RESEARCH ARTICLE OPEN ACCESS

Addressing the Exception Prioritization Problem in Continuous Auditing Systems With Thresholding

Jan Svanberg[1] | Peter Öhman[2] | Isak Samsten[3] | Presha E. Neidermeyer[4] | Tarek Rana[5] | Mats Danielson[6,3]

[1]Department of Business and Economics Studies, University of Gävle, Gävle, Sweden | [2]Department of Economics, Geography, Law and Tourism, Centre for Research on Economic Relations, Mid Sweden University, Sundsvall, Sweden | [3]Department of Computer and Systems Sciences, Stockholm University, Stockholm, Sweden | [4]John Chambers College of Business and Economics, West Virginia University, Morgantown, West Virginia, USA | [5]School of Accounting, Information Systems and Supply Chain, The Royal Melbourne Institute of Technology, Melbourne, VIC, Australia | [6]International Institute for Applied Systems Analysis (IIASA), Laxenburg, Austria

Correspondence: Jan Svanberg (jan.svanberg@hig.se)

Received: 23 February 2024 | **Revised:** 15 October 2025 | **Accepted:** 21 November 2025

Keywords: continuous auditing | exceptions | machine learning | overload | prioritization | thresholding

ABSTRACT

Continuous auditing research has grappled with the challenge of managing the abundance of detected exceptions in internal audit applications for the past 30 years. A key issue in continuous auditing involves the uncontrolled proliferation of exceptions, where the sheer volume makes manual follow-up impractical, undermining the viability of the technology. The root cause of this problem is the combination of strong class imbalance and the predominant rule-based systems design. Prior investigations have attempted ad hoc remedies like introducing additional layers to prioritize the most suspicious exceptions or aggregating data. Currently, there is no universal method to address this prioritization challenge, leaving internal auditors without a means to focus specifically on exceptions most likely to represent genuine faults. Our research explores the origin of this prioritization dilemma and proposes a systems design that can deal appropriately with class imbalance. This solution allows full control of the exception volume by a simple approach in machine learning called thresholding and combined with methods to interpret the output of a continuous auditing system our design effectively focuses the internal auditors' attention on the most significant exceptions. We discuss the implications of thresholding for practice and the literature.

1 | Introduction

Traditional audit sampling examines only hundreds of records from millions, risking undetected errors. While risk-based auditing targets high-risk transactions, detecting rare errors in large datasets remains challenging (Teitlebaum and Robinson 1975), and judgment-based approaches suffer from bias (Hall et al. 2000). An alternative, made possible by advances in data technology, is to forego sampling and instead review the entire population (Issa 2013; Li et al. 2016; No et al. 2019). These techniques involve calculating the likelihood of a transaction being incorrect and then manually investigating the most suspicious cases. Such use of computers in auditing is called continuous auditing (CA). CA, pioneered by Groomer and Murthy (1989) and Vasarhelyi and Halper (1991), focuses on detecting exceptions—suspicious or irregular transactions—for internal auditor follow-up. While our application focuses on internal audit activities that could be characterized as continuous monitoring, we maintain the CA framework to align with the established literature that consistently positions internal audit exception detection under the CA umbrella (Vasarhelyi and Halper 1991; Alles et al. 2006; Eulerich and Kalinichenko 2018). Despite extensive conceptual development and numerous definitions (Kogan et al. 1999; Rezaee et al. 2002; Vasarhelyi and Halper 1991; Woodroof and Searcy 2001), CA primarily serves to identify exceptions for internal auditor investigation (Byrnes and

Mcquilken 2012; Eulerich and Kalinichenko 2018; Vasarhelyi et al. 2012). CA systems have limited adoption. Eulerich and Kalinichenko (2018) note that while CA systems can identify anomalies based on predefined criteria, they are "not commonly used by organizations" (p. 142). Few studies examine actual CA implementations (El-Masry and Reck 2008; Gonzalez et al. 2012; Rikhardsson and Dull 2016; Vasarhelyi et al. 2012), with real-world data usage often highlighted as a contribution (Wei et al. 2024).

The gap between CA theory and practice explains implementation difficulties. While CA theoretically increases efficiency and effectiveness, a critical practical challenge is controlling output volume. CA systems often generate overwhelming numbers of exceptions, leaving internal auditors uncertain which to investigate (Alles et al. 2006, 2008; Perols and Murthy 2012). This exception overload undermines CA's practical benefits.

While some CA designs may avoid exception overload—such as process mining for internal control evaluation or machine learning for fraud risk assessment (Ding et al. 2019; Perols et al. 2017)—the literature lacks evidence comparing design effectiveness in managing output volume.

This study addresses why CA systems generate overwhelming exception volumes and proposes thresholding methods enabling internal auditor control. Unlike previous rule-based approaches requiring organization-specific expertise (Issa 2013; Li et al. 2016), our machine learning-based method provides generally applicable prioritization through probability estimates combined with SHAP interpretation methods.

Random sampling, which is to treat all exceptions equally as if there is no information about the likelihood that an exception is true or not, may have the merits of simplicity and ease of use, representative coverage of a whole dataset, and avoidance of overfitting a more precise prioritization method and thus avoidance of a too narrow audit focus. The latter may therefore be more compliant with audit standards than a method that can identify true exceptions more effectively and efficiently. Even if the internal audit designs its methods discretionarily, compliance with external audit standards makes the internal audit more reliable and useful for the external auditors (Malaescu and Sutton 2015) and may therefore be an issue to consider when choosing CA architecture. However, we do not consider random sampling an effective prioritization method, as it does not enhance the quality of the sample by identifying a higher proportion of true exceptions than what exists in the overall population. To improve the efficiency of manual follow-ups, it is crucial for CA system development to adopt more effective methods for prioritizing exceptions. Relying on random sampling leads to inefficient use of internal auditors' time.

Issa (2013) investigated alternatives to random sampling in a doctoral thesis motivated by the scarcity of studies that address the problem of processing large numbers of identified exceptions and proposing methodologies to detect and subsequently prioritize such exceptions with the purpose of avoiding exception overload. These prioritization techniques were intended to direct auditors' and management's investigations towards the more suspicious cases. These methodologies were all based on rules that incorporated the internal auditors' expertise. In a rule-based CA system the main idea was to have auditors assign a suspicion score based on what rule the transaction violates. A later study by Li et al. (2016) investigated a similar solution on real company data and found that the addition of extra layers of rules (suspicion scores) results in effective prioritization of exceptions. Results show that the methods designed to enhance the prioritization performance of rule-based implementations of CA systems offer improved prioritization, saving the internal audit costs and increasing its ability, and adding a learning feature that updates the suspicion scores, adding further capacity compared to simply random sampling. This more structured, adaptive, and risk-focused prioritization is clearly superior to random sampling. The downside of, at least the Li et al. (2016) implementation, is its technical complexity, the initial setup process, the need for adaptation and maintenance, and its organization-idiosyncratic nature.

Methods that require a lot of situation-specific development cannot be assumed to function equally well in all organizations. The fact that Issa (2013) and Li et al. (2016) were successful with their rule-based CA systems that prioritize exceptions with suspicion scores does not guarantee that the specific conditions in the investigated data or organization did not present themselves as a good fit with the employed methods and that the method therefore is highly organization-idiosyncratic. The rules and the suspicion scores could be very different from those presented by Issa (2013) and by Li et al. (2016) when developed in other organizations and there is no guarantee that a CA system prioritizing exceptions well enough can be developed.

Another limitation of their proposed solution is that the development of a functional rule-based CA system requires extensive collaboration with internal audit, as well as the audit team's ability to provide the CA developers with relevant information. However, this is not always something that can be relied upon. There is no guarantee that a rule-based CA system delivers sufficient precision and that suspicion scores add to the performance of the system so that the prioritized exceptions are a good enough sample of the transactions for auditors' follow-up. The ad hoc solutions proposed by Issa (2013) and Li et al. (2016) advocate the combination of prediction methods in a layered fashion, where the output of one layer serves as the input for the subsequent one. Layers of classifiers pose challenges in terms of interpretation, being potentially intricate, time-consuming, expensive, or even unfeasible for the internal auditors to understand. A layered rule-based CA system has a trade-off between predictive performance and interpretability for which there is no guarantee with the layered design that interpretability can be maintained at acceptable levels of exception prioritization. There is a risk that developers need to add such complex detection and prioritization methods that interpretability is lost. Finally, the main objection to the CA system designs discussed by Issa (2013) and Li et al. (2016) is that they are rule-based. Transactions may be hundreds of thousands, but the errors that internal auditors look for are typically few. In machine learning terms, this is referred to as class imbalance. A rule-based design is, in our view, harder to make work effectively than other designs when there is significant class imbalance. The reason is that the rule-based design offers no easily implementable method to set a threshold for the probability at which a transaction should be classified as an exception. This is why it is difficult to prioritize exceptions in a rule-based design.

This argumentation is straightforward but literature reviews offer little insight into the information overload problem. Reviews focus on themes such as reasons for using CA (Brown et al. 2007; Hassan et al. 2023), statistical, mathematical, and IT instruments as enabling technologies for CA (cf. Eulerich and Kalinichenko 2018) and some basic tools that developers would know about, for example, time series, cross-sectional regressions, and continuity equations are mentioned briefly by Brown et al. (2007). When discussing enabling technologies Brown et al. (2007) mainly cites a conceptual paper by Rezaee et al. (2002) and a key concept for prioritizing exceptions; the belief functions or suspicion scores are only mentioned in passing. It seems as though the exception overload issue is unknown to the literature reviews. There are of course many other practical challenges with new and complicated internal audit systems such as CA, for example, extracting data from an ERP system (Wang and Kogan 2020), but the reasons why CA systems can generate such a large volume of exceptions that the technology becomes unusable for internal auditing is a key issue for the future of CA systems that cannot be ignored. The present study contributes to the literature (1) by introducing thresholding as a method to prioritize detected exceptions in CA applications. The study demonstrates how probability estimates for the exceptions can be used as assessments of the likelihood that a suspicious exception is a true exception. In addition to prioritizing exceptions based on likelihood estimates, the study contributes to the literature by (2) describing how the SHapley Additive exPlanations (SHAP) interpretation methodology (Lundberg et al. 2020; Lundberg and Lee 2017) can provide internal auditors with further support in assessing which exceptions should be prioritized, as exceptions can be understood in a causal context.

We examine CA application development at Sweden's National Government Employee Pensions Board (SPV), serving 1,100,000 employees and pensioners across 250 employers. The pilot project developed methods to identify errors in pension registers, testing whether CA could enhance internal audit efficiency. SPV had no prior CA experience. Our CA application differs from previous systems by using standard machine learning algorithms without organization-specific rule-based constraints, providing generally applicable exception prioritization through probability estimates and variable importance metrics for auditor interpretation on comprehensive population data.

This paper proceeds with a literature review on CA volume control, followed by our case study findings and threshold accuracy analysis demonstrating SHAP's role in auditor interpretation.

2 | Literature Review

When introducing full-population auditing methods, auditors are faced with the choice between a rule-based design and an automated design. In a rule-based design, auditors establish specific rules to flag suspicious events, functioning similarly to expert systems. This approach is appealing because it leverages auditors' expertise, making the process intuitive and easier to comprehend. Alternatively, auditors can use data science techniques to detect errors with a higher level of technical autonomy, either through unsupervised learning (outlier detection) or supervised learning. Unsupervised learning is entirely data-driven (Alghushairy et al. 2021) and can be effectively applied in CA systems (Wei et al. 2024), whereas supervised learning requires auditors to provide a preclassified dataset of true and false exceptions (Caroline and Thomas 2021).

In the following, exceptions refer to deviations that may represent actual errors in the data or simply anomalies that deviate from a normal pattern. We also use the term error to refer specifically to actual data errors. An exception can thus be either an error or a false alarm. Determining which is present typically requires manual verification. A CA system that identifies exceptions can therefore return both errors and false alarms. The term anomalies is used in its lexical sense.

2.1 | Exception Overload: A Core Challenge in CA Implementation

An obstacle preventing widespread CA adoption is exception overload—the generation of more suspicious transactions than internal auditors can practically investigate. Alles et al. (2006, 2008) documented this problem in pilot implementations, where CA systems produced overwhelming volumes of alerts that exceeded audit department capacity. Perols and Murthy (2012) confirmed that even modest exception volumes can surpass auditors' analytical capabilities, leading to system abandonment. This overload problem directly undermines CA's theoretical benefits. When auditors cannot investigate detected exceptions due to volume constraints, the economic value of comprehensive population testing disappears. Organizations consequently revert to traditional sampling methods or disable CA systems entirely, explaining the limited practical adoption despite decades of research (Eulerich and Kalinichenko 2018).

The effectiveness of full-population auditing depends on auditors' expertise in identifying risk factors and assigning appropriate weights when determining suspicion levels. However, this reliance on judgment creates a fundamental dilemma. Unsupervised outlier detection, while promising, cannot distinguish between relevant and irrelevant deviations, often producing overwhelming numbers of potentially insignificant exceptions. This creates "exception overload"—the core challenge preventing CA adoption in practice. Exception overload occurs when CA systems generate more suspicious transactions than auditors can investigate (Kim and Vasarhelyi 2012; Thiprungsri and Vasarhelyi 2011). When investigation becomes impractical due to volume, CA's economic benefits disappear, explaining limited adoption despite decades of research (Eulerich and Kalinichenko 2018). Issa and Kogan (2014) argued that managing large volumes of exceptions detected for reasons unknown to auditors is challenging, supported by behavioral studies showing humans struggle to analyze such exceptions effectively (Iselin 1988; Kleinmuntz 1990). Therefore, even modest exception volumes can exceed auditors' capacity to address them effectively.

The issue of overload is not universally applicable to all CA systems. By focusing CA on business process controls, it is possible to limit the number of alerts through a technique known as process mining. Although process mining is well-established, few studies have examined its application in accounting and auditing (Duan et al. 2024; Jans and Hosseinpour 2019). This methodology has proven effective for evaluating internal controls as well as financial statements (Werner et al. 2021).

While process mining may be pivotal for future CA implementations, our focus is on error detection at the transactional level. This level involves a vast number of observations, often numbering in the millions. Analyzing such large datasets is the focus of numerous studies, including Zhaokai and Moffitt (2019), who developed a contract analytics framework to facilitate the analysis of entire populations of contracts, traditionally assessed through sampling. Visualization is another powerful tool that can enhance the audit process (Abdullah 2015). While process mining offers an alternative for assessing internal control effectiveness through event log data, it is not designed for transactional analysis (Jans et al. 2013).

At the transactional level, addressing the overload issue in full-population testing is particularly challenging due to the sheer volume of transactions and the corresponding number of potential errors. Recent literature has begun to explore the problem of exception overload and exception detection in this context. Li et al. (2016) propose a method that assigns suspicion scores to each transaction based on violations of predefined expert rules, setting a threshold for further investigation. While this rule-based design is intuitive for auditors, it relies on accurate expert knowledge of error characteristics and lacks a mechanism to control the volume of detected exceptions, making overload a significant risk. Furthermore, while the belief functions are set by internal auditors, their definition involves elements that are not suitable for decision-making and they are eventually updated by a learning feature that makes interpretation of the prioritized exceptions more and more difficult as the systems learn (Rozario and Issa 2020).

Similarly, No et al. (2019) introduce a rule-based approach that employs weighted filters based on risk factors for suspicion scoring. In their study, the suspicion scores serve as proxies for auditors' judgments, aiming to reduce exception volume through serially linked rule-based classifiers resembling a previous study by Issa (2013). In a more recent study, Freiman et al. (2022) utilize a multidimensional audit data sampling methodology on a real-world general ledger dataset, demonstrating its effectiveness in managing exception overload.

The discussion is not new. Alles et al. (2006) noted that even in control-oriented designs, the volume of generated alarms can overwhelm internal auditors (Jans and Hosseinpour 2019; Perols and Murthy 2012). As data processing increases and the effectiveness of detection techniques diminishes, the number of alarms is likely to rise. The system's ability to accurately identify suspicious events hinges on the technology used for data classification, directly impacting audit efficiency. Limited resources within audit departments further constrain the capacity to investigate exceptions (Chan and Vasarhelyi 2011).

It is crucial to distinguish between control-oriented and data-oriented CA systems, as the former typically generates fewer exceptions than the latter due to the nature of the task. This distinction, highlighted by Kogan et al. (1999), involves finding a trade-off between control-oriented and data-oriented CA procedures, which relates to the level of data aggregation employed. Selecting an appropriate level of data aggregation can help manage the volume of detected exceptions. Most literature has focused on control-oriented CA (Wei et al. 2024), operating at a higher level of aggregation.

A recent development by Yoon et al. (2021) illustrates how aggregation can mitigate the overload problem. Their approach utilizes three layers: The first identifies unusual transactions as nonroutine errors, the second flags transactions violating internal controls as exceptions, and the third detects transactions deviating from standard business behaviors as anomalies. While their study found that aggregated data can reduce audit effectiveness, it emphasizes the need to find the right level of aggregation for effective CA systems. The challenge remains that increasing abstraction may cause many errors to go undetected, compromising the CA system's reliability as a component of internal audit.

Another recent study that deals with prioritization is Rozario and Issa (2020). This study adopted and modified Issa's (2013) framework, according to which prioritization is based on weights defined by expert knowledge from domain specialists. For auditors, simple weights that they themselves define may be easier to interpret compared to the less transparent belief functions methodology used by Li et al. (2016). The study was conducted within the internal audit department of a US county, using real data with the task of identifying duplicate payments. Results of the case study showed that the proposed framework significantly improves both efficiency and effectiveness compared to traditional methods auditors use to address duplicate payments, such as visually scanning transactions or reviewing a sample. Whereas sampling and visual scanning can only find the same proportion of errors that the dataset has, the analytics for prioritization enabled auditors to find about 4/7th of the errors or 3/4th of the errors when manually following up only 15% of the suspicious exceptions after prioritization. Nevertheless, the proposed method includes idiosyncratic elements because the effectiveness of the method in finding true errors depends on the ability of internal auditors to identify relevant indicators that predict the errors and define accurate weights on the indicators to maximize the performance of the classifier used for prioritization. This ability will vary from organization to organization.

2.2 | Rule-Based vs. Probability-Based Exception Detection: Design Differences

Exception overload in CA systems stems from design choices in detection architectures. Rule-based systems, dominant in CA literature (Kogan et al. 1999; Li et al. 2016; No et al. 2019), detect exceptions through explicit logical rules defined by domain experts. Probability-based systems use machine learning algorithms to assign likelihood scores based on patterns learned from historical data. These approaches differ in their mechanisms for controlling exception volume.

Rule-based CA systems evaluate transactions against programmed detection rules such as "IF transaction amount

> $50,000 THEN flag as exception" (Issa 2013; Li et al. 2016). Multilayered implementations prioritize detected exceptions using belief functions or suspicion scores (Li et al. 2016; Rozario and Issa 2020). Auditors can trace which rules a transaction violated.

Threshold-setting challenges emerge in multidimensional contexts. A procurement system might flag: (1) transactions > $50,000, (2) vendors < 2 years old, (3) noncompetitive bids, and (4) tax haven locations. A transaction at $49,999 to a 25-month-old vendor in the Cayman Islands evades Rules 1–2 while violating 3–4. A legitimate $75,000 IT investment with a startup triggers Rules 1–2. Distinguishing these requires additional exception rules or manual review. Because these "automated rules are strictly formal, the existing rules have a significant amount of imprecision" (Alles et al. 2004, 7)—a statement that underlines the difficulty with using a combination of rules to control the exception volume. Volume control requires trial and error. If $50,000 generates 10,000 exceptions, auditors might adjust to $75,000 but cannot predict the resulting volume without rerunning the system. Adjustments in one dimension interact unpredictably with other rules. Dynamic business environments compound this—organizational growth changes transaction distributions, requiring threshold recalibration. Alles et al. (2004, 6) state that rules "has to be reexamined and updated on a regular basis" in order to ensure appropriate thresholds. Hayes-Roth et al. (1983) documented this knowledge acquisition bottleneck for expert systems. It follows from the logic of rule-based designs that you have to adjust one or more rules in order to change the detection volume, and it can be difficult to know how the rule changes transform into a detection volume change.

Probability-based systems such as supervised machine learning algorithms learn patterns from historical labeled data, producing probability scores (0–1) for each transaction representing exception likelihood (Ngai et al. 2011; West and Bhattacharya 2016). Auditors set a single probability threshold (e.g., 0.7 = flag transactions ≥ 70% probability). Precision–recall curves show exact trade-offs between false positives and true positives at every threshold value before implementation (Zou et al. 2016). Thus, the designer does not have to guess the detection volume; it can be an easily controlled aspect of the design of the detection system. Algorithms implicitly learn complex, nonlinear relationships without explicit rule specification. They consider hundreds of variables simultaneously, weighting each according to empirical predictive power. Models retrain on updated data to adapt to changing conditions automatically. This addresses combinatorial explosion—learning from data which patterns predict exceptions rather than requiring explicit specification.

Furthermore, with this type of method interpretability requires post hoc methods. SHAP values quantify each variable's contribution to individual predictions (Lundberg and Lee 2017), revealing which characteristics influenced probability scores.

We can now compare the two types of approaches and identify applicable conditions. Probability-threshold advantages manifest when: (1) exception volumes exceed capacity by orders of magnitude, (2) multidimensional risk patterns involve complex interactions, (3) class imbalance is severe (error rates < 5%), (4) business environments change frequently, and (5) domain expertise for complete rule specification is limited. Rule-based approaches remain applicable when volumes are manageable (< 500), patterns are simple, regulations require explicit criteria, or historical training data are insufficient. The choice involves trade-offs between volume control predictability and inherent interpretability.

From a control-oriented CA perspective, implementing a rule-based program and prioritizing with belief functions, akin to past expert systems, is both logical and intuitive. Previous literature frequently advocates for rule-based systems to identify suspicious items based on expert knowledge (Li et al. 2016; Perols and Murthy 2012). However, this approach has notable downsides. To clarify how different approaches address the exception volume challenge, Table 1 summarizes the main methods used in CA systems along with their capabilities for volume control.

As Table 1 demonstrates, existing approaches either sacrifice detection capability (sampling, aggregation) or lack direct volume control mechanisms (rule-based systems). Our thresholding approach uniquely combines precise volume control with probability-based prioritization.

Rule-based CA systems require extensive process documentation and exhaustive error definitions, making them deterministic rather than probabilistic. While most CA designs are rule-based (Kogan et al. 1999; Murthy 2004; Rezaee et al. 2002; Woodroof and Searcy 2001), they lack exploratory capabilities and often require multiple classifier layers for adequate performance. This results in organization-specific solutions that may be impractical across different contexts.

Organization-specific prioritization methods create inconsistencies (Issa and Kogan 2014). These approaches assume auditors can assess all detected exceptions, making feasibility dependent on exception volume—viable in some organizations but unmanageable in others. Data aggregation reduces exception volume by treating thousands of events as one but causes significant information loss (Yoon et al. 2021). While popular for audit planning rather than substantive testing (Eulerich et al. 2020), highly aggregated analytical procedures can achieve accuracies as low as 50%.

Understanding why transaction-level CA designs in the literature often suffer from exception overload requires insight into rule-based implementations. In these designs, exception volume is typically controlled by adding or tightening rules, which reduces detections gradually. Data aggregation, another rule-based technique, raises abstraction by ignoring case differences. Studies like Li et al. (2016) and Issa (2013) highlight the appeal of rule-based systems for auditors but reveal their limitations: they lack mechanisms to control exception volume through probability thresholds and have suboptimal accuracy. We see three main problems with the previous literature's focus on rule-based designs and data aggregation. Firstly, rule-based systems do not provide a "volume control" to adjust the number of exceptions, forcing auditors to manually tweak rules and check outcomes only after running the system, offering limited predictability and control.

TABLE 1 | Comparison of continuous auditing approaches for exception volume management.

Approach	Description	Advantages	Disadvantages	Volume control method	Interpretability
Traditional sampling	Random/judgmental selection	Simple, established practice	Misses rare errors	Manual sample size selection	High
Rule-based CA	Expert-defined detection rules	Intuitive for auditors	No probability estimates	Rule adjustment only	Medium
Multilayer rules (Li et al. 2016)	Layered classifiers with belief functions	Improved prioritization	Complex, organization-specific	Indirect through layers	Low
Data aggregation (Yoon et al. 2021)	Analysis at higher abstraction levels	Reduces alert volume	Loss of transactional detail	Aggregation level selection	Medium
Process mining	Event log pattern analysis	Control-focused insights	Not transaction-level	Process scope definition	High
ML + thresholding (this study)	Probability-based classification	Direct volume control with prioritization	Requires labeled training data	Probability-threshold adjustment	High (with SHAP)

Secondly, reducing detections in traditional CA designs often requires layering rules, which complicates the interpretability of results. While individual rules are easy to understand, the combined effect of multiple interacting rules becomes complex, making it harder to assess the reasons behind detections.

Thirdly, evidence on CA systems using disaggregated data is scarce. Errors can increase with higher levels of aggregation, supporting the idea that accuracy improves with more granular data (Chen and Leitch 1999; Hoitash et al. 2006; Leitch and Chen 2003). This suggests that refraining from data aggregation is critical for achieving high accuracy in CA systems. While aggregation is often used to manage exception overload, it comes at the cost of accuracy. As Yoon et al. (2021) state, "A large volume of alarms in a CA system can reduce audit efficiency... leading to reliance on aggregated data with low accuracy as the effect" (p. 14). This trade-off is inherent in rule-based designs, which regulate exception volume through rules and aggregation, but may not apply to other designs.

Our proposed CA design incorporates thresholding for direct exception volume control, allowing auditors to manage detected exceptions like adjusting radio volume. Thresholding sets classification boundaries based on probability scores (Sibiya and Sumbwanyambe 2021), ensuring only exceptions surpassing chosen thresholds are flagged. This provides flexible, data-driven exception management—more effective and adaptable than current rule-based literature approaches.

CA classification models output probability scores indicating the likelihood of genuine faults. Thresholds determine which exceptions warrant follow-up, creating precision–recall trade-offs. These readily available metrics provide internal auditors with complete insight for threshold decisions.

While uncommon in CA designs, thresholding is essential for class imbalance—where one class significantly outweighs another in binary datasets. Class imbalance occurs in practical scenarios like fraud detection (Krambia-Kapardis et al. 2010), where the minority group represents the focal class of interest (Johnson and Khoshgoftaar 2019). This imbalance causes overclassification of majority groups due to higher prior probability (Chawla et al. 2004), with challenges exacerbated by big data complexities and class rarity. Model performance deteriorates as imbalance increases (Weiss and Provost 2001), making methods for severe class imbalance imperative for effective CA exception detection.

In CA, class imbalance creates exception detection overload through poor classification accuracy—either excessive false positives or insufficient true positives. Both scenarios render methods practically ineffective. Too few exceptions allow errors to go undetected (like nets with overly large mesh catching no fish), while too many exceptions overwhelm auditors with false alarms (like nets with small mesh catching unwanted fish). For internal auditors, the optimal level involves identifying enough errors to make manual follow-up practically feasible, provided the detection method has sufficient predictive performance (precision and recall). When the alternative is random selection from CA output, prioritization methods offer significant value.

The application of thresholding with machine learning algorithms has been extensively employed to enhance classification outcomes (Chen et al. 2006; Zou et al. 2016). As an algorithm-level technique, it adjusts the bias towards a specific class by modifying the classification threshold utilized for assigning class labels to probability estimates. A decision threshold of 0.5 means that the positive class label is assigned when the classifier estimates a posterior probability greater than or equal to 0.5. Lowering the threshold causes the classifier to assign the positive class label to observations with lower confidence, with more positives as a result. At threshold 0, all cases are positives (declaring all accounting as suspicious does not help the auditor much), and at 1, they are all negatives (handing the auditor no detected exceptions is equally useless). Consequently, the trade-off between class-wise performance scores must be taken into account. Arguably, it is a critical mistake to use the default threshold 0.5 when data are imbalanced because in auditing it is in the nature of the task that data are imbalanced with the resulting overclassification bias causing a need for a decision about the threshold. Nevertheless, thresholding is not a concept that the CA literature has employed.

A reason for not adopting thresholding might be that the data are often too imbalanced. A critical gap might therefore exist between technical capability and practical adoption in auditing applications of machine learning. Despite decades of research in fraud detection using anomaly detection techniques, few organizations implement these methods in practice. This implementation paradox—where technically sound methods fail in practice—stems from fundamental data limitations that must be examined in the context of CA applications. Machine learning approaches to fraud detection face a fundamental obstacle: fraud rates are typically so low ($< 0.01\%$) that training effective models becomes impossible. Even with sophisticated training methods, the scarcity of fraud cases prevents model development, explaining why this research stream, despite being decades old, sees limited practical application in audit and assurance activities.

Our CA application addresses data quality errors rather than fraud, creating fundamentally different implementation conditions. Data quality errors occur at substantially higher rates (3%–7% in our study) compared to fraud ($< 0.01\%$), providing sufficient positive examples for model training. Thus, in some other situations the data may be too imbalanced for our method. Additionally, organizations typically maintain historical correction records that serve as natural training labels, unlike fraud detection where true labels are rarely available.

Successful implementation nevertheless requires specific organizational conditions. Organizations need historical error correction data with a minimum of 1000 labeled errors and error rates exceeding 1% for adequate model training. Stable data structures and business processes are essential, along with systematic error identification and correction procedures. Conversely, organizations should avoid this approach when error rates fall below 0.5% due to insufficient training data, when business processes change rapidly causing model instability, when no systematic error correction history exists, or when data governance maturity is insufficient to support the implementation.

While thresholding is a well-established technique in machine learning for handling class imbalance, its systematic application to CA exception volume control has not been explored in the literature. Previous CA research has focused primarily on improving detection accuracy through rule refinement or layered classifiers but has not addressed the volume control problem that prevents practical implementation. The contribution of thresholding to CA lies in providing probability-based prioritization with direct volume control, where auditors can adjust exception volume similarly to adjusting classification thresholds in machine learning applications. This approach differs from organization-specific rule-based solutions by offering generally applicable, threshold-adjustable classification methods that address the exception overload problem documented in CA implementations.

To fill a void in CA research our study systematically explores the application of thresholding to address class imbalance in significantly imbalanced CA data. The CA dataset in this study exhibits a class distribution where the erroneous entries constitute 5% of the entire dataset. We conduct experiments with different threshold levels to demonstrate how the volume of detected exceptions can be controlled using this technique. In machine learning literature, there is a conceptual idea of an optimal threshold level, but in the context of CA, this level depends on parameters such as the cost of following up on an exception and the value of detecting a genuine anomaly. Unfortunately, in this study, we do not have access to precise values for these parameters.

In an ideal scenario, these parameters could be calculated or estimated, and an optimal threshold level would ideally then be determined so that the cost of following up on exceptions at the threshold level equals the economic value of detecting them. Unfortunately, in practice, it is not always possible to precisely determine these values, and there remains a challenge of adjusting the threshold level in a way that balances the cost of follow-up and the value of anomaly detection. However, this problem is a luxury that should be afforded to the internal auditors because without thresholding the auditors may be unable to control the exception volume.

We address these issues with previous CA systems research by describing the development process leading up to a data level CA system that relies on generally applicable machine learning algorithms that allow thresholding and builds on state-of-the-art interpretation methods in machine learning. We investigate two research questions:

> R1: How can thresholding be used as a method to control the volume of detected exceptions of a data level CA application?
>
> R2: What contribution can a variable importance measure provide to the organization's understanding of the causes of register data errors?

3 | Method and Data

3.1 | The Organizational Context

Statens Pensionsverk (SPV), which is the state government's occupational pensions agency, stands as one of Sweden's major providers of pension services, overseeing the administration of occupational pensions for more than 1,100,000 people, including current and former government employees and pensioners. Additionally, SPV manages pension data from approximately 250 authorities and selects companies with government contracts. Its responsibilities encompass calculating pension premiums, providing forecasts and statistics to the government, and serving about 250 employees. This agency plays a pivotal role in responding to inquiries from pensioners, government employees, and employers. Its tasks extend to the computation and disbursement of occupational pensions, calculation of liability and premiums for employers, and the compilation of pension statistics.

In accordance with government regulation (1997:909), SPV is entrusted with the administration of pensions for a substantial portion of government employees in Sweden. This involves the management of a service register for individuals covered by permanent occupational pension regulations. State authorities, serving as employers, are mandated to furnish SPV with the requisite information for maintaining this register. SPV has established regulations (AgVFS 2016:1 A1) delineating the process by which employers must provide accurate and complete monthly information on employment, in line with the agency's stipulated requirements outlined in a document known as Transfer Requirements. The organizational context is meticulously regulated, necessitating employers reporting to SPV to adhere to prescribed methods for reporting employment information. Accurate reporting is a prerequisite for ensuring that government employees in authorities receive their entitled pension upon retirement. Keeping track of how much people have worked and how much they were paid is a vital part of SPV's obligations, so SPV needs to keep this record as accurately as possible. The purpose of this project from SPV's perspective is to develop a method that can assist SPV in identifying registry entries that justify manual follow-up. The project is explorative and the interest of SPV was to see whether machine learning technology could benefit the manual work of internal auditors.

The reporting of employment conditions is an ongoing process, with certain reports carrying particular significance. One crucial instance is the termination of employment. Despite government authorities being mandated to report such terminations and their reasons, a noteworthy number of errors persist, posing challenges for SPV in identification. SPV has published a document on its website outlining the agency's requirements for employers' reporting of employment information. This document stipulates that, upon termination of employment, the monthly report submitted by the employer must include the date of termination, along with the reason for termination. The document details various reasons, each with distinct implications for pension outcomes. Examples include retirement with a disability pension, retirement with an old-age pension for specific professions, retirement before the age of 65, retirement with pension compensation, retirement in accordance with transitional provisions, deceased status, and other retirement scenarios. The document lists 20 different reasons for leaving that employers must specify. Termination of employment holds not only significant importance for SPV's accurate registration but also provides access to labeled data.

SPV maintains an internal control framework that regularly conducts internal audits. However, challenges arise in rectifying errors related to the absence of reports for employment terminations, making it difficult for SPV to detect and address such discrepancies. Instances persist where individuals remain in the system despite having concluded their employment several years prior. Typically, employers are the ones to identify cases where employment has ceased but has not been reported to SPV, although there is no guarantee that this oversight will consistently occur. Identifying such inaccuracies involves a labor-intensive process, and the authority faces constraints in internal audit resources to systematically detect these exceptions on a large scale without adequate system support. Consequently, SPV requires a CA system to facilitate the identification of unreported employment terminations. The system needs not report exceptions in real time but can be used as an audit analytic when the employment data register is audited. If compared to the idea of a CA system providing real-time assurance, the application we are studying here is a more limited application that is tested in a pilot project with the purpose to demonstrate whether a machine learning application can be useful for internal auditors as they seek to improve the quality of SPV's data.

Section 3.2 delineates how SPV can identify inaccurate entries of employment terminations in its registers. During the second quarter of 2022, information for over 280,000 individuals was reported to SPV. Registers of previously self-corrected employment terminations provide valuable resources for the CA system's development. In collaboration with SPV, our research group has identified the most pertinent indicators to form the cornerstone of the CA system, with Section 3.2 providing comprehensive details on the data and research design. Identifying indicators with predictive power for detecting true errors is helped if auditors are able to highlight contexts that may correlate with the occurrence of actual errors. It is not even necessary that they can do that because developers can investigate models with hundreds of indicators and narrow-down to the most important ones in a trial and error fashion. In contrast, this task is much simpler than the one auditors face in a rule-based design, where they must possess detailed knowledge of the specific conditions that cause errors and the exact indicator levels likely associated with errors. In a rule-based design, auditors are required to formalize the model, including assigning weights to parameters and setting threshold levels for indicators to identify exceptions. This means they need a deep, precise understanding of the rules and how each factor interacts, whereas a predictive model can identify patterns without requiring the same level of explicit detail about the causes and specific thresholds for each indicator.

3.2 | Data and Research Design

The purpose of the CA system examined in this study is to assist the internal audit in its efforts to identify errors in the reported employment data. We do not have access to an integrated IT infrastructure, which is a prerequisite for real-time monitoring of internal controls, so we choose a design that provides assurance through analytical procedures by investigating one registry, albeit a large one, that would require years of work to fully investigate manually. We avoid rule-based classifiers for the reasons explained regarding their deficiencies when it comes to providing a likelihood estimate because such a design does not normally give us probability estimates for each detected exception. Instead, we use supervised machine learning models that provide probability estimates for each detected exception and identify qualified exceptions as those with a suspicious score above a threshold. We are interested only in models that allow us to control the volume of detected exceptions with the threshold level.

Before describing the research design, we describe data. SPV's internal audit expert provided a dataset with more than 130 indicators for each employee in the register. Most indicators are relevant only to particular employee categories, which means that a lot of indicators are missing for a particular employee. The dataset covers data for 131,039 individuals of which 6538, approximately 5%, had been labeled as exceptions through a labelling procedure by the internal audit expert. SPV's expert developed an analytical formula with which to define termination dates as retroactively added or removed. This defines a register error that has been retroactively corrected, and we use this as our proxy for a register error in this study. The presence of retroactive corrections signals that a register error has been in place for some time that has eventually been identified by SPV or the employer. Thus, the retroactive corrections can be used as labels of incorrect entries but do not cover all register errors because they may not have been detected and corrected. The discovered and corrected entries most likely constitute only a fraction of the actual number of incorrect entries because there is no viable method to detect them all manually so the purpose of the CA system is to learn a method to detect not only these 6538 errors but to ensure data quality in the registers and ultimately to assure the quality of the services provided by SPV by detecting a large fraction of the now unknown register errors. Since new registry data is continuously entered, there is also a need for a method that allows for the continuous review of incoming information to correct errors in real time and even prevent errors. The register contains many features that each employee may or may not have or may be related to: Employment categories such as managers covered by regulations, air traffic controllers, certain military positions, reserve officers, train drivers, professional officers at the age of 60, pilots, combined positions at universities and colleges, and insurance medical and dental advisors within the insurance agency. Different age categories apply for different employment categories. Employments are in three main types of contracts—with monthly payment, without monthly payment and with less than 20% of full time, without monthly payment and with more than 20% of full time. A number of reasons for being absent from work are given such as illness, part-time retirement, on leave for military service within the armed forces, unpaid leave, fully on leave for another employment with entitlement to occupational pension, on part-time leave for another employment, or fully on leave for another employment without entitlement to occupational pension.

The retroactive corrections, our labels, are selected using several criteria, some of which are difficult to decipher without detailed knowledge of how the register is constructed. We relied on the internal audit expert for this assessment. The register errors relate only to the situation when the employer has not reported termination of employment or has reported termination although the employment is still there. There are several limitations to which errors we focus on. Firstly, only employment periods corresponding to employment with at least 20% employment are included. Small part-time employments are excluded from the study because dealing with them in the register would require manual preparation of the register so that it is consistent with our definition of retroactively adding or removing an employment termination date as our definition of a register entry error. Secondly, we also used the distinction that for an employment to be considered to have an expiry date, there must be no employment for at least 30 days after the employment's last registered entries. Thus, there should be a period in which the person has apparently not worked at all for the employer. Thirdly, for an expiry date to exist as the originally reported expiry date, there must exist an entered expiry date when considering only data valid at the end of the second month after the expiry date. This criterion targets falsely registered expiries, cases in which employment continues despite that the employer has reported a termination at some point.

We anticipate that the incorrect entries have heterogeneous, nonlinear and multidimensional relationships with the indicators identified by the internal audit as potentially related to the likelihood that an entry is incorrect. We therefore expect that an estimation method with a high capacity for capturing complexity and nonlinearity should be more effective at learning to predict the incorrect entries than linear models or models used in statistics. For these reasons, we find the ML models that are known for their complexity-handling capacity more favorable for assessing whether an entry is incorrect or not than the often-used rule-based methods or statistical methods. ML allows the prediction of exceptions when distributions are unknown and when the estimated relationships are severely nonlinear (Duda et al. 2001). This methodology has been used in similar areas, for example, in finance for credit scoring (Cleofas-Sánchez et al. 2016) and bankruptcy prediction (Gerlein et al. 2016), which are applications having features in common with our problem, for example, class imbalance. A broad spectrum of algorithms can be deployed to classify our type of data with varying results, such as Nearest Neighbor, Linear Support Vector Machine, Radial Basis Function Support Vector Machine, Random Forest, Logistic Regression, Artificial Neural Network, Gradient Boosting, Naïve Bayes, and Quadratic Discriminant Analysis, but because there is a close relationship between the tree-based algorithms and the interpretation method SHAP we opted for the algorithm XGBoost, that is, tree-based and suited for nonlinear and complex data.

XGBoost in a CA system allows for thresholding by outputting probabilistic scores or confidence levels for each prediction, which auditors can use to set flexible thresholds based on risk tolerance. For example, auditors can decide to investigate only transactions that exceed a certain probability of being an exception (e.g., 80% or higher). This thresholding capability enables a tailored prioritization of exceptions, where the cutoff can be adjusted based on factors such as audit resources, risk levels, and specific case requirements. Prioritization of exceptions can therefore be achieved through thresholding. Our reason for using an algorithm such as XGBoost becomes evident when contrasted with rule-based CA systems. A rule-based implementation lacks this flexibility because it operates on binary logic. Each rule either flags a transaction as an exception or it does not, without any associated confidence score or gradient. As a result, auditors must rely solely on the set rules to filter exceptions and cannot easily adjust sensitivity based on probability or risk level. If auditors want to change the strictness of detection, they need to redefine the rules themselves, which is more rigid and less adaptable than the dynamic thresholding that XGBoost offers.

We sort the persons in the register into two (disjoint) categories based on whether they are considered false entries or not and denote false entries as positive cases and correct entries as negative cases. Our goal is to capture the attribute interactions that describe a register post as high risk of being a false entry.

XGBoost, or eXtreme Gradient Boosting, is a powerful and popular open-source machine learning algorithm that is widely used for supervised learning tasks such as classification and regression (Qiu et al. 2022). It is an ensemble algorithm that combines the strengths of multiple decision tree models to improve predictive accuracy (Le et al. 2019; Zhou et al. 2020).

In XGBoost, decision trees are constructed iteratively in a process called boosting. Each tree is built to correct the errors made by the previous tree, with the final prediction being a weighted combination of all the trees in the ensemble. The "extreme" in XGBoost refers to the fact that it employs a regularized form of gradient boosting that can handle complex datasets and prevent overfitting (Zhang et al. 2020).

One of the key advantages of XGBoost is its speed and scalability (Qiu et al. 2022). It is designed to be highly efficient, with parallel processing capabilities that enable it to handle large datasets with millions of features and billions of rows. XGBoost also provides built-in support for missing values and can handle a variety of data types.

XGBoost has been successfully applied to a wide range of machine learning tasks, including image classification, natural language processing, and fraud detection. It is widely used in industry (Qiu et al. 2022) and has won numerous machine learning competitions on platforms such as Kaggle. Gradient boosting is one of the strongest classifiers for various tasks (Sigrist and Hirnschall 2019).

3.3 | Experiments

To assess the predictive performance of ML algorithms on a previously unseen set of test instances, indicative of individual data in the register, a common practice is to partition the dataset into training and test sets. The training set is used to train the ML algorithm, while the test set evaluates its performance on independent data. However, when data scarcity or a more robust estimate of generalization performance is required, an alternative method is warranted. In addressing this, a recommended approach is *k*-fold cross-validation. This method involves dividing the dataset into *k* disjoint partitions (folds) and iteratively training the learning model on *k*–1 folds, reserving onefold for testing. The outcome yields *k* performance measures, and their mean provides a reliable estimate of generalization performance. Although various approaches exist for selecting the number of folds, *k*, this study adopts the widely used value of 10, resulting in tenfold cross-validation (Guyon 1997). To ensure an equal representation of positive and negative cases in each test set, a stratified tenfold cross-validation strategy is employed in this study. For the ML algorithm, the hyperparameter chosen is a learning rate of 0.1, while the remaining parameters adhere to the default values for SciKit-learn version 0.22 (Pedregosa et al. 2011).

The evaluation of machine learning models involves the use of diverse metrics, each capturing distinct aspects of learning ability (Alpaydin 2010). In this study, we employ five widely accepted performance measures: precision, recall, *F*-measure, the area under the ROC curve, and precision–recall curve (PRC). To assess and compare the performance of machine learning algorithms, we initially compute precision and recall for the estimators.

Precision and recall are defined using key metrics such as true positive (TP), false positive (FP), true negative (TN), and false negative (FN), as outlined in Equations (1–5) (refer to Table 2). Precision, representing the fraction of true positives relative to the total number of positive case predictions, mirrors the precision of a dart player hitting the target concerning the attempts made. Recall, on the other hand, signifies the fraction of true-positive predictions relative to all positive cases in the data, indicating the predictor's ability to identify the largest possible fraction of incorrectly labeled entries in our dataset.

Precision serves as a metric to gauge the classifier's sensitivity, specifically its accuracy in predicting controversy and noncontroversy classes. It is calculated as the correct positive fraction divided by the total number of positive predictions (Equation 1). Given the conflicting nature of precision and recall—for instance, a classifier predicting every company as having a controversy would yield a recall positive of 100%—the *f*-measure (Equation 3) effectively captures the trade-off between these two metrics.

The area under the ROC curve is determined by plotting true positives (Equation 4) against false positives (Equation 5). This measure estimates the probability of a classifier ranking a true-positive instance ahead of a false-positive instance, providing insights into its ranking performance. Similarly, the PRC evaluates the mean precision for multiple recall thresholds, akin to the *f*-measure, offering a perspective on the precision–recall trade-off. The area under the PRC is defined as the region beneath the plot of precision (Equation 1) versus recall (Equation 2). Notably, both the area under the ROC curve and the PRC are

TABLE 2 | Basic performance measures.

Measures of performance	Equation
$Precision = \frac{TP}{TP+FP}$	(Equation 1)
$Recall = \frac{TP}{TP+FN}$	(Equation 2)
$F - measure = \frac{2\times Precision\times Recall}{Precision+Recall}$	(Equation 3)
$True_{positive} = \frac{TP}{TP+FN}$	(Equation 4)
$False_{positive} = \frac{FP}{FP+TN}$	(Equation 5)

advantageous as they remain insensitive to the class distribution in the training and testing data, in contrast to accuracy.

4 | Results

In an organization that implements CA the auditors can be notified immediately and may initiate detailed investigation as an exception is detected. The auditor may then decide to correct the error before the next round of audit starts, and if the exception detection system is good enough, it will enable auditors to get rid of a lot of the errors prior to the next round of auditing. This could be on a daily basis or as seldom as once per month. In our experiments we do not use real-time error correction but test our exception detection methods on uncorrected data. Exception detection would perform better on data that has been gradually cleaned from exceptions because some of the errors are noise that tend to be detected as false positives. Our purpose is to propose and validate a one-step, standard application framework that prioritizes exceptions in the CA environment well enough to not require further layers of ad hoc exception prioritization that is the approach in previous studies (cf. Li et al. 2016). Our purpose is also to demonstrate a method of detecting and prioritizing exceptions that explains reasons for identifying an exception in a manner intelligible to an internal auditor. Prioritization of exceptions enables auditors to focus on those suspicious register entries that are more likely to be irregular transactions and other suspicious entries. Prioritization in most CA setups would depend on both how high accuracy the CA system can achieve after training and the extent that this can be maintained over time. The latter is because the process the ML model learns to represent may not be stationary, and in most cases, it is not. The levels and dynamics of the activity of, for example, accounts payable can shift rather dramatically over time. In our case, we do not have this problem. We treat the whole dataset as if it was entered into the books at the same time and could all be audited as it came in. Thus, our case is similar to what an auditor would confront with the audit at the end of a financial period. All data are already on the table. In our case, it means that we do not have to treat data as a flow generated from a nonstationary model, but as a time-independent phenomenon where all we need to do is to capture the process through which incorrect entries occur. The experiment therefore relies not on two criteria to evaluate the performance of the framework as in the normal CA case: Normally, the first criterion is the ability to effectively prioritize erroneous exceptions higher than non-erroneous exceptions or, in ML terminology, prioritize true positives higher than false positives. The second criterion adopted in the normal CA case is the framework's ability to improve its prioritization performance after each iterative run. Because we do not have a nonstationary process, we neglect the second criterion—we do not adopt continuous learning.

The predictive performance of the learning algorithms is presented in Table 3. An initial understanding of how the identification of suspected errors operates can be obtained from the values. The values illustrate that, in its default configuration, the algorithm chooses to flag 18% of the total 6538 suspected errors found in the file. Additionally, we observe a precision of 0.70, indicating that the algorithm makes a correct assessment in 70% of the cases where it deems an entry to be an error. A recall of 0.18 signifies that, with its default settings, the algorithm captures 18% of the labeled errors present in the registry (18% of 6538).

TABLE 3 | Performance measures.

	Precision	Recall	f1	Numbers
Positives	0.96	1.00	0.98	124,501
Negatives	0.70	0.18	0.28	6538
Accuracy			0.96	131,039
Macro average	0.83	0.59	0.63	131,039
Weighted average	0.95	0.96	0.94	131,039

These values suggest the feasibility of capturing the suspected errors using an XGBoost algorithm. However, it may seem practically unrealistic to follow up as many suspected deviations as 18% of 6538, which, when scaled to the entire registry, would result in a significantly larger number of suspected errors. Therefore, it could be of practical relevance to explore how the number of suspected errors can be adjusted to any level with which internal auditors are comfortable.

To illustrate the possibility of adjusting the model's probability threshold we conducted multiple runs with different threshold values in Table 4. In contrast to rule-based classification, which typically does not provide a probability value for each identified suspected error in a registry, machine learning algorithms normally have this capability; and in Table 4, the advantage of using such algorithms rather than a rule-based classifier becomes apparent. The threshold value represents the minimum probability at which the model classifies a register entry as an error. The threshold can be set to anything between 0 and 1 by the internal auditor who is using the CA system. As the threshold value increases, the model enhances its accuracy by demanding higher certainty in its classifications, leading to a rise in precision albeit at the cost of recall. Notably, the algorithm faces challenges due to significant class imbalance, struggling to attain a high confidence level for identified errors.

Despite these challenges, the model attains a precision of 0.73 when the threshold value is set at 0.5. Determining the feasibility of identifying a specific percentage of registry errors requires a comprehensive cost–benefit analysis. If the expense of investigating the detected exceptions is outweighed by the benefit of accurately identifying 73% of errors, the procedure is deemed efficient. At the 0.5 threshold level, the model successfully identifies 18% of all errors in the file, comprising 1177 accurately identified errors and 435 false alarms. Consequently, auditors must evaluate whether they can manage the 1612 identified exceptions and, if not, explore strategies to enhance efficiency or choose to investigate a smaller set of exceptions that costs less to follow up because it contains a lower percentage of false alarms.

TABLE 4 | Thresholds and exception detection volumes.

Threshold $p = 0.1$				
	Precision	**Recall**	**f1**	**Sample**
Non-errors	0.98	0.89	0.98	124,501
Errors	0.26	0.70	0.28	6538
Accuracy			0.96	131,039
Macro average	0.62	0.80	0.66	131,039
Weighted average	0.95	0.88	0.91	131,039
Threshold $p = 0.15$				
	Precision	**Recall**	**f1**	**Sample**
Non-errors	0.98	0.93	0.95	124,501
Errors	0.31	0.62	0.41	6538
Accuracy			0.91	131,039
Macro average	0.64	0.77	0.68	131,039
Weighted average	0.94	0.91	0.92	131,039
Threshold $p = 0.20$				
	Precision	**Recall**	**f1**	**Sample**
Icke-fel	0.97	0.95	0.96	124,501
Fel	0.36	0.51	0.42	6538
Accuracy			0.93	131,039
Macro average	0.67	0.73	0.69	131,039
Weighted average	0.94	0.93	0.93	131,039
Threshold $p = 0.25$				
	Precision	**Recall**	**f1**	**Sample**
Non-errors	0.97	0.98	0.97	124,501
Errors	0.47	0.35	0.40	6538
Accuracy			0.95	131,039
Macro average	0.72	0.66	0.69	131,039
Weighted average	0.94	0.95	0.94	131,039
Threshold $p = 0.30$				
	Precision	**Recall**	**f1**	**Sample**
Non-errors	0.96	0.99	0.97	124,501
Errors	0.53	0.29	0.38	6538
Accuracy			0.95	131,039
Macro average	0.75	0.64	0.68	131,039
Weighted average	0.94	0.95	0.94	131,039
Threshold $p = 0.35$				
	Precision	**Recall**	**f1**	**Sample**
Non-errors	0.96	0.99	0.98	124,501
Errors	0.59	0.26	0.36	6538
Accuracy			0.96	131,039
Macro average	0.77	0.62	0.67	131,039
Weighted average	0.94	0.95	0.94	131,039
Threshold $p = 0.40$				
	Precision	**Recall**	**f1**	**Sample**
Non-errors	0.96	0.99	0.98	124,501
Errors	0.65	0.23	0.33	6538
Accuracy			0.95	131,039
Macro average	0.80	0.61	0.66	131,039
Weighted average	0.94	0.95	0.94	131,039
Threshold $p = 0.45$				
	Precision	**Recall**	**f1**	**Sample**
Icke-fel	0.96	1.00	0.98	124,501
Fel	0.69	0.20	0.31	6538
Accuracy			0.95	131,039
Macro average	0.82	0.60	0.64	131,039
Weighted average	0.95	0.95	0.94	131,039
Threshold $p = 0.50$				
	Precision	**Recall**	**f1**	**Sample**
Non-errors	0.96	1.00	0.98	124,501
Errors	0.73	0.18	0.29	6538
Accuracy			0.96	131,039
Macro average	0.84	0.59	0.63	131,039
Weighted average	0.95	0.96	0.94	131,039

In the absence of a method that offers precision the internal auditors only have the option to prioritize by random sampling. They would have to randomly select the number of exceptions from the output of the CA system that they are able to follow up and all detected exceptions have the same likelihood of being selected for follow-up. This procedure is obviously less efficient than restricting the number of exceptions to those with the highest likelihood of being the true errors the auditors are looking for, with output levels illustrated in Table 4. The latter approach would minimize the internal auditors' follow-up of false alarms.

The control of the volume of detected exceptions by internal auditors is achieved by shifting the detection model's performance along the horizontal axis where threshold values are indicated from 0 to 0.5 in Figure 1. The higher the confidence required by the model, the more selective it must be in choosing what exceptions to output. As precision increases (blue curve) with increasing threshold values, the proportion

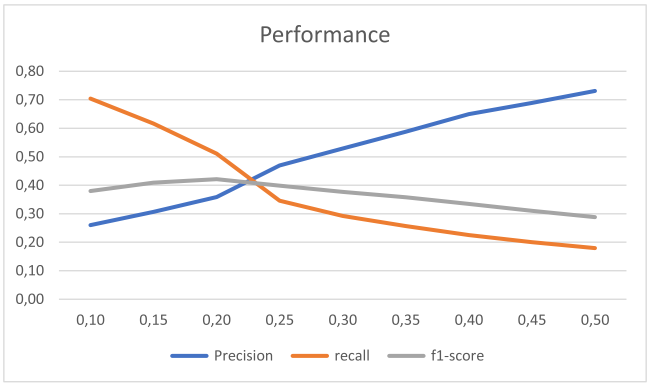

FIGURE 1 | The precision–recall trade-off and thresholding.

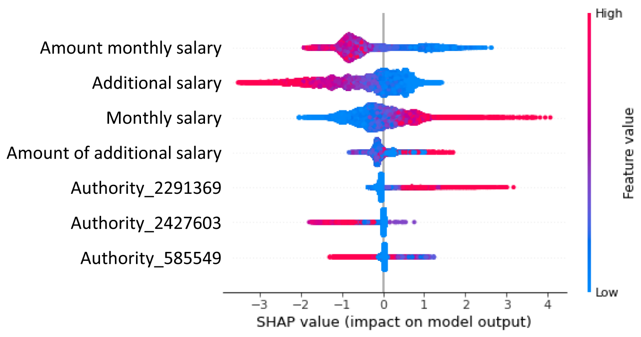

FIGURE 2 | A global SHAP-plot summary of how all employments were classified as exceptions or not.

of exceptions that the model correctly identifies (orange recall curve) decreases from the total 6513. The point at which the precision and recall measures meet is about 0.225 where the classifier offers a precision of about 0.4. This metric may be much too low if manual follow-up is considered necessary for all detected exceptions because most of the exceptions are false alarms at this low precision. It is clearly visible from Figure 1 however that an internal audit function can choose whatever combination of precision- and recall levels it finds appropriate, considering its resources and its cost benefit trade-off for manual follow-up.

Finally, internal auditors need additional information on detected exceptions to decide whether to follow up. In complex CA systems, if auditors see the output as a "black box" with no insight into why exceptions were flagged, they treat all exceptions equally. Even when classifiers provide likelihood estimates, errors vary in materiality or severity. Without detailed information, auditors are forced to choose exceptions blindly, leading to inefficient investigations. In multilayer rule-based CA systems, this often results in reliance on random sampling. Alternatively, the auditors may be able to address the exception overload problem by turning off entire groups of controls (rules) and thus ignore what the auditors think are the least severe exceptions (Alles et al. 2006). Fortunately, exception characterization is possible using metrics like SHAP values, which show how much each variable contributes to classifying a registry item as an exception. SHAP values can be calculated for all variables, such as employment data in a pension register, to indicate their importance in detecting errors. These values can be displayed in local SHAP plots for individual cases or global SHAP plots for an overview of all exceptions. A global plot for our data is shown in Figure 2.

The horizontal axis shows SHAP values for each variable's contribution to classifying employment contracts as exceptions. Each contract is represented by a colored dot (blue for low values, red for high). A positive SHAP value increases the likelihood of an exception, while a negative value decreases it. For example, employees of Authority_585549 are less likely to have errors in SPV's register, indicating reliable data reporting, whereas Authority_22913269 has more exceptions, signaling poor data quality. High monthly or additional salary increases the likelihood of classification as an exception. Local SHAP plots show contributions for individual data points, helping auditors prioritize investigations based on the likelihood and nature of the error. The SHAP values provide a "fingerprint" of each exception, aiding auditors in assessing which cases to focus on, much like risk-based audit planning.

Figure 2 gives an overview of the model's behavior across all data points, while a corresponding but local SHAP plot would show how each variable influences the classification of a single data point as correct or an error. This information helps auditors assess each exception. Using their experience, auditors can determine which cases need attention and which can be resolved through routine communication or ignored. In many CA systems, multiple SHAP values provide a detailed "fingerprint" of each exception, allowing auditors to prioritize investigations similarly to risk-based audit planning.

SHAP provides unique value in CA applications compared to other audit contexts for several specific reasons. First, CA systems generate continuous streams of exceptions where auditors need rapid interpretation capabilities—SHAP's real-time variable importance explanations enable immediate prioritization decisions without requiring extended analysis periods typical in traditional auditing. Second, unlike periodic audit applications, CA environments benefit from SHAP's ability to incorporate new variables into updated results seamlessly as business processes evolve, maintaining system relevance without complete redesign. Third, the high-volume, time-sensitive nature of exception handling in CA makes SHAP's automated interpretation particularly valuable compared to the manual analysis required in traditional audit contexts. Each exception receives an immediate "fingerprint" of contributing factors, enabling auditors to make informed decisions about investigation priorities within the operational timeframes required by continuous monitoring.

Furthermore, SHAP enables auditors to build institutional knowledge about error patterns over time, creating learning effects not available in point-in-time audit applications. As auditors observe recurring SHAP patterns associated with true errors, they develop enhanced judgment about which exceptions warrant immediate attention versus those that can be addressed through routine procedures. This cumulative learning aspect distinguishes CA applications from traditional auditing where interpretation insights are typically project-specific rather than building systematic organizational knowledge about error characteristics.

5 | Discussion and Conclusion

This study proposes using thresholding, a simple feature in many machine learning algorithms, to help internal auditors manage the number of exceptions detected in CA systems. Unlike methods described in the literature, this approach directly reduces exception overload. While auditors may sometimes use random sampling to reduce exceptions, this does not prioritize the most likely or serious errors and thus is not a true prioritization method. Overall, the literature presents three main strategies to make manual follow-up more manageable.

The common approach in the literature is to layer rule-based classifiers and adopt belief functions or Delphi weights to finally filter detected exceptions to a manageable volume. Perols and Murthy (2012) proposed a multilayered system starting with rule-based detection, followed by classifiers to reduce exceptions. However, this adds complexity, making the system opaque and costly to develop, especially with supervised learning requiring labeled data. Li et al. (2016) introduced an exception prioritization method using Dempster–Shafer belief functions, which assigns suspicion scores and iteratively ranks exceptions for manual classification, creating an active learning system.

Yoon et al. (2021) suggested data aggregation as an alternative, which simplifies data by grouping issues but risks losing crucial details, making the system appear efficient while potentially missing key errors.

A more general approach is to frame the issue as process mining rather than just exception detection. While process mining methods exist in machine learning literature, no evidence supports that they can solve the exception overload problem. Process mining is a control-focused CA design that does not quite address the same problem as transaction-level exception detection and therefore is not quite comparable. Furthermore, conceptual descriptions (Jans et al. 2011; Jans et al. 2013; Jans and Hosseinpour 2019) suggest easy implementation and some evidence of usefulness exists (Duan et al. 2024), but until proven effective, simpler solutions should be considered. One such option is thresholding to control the CA system outputs, which would require shifting from rule-based systems to other machine learning methods. The need for thresholding is evident because in extreme cases, researchers have even suggested turning off alarms to reduce exceptions (Alles et al. 2006, 2008).

In practice, however, auditors may have to rely on traditional approaches that auditors follow in addressing the problem of duplicate payments, which comprise either visually scanning the transactions or examining a sample. Either way, the auditors will not be able to obtain a higher precision than the proportion of items in the CA output that are true errors, which could be in the range of 7/6980 as in Rozario and Issa (2020), that is, not more than 0.1%, which means that manual follow-up needs to cover a lot of ground. With their solution the auditors would not know just by looking at the output which of the detected exceptions have the highest likelihood of being true errors rather than false alarms, so auditors are suggested by Rozario and Issa (2020) to manually investigate all 800 with the expected outcome of finding 4 out of the totally 7 true errors that existed in the population. That is a lot better than manually following up the 6980 but with thresholding they would clearly be better off. Because they would have to follow up only a small fraction of the 800 and, because their efforts can be more intelligently invested, find a higher fraction of the errors that exist in the population. It is the class imbalance that makes the internal auditors' job so difficult and therefore it is unintelligible why they should not use the commonly recognized method for dealing with class imbalance.

What would the internal auditors do with thresholding? They would still have a tough job but would not have to work blindfolded. First, they would adjust the number of detected exceptions in the same way they adjust their car radio. Second, they would determine the optimal output number as a trade-off between the effort they can put into manual follow-up and the benefit of finding a true error. The CA system with thresholding ensures that they always obtain the most likely true errors whatever volume they choose. With all previous techniques using rule-based classifiers, including belief functions, the auditors cannot without redesigning the system adjust output volume. Third, internal auditors not only obtain the detected exceptions that are the most likely of being true errors, but also, for each exception, they get the likelihood estimate that the exception is a true error. It is obvious that, if the estimates are correct, the auditors do not start with the ones having the lowest likelihood. They obviously take the ones with the highest likelihood first. No other method known to the literature can do this. Auditors can see immediately on the screen the risk they take by ignoring a certain exception. No other method offers such maneuverability. Fourth, by adding the feature importance estimate, unique to each exception, auditors can deviate from the likelihood estimates as they plan how they invest their efforts. A local SHAP plot tells the auditors which circumstances the CA system considered most crucial for identifying an item as an exception. The experienced auditors may determine the seriousness or materiality of a potential error by just looking at the SHAP plot, which is delivered in real time with each exception. As we see it, there is no other method in CA literature that offers the same level of interpretability of detected exceptions.

These aspects of thresholding and SHAP plots stand in stark contrast to traditional rule-based systems, which neither align well with thresholding nor offer an effective initial interpretation of exceptions. Designs like those of Li et al. (2016) require extensive customization based on auditors' expertise, making them difficult to generalize. In contrast, our approach, which leverages supervised learning with machine learning algorithms, offers off-the-shelf solutions to these fundamental challenges in CA systems.

A question regarding our approach concerns the trade-off between data-driven exception prioritization and expertise-based threshold setting. Domain expertise can inform threshold calibration, and under certain conditions, expertise-based approaches may outperform generalized methods. "General applicability" means the algorithmic framework—supervised learning with probability-based thresholding—can be deployed across contexts without organization-specific rule engineering for each implementation. The method provides a standardized technical approach, while learned patterns adapt to organization-specific characteristics through training data.

Probability-based thresholding provides advantages when (1) articulating complete detection rules is difficult, while providing labeled examples is feasible; (2) exception patterns involve complex interactions among many variables; (3) business processes evolve, requiring frequent recalibration; and (4) organizations need standardized methodology across multiple units. Expert-based thresholding applies when (1) exception patterns are simple and stable; (2) regulatory requirements demand explicit decision criteria; (3) organizations possess readily articulable domain knowledge; and (4) historical training data are insufficient (error rates $<0.5\%$, <1000 labeled examples). The approaches are complementary. Our method incorporates expertise through: feature engineering (experts identify relevant variables), threshold refinement (initial thresholds adjusted based on organizational context), exception interpretation (SHAP explanations enable expert validation), and hybrid architectures (rule-based filters combined with ML prioritization). Our contribution demonstrates that probability-based thresholding addresses documented limitations of rule-based designs—exception overload, calibration difficulties, and adaptation challenges—in transaction-level applications with severe class imbalance.

Our study faces several limitations, particularly in the application of supervised learning within the CA implementation. One notable challenge is the use of binary CA indicators, which, while useful in specific contexts, may lack universal applicability across all accounting processes, such as booking on CPD accounts in ERP systems. The binary classification approach simplifies decision-making but may oversimplify complex financial transactions, leading to a loss of nuance. Additionally, our explanation of these indicators may not fully capture the variability in how they can or should be applied to different systems. Future work should explore more flexible, multiclass models and refine the application of CA indicators to better account for the diverse nature of organizational systems, ensuring that supervised learning is adapted to varying accounting environments.

Data Availability Statement

The data that support the findings of this study are available from Participating company. Restrictions apply to the availability of these data, which were used under license for this study. Data are available from the author(s) with the permission of Participating company.

References

Abdullah, A. 2015. “The Application of Data Visualization in Auditing.” In *Graduate School-Newark Rutgers, the State University of New Jersey*. Rutgers University.

Alghushairy, O., R. Alsini, T. Soule, and X. Ma. 2021. “A Review of Local Outlier Factor Algorithms for Outlier Detection in Big Data Streams.” *Big Data and Cognitive Computing* 5, no. 1: 1–24. https://doi.org/10.3390/BDCC5010001.

Alles, M., A. Kogan, and M. A. Vasarhelyi. 2004. “Principles of Analytic Monitoring for Continuous Assurance.” *Journal of Emerging Technologies in Accounting* 1, no. 1: 1–21. https://doi.org/10.2308/jeta.2004.1.1.1.

Alles, M. G., A. Kogan, and M. A. Vasarhelyi. 2008. “Putting Continuous Auditing Theory Into Practice: Lessons From Two Pilot Implementations.” *Journal of Information Systems* 22, no. 2: 195–214. https://doi.org/10.2308/JIS.2008.22.2.195.

Alles, M., G. Brennan, A. Kogan, and M. A. Vasarhelyi. 2006. “Continuous Monitoring of Business Process Controls: A Pilot Implementation of a Continuous Auditing System at Siemens.” *International Journal of Accounting Information Systems* 7, no. 2: 137–161. https://doi.org/10.1016/j.accinf.2005.10.004.

Alpaydin, E. 2010. *Introduction to Machine Learning*. Second ed. MIT Press.

Brown, C. E., J. A. Wong, and A. A. Baldwin. 2007. “A Review and Analysis of the Existing Research Streams in Continuous Auditing.” *Journal of Emerging Technologies in Accounting* 4, no. 1: 1–28. https://doi.org/10.2308/JETA.2007.4.1.1.

Byrnes, P. E., and D. Mcquilken. 2012. *The Current State of Continuous Auditing and Continuous Monitoring*. Aicpa October.

Caroline, C., and G. Thomas. 2021. “An Outlier Detection Approach on Credit Card Fraud Detection Using Machine Learning: A Comparative Analysis on Supervised and Unsupervised Learning.” *Advances in Intelligent Systems and Computing* 1167: 125–135. https://doi.org/10.1007/978-981-15-5285-4_12.

Chan, D. Y., and M. A. Vasarhelyi. 2011. “Innovation and Practice of Continuous Auditing.” *International Journal of Accounting Information Systems* 12, no. 2: 152–160. https://doi.org/10.1016/J.ACCINF.2011.01.00110.1016/J.ACCINF.2011.01.001.

Chawla, N. V., N. Japkowicz, and A. Kotcz. 2004. “Editorial.” *ACM SIGKDD Explorations Newsletter* 6, no. 1: 1–6. https://doi.org/10.1145/1007730.1007733.

Chen, J. J., C. A. Tsai, H. Moon, H. Ahn, J. J. Young, and C. H. Chen. 2006. “Decision Threshold Adjustment in Class Prediction.” *SAR and QSAR in Environmental Research* 17, no. 3: 337–352. https://doi.org/10.1080/10659360600787700.

Chen, Y., and R. A. Leitch. 1999. “An Analysis of the Relative Power Characteristics of Analytical Procedures.” *Auditing* 18, no. 2: 35–69. https://doi.org/10.2308/AUD.1999.18.2.35.

Cleofas-Sánchez, L., V. García, A. I. Marqués, and J. S. Sánchez. 2016. “Financial Distress Prediction Using the Hybrid Associative Memory With Translation.” *Applied Soft Computing Journal* 44: 144–152. https://doi.org/10.1016/j.asoc.2016.04.005.

Ding, K., X. Peng, and Y. Wang. 2019. “A Machine Learning-Based Peer Selection Method With Financial Ratios.” *Accounting Horizons* 33, no. 3: 75–87. https://doi.org/10.2308/ACCH-52454.

Duan, H. K., M. A. Vasarhelyi, and M. Codesso. 2024. “Integrating Process Mining and Machine Learning for Advanced Internal Control Evaluation in Auditing.” *Journal of Information Systems* 1–21: 55–75. https://doi.org/10.2308/ISYS-2022-028.

Duda, R. O., P. E. Hart, and D. G. Stork. 2001. *Pattern Classification*. John Wiley.

El-Masry, E. H., and J. L. Reck. 2008. “Continuous Online Auditing as a Response to the Sarbanes-Oxley Act.” *Managerial Auditing Journal* 23, no. 8: 779–802. https://doi.org/10.1108/02686900810899527.

Eulerich, M., C. Georgi, and A. Schmidt. 2020. “Continuous Auditing and Risk-Based Audit Planning—An Empirical Analysis.” *Journal of Emerging Technologies in Accounting* 17, no. 2: 141–155. https://doi.org/10.2308/JETA-2020-004.

Eulerich, M., and A. Kalinichenko. 2018. “The Current State and Future Directions of Continuous Auditing Research: An Analysis of the Existing Literature.” *Journal of Information Systems* 32, no. 3: 31–51. https://doi.org/10.2308/ISYS-51813.

Freiman, J. W., Y. Kim, and M. A. Vasarhelyi. 2022. “Full Population Testing: Applying Multidimensional Audit Data Sampling (MADS) to General Ledger Data Auditing.” *International Journal of Accounting*

Information Systems 46: 100573. https://doi.org/10.1016/J.ACCINF.2022.100573.

Gerlein, E. A., M. McGinnity, A. Belatreche, and S. Coleman. 2016. "Evaluating Machine Learning Classification for Financial Trading: An Empirical Approach." *Expert Systems With Applications* 54: 193–207. https://doi.org/10.1016/j.eswa.2016.01.018.

Gonzalez, G. C., P. N. Sharma, and D. Galletta. 2012. "Factors Influencing the Planned Adoption of Continuous Monitoring Technology." *Journal of Information Systems* 26, no. 2: 53–69. https://doi.org/10.2308/ISYS-5025910.2308/ISYS-50259.

Groomer, S. M., and U. S. Murthy. 1989. "Continuous Auditing of Database Applications: An Embedded Audit Module Approach." *Journal of Information Systems* 3, no. 2: 53–69.

Guyon, I. 1997. "A Scaling Law for the Validation-Set Training-Set Size Ratio." AT&T Bell Laboratories, 1–11. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.33.1337&rep=rep1&type=pdf.

Hall, T. W., J. E. Hunton, and B. J. Pierce. 2000. "The Use of and Selection Biases Associated With Nonstatistical Sampling in Auditing." *Behavioral Research in Accounting* 12: 231–255.

Hassan, A., N. Salleh, M. N. Ismail, M. N. Ahmad, and A. R. C. Hussin. 2023. "Empirical Evaluation of Continuous Auditing System Use: A Systematic Review." *International Journal of Electrical and Computer Engineering (IJECE)* 13, no. 1: 796–808. https://doi.org/10.11591/ijece.v13i1.pp796-808.

Hayes-Roth, F., D. A. Waterman, and D. B. Lenat. 1983. *Building Expert Systems*. Addison-Wesley Publishing Company.

Hoitash, R., A. Kogan, and M. A. Vasarhelyi. 2006. "Peer-Based Approach for Analytical Procedures." *Auditing* 25, no. 2: 53–84. https://doi.org/10.2308/AUD.2006.25.2.53.

Iselin, E. R. 1988. "The Effects of Information Load and Information Diversity on Decision Quality in a Structured Decision Task." *Accounting, Organizations and Society* 13, no. 2: 147–164. https://doi.org/10.1016/0361-3682(88)90041-4.

Issa, H. 2013. "Exceptional Exceptions."

Issa, H., and A. Kogan. 2014. "A Predictive Ordered Logistic Regression Model as a Tool for Quality Review of Control Risk Assessments." *Journal of Information Systems* 28, no. 2: 209–229. https://doi.org/10.2308/ISYS-50808.

Jans, M., and M. Hosseinpour. 2019. "How Active Learning and Process Mining Can Act as Continuous Auditing Catalyst." *International Journal of Accounting Information Systems* 32: 44–58. https://doi.org/10.1016/j.accinf.2018.11.002.

Jans, M., J. M. Van Der Werf, N. Lybaert, and K. Vanhoof. 2011. "Abusiness Process Mining Application for Internal Transaction Fraud Mitigation." *Expert Systems with Applications* 38, no. 10: 13351–13359. https://doi.org/10.1016/j.eswa.2011.04.159.

Jans, M., M. Alles, and M. Vasarhelyi. 2013. "The Case for Process Mining in Auditing: Sources of Value Added and Areas of Application." *International Journal of Accounting Information Systems* 14, no. 1: 1–20. https://doi.org/10.1016/J.ACCINF.2012.06.015.

Johnson, J. M., and T. M. Khoshgoftaar. 2019. "Survey on Deep Learning With Class Imbalance." *Journal of Big Data* 6, no. 1: 1–54. https://doi.org/10.1186/S40537-019-0192-5.

Kim, Y., and M. A. Vasarhelyi. 2012. "A Model to Detect Potentially Fraudulent/Abnormal Wires of an Insurance Company: An Unsupervised Rule-Based Approach." *Journal of Emerging Technologies in Accounting* 9, no. 1: 95–110. https://doi.org/10.2308/jeta-50411.

Kleinmuntz, B. 1990. "Why We Still Use Our Heads Instead of Formulas: Toward an Integrative Approach." *Psychological Bulletin* 107, no. 3: 296–310. https://doi.org/10.1037/0033-2909.107.3.296.

Kogan, A., E. F. Sudit, and M. A. Vasarhelyi. 1999. "Continuous Online Auditing: A Program of Research." *Journal of Information Systems* 13, no. 2: 87–103. https://doi.org/10.2308/JIS.1999.13.2.8710.2308/JIS.1999.13.2.87.

Krambia-Kapardis, M., C. Christodoulou, and M. Agathocleous. 2010. "Neural Networks: The Panacea in Fraud Detection?" *Managerial Auditing Journal* 25, no. 7: 659–678. https://doi.org/10.1108/02686901011061342.

Le, L. T., H. Nguyen, J. Zhou, J. Dou, and H. Moayedi. 2019. "Estimating the Heating Load of Buildings for Smart City Planning Using a Novel Artificial Intelligence Technique PSO-XGBoost." *Applied Sciences* 9, no. 13: 2714. https://doi.org/10.3390/app9132714.

Leitch, R. A., and Y. Chen. 2003. "The Effectiveness of Expectation Models in Recognizing Error Patterns and Generating and Eliminating Hypotheses While Conducting Analytical Procedures." *Auditing* 22, no. 2: 147–170. https://doi.org/10.2308/AUD.2003.22.2.147.

Li, P., D. Y. Chan, and A. Kogan. 2016. "Exception Prioritization in the Continuous Auditing Environment: A Framework and Experimental Evaluation." *Journal of Information Systems* 30, no. 2: 135–157. https://doi.org/10.2308/ISYS-51220.

Lundberg, S. M., G. Erion, H. Chen, et al. 2020. "From Local Explanations to Global Understanding With Explainable AI for Trees." *Nature Machine Intelligence* 2, no. 1: 56–67. https://doi.org/10.1038/S42256-019-0138-9.

Lundberg, S. M., and S. I. Lee. 2017. "A Unified Approach to Interpreting Model Predictions." Advances in Neural Information Processing Systems, 2017-December, 4766–4775.

Malaescu, I., and S. G. Sutton. 2015. "The Reliance of External Auditors on Internal Audit's Use of Continuous Audit." *Journal of Information Systems* 29, no. 1: 95–114. https://doi.org/10.2308/ISYS-50899.

Murthy, U. S. 2004. "An Analysis of the Effects of Continuous Monitoring Controls on e-Commerce System Performance." *Journal of Information Systems* 18, no. 2: 29–47. https://doi.org/10.2308/JIS.2004.18.2.2910.2308/JIS.2004.18.2.29.

Ngai, E. W. T., Y. Hu, Y. H. Wong, Y. Chen, and X. Sun. 2011. "The Application of Data Mining Techniques in Financial Fraud Detection: A Classification Framework and an Academic Review of Literature." *Decision Support Systems* 50, no. 3: 559–569.

No, W. G., K. Lee, F. Huang, and Q. Li. 2019. "Multidimensional Audit Data Selection (MADS): A Framework for Using Data Analytics in the Audit Data Selection Process." *Accounting Horizons* 33, no. 3: 127–140. https://doi.org/10.2308/ACCH-52453.

Pedregosa, F., V. Michel, O. Grisel, et al. 2011. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research* 12, no. 85: 2825–2830. http://scikit-learn.sourceforge.net.

Perols, J. L., R. M. Bowen, C. Zimmermann, and B. Samba. 2017. "Finding Needles in a Haystack: Using Data Analytics to Improve Fraud Prediction." *Accounting Review* 92, no. 2: 221–245. https://doi.org/10.2308/ACCR-51562.

Perols, J. L., and U. S. Murthy. 2012. "Information Fusion in Continuous Assurance." *Journal of Information Systems* 26, no. 2: 35–52. https://doi.org/10.2308/ISYS-50216.

Qiu, Y., J. Zhou, M. Khandelwal, H. Yang, P. Yang, and C. Li. 2022. "Performance Evaluation of Hybrid WOA-XGBoost, GWO-XGBoost and BO-XGBoost Models to Predict Blast-Induced Ground Vibration." *Engineering With Computers* 38, no. 5: 4145–4162. https://doi.org/10.1007/S00366-021-01393-9/FIGURES/11.

Rezaee, Z., A. Sharbatoghlie, R. Elam, and P. L. McMickle. 2002. "Continuous Auditing: Building Automated Auditing Capability." *Auditing* 21, no. 1: 147–163. https://doi.org/10.2308/aud.2002.21.1.147.

Rikhardsson, P., and R. Dull. 2016. "An Exploratory Study of the Adoption, Application and Impacts of Continuous Auditing

Technologies in Small Businesses." *International Journal of Accounting Information Systems* 20: 26–37. https://doi.org/10.1016/J.ACCINF.2016.01.003.

Rozario, A. M., and H. Issa. 2020. "Risk-Based Data Analytics in the Government Sector: A Case Study for a U.S. County." *Government Information Quarterly* 37, no. 2: 101457. https://doi.org/10.1016/J.GIQ.2020.101457.

Sibiya, M., and M. Sumbwanyambe. 2021. "Automatic Fuzzy Logic-Based Maize Common Rust Disease Severity Predictions With Thresholding and Deep Learning." *Pathogens* 10, no. 2: 131. https://doi.org/10.3390/PATHOGENS10020131.

Sigrist, F., and C. Hirnschall. 2019. "Grabit: Gradient Tree-Boosted Tobit Models for Default Prediction." *Journal of Banking and Finance* 102: 177–192. https://doi.org/10.1016/j.jbankfin.2019.03.004.

Teitlebaum, A. D., and C. F. Robinson. 1975. "The Real Risks in Audit Sampling." *Journal of Accounting Research* 13: 70. https://doi.org/10.2307/2490480.

Thiprungsri, S., and M. Vasarhelyi. 2011. "Cluster Analysis for Anomaly Detection in Accounting Data: An Audit Approach." *International Journal of Digital Accounting Research* 11: 69–84. https://doi.org/10.4192/1577-8517-V11_410.4192/1577-8517-V11_4.

Vasarhelyi, M. A., M. Alles, S. Kuenkaikaew, and J. Littley. 2012. "The Acceptance and Adoption of Continuous Auditing by Internal Auditors: A Micro Analysis." *International Journal of Accounting Information Systems* 13, no. 3: 267–281. https://doi.org/10.1016/J.ACCINF.2012.06.01110.1016/J.ACCINF.2012.06.011.

Vasarhelyi, M. A., and F. B. Halper. 1991. "The Continuous Audit of Online Systems." *Auditing: A Journal of Practice & Theory* 10, no. 1: 110–125.

Wang, Y., and A. Kogan. 2020. "Cloud-Based In-Memory Columnar Database Architecture for Continuous Audit Analytics." *Journal of Information Systems* 34, no. 2: 87–107. https://doi.org/10.2308/ISYS-52531.

Wei, D., S. Cho, M. A. Vasarhelyi, and L. Te-Wierik. 2024. "Outlier Detection in Auditing: Integrating Unsupervised Learning Within a Multilevel Framework for General Ledger Analysis." *Journal of Information Systems* 38, no. 2: 123–142. https://doi.org/10.2308/ISYS-2022-026.

Weiss, G. M., and F. Provost. 2001. The Effect of Class Distribution on Classifier Learning: An Empirical Study. Technical Report ML-TR-44, Department of Computer Science, Rutgers University.

Werner, M., M. Wiese, and A. Maas. 2021. "Embedding Process Mining Into Financial Statement Audits." *International Journal of Accounting Information Systems* 41: 100514. https://doi.org/10.1016/j.accinf.2021.100514.

West, J., and M. Bhattacharya. 2016. "Intelligent Financial Fraud Detection: A Comprehensive Review." *Computers & Security* 57: 47–66. https://doi.org/10.1016/j.cose.2015.09.005.

Woodroof, J., and D. Searcy. 2001. "Continuous Audit." *International Journal of Accounting Information Systems* 2, no. 3: 169–191. https://doi.org/10.1016/S1467-0895(01)00019-710.1016/S1467-0895(01)00019-7.

Yoon, K., Y. Liu, T. Chiu, and M. A. Vasarhelyi. 2021. "Design and Evaluation of an Advanced Continuous Data Level Auditing System: A Three-Layer Structure." *International Journal of Accounting Information Systems* 42: 100524. https://doi.org/10.1016/J.ACCINF.2021.100524.

Zhang, X., H. Nguyen, X. N. Bui, et al. 2020. "Novel Soft Computing Model for Predicting Blast-Induced Ground Vibration in Open-Pit Mines Based on Particle Swarm Optimization and XGBoost." *Natural Resources Research* 29, no. 2: 711–721. https://doi.org/10.1007/s11053-019-09492-7.

Zhaokai, Y., and K. C. Moffitt. 2019. "Contract Analytics in Auditing." *Accounting Horizons* 33, no. 3: 111–126. https://doi.org/10.2308/ACCH-52457.

Zhou, J., Y. Qiu, S. Zhu, D. J. Armaghani, M. Khandelwal, and E. T. Mohamad. 2020. "Estimation of the TBM Advance Rate Under Hard Rock Conditions Using XGBoost and Bayesian Optimization." *Underground Space* 6, no. 5: 506–515. https://doi.org/10.1016/j.undsp.2020.05.008.

Zou, Q., S. Xie, Z. Lin, M. Wu, and Y. Ju. 2016. "Finding the Best Classification Threshold in Imbalanced Classification." *Big Data Research* 5: 2–8. https://doi.org/10.1016/J.BDR.2015.12.001.