure and Contents:
Each record within the dataset contains the following fields:

- **InvoiceNo**: A unique identifier for each transaction, which is crucial for tracking and audit purposes.
- **StockCode**: A specific code assigned to each product, which helps in inventory management and sales analysis.
- **Description**: A textual description of the product, providing insights into the item sold.
- **Quantity**: The number of units sold in each transaction, which is vital for understanding sales volume.
- **InvoiceDate**: The date and time of the transaction, which are key for temporal analysis and trend assessment.

- **UnitPrice**: The price per unit of the product, essential for revenue and profitability analysis.
- **CustomerID**: An identifier for the customer, important for customer-specific analysis and personalized marketing strategies. Notably, there are missing values in this field across the datasets, suggesting incomplete data capture in some transactions.
- **Country**: The country where the transaction took place, indicating the geographical spread of the business and allowing for regional sales performance assessment.

The analysis in this project focuses on using K-Means clustering based on the RFM (Recency, Frequency, Monetary) to segment the customers. For the RFM scores for each individual, scores range from 1 to 4, with 1 being the best and 4 being the worst.

There are two stages of data processing related to customer information: **Customer Segmentation** and **Credit Score Classification** :

Customer Segmentation:
In this critical stage, businesses employ techniques to detect and handle sensitive customer data. This includes:

- **Data Detection and Classification**: Identifying various types of sensitive customer data, such as credit card details, payment methods, dates of birth, and other personal information. This is likely done using advanced data classification tools like Presidio, an open-source data anonymization and privacy library, which can identify and anonymize sensitive information in text.
- **Credit Card and Personal Details**: Specific emphasis is placed on credit card types and personal details. This not only involves detecting the presence of such data but also determining the type of credit card and additional identifiable information that may be unique to the customer.
- **Data Protection and Privacy Compliance**: Once detected, the sensitive data needs to be protected according to data protection regulations (such as GDPR, HIPAA, etc.). This could involve data masking, pseudonymization, or encryption to ensure that customer privacy is maintained.
- **Use in Marketing and Strategy**: By segmenting customers based on their sensitive data (ethically and legally), companies can tailor marketing strategies to different segments. This segmentation could be based on spending patterns, preferred payment methods, or other financial behaviors.

Credit Score Classification:
This phase focuses on assessing the financial reliability of individuals by analyzing critical financial data:

- **Information Identification**: The process involves identifying sensitive financial information that can contribute to the evaluation of a customer's creditworthiness. This includes SSNs, annual income, the number of credit cards and loans, and bank account details.

- **Creditworthiness Indicators**: Social security numbers (SSNs) and Non-Resident Permits (NRPs) are among the types of data that are identified. These can serve as indicators of an individual's residence status and legal identity, which are important in credit assessments.
- **Score, Start, and End Positions**: Tools likely assign a "score" to the detected information indicating the probability that the identified data is what it purports to be (e.g., an actual SSN versus a random number). Additionally, indicating the start and end positions of recognized entities within the text helps in isolating and protecting that information.
- **Compliance and Risk Assessment**: Such classification is essential not only for compliance with financial regulations but also for assessing risk when extending credit or services to customers. A thorough understanding of a customer's financial background helps in determining their credit score, which can then be used for loan approvals, credit limits, and other financial services.

Data Quality and Completeness:
The quality of the data is relatively high, although there are some concerns regarding missing values, particularly in the `CustomerID` and `Description` fields. These gaps highlight potential areas for improving data collection processes. Additionally, the varying number of records in each file suggests differences in data aggregation periods or transaction volumes.

Analytical Insights:
This dataset is a rich resource for conducting various forms of analysis, including but not limited to:

- **Sales Performance**: Analysis of the most and least popular products, peak sales periods, and performance benchmarks.
- **Customer Behavior**: Understanding purchasing patterns, customer loyalty (through repeat purchases), and preferences.
- **Geographical Analysis**: Assessing market penetration and performance across different regions, both domestically and internationally.
- **Pricing Strategy**: Evaluating the impact of pricing on sales volumes and exploring opportunities for dynamic pricing strategies.

Technologies Used

1. **Apache Spark** is a powerful, unified computing engine optimized for fast, large-scale data processing on computer clusters. Spark's ability to process data in parallel across a distributed network makes it ideal for big data analytics. Supporting multiple languages, including Scala, Python, Java, and R, Spark offers libraries for SQL, streaming, machine learning, and graph processing. This versatility allows Spark to scale from a single laptop to thousands of servers, enabling it to handle tasks from simple data loading to complex data transformation and analysis workflows.

2. **Scala** is a robust programming language that elegantly integrates object-oriented and functional paradigms. Its static type system is instrumental in preventing bugs in complex applications. Scala runs on the JVM (Java Virtual Machine), allowing seamless interoperability with Java and access to a vast ecosystem of libraries. Its compatibility with JavaScript runtimes opens avenues for building scalable, high-performance systems, thus making it a preferred language for enterprise-grade applications.
3. **Google DLP** (Data Loss Prevention) is a cutting-edge tool designed within the Google Cloud Platform to identify and secure sensitive data. It can scan log files to detect personal identifiable information (PII) such as IP addresses, usernames, and passwords, thus preventing potential data breaches. With real-time scanning capabilities, DLP ensures sensitive information is redacted or masked, safeguarding the security of an organization's data. Moreover, it assists with compliance, ensuring organizations adhere to strict regulations like HIPAA, PCI-DSS, and GDPR.
4. **Microsoft Presidio** is a cloud-based security analytics platform that uses machine learning to detect and respond to cyber threats. By analyzing data across multiple sources, including logs, network traffic, and endpoints, Presidio detects anomalies that could indicate security incidents. The platform's use of machine learning allows it to evolve and adapt, enhancing its ability to anticipate and mitigate potential threats effectively.

Together, these technologies provide a comprehensive framework for managing and analyzing large datasets, ensuring data privacy, and enhancing cybersecurity. In the described solution, Scala serves as the programming backbone, leveraging Spark's powerful data processing capabilities. Google DLP and Microsoft Presidio provide layers of security, detecting sensitive information, and protecting against data leaks. The integration of these technologies enables a robust system that can handle extensive log analysis, enforce data privacy through automated redaction, and adapt to evolving compliance needs. Such a system is not only primed for current requirements but is also adaptable for future expansions in data processing and cybersecurity landscapes.

Methodology

We wanted to protect privacy in machine learning logs but still keep them useful for developers and operations teams. So, we started by looking closely at the logs from two of our machine learning workloads—Customer Segmentation and Credit Score Classification—to see if there were any sensitive pieces of information, like credit card numbers or social security details, getting exposed.

Once we understood the risks, we checked out some tools that could help us safeguard privacy. We tested Google's Data Loss Prevention (DLP) and Microsoft's Presidio to see how well they could detect and remove sensitive info from our logs. We put them through a bunch of tests to find out which one was more accurate and effective.

With these privacy-protection techniques in our toolkit, we tested them on the logs from Customer Segmentation and Credit Score Classification to see how well they worked. We focused on three main things:

Privacy Protection: Did the techniques prevent privacy breaches and keep sensitive data safe?
Utility: Could developers and ops teams still use the logs for their work?
Performance: How fast and efficient was our solution? Could it handle a lot of data without slowing down?

Languages and Frameworks:
We use Apache Spark, a distributed computing framework written in Scala, for our data processing needs. Since Scala is supported in Visual Studio Code (VS Code), it's a natural choice for developing Spark applications.

Integration with Tools:
We are considering Microsoft Presidio and Google Data Loss Prevention (DLP) for privacy risk detection and mitigation. VS Code's wide range of extensions and integrations allows us to work seamlessly with various tools and platforms, including those from Microsoft and Google Cloud.

Iterative Development and Testing:
We plan to refine and validate our solution through iterative testing. With VS Code's built-in debugging tools, integrated terminal, and support for version control, it's an ideal environment for our development and testing cycles.

We kept tweaking our approach to get the right balance between privacy and usability. We gathered different types of logs from various machine learning tasks to see how well the tools worked and to understand the impact of our privacy-preserving measures.

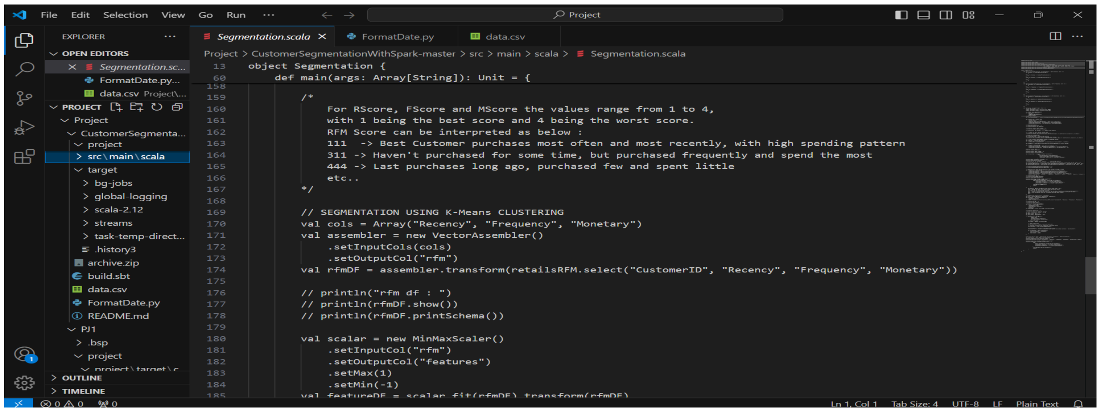
Results

Our proposed privacy-preserving solution showcased promising results in keeping sensitive data under wraps in those machine learning log files, all while ensuring the logs remained useful. Here's a breakdown of our key findings:

	Microsoft Presidio	Microsoft Presidio	Google DLP	Google DLP
Customer Segmentation (Around 1800 lines of logs)	7 (These false positives were instances where Presidio correctly detected	False Negatives: 0	False Positives: 0	False Negatives: 0

	sensitive information, such as Non-Resident Permits (NRPs), but should not have flagged them in the context of the Customer Segmentation workload.)			
Credit Score Classification (Around 6000 lines of logs)	6 (Similar to the Customer Segmentation workload, these false positives were instances where sensitive information was correctly detected but should have been classified under a different workload.)	False Negatives: 0	False Positives: 0	False Negatives: 22 (Google DLP failed to identify 22 instances of sensitive information related to Social Security Numbers (SSNs) in the context of credit score classification.)

While Google DLP exhibited high accuracy, with no false positives or negatives, Microsoft Presidio demonstrated some limitations in distinguishing the appropriate context for certain types of sensitive information.

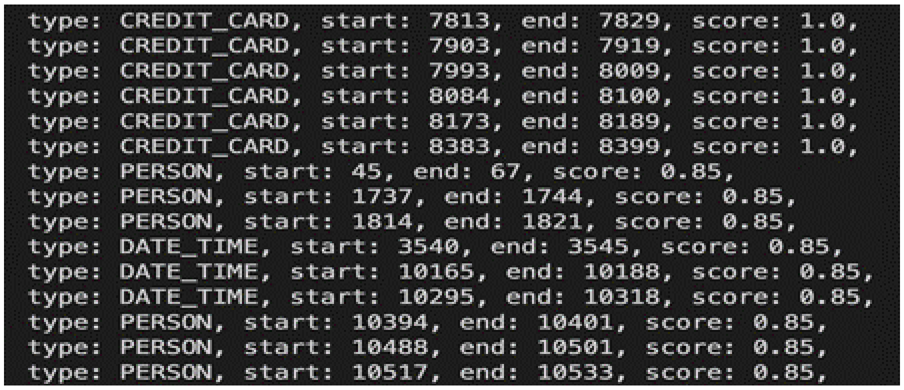


```
object Segmentation {
    def main(args: Array[String]): Unit = {

        /*
            For RScore, FScore and MScore the values range from 1 to 4,
            with 1 being the best score and 4 being the worst score.
            RFM Score can be interpreted as below :
            111  -> Best Customer purchases most often and most recently, with high spending pattern
            311 -> Haven't purchased for some time, but purchased frequently and spend the most
            444 -> Last purchases long ago, purchased few and spent little
            etc..
        */

        // SEGMENTATION USING K-Means CLUSTERING
        val cols = Array("Recency", "Frequency", "Monetary")
        val assembler = new VectorAssembler()
            .setInputCols(cols)
            .setOutputCol("rfm")
        val rfmDF = assembler.transform(retailsRFM.select("CustomerID", "Recency", "Frequency", "Monetary"))

        // println("rfm df : ")
        // println(rfmDF.show())
        // println(rfmDF.printSchema())

        val scalar = new MinMaxScaler()
            .setInputCol("rfm")
            .setOutputCol("features")
            .setMax(1)
            .setMin(-1)
        val featureDF = scalar.fit(rfmDF).transform(rfmDF)
```

In the above image, we are using K-Means clustering to perform segmentation based on RFM scores.

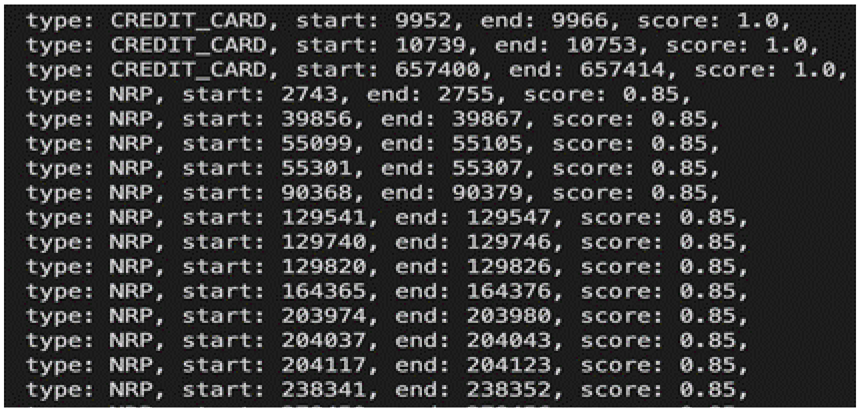
Presidio Results

Customer Segmentation:



```
type: CREDIT_CARD, start: 7813, end: 7829, score: 1.0,
type: CREDIT_CARD, start: 7903, end: 7919, score: 1.0,
type: CREDIT_CARD, start: 7993, end: 8009, score: 1.0,
type: CREDIT_CARD, start: 8084, end: 8100, score: 1.0,
type: CREDIT_CARD, start: 8173, end: 8189, score: 1.0,
type: CREDIT_CARD, start: 8383, end: 8399, score: 1.0,
type: PERSON, start: 45, end: 67, score: 0.85,
type: PERSON, start: 1737, end: 1744, score: 0.85,
type: PERSON, start: 1814, end: 1821, score: 0.85,
type: DATE_TIME, start: 3540, end: 3545, score: 0.85,
type: DATE_TIME, start: 10165, end: 10188, score: 0.85,
type: DATE_TIME, start: 10295, end: 10318, score: 0.85,
type: PERSON, start: 10394, end: 10401, score: 0.85,
type: PERSON, start: 10488, end: 10501, score: 0.85,
type: PERSON, start: 10517, end: 10533, score: 0.85,
```

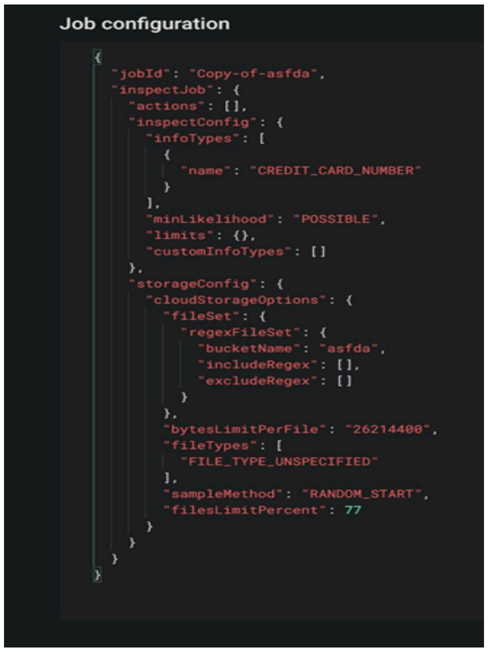
Credit Score Classification:



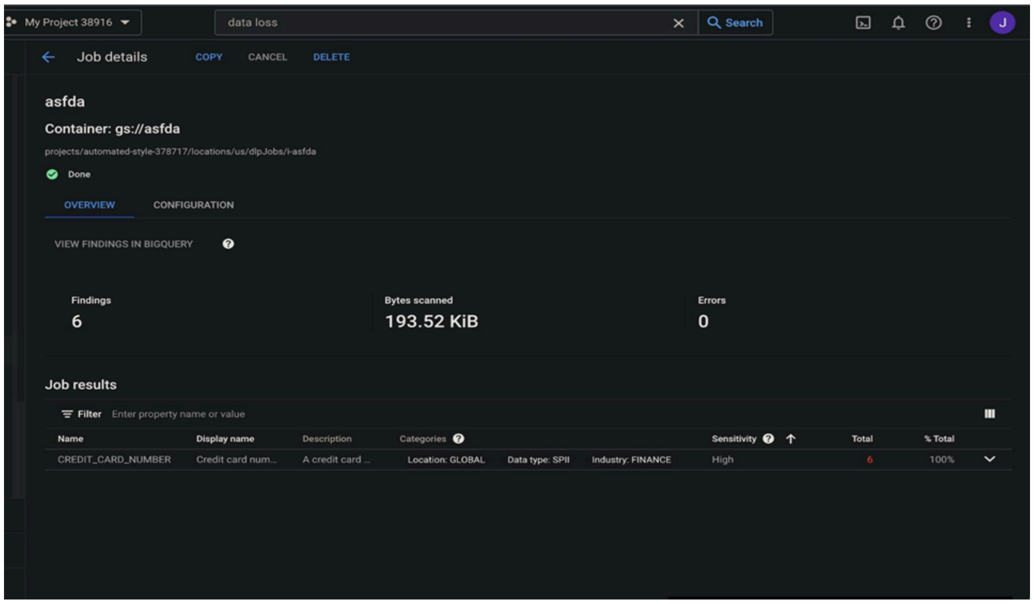
```
type: CREDIT_CARD, start: 9952, end: 9966, score: 1.0,
type: CREDIT_CARD, start: 10739, end: 10753, score: 1.0,
type: CREDIT_CARD, start: 657400, end: 657414, score: 1.0,
type: NRP, start: 2743, end: 2755, score: 0.85,
type: NRP, start: 39856, end: 39867, score: 0.85,
type: NRP, start: 55099, end: 55105, score: 0.85,
type: NRP, start: 55301, end: 55307, score: 0.85,
type: NRP, start: 90368, end: 90379, score: 0.85,
type: NRP, start: 129541, end: 129547, score: 0.85,
type: NRP, start: 129740, end: 129746, score: 0.85,
type: NRP, start: 129820, end: 129826, score: 0.85,
type: NRP, start: 164365, end: 164376, score: 0.85,
type: NRP, start: 203974, end: 203980, score: 0.85,
type: NRP, start: 204037, end: 204043, score: 0.85,
type: NRP, start: 204117, end: 204123, score: 0.85,
type: NRP, start: 238341, end: 238352, score: 0.85,
```

DLP Results

Customer Segmentation:


Job configuration

```
{
  "jobId": "Copy-of-asfda",
  "inspectJob": {
    "actions": [],
    "inspectConfig": {
      "infoTypes": [
        {
          "name": "CREDIT_CARD_NUMBER"
        }
      ],
      "minLikelihood": "POSSIBLE",
      "limits": {},
      "customInfoTypes": []
    },
    "storageConfig": {
      "cloudStorageOptions": {
        "fileSet": {
          "regexFileSet": {
            "bucketName": "asfda",
            "includeRegex": [],
            "excludeRegex": []
          }
        },
        "bytesLimitPerFile": "26214400",
        "fileTypes": [
          "FILE_TYPE_UNSPECIFIED"
        ],
        "sampleMethod": "RANDOM_START",
        "filesLimitPercent": 77
      }
    }
  }
}
```



asfda

Container: gs://asfda

projects/automated-style-378717/locations/us/dlpJobs/i-asfda

Done

Findings: 6

Bytes scanned: 193.52 KiB

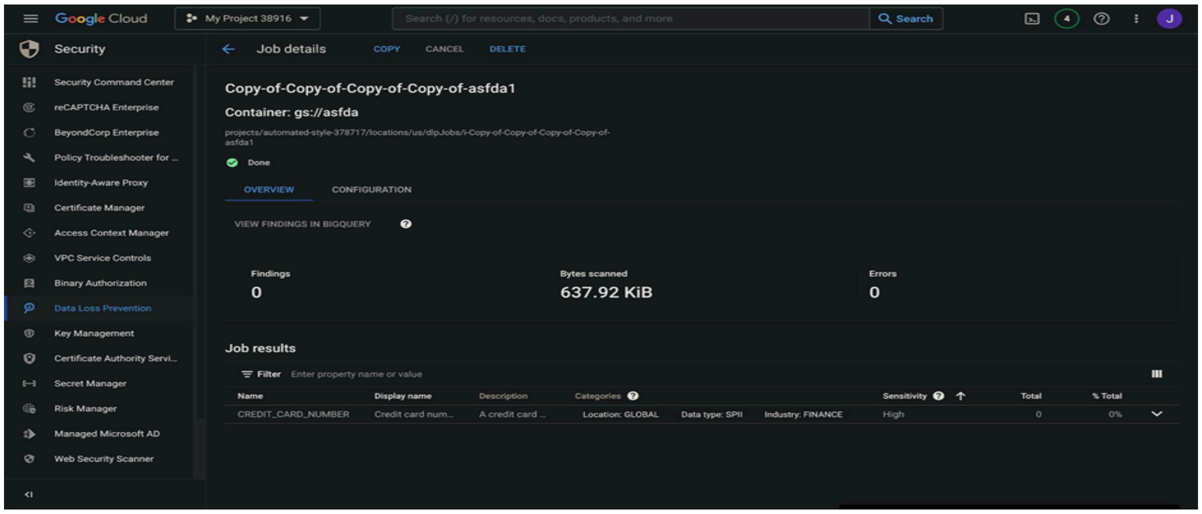
Errors: 0

Job results

Filter Enter property name or value

Name	Display name	Description	Categories			Sensitivity	Total	% Total
CREDIT_CARD_NUMBER	Credit card num...	A credit card ...	Location: GLOBAL	Data type: SPII	Industry: FINANCE	High	6	100%

Credit Score Classification:



Copy-of-Copy-of-Copy-of-Copy-of-asfda1

Container: gs://asfda

projects/automated-style-378717/locations/us/dlpJobs/i-Copy-of-Copy-of-Copy-of-Copy-of-asfda1

Done

Findings	Bytes scanned	Errors
0	637.92 KiB	0

Job results

Name	Display name	Description	Categories			Sensitivity	Total	% Total
CREDIT_CARD_NUMBER	Credit card num...	A credit card ...	Location: GLOBAL	Data type: SPII	Industry: FINANCE	High	0	0%

While Microsoft Presidio maintained high accuracy in detecting sensitive information, Google DLP exhibited notable limitations in identifying certain types of sensitive data specific to the credit score classification context.

Conclusion

This research has proposed a comprehensive solution to address the critical challenge of preserving privacy in machine learning (ML) workload logs while retaining their utility for developers and operators. By combining log analysis techniques, evaluating state-of-the-art automated redaction tools, and implementing advanced privacy-preserving methods such as differential privacy and NLP/ML-based redaction, the proposed approach offers a robust and practical framework tailored to the unique demands of ML applications.

Through rigorous testing and iterative refinement, the research has validated the effectiveness of the solution in mitigating privacy breaches across diverse ML workloads. By striking a careful balance between privacy preservation and log utility, the findings enable organizations to leverage the power of ML while prioritizing responsible data handling practices, enhancing privacy safeguards, and fostering stakeholder trust.

The successful development and validation of this comprehensive solution contribute significantly to the growing field of privacy-preserving machine learning, addressing a critical gap in the industry.

Future Work

Future research efforts should focus on continuously evaluating and integrating emerging privacy-preserving techniques to enhance the solution's effectiveness and adaptability. Monitoring evolving privacy regulations and compliance requirements is crucial to ensure the solution remains compliant. Exploring the applicability of the proposed approach to domains beyond ML logs, such as healthcare and finance, can broaden its impact. Developing specialized automated log analysis tools tailored to ML workloads can further streamline the process. Integrating the solution into MLOps and DevOps pipelines enables continuous privacy risk mitigation throughout the ML lifecycle. Conducting user studies and gathering practitioner feedback will aid in refining usability and aligning the solution with real-world use cases. By pursuing these directions, the proposed privacy-preserving log analysis solution can continue to evolve, contributing to the development of secure, trustworthy, and responsible AI systems.

References

[1] P. Mohassel and Y. Zhang, "SecureML: A System for Scalable Privacy-Preserving Machine Learning," 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 2017, pp. 19-38, doi: 10.1109/SP.2017.12.
[2] Author(s), "Privacy-Preserving Machine Learning: Methods, Challenges and Directions," arXiv:2108.04417 [cs.LG], 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2108.04417
[3] Author(s), "BLAZE: Blazing Fast Privacy-Preserving Machine Learning," arXiv:2005.09042 [cs.CR], 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2005.09042
[4] Mannhardt, F., Koschmider, A., Baracaldo, N. et al. Privacy-Preserving Process Mining. Bus Inf Syst Eng 61, 595–614 (2019). https://doi.org/10.1007/s12599-019-00613-3.