**Isolation Forest Algorithm**

**Emanuel-Vasile Puțura**

Abstract

This paper conducts a thorough examination of the Isolation Forest algorithm, a fundamental tool in identifying outliers within data analysis. Outliers, also known as anomalies, represent data points that significantly differ from the majority of the dataset. This study starts by presenting the significance of anomaly detection in data analysis and situating the Isolation Forest algorithm within the broader context of this field. It subsequently explores the theoretical foundations of the algorithm, providing an exhaustive analysis that encompasses its original iteration, state-of-the-art advancements, and various variations, including those specialized for handling data streams. This study further presents some applications and key limitations of the algorithm. Finally, the paper outlines the main findings and insights about the Isolation Forest algorithm, showcasing the importance of this method in the realm of data analysis and particularly anomaly detection.

1. Introduction

Anomaly detection, a popular topic in data analysis, studies the problem of finding irregular patterns in data that deviate from anticipated behavior [1]. These deviations vary in terminology across different application domains, with the terms anomalies and outliers being the most frequently used, particularly in the context of anomaly detection. Among the methods used for this problem, the Isolation Forest algorithm stands out as one of the most popular algorithms, that operates on the principle of isolating anomalies rather than explicitly modeling normal behavior, as many other algorithms do.

Introduced by Liu et al. in 2008 [2], the Isolation Forest algorithm is known for its efficiency and effectiveness in identifying outliers in large datasets. It uses binary trees to detect anomalies, resulting in a linear time complexity and low memory usage, which makes it well-suited for processing large datasets.

This paper delves into an in-depth exploration of the Isolation Forest algorithm, aiming to comprehensively study its principles, recent advancements, state-of-the-art applications, and variants. As a pivotal player in the realm of anomaly detection, the Isolation Forest algorithm has garnered considerable attention for its efficacy and versatility. Anomalies, or deviations from expected patterns, pose a significant challenge in diverse fields, ranging from cybersecurity [2] to finance [3] and beyond. Consequently, understanding the intricacies of the Isolation Forest algorithm becomes crucial for researchers, practitioners, and enthusiasts seeking robust solutions to anomaly detection.

The paper has been organized into six distinct sections, including the introduction and the concluding section, each meticulously designed to provide an exhaustive exploration and in-depth examination of the isolation forest algorithm within the realm of data analysis, particularly anomaly detection methods. Through the composition of this study, our examination actively contributes to the ever-changing field of outlier detection, facilitating the emergence of more resilient and finely nuanced methodologies for identifying anomalies.

The second section aims to integrate the approached topic, i.e., the Isolation Forest algorithm, into the data analysis field and particularly into the anomaly detection methods field. It provides an overview of the anomaly detection problem and how isolation forest algorithm aligns with and contributes to the resolution of this problem. Also, this section provides a very brief comparison of the Isolation Forest algorithm with other popular anomaly detection methods, such as Local Outlier Factor, and One-Class SVM (Support Vector Machine).

The third section serves as a detailed exploration of the algorithm itself, starting with its initial version [4], progressing to encompass the contemporary state-of-the-art iterations, and delving into various adaptations of the algorithm, including some versions specialized in handling data streams. By presenting this comprehensive overview of the algorithm, we afford readers the opportunity to grasp the historical challenges and evolutions that have shaped it, as well as the current potential limitations and improvements that persist to this day.

Subsequently, the following sections delve into the practical applications of the isolation forest algorithm across diverse industries, including but not limited to cybersecurity, finance, and healthcare. These segments expound upon the algorithm's efficacy and utility in real-world scenarios within these specific domains.

Concomitantly, a dedicated section is allocated to an exhaustive discussion of the inherent limitations associated with the isolation forest algorithm. This critical analysis sheds light on the algorithm's constraints and potential shortcomings, offering a balanced perspective on its applicability and efficacy in certain contexts.

1. Isolation Forest in the Context of Anomaly Detection Methods

In the realm of anomaly detection, outliers play a pivotal role as key indicators of deviations from expected patterns within a dataset. Outliers, also known as anomalies, represent data points that significantly differ from the majority of the data, exhibiting behaviors that are atypical or unexpected. These anomalies can arise due to various factors such as errors, novel events, or genuine irregularities within the underlying system. Identifying outliers is crucial for anomaly detection systems, as these anomalous instances often hold valuable information or signal potential issues that merit closer examination. Anomaly detection is essential in situations where normal patterns are well understood, and deviations from these norms may signify potential issues or noteworthy events. Outliers may manifest in diverse forms, ranging from data points with extreme values to patterns that defy the general trends observed in the dataset. The challenge in anomaly detection lies in distinguishing between legitimate outliers that signify meaningful information and those that result from noise or errors in the data.

The presence of outliers can have significant implications across numerous domains, including finance, healthcare, cybersecurity, and industrial processes. In financial systems, for instance, the identification of unusual trading activities or transactions can signal fraudulent behavior. In healthcare, outliers in patient data may indicate rare medical conditions or adverse reactions to treatments. In environmental monitoring, outliers are used to identify unusual events or changes in environmental parameters, which is crucial for early detection of natural disasters or environmental hazards. Thus, the accurate detection and interpretation of outliers are essential for maintaining the integrity and reliability of systems in various applications.

Various statistical and machine learning techniques are employed to detect outliers, with algorithms such as the Isolation Forest, Local Outlier Factor, and One-Class SVM (Support Vector Machine) being commonly utilized. These methods aim to differentiate between normal and anomalous behavior, providing a foundation for robust anomaly detection systems that can effectively manage the complexities introduced by outliers in diverse datasets. Comparing these methods, Isolation Forest is advantageous in terms of computational efficiency and scalability, making it suitable for large datasets. Local Outlier Factor excels in capturing local anomalies within varying density clusters, offering flexibility in scenarios where anomalies may not be uniformly distributed. One-Class SVM, while powerful, demands careful parameter tuning and might be more suitable for scenarios where anomalies are well-separated in the feature space. The choice between these methods ultimately depends on the characteristics of the data and the specific requirements of the anomaly detection task at hand. In this paper, we aim to provide an overview of the Isolation Forest algorithm.

1. The Isolation Forest Algorithm

The Isolation Forest algorithm is a machine learning anomaly detection technique designed to efficiently identify outliers or anomalies within a dataset. Introduced by Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou in 2008 in [2], the algorithm operates on the principle that anomalies are often easier to isolate than normal instances in high-dimensional spaces. Its ability to efficiently handle large datasets and high-dimensional spaces makes it particularly suitable for applications such as fraud detection, network security, and intrusion detection.

* 1. Overview

The key idea behind Isolation Forests is to isolate anomalies by randomly selecting features and creating isolation trees. An isolation tree is constructed by recursively partitioning the data until each data point is isolated in its own leaf node. The rationale is that anomalies are typically isolated more quickly than normal instances during this random partitioning process. As a result, the path length from the root of the tree to the point of isolation serves as a measure of anomaly-ness. Shorter paths indicate anomalies, while longer paths correspond to normal instances.

The aggregation of multiple isolation trees forms an Isolation Forest. Anomalies will have shorter average path lengths across the trees, making them stand out when compared to normal instances. During the anomaly detection phase, a decision function is applied to determine the anomaly score for each data point based on the average path length. A lower average path length is indicative of a higher anomaly score.

Isolation Forests exhibit several advantages, including their ability to efficiently handle high-dimensional data and their relative insensitivity to the scaling of features. Moreover, they require minimal hyperparameter tuning, making them easy to use and deploy. This algorithm has found applications in various domains such as cybersecurity, fraud detection, and industrial quality control, where the identification of anomalies is crucial for maintaining system integrity and security.

* 1. Initial Version of the Algorithm

The first version of the Isolation Forest algorithm that was proposed is the IForest method [4]. This approach employs a collection of random and independent trees, referred to as itrees, forming a so called random forest. To compute a score for each data point, IForest utilizes all the trees within the forest. The calculation of the data score involves two input parameters: *ψ*, representing the size of the randomly selected sample from the entire dataset, and *t*, denoting the number of trees in the forest. Each tree is constructed independently through dataset sampling, with the number of trees corresponding to the number of samples.

IForest consists of two distinct stages, i.e., the training phase, which essentially entails the construction of the forest, and the scoring phase, where a score is generated for each item in the dataset. Let *n* denote the number of datapoints in a dataset *X(n, m) ⊂ Rm* and *m* the dimension of the data.

* + 1. Training Phase

The primary goal during this phase of the algorithm is to construct a forest comprised of random and independent itrees. IForest starts by randomly selecting a subset of size *ψ* from the data, without replacement, forming the sample used for building an individual tree. This step serves as the cornerstone for the efficacy and efficiency of IForest, having a pivotal importance. Given the random selection of data and the assumption that anomalous data are significantly outnumbered by normal data, a sample may contain predominantly normal data or a combination of mostly normal data.

The isolation tree, i.e., the itree, takes the form of a binary tree. In the beginning, the root node encompasses all sample data. As the tree is built, each internal node undergoes a split into two subnodes until achieving complete data isolation or reaching a maximum tree depth of *log2(ψ)*. Data is considered isolated when residing alone in its node. To split a node *i*, IForest randomly selects a dimension *di*, and subsequently, the split value *vi* is chosen randomly within the range of minimum to maximum values for the data within the addressed node, specifically for dimension *di*. The elements of node *i* are then partitioned into two groups by comparing their values to *vi*.

The two steps of sampling and tree construction are iterated *t* times, which leads to constructing the *t* trees in the forest. Consequently, each tree possesses its dedicated sample, with the sample for each itree drawn from the entire dataset. The complexity of the training phase is expressed as *O(t ψ log ψ)*, as every item among the *ψ* subsets across the *t* trees must undergo isolation within the associated tree. The parameter *t*, representing the number of trees, emerges as a critical factor influencing the performance of IForest.

* + 1. Scoring Phase

In this phase, the score of each item in the dataset, i.e., the degree of similarity between the respective item and the others, is computed. During the computation of this score, an item *x* is processed by each tree within the forest. Eventually, *x* is invariably situated in an external node of each tree, contingent upon the established split criteria. The count of nodes traversed by *x* from the root node to its ultimate external node is denoted as *h(x)*. Following *x*'s traversal through all the trees in the forest, IForest proceeds to determine the average length of the *t* paths of *x*, denoted as *E(h(x))*. Let

*H(i) = ln(i) + 0.5772156649 (Euler constant)*

be a harmonic value and consider

*C(n) = 2 H(n – 1) – 2*

which is the average length of the paths of an unsuccessful search in a binary search tree. Then, the score *s(x, n)* for *x* is given by the following formula, as presented in [4]:

*s(x, n) = 2-*

Being given this score formula, we identify several distinct cases:

* Case 1: *E(h(x)) → C(n), s → 0.5*. When the score is ~0.5 for all the instances, there are no outliers in the dataset.
* Case 2: *E(h(x)) → 0, s → 1*. Any *x* that has a score that is very close to 1 is an outlier.
* Case 3: *E(h(x)) → n - 1, s → 0*. Any *x* having a score that is much lower than 0.5 is normal data.

If *E(h(x))* is short, it suggests that the ensemble of *t* trees collectively categorizes *x* as an anomaly. The computational complexity of the scoring phase is expressed as *O(n t log ψ)*, as each item within the dataset of size *n* undergoes processing by the *t* trees in the forest. This processing involves evaluating its path length, subsequently determining its average depth, and ultimately generating its score. Therefore, IForest exhibits linear complexity, directly proportional to the dataset size *n*, with *t* and *ψ* serving as constant input parameters.

IForest grapples with the challenges posed by swamping and masking, demonstrating resilience up to a certain threshold. Swamping occurs when normal data are misclassified as anomalies due to their similarity. In scenarios where normal data closely resemble anomalies, an increased number of partitions is necessary to distinguish between the two, rendering the task of differentiating anomalies from normal data more challenging. Masking, on the other hand, arises from the presence of numerous similar anomalies, making it difficult to isolate them individually. The difficulty intensifies when a large and dense cluster of anomalies requires an escalating number of partitions for individual isolation. Consequently, such anomalies may be mistakenly identified as normal data. IForest mitigates these challenges by relying on random samples rather than the entire dataset to construct the forest of random trees. The sampling strategy allows the consideration of data with lower density compared to the full dataset, facilitating a more effective separation between anomalies and normal data, as well as among anomalies themselves. Additionally, each tree in the forest is constructed with its unique sample, implying that trees may not necessarily isolate the same anomalies, and some samples may not even contain any anomalies. As a result, IForest exhibits robustness against the adverse effects of swamping and masking.

* 1. Variations and State-of-the-art Improvements of the Algorithm

Since the IForest algorithm was initially proposed in 2008, many limitations of this method have been discovered and multiple improvements have been proposed.

One such example is the SciForest method, introduced in 2010 in [5], which, akin to distance and density-based approaches, is designed to identify clusters within data. SCiForest represents an advancement of IForest, specifically tailored for clustering to detect local anomalies. The method randomly selects a hyperplane to partition the data within a node, enabling the consideration of anomalies associated with multiple attributes simultaneously. To delineate distinct clusters, SCiForest establishes a split criterion for each node that takes into consideration the standard deviation of the data within that node. Despite its adaptability to complex data, it is worth noting that SCiForest is burdened by a high computational complexity, serving as a notable drawback for this method.

Another example is the Simulated Annealing IForest (SA-IForest), introduced in 2017 in [6], that represents an enhancement to the original IForest algorithm, aiming to create a more robust and stable forest during the training phase. SA-IForest achieves this by initially constructing a forest using the IForest approach. Subsequently, it assesses the similarity among the trees within the forest. Precision metrics for the trees are determined through cross-validation. Following this process, only the trees exhibiting high precision are preserved. It is important to note that the validation step necessitates the labeling of the training data, which can pose challenges and serves as a limitation to this semi-supervised method.

The Probabilistic Generalization of Isolation Forest (PGIF) is introduced in [9] as an intuitively appealing and efficient enhancement of the original Isolation Forest approach. The proposed generalization is grounded in the nonlinear dependence of segment-cumulated probability on the segment's length. By incorporating this generalization, the method achieves more effective splits, specifically between clusters—regions characterized by dense formations of datapoints, rather than within them. Through a comprehensive series of experiments, it is demonstrated that the PGIF method enhances the detection of anomalies concealed between clusters. Furthermore, the study illustrates that this approach significantly improves anomaly detection quality across both artificial and real datasets. Importantly, the time complexity of the method remains comparable to the original approach, as the generalization specifically pertains to the tree-building process, while the scoring procedure, which consumes the majority of the processing time, remains unchanged.

* 1. Data Stream Based Variations of the Isolation Forest Algorithm

A limitation of the original Isolation Forest algorithm and of its variants presented above is that they are not able to handle data streams, i.e., data being volatile and received continuously.

The significance of an Isolation Forest algorithm's capability to handle data streams lies in its relevance to real-time anomaly detection and adaptability to dynamic environments. In scenarios where data is continuously streaming, such as in IoT (Internet of Things) applications or financial transactions, the ability to detect anomalies in real-time is crucial. An Isolation Forest algorithm tailored for data streams ensures that anomalies can be identified promptly as they occur, facilitating timely intervention or response to potential issues.

Moreover, data streams often exhibit dynamic patterns, where the underlying characteristics of the data may change over time. This phenomenon is known as concept drift. Isolation Forest algorithms adapted for data streams are designed to cope with concept drift, allowing them to adjust and remain effective in identifying anomalies even as the data patterns evolve. This adaptability is essential for maintaining the accuracy of anomaly detection models in environments where the normal behavior of the system may change.

Another key advantage is resource efficiency. Traditional machine learning models might necessitate retraining on the entire dataset when faced with changes in data distribution. Isolation Forest algorithms for data streams, however, often employ techniques like sliding windows. This approach enables them to adapt to changes efficiently without requiring a full retraining of the model, thereby conserving computational resources. Additionally, the scalability of Isolation Forest algorithms makes them well-suited for handling large volumes of streaming data efficiently.

The early detection of anomalies is facilitated by the real-time nature of Isolation Forest algorithms for data streams. This early detection is critical in applications where prompt action is necessary to prevent or mitigate issues before they escalate. Furthermore, some adaptations of Isolation Forest algorithms for data streams, such as sliding window techniques, contribute to reduced memory requirements by focusing on recent data, making them more resource-efficient compared to models that consider the entire historical dataset. Isolation Forest variants such as the Isolation Forest Algorithm for Streaming Data (IForest ASD) [7], and Half-Space Trees (HSTrees) [8], represent adaptations of IForest to the realm of data streams.

IForest ASD employs a sliding window technique to retrieve data, executing the IForest method within each window to detect anomalies based on a model previously established with data from the preceding windows. In the event of drift, this model undergoes reinitialization. HSTrees, an advancement of IForest tailored for streaming, employs node splitting based on the average of node items for randomly selected attributes. Unlike IForest ASD, HSTrees automatically manages concept drift without updating its model through reinitialization. Notably, HSTrees is faster than IForest ASD and constructs its model independently of the specific dataset. Conversely, IForest ASD is closely tied to the dataset. To handle concept drift, IForest ASD maintains an input value *μ* representing the anomaly rate. When, within a given window, the anomaly rate surpasses *μ*, IForest ASD assumes a change in the normal data behavior (drift) and updates the model with data from the current window. This update involves discarding the existing model and constructing a new one based on the data from the current window. However, this approach is deemed inefficient as it results in the loss of the entire history of normal behavior with each occurrence of concept drift.

1. Applications

The Isolation Forest algorithm has found diverse applications across various domains due to its effectiveness in detecting anomalies and outliers. The algorithm's widespread applications can be attributed to its adaptability and effectiveness in addressing the dynamic challenges posed by anomalies. In essence, the Isolation Forest algorithm excels in identifying instances that deviate significantly from the norm, making it invaluable in scenarios where the detection of unusual patterns is paramount. Its underlying mechanism of isolating anomalies by recursively partitioning the data, often with a minimal number of steps, results in a fast and scalable solution, rendering it suitable for real-time applications and large-scale datasets.

The Isolation Forest algorithm has become a cornerstone in the field of cybersecurity, playing a pivotal role in fortifying digital systems against evolving threats. One of its primary applications lies in the detection of anomalies within network traffic. Cybersecurity professionals leverage the algorithm to scrutinize network activities, isolating and highlighting potential threats such as malicious activities or cyber attacks. By swiftly pinpointing anomalies, the algorithm enables a proactive response, helping to mitigate the impact of security breaches. Its effectiveness in detecting deviations from normal network behavior makes it an invaluable tool for identifying both known and novel cyber threats. The algorithm's adaptability to changing patterns in network traffic is particularly advantageous in the realm of cybersecurity, where threats are constantly evolving. It can quickly adapt to new attack methodologies and anomalous behaviors, making it a robust solution for staying ahead of cyber adversaries.

In the realm of finance, the algorithm proves valuable for fraud detection. It excels in isolating fraudulent transactions or activities that deviate from regular patterns, offering financial institutions a robust tool to safeguard against illicit financial activities. Financial institutions leverage the Isolation Forest algorithm to scrutinize vast datasets of transactions, identifying patterns and behaviors that deviate from established norms. This capability proves indispensable in isolating fraudulent transactions, whether they involve stolen credit card information, identity theft, or other deceptive financial practices. The algorithm's effectiveness lies in its capacity to discern anomalies efficiently, providing financial institutions with a proactive means of detecting and mitigating fraud. The dynamic and evolving nature of fraudulent behavior poses an ongoing challenge for financial institutions. Traditional fraud detection methods may struggle to keep pace with the constantly changing tactics employed by fraudsters. Here, the Isolation Forest algorithm's adaptability shines. Its ability to quickly adapt to new patterns and identify anomalies in real-time makes it particularly well-suited for addressing the dynamic landscape of fraudulent activities. By efficiently isolating emerging patterns indicative of potential fraud, the algorithm empowers financial institutions to stay one step ahead of evolving threats.

In the rapidly advancing landscape of healthcare, the Isolation Forest algorithm emerges as a revolutionary force, offering a sophisticated solution to the intricate challenge of anomaly detection in medical data. This algorithm's application becomes particularly pivotal in the context of patient care, where the ability to discern subtle deviations from normal health metrics or behaviors can be the key to early disease detection and proactive intervention. Sudden changes in vital signs or anomalies in medical records can signal potential health issues, allowing for early intervention and improved patient outcomes. The Isolation Forest algorithm analyzes vast datasets of medical information and, with its unique capacity to isolate abnormal patterns, it is able to flag unusual patient behaviors or health metrics that may serve as early indicators of potential health risks. This proactive approach marks a paradigm shift in healthcare, where the emphasis is shifting from reactive treatments to preventative measures. Early disease detection is a cornerstone in the quest for improved patient outcomes, and the Isolation Forest algorithm stands at the forefront of this endeavor. By swiftly identifying anomalies in patient data, healthcare professionals gain valuable insights into potential health risks, allowing for timely intervention and the implementation of preventative measures. This not only enhances the chances of successful treatment but also contributes to reducing the overall burden on healthcare systems by addressing health issues at their nascent stages. Monitoring patient well-being takes on a new dimension with the incorporation of the Isolation Forest algorithm. Its ability to navigate through complex medical data sets with efficiency empowers healthcare providers to offer personalized and timely interventions, fostering a patient-centric approach.

These applications highlight the versatility of the Isolation Forest algorithm, showcasing its efficacy in diverse domains where anomaly detection is paramount for ensuring security, reliability, and optimal performance.

1. Limitations

The Isolation Forest algorithm, while effective in certain scenarios, has its limitations that should be considered when applying it to real-world problems. One significant limitation is its sensitivity to the dimensionality of the data. As the number of features increases, the effectiveness of the Isolation Forest may decrease. This is because in high-dimensional spaces, it becomes easier for the algorithm to create isolation trees that isolate normal instances rather than anomalies, leading to a decrease in detection accuracy.

Another limitation lies in its assumption of the independence of features when constructing isolation trees. In real-world datasets, features are often correlated, and the algorithm may struggle to accurately isolate anomalies when strong correlations exist. The Isolation Forest relies on the concept of randomly selecting features at each split, assuming that outliers have attributes that are noticeably different from those of normal instances.

Additionally, the Isolation Forest is sensitive to the choice of hyperparameters, such as the number of isolation trees and the subsampling size. Tuning these parameters can be challenging, and suboptimal choices may result in a decreased ability to detect anomalies or an increased rate of false positives. Furthermore, the Isolation Forest may struggle with imbalanced datasets, where the number of normal instances significantly outweighs the number of anomalies or vice versa.

In summary, while the Isolation Forest is a powerful algorithm for anomaly detection in certain situations, its limitations in handling high-dimensional data, correlated features, sensitivity to hyperparameters, and challenges with imbalanced datasets should be carefully considered when choosing an appropriate method for anomaly detection in a specific application.

1. Conclusions

In conclusion, our exploration of the Isolation Forest algorithm within the context of anomaly detection has revealed its significance and versatility in addressing the challenges posed by identifying outliers in diverse datasets. The algorithm, originating as a novel approach to isolation-based anomaly detection, has evolved into a state-of-the-art method, demonstrating its efficacy in detecting anomalies efficiently and with remarkable accuracy. By leveraging the inherent structure of normal instances, the Isolation Forest algorithm exhibits a unique ability to isolate anomalies, making it particularly well-suited for high-dimensional datasets where traditional methods may falter.

Throughout our study, we delved into the intricacies of the original Isolation Forest and explored various adaptations and enhancements that have emerged in the literature. These variations have further extended the algorithm's applicability to different domains and datasets, showcasing its adaptability to diverse data structures and anomaly patterns. From extended forests to feature-specific modifications, researchers have continually innovated upon the original algorithm, enhancing its performance and applicability across various scenarios.

Examining practical applications has demonstrated the algorithm's utility in real-world scenarios, ranging from cybersecurity to finance, where the ability to promptly identify anomalies is paramount. Despite its numerous strengths, it is essential to acknowledge the limitations of the Isolation Forest algorithm. Sensitivity to the choice of hyperparameters, potential challenges with skewed datasets, and varying degrees of success in handling certain types of anomalies highlight areas for further research and improvement.

In conclusion, the Isolation Forest algorithm stands as a robust and valuable tool in the anomaly detection toolbox. Its ability to efficiently identify outliers in high-dimensional datasets, coupled with its adaptability and versatility, positions it as a significant player in the ever-evolving landscape of machine learning-based anomaly detection methods. As the field continues to advance, ongoing research and development in this area hold promise for refining the algorithm's capabilities and expanding its applicability to an even broader array of domains and challenges.

1. Bibliography

[1] Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: A survey. ACM Comput. Surv. 41, 15:1–15:58.

[2] Karev, D., McCubbin, C., Vaulin, R., 2017. Cyber threat hunting through the use of an isolation forest, in: Rachev, B., Smrikarov, A. (Eds.), Proceedings of the 18th International Conference on Computer Systems and Technologies, CompSysTech 2017, Ruse, Bulgaria, June 23-24, 2017, ACM. pp. 163–170.

[3] John, Hyder, and Sameena Naaz. "Credit card fraud detection using local outlier factor and isolation forest." Int. J. Comput. Sci. Eng 7.4 (2019): 1060-1064.

[4] Liu, F.T., Ting, K.M., Zhou, Z., 2008. Isolation forest, in: Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy, IEEE Computer Society. pp. 413–422.

[5] F. T. Liu, K. M. Ting, and Z.-H. Zhou. On detecting clustered anomalies using sciforest, in: Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases, in Lecture Notes in Computer Science, vol. 6322. Berlin, Germany: Springer, 2010, pp. 274–290.

[6] D. Xu, Y. Wang, Y. Meng, and Z. Zhang. An improved data anomaly detection method based on isolation forest, in: Proc. 10th Int. Symp. Comput. Intell. Design (ISCID), Dec. 2017, pp. 287–291.

[7] Z. Ding and M. Fei. An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window, IFAC Proc. Volumes, vol. 46, no. 20, pp. 12–17, 2013.

[8] S. C. Tan, K. M. Ting, and T. F. Liu. Fast anomaly detection for streaming data, in: Proc. 22nd Int. Joint Conf. Artif. Intell., North America, Jun. 2011.

[9] Tokovarov, M., Karczmarek, P., 2022. A probabilistic generalization of isolation forest. Inf. Sci. 584, 433–449.