Enhancing Security of Databases through Anomaly Detection in Structured Workloads

Charanjeet Dadiyala*, Faijan Qureshi, Kritika Anil Bhattad, Sourabh Thakur, Nida Tabassum Sharif Sheikh & Kushagra Anil Kumar Singh

Shri Ramdeobaba College of Engineering and Management Nagpur, 440013, India
*E-mail: dadiyalacs@rknec.edu

Abstract. In today's world, the protection of databases in any global organization has become paramount due to the rapid growth of data and the new generations of cyber threats. This highlights the need for more enhanced security precautions to secure these databases containing sensitive information. One of the most advanced ways of enhancing database security is using an anomaly detection system, especially for structured workloads. Structured workloads typically exhibit predictable patterns of data access and usage, making them susceptible to displaying anomalies that may indicate unauthorized access, data manipulation, or other security breaches. Anomaly detection methods can identify patterns that are unusual, an indication of malicious activity, or a data security breach. The present research utilized the Isolation Forest algorithm to detect outliers in high-dimensional data sets. The main contribution and novelty of this research lies in leveraging the Isolation Forest algorithm for structured database workloads to proactively identify and mitigate potential security threats. Our study showed that the proposed model, with an accuracy of 85%, outperformed various state-of-the-art methods. Furthermore, anomaly detection systems powered by advanced algorithms and machine learning enable real-time database activities analysis, addressing challenges like preprocessing, model training and scalability.

Keywords: *anomaly detection; database security; Isolation Forest; machine learning; MySQL; structured workloads.*

1 Introduction

In today's interconnected world, databases serve as the backbone of numerous applications, handling vast amounts of sensitive data. However, with the proliferation of cyber threats, ensuring the security of databases has become increasingly challenging. Unauthorized access, data breaches, and malicious activities pose significant risks to the integrity and confidentiality of stored information. Traditional security measures [1], such as access control (role-based access), encryption of stored or transmitted data, user authentication, use of firewalls etc., are often insufficient to detect and respond to sophisticated attacks effectively. These methods may fall short in dealing with sophisticated and evolving threats such as insider threats, advanced persistent threats (APTs), and subtle anomalies indicative of malicious activity. This study also addressed

Received April 29th, 2024, Revised November 26, 2024, Accepted for publication December 23rd, 2024.
Copyright © 2025 Published by IRCS-ITB, ISSN: 2337-5787, DOI: 10.5614/itbj.ict.res.appl.2025.18.3.2

traditional methods and their challenges in scalability and advanced attack patterns using machine learning [2].

To address these challenges, there is a growing interest in leveraging anomaly detection techniques to enhance the security of databases to identify unusual patterns or behaviors within data sets that deviate from the norm, signaling potential security threats [3]. In the context of structured workloads, which encompass various user activities and transactions, anomaly detection plays a crucial role in safeguarding database integrity. The focus of this research was to explore the application of anomaly detection in structured workloads to enhance the security of databases [4]. Specifically, we investigated the use of the Isolation Forest algorithm, renowned for its effectiveness in detecting outliers in high-dimensional data sets, such as database activity logs. By isolating anomalies and identifying deviations from normal patterns, the Isolation Forest algorithm offers a promising approach to detecting and mitigating security threats in structured workloads [5].

This paper presents a comprehensive methodology for enhancing database security through anomaly detection in structured workloads. We began by establishing a connection to the MySQL database and retrieving relevant data sets for analysis [6]. SQL queries were employed to extract user activities, timestamps, and other pertinent features, which were then pre-processed to prepare them for the anomaly detection algorithm. Central to our approach is the utilization of the Isolation Forest algorithm, which was configured and trained on the pre-processed data set to identify anomalies effectively [7]. The algorithm's ability to isolate outliers by randomly selecting features and split values makes it well-suited for detecting anomalous behavior within structured workloads. Furthermore, we incorporated user verification mechanisms to validate anomalies related to bulk deletion, thereby reducing the likelihood of false positives [8]. The detected anomalies and their details are presented on a dashboard for easy monitoring and analysis, enabling administrators to take proactive measures to mitigate security risks. It also contributes to advancing anomaly detection techniques, offering insights for improving database security practices across industries.

The primary objective of this research was to develop an anomaly detection system tailored for structured workloads in databases to enhance database security by accurately detecting anomalous activities and alerting administrators to them, thereby mitigating potential security risks and ensuring data confidentiality and integrity. More specific research objectives included:

1. To analyze anomalies and attack patterns in structured workloads to comprehensively understand potential security risk.

2. To design an Isolation Forest-based method for anomaly detection tailored to the unique traits of structured database workloads.
3. To assess the anomaly detection system's real-time performance in identifying and mitigating security vulnerabilities.
4. To improve the scalability of the anomaly detection system to ensure effectiveness across database workloads.

2 Literature Review

This section contains a review of the latest relevant literature on data stream anomaly detection, highlighting the work done, the observations made, and the scope for improvement. This section also presents a summarized literature survey in a tabular format (Table 1).

The research on data stream anomaly detection reported in [9] categorized methods into offline, semi-online, and online learning-based methods, each with theoretical support. However, there is a gap in applying these methods to specific environments such as structured databases and real-time detection remains a key focus for future advancements. Another study [10] explored a framework for anomaly detection and real-time reliability evaluation in cloud logs. Using traditional machine learning and ensemble methods, it was able to enhance the prediction accuracy while integrating reliability metrics for robust system management. The next study [11] used K-means clustering to detect anomalies in log data, focusing on patient events involving multiple doctors versus single-doctor visits. Anomalies were identified through monthly log analyses and boxplots, revealing outliers like high event counts for individual patients. Future work aims to automate log processing and evaluation, exploring machine learning techniques like SVM and decision tree for enhanced anomaly detection. Another study [12] proposed a system for real-time anomaly detection in NoSQL systems by analyzing resource usage and process data, with LSTM-RNN outperforming ARIMA and SARIMA in CPU usage forecasting. The experimental results using YCSB workloads highlighted its effectiveness.

One study [13] presents an anomaly detection system for database access patterns using user behavior analysis, rule-based controls, and machine learning techniques like one-class SVM, Naïve Bayes, and Nearest Neighbor algorithms. The experimental results highlight one-class SVM as the most effective, demonstrating the potential of machine learning for accurate anomaly detection in database access. A similar study [14] proposed a workload-aware anomaly detection method for web applications, utilizing incremental clustering to group workload patterns based on access behavior and request volume. By integrating Local Outlier Factor (LOF), it was able to enhance anomaly detection accuracy compared to traditional methods that neglect workload dynamics. The study reported in [15] introduced a method for detecting anomalous database access patterns using models that analyze SQL query patterns from logs at different granularities. Role-based classifiers and clustering algorithms were employed to identify anomalies, even without explicit role data. The experiments on real and synthetic data sets confirmed its efficacy.

Table 1 Tabular summary of the relevant and latest research papers considered in this study.

Technique	**Defense**	**Attack Type**	**Attack method**	**Data set type**	**Research Gap & citation**
Offline, semi-online, online learning	Detection of data stream anomalies	Generic	Learning-based	Data streams	Lack of application in specific environments such as structured databases [9]
Anomaly detection with logs	Real-time reliability evaluation	Cloud platform attacks	Log analysis	Cloud platform logs	Limited focus on generalization across database environments [10]
SQL database analysis	Anomaly detection in SQL databases	SQL injection, misuse	Retros-pective anomaly detection	SQL databases	Limited exploration of real-time monitoring and contextual anomalies [11]
Resource usage monitoring	Real-time NoSQL anomaly detection	Resource overuse	Monitoring resource usage	NoSQL databases	Scalability and heterogeneity of data sets not fully addressed [12]
Access pattern analysis	Detection of anomalous access patterns	Access anomalies	Pattern analysis with SVM	Relational databases	Limited consideration of workload-specific anomaly traits [13]
Workload-aware detection	Identifying anomalies in web apps	Workload anomalies	Statistical and ML methods	Web applications	Lack of exploration in non-web database systems [14]
Unsupervised contextual detection	Context-aware anomaly detection	Contextual anomalies	Meta-learning approaches	Database systems	High computational costs in real-time implementations [3]
Isolation Forest optimization	Intrusion detection for IIoT networks	IoT anomalies	Optimized Isolation Forest	Industrial IoT data	Scalability for heterogeneous data sets [7]
Meta learning-based VAE	Facility management anomaly detection	Facility operations	Variational autoencode r	Building facility data	Lack of focus on database anomaly detection techniques [6]
Present paper	Insider threats, APTs	Anomaly detection via Isolation Forest	Structured workloads	SQL data sets: healthcare data set-diabetes, another Cricket data set	Outperforms the state-of-the-art models covered in the section V

This paper provides a thorough review of anomaly detection algorithms for data streams, categorizing them into offline, semi-online, and online learning approaches. It offers theoretical insights into the feasibility of each algorithm and identifies promising research directions for future investigations in this field. The study further highlights the ongoing focus on real-time anomaly detection in streaming environments across various disciplines, underscoring the importance of advancing methods for enhancing security and reliability in data stream scenarios.

3 Research Methodology

The main objective of this research was to improve database security by implementing an anomaly detection system for structured workloads. The proposed system uses the Isolation Forest algorithm, known for its effectiveness in detecting outliers in high-dimensional data sets. It uses XAMPP, a popular open-source cross-platform web server solution stack package, as the database server, while Python serves as the primary programming language for the implementation due to its versatility and extensive libraries for data analysis and machine learning.

3.1 Workflow Diagram

This section explores the overall working of the model and the interaction between different components. Figure 1 illustrates the workflow for anomaly detection using the Isolation Forest algorithm with data sourced from a MySQL database. The process begins with the collection of data from the MySQL database, followed by the construction of a model using the Isolation Forest algorithm. This model is then employed to predict anomalies within the data set, with negative predictions indicating the presence of anomalies. In the event of anomalies related to bulk deletion, the process advances to user verification to confirm the responsible user's identity. Finally, the detected anomalies and their details are displayed on a dashboard, providing a clear and accessible overview of the anomalies for monitoring and further action.

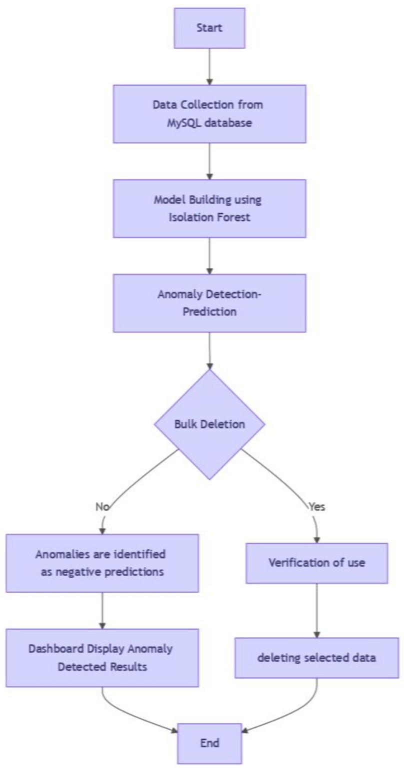

Start
Data Collection from MySQL database
Model Building using Isolation Forest
Anomaly Detection-Prediction
Bulk Deletion
No
Yes
Anomalies are identified as negative predictions
Verification of use
Dashboard Display Anomaly Detected Results
deleting selected data
End

Figure 1 Workflow Diagram of the proposed model.

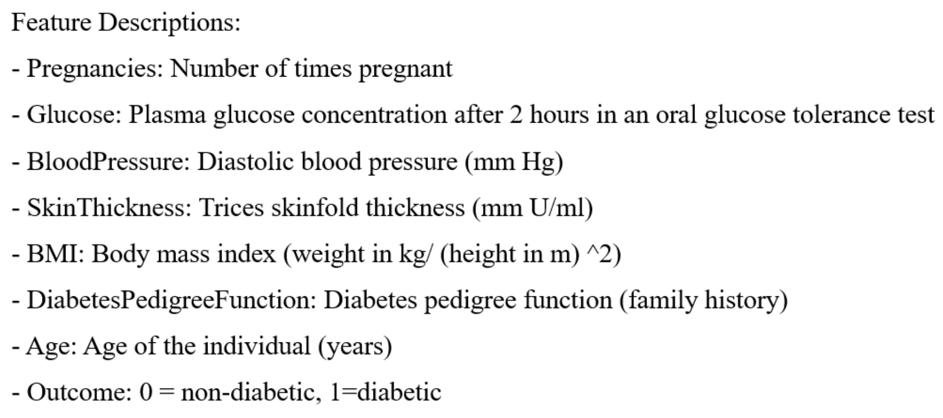
3.2 Data Set Collection

3.2.1 Name of Data Set: Diabetes

The database schema includes a table named ‘Diabetes’ containing information related to diabetes patients, including columns for the number of pregnancies, glucose levels, blood pressure, skin thickness, insulin levels, BMI, diabetes pedigree function, and age. This structured data set provides meaningful data for anomaly detection, aiding in the analysis of factors related to diabetes. The data set description and class distribution are depicted in Figures 2 and 3, respectively. Also, in Figure 4, a heatmap of the correlation between the variables in the data set is shown for better understanding.

The chosen attributes and their properties are outlined below:

1. **Pregnancies:** The number of pregnancies a patient has had can provide insights into the patient's medical history as well as reveal potential risks associated with diabetes.
2. **Glucose Levels:** The plasma glucose concentration after a two-hour oral glucose tolerance test is a crucial indicator for diabetes diagnosis and management. Anomalies in glucose levels could indicate potential health issues or measurement errors.
3. **Blood Pressure:** Diastolic blood pressure is another important health indicator that, when abnormal, could be a sign of underlying health conditions related to diabetes.
4. **Skin Thickness:** The triceps skin fold thickness can be a factor in assessing body composition and health risks associated with diabetes.
5. **Insulin Levels:** Two-hour serum insulin levels provide information about the body's insulin production and can help to diagnose insulin resistance, a common condition in diabetes.
6. **BMI:** The body mass index (BMI) is a measure of body fat based on height and weight. Anomalies in BMI could indicate obesity, a risk factor for type 2 diabetes.
7. **Diabetes Pedigree Function:** This function scores the likelihood of diabetes based on family history, providing insights into genetic predispositions to the disease.
8. **Age:** The age of the patient is a significant factor in diabetes diagnosis and management, as the risk of developing diabetes increases with age.


Feature Descriptions:

- Pregnancies: Number of times pregnant

- Glucose: Plasma glucose concentration after 2 hours in an oral glucose tolerance test

- BloodPressure: Diastolic blood pressure (mm Hg)

- SkinThickness: Trices skinfold thickness (mm U/ml)

- BMI: Body mass index (weight in kg/ (height in m) ^2)

- DiabetesPedigreeFunction: Diabetes pedigree function (family history)

- Age: Age of the individual (years)

- Outcome: 0 = non-diabetic, 1=diabetic

Figure 2 Data set description of the SQL data set 'diabetes.csv'.

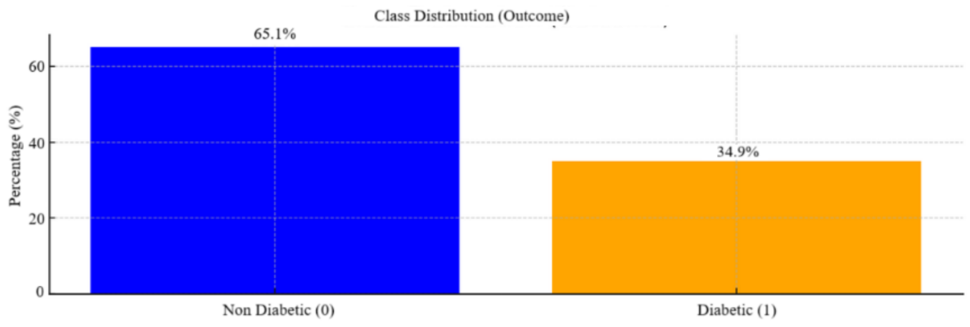


Figure 3 Class distribution (outcome) of the SQL data set ‘diabetes.csv’.

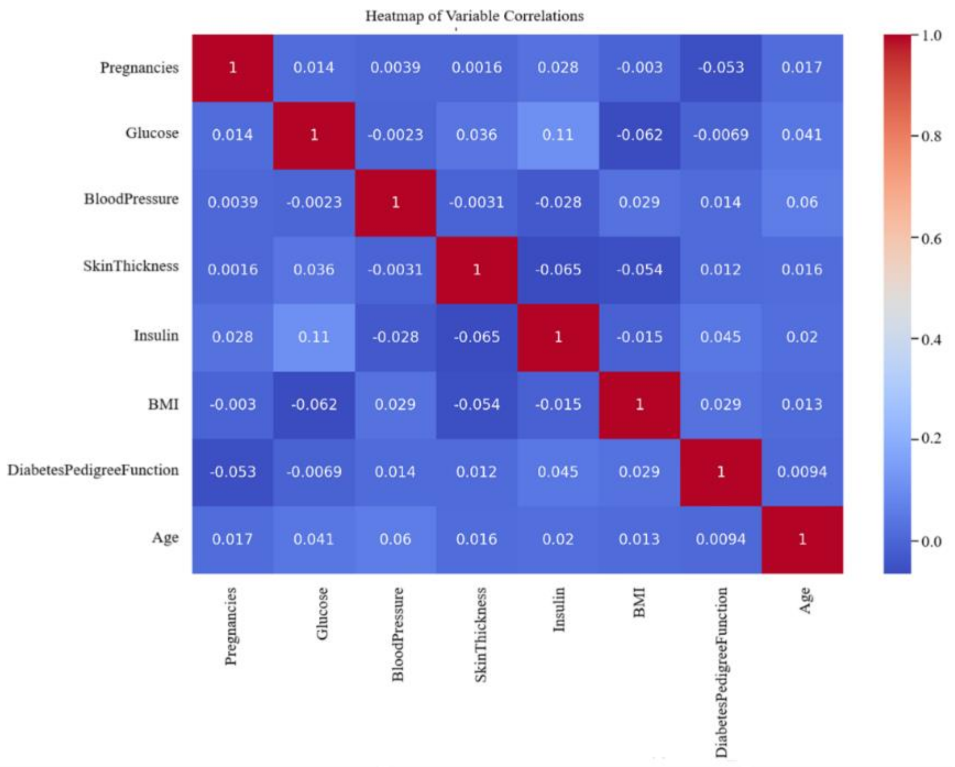


Figure 4 Heatmap of the correlation between variables of the SQL data set ‘diabetes.csv’

3.2.1.1 Data Loading and Preprocessing

Following data collection, the data set undergoes meticulous loading into memory from either the database or a CSV file. Missing values are handled using

Pandas methods such as ‘fillna()’ or ‘dropna()’ to maintain data set integrity. Feature selection techniques like SelectKBest or Recursive Feature Elimination (RFE) may be applied to optimize anomaly detection performance. This preparatory phase ensures that the data set is appropriately formatted for subsequent analysis and model building.

3.2.1.2 Development of the Proposed Model

In the model building phase, the Isolation Forest algorithm, a powerful tool available in the sklearn.ensemble module, takes center stage for anomaly detection tasks. An instance of the Isolation Forest class is meticulously created with careful consideration of parameters such as n_estimators, contamination, and max_features, tailored to best suit the data set’s characteristics. The percentage distribution of the data set for training, testing, and evaluation are 70%, 15% and 15% respectively. The model is then meticulously trained on the preprocessed data set using the fit method, a crucial step in establishing its efficacy for anomaly detection tasks. This phase forms the backbone of the anomaly detection pipeline, laying the groundwork for subsequent detection and visualization processes.

3.2.1.3 Anomaly Detection and Its Measurements

Trained on the data set, the Isolation Forest model is utilized to predict anomalies using the predict method. Anomalies are identified based on negative predictions, indicating outliers within the data set. This step is critical for flagging potential anomalies and enabling further investigation and analysis. To calculate the anomaly score of a data point *x*, is given as in Eq. (1):

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{1}$$

where *E(h(x)* is average path length of *x* across all trees in the forest, *c(n)* is the average path length for a data set of size *n*.

1. If *s(x,n)* is close to 1: *x* is highly likely to be an anomaly.
2. If *s(x,n)* is close to 0.5: *x* is likely a normal point.

3.2.1.4 Visualization and Dashboard Creation

To further enhance anomaly detection analysis, Matplotlib is used for visualizing the results, offering insights into anomaly distribution and patterns. A Tkinter-based GUI dashboard presents these visualizations in an intuitive and user-friendly layout. This approach enables stakeholders to easily interpret findings and make informed decisions effectively.

3.2.1.5 Performance Evaluation and Execution

The proposed model automates the anomaly detection process by connecting to a data set, train the model, identify the anomalous patterns, and present the results in a dashboard and in various visualizations, enabling continuous monitoring and timely mitigation of potential risk.

4 Results and Observations

Anomaly detection models often assign anomaly scores to each data point, indicating how anomalous or different it is compared to the rest of the data. Higher anomaly scores suggest a higher likelihood of being an anomaly. Based on the anomaly scores, data points are classified as anomalies or normal data. Anomalies are flagged as outliers or unusual observations that deviate significantly from the expected behavior of the system. For better visualization of anomalies, another simple data set from the Cricket SQL database was used. A heatmap of the results is depicted in Figure 5, where the deviation is clearly highlighted in orange and red colors.

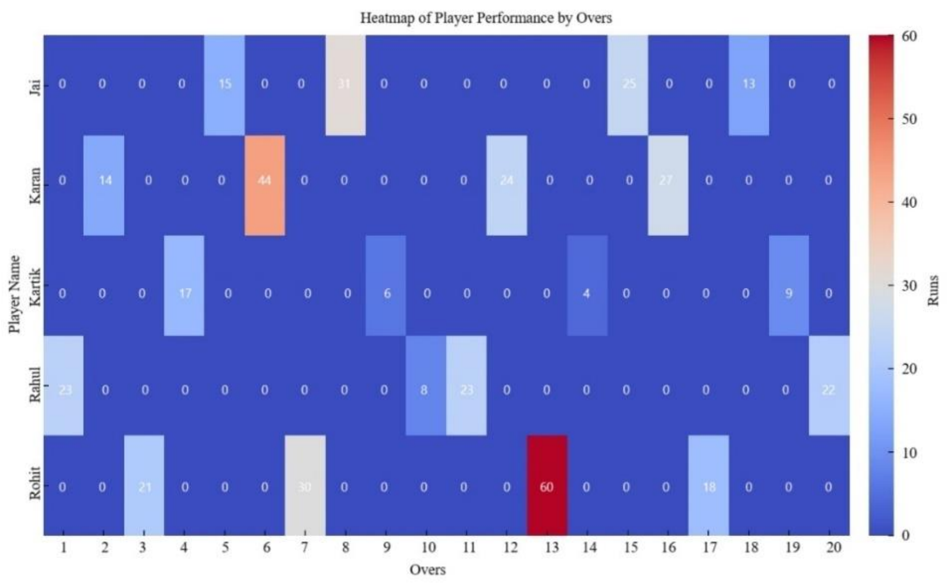


Figure 5 Heatmap of player performance by overs in Cricket SQL database.

For example, if the total number of runs in a particular over exceeds the permissible limit of 36 runs, anomalies are identified. For instance, consider column 6, where a total of 44 runs can be seen, surpassing the expected threshold, so anomalies were detected and highlighted for further investigation. This feature selection of runs for anomaly detection is chosen by the model itself and no mathematical conditions are required. In this way the model can handle any kind of data set without any prior conditions. It also predicts anomaly scores for each data point, indicating how anomalous or different it is compared to the rest of the data. Higher anomaly scores suggest a higher likelihood of being an anomaly. Figure 6 depicts a line chart for the Cricket SQL database that shows the anomaly scores over Overs by Player, which visualizes the anomaly scores varying from each player, indicating trends that deviate significantly.

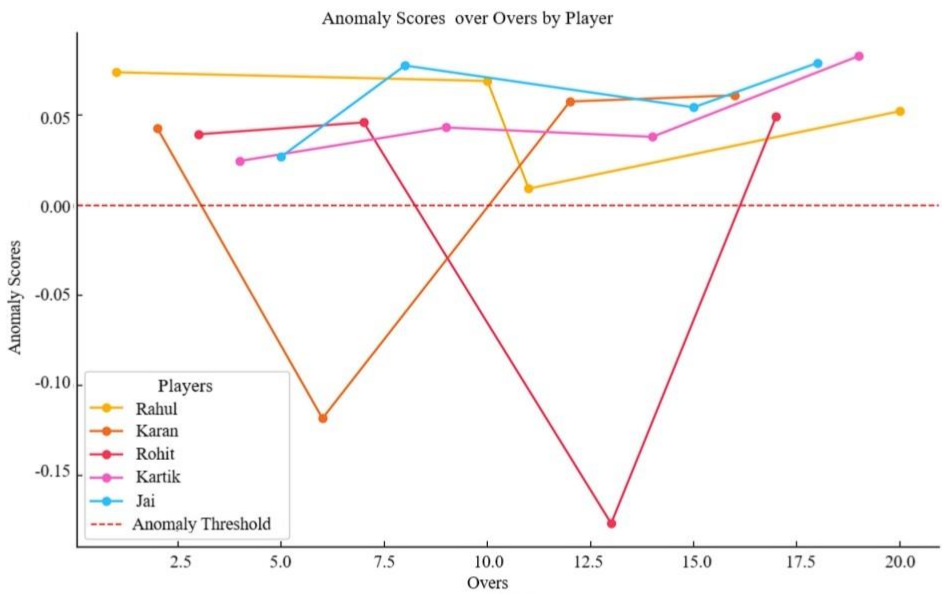


Figure 6 Line chart of anomaly scores over Overs by Player in Cricket SQL database.

While detecting anomalies in the diabetes database, anomalies are denoted by a value of -1 in the anomaly detection results. This signifies instances where the observed data points deviate significantly from the expected patterns, as depicted in the box plot in Figure 7 below. For example, if the glucose level of a patient exceeds a certain threshold that is considered abnormal, it will be flagged as an anomaly. For instance, consider a row where the glucose level is extremely high compared to the normal range, indicating a potential anomaly that warrants further investigation.

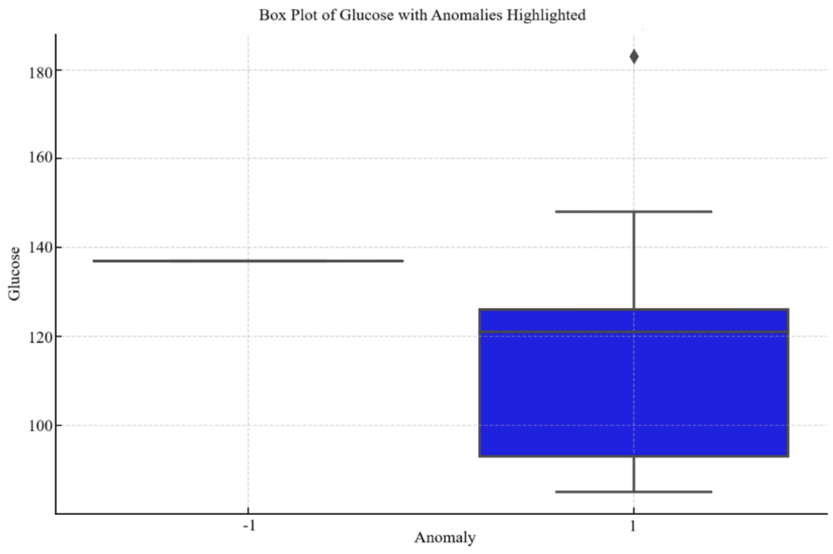


Figure 7 Box plot for glucose from the Diabates SQL database with anomalies highlighted.

For better visualization, the following figures (Figures 8 and 9) were generated, showing histograms of blood pressure and BMI with anomalies highlighted, respectively.

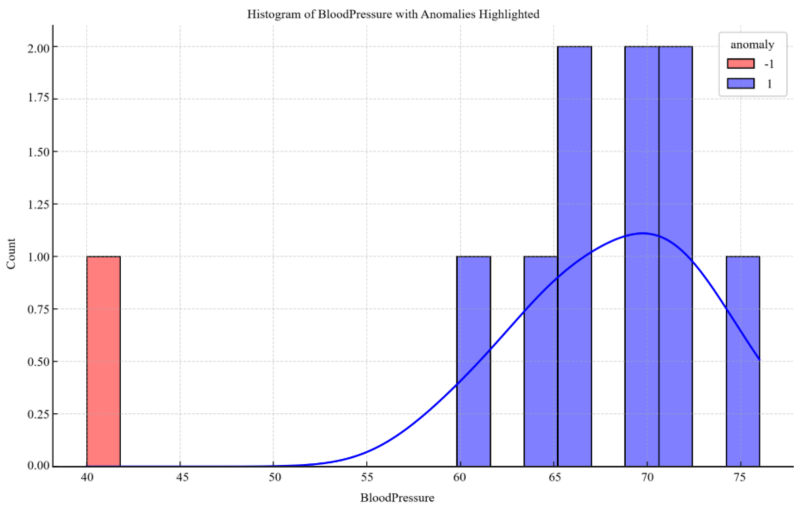


Figure 8 Histogram of blood pressure from the Diabates SQL database with anomalies highlighted.

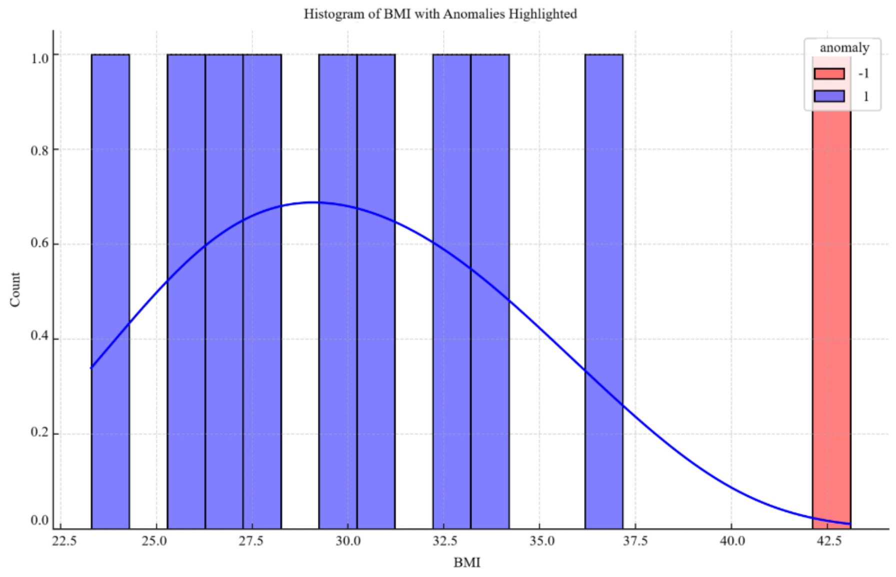


Figure 9 Histogram of BMI from the Diabates SQL database with anomalies highlighted.

5 Model Performance Evaluation Metrics

To evaluate the effectiveness of the proposed model's performance, four key metrics were computed: precision, recall, F1-score and accuracy. After that, its performance was analyzed using confusion-matrix based calculations. Given a total of 768 samples in the Diabetes data set, with approximately 268 cases of diabetes (positive class) and 5000 non-diabetes cases (negative class), the confusion matrix was calculated as follows:

1. True Positive (TP) = Recall x Total Positives = 201
2. False Positive (FP) = Total Positives – TP = 67
3. Precision = TP/ (TP+FP) = 50
4. True Negatives (TN) = Total Negatives – FP = 450

5.1 Key Metrics Calculations:

1. Accuracy:= $\frac{TP+TN}{Total\ Samples} = \frac{201+450}{768} = 0.85$
2. Precision = $\frac{TP}{TP+FP} = \frac{201}{201+50} = 0.80$

3. Recall $= \frac{TP}{TP+FN} = \frac{201}{201+67} = 0.75$
4. F1-score $= \frac{2\ X\ Precision\ x\ Recall}{Precision+Recall} = \frac{2\ x\ 0.80\ x\ 0.75}{0.80+075} = 0.77$

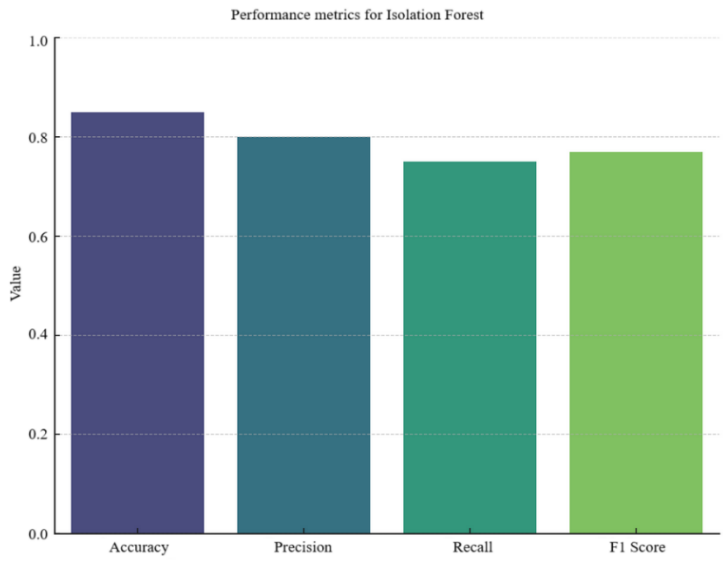


Figure 10 Performance metrics for the proposed model.

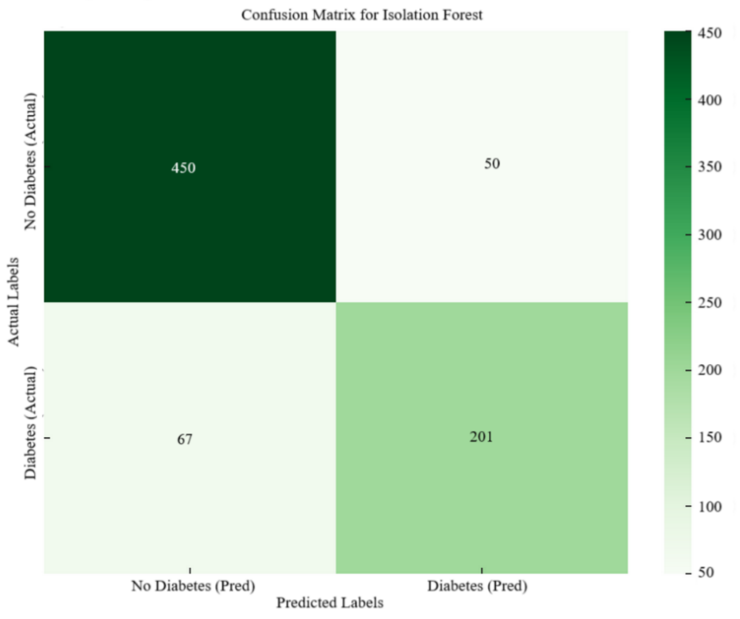


Figure 11 Confusion matrix for the proposed model.

5.2 Comparison with State-of-the-Art Methods

Additionally, state-of-the-art models like Support Vector Machine (SVM), Random Forest, and Gradient Boosting were trained to compare them with the proposed model using the same data set and preprocessing steps for a fair comparison, as shown in Table 2.

Tabel 1 Comparison of the performance of the proposed model with state-of-the-art models.

Model	Accuracy	Precision	Recall	F1-Score
Isolation Forest	0.853	0.801	0.752	0.771
Random Forest [16]	0.753	0.639	0.663	0.650
Gradient Boosting [16]	0.749	0.631	0.663	0.646
One-class SVM [16]	0.346	0.346	1.000	0.514

Figure 12 is a common visualization for evaluating the performance of classification models. Here, the curve shape depicts the trade-off between recall (X-axis) and precision (Y-axis). In the figure we can observe high precision. The low recall means the model is very confident in its predictions, but it missed some actual positives. Also, high recall but low precision means the model catches most positives but includes a few false positives too. The top-left corner shows the ideal performance where both precision and recall are maximized but where the steep drops indicate regions where the model sacrifices precision to improve recall or vice versa.

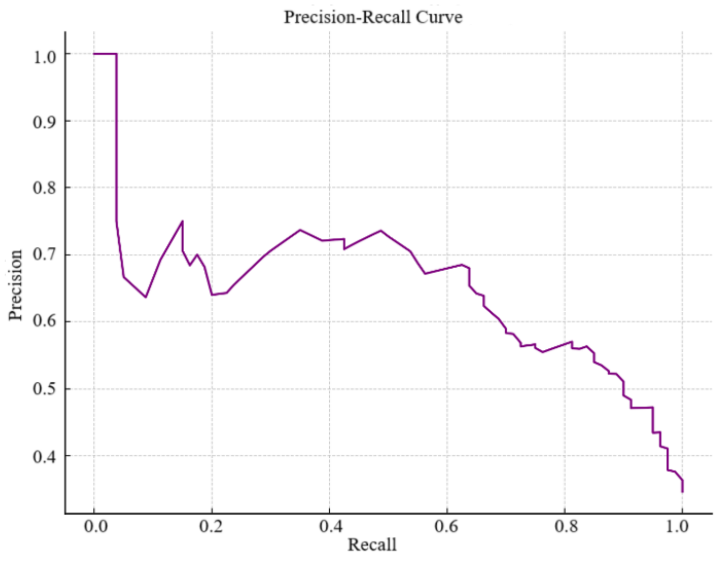


Figure 12 Precision-Recall curve.

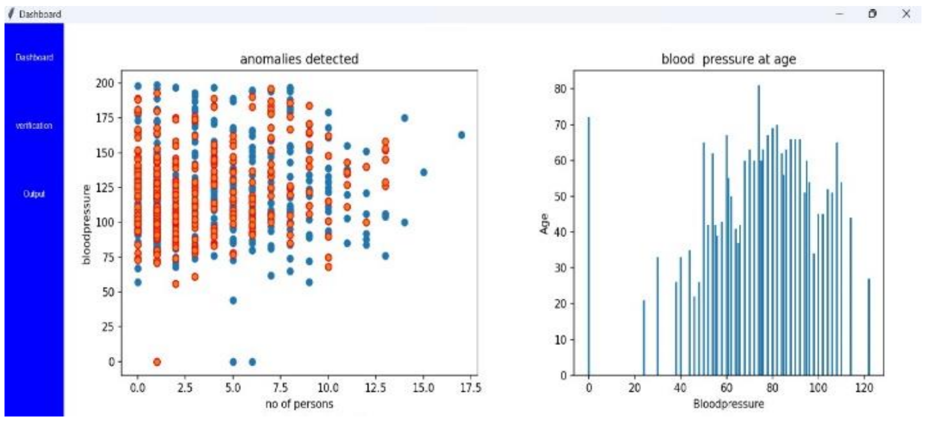


Figure 13 Tkiner-based GUI dashboard.

In this study, the performance of various machine learning models for anomaly detection in diabetes diagnoses was explored, i.e., Isolation Forest, Random Forest, Gradient Boosting and One-Class SVM. The results conclusively demonstrated that the Isolation Forest model outperformed the state-of-the-art methods across multiple evaluation metrics. With an AUC-PR of 0.81 (Figure 14), Isolation Forest showed a strong ability to balance precision and recall, making it highly effective in distinguishing between diabetic and non-diabetic cases. This performance was particularly significant for imbalanced data sets, where identifying rare positive cases is a challenging task, The model achieved superior metrics with an accuracy of 85%, precision of 80%, recall of 75% and F1- score of 77%, indicating its robustness in reliably predicting the positive class while minimizing false positives and negatives.

Compared to other models, such as Random Forest and Gradient Boosting, Isolation Forest demonstrated a higher capability to handle the nuances of the data set. Furthermore, the model's ability to deliver a consistently high-performance score highlights its suitability for practical deployments in real-world medical anomaly detection systems.

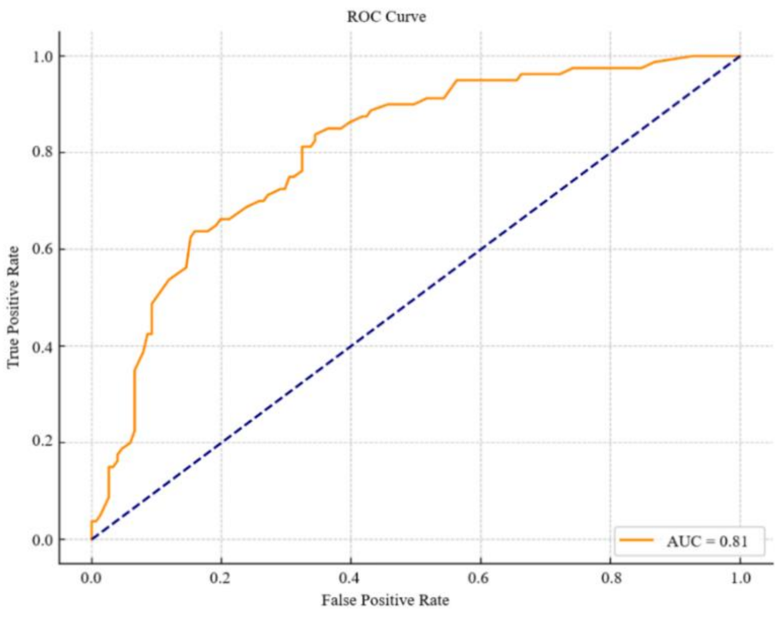


Figure 1 ROC curve.

6 Conclusion and Future Scope

This research affirms that Isolation Forest is an effective and reliable model for anomaly detection, outperforming other state-of-the-art models in this task. Future work should focus on adapting Isolation Forest for other data sets to evaluate its generalizability and robustness across diverse medical conditions. Additionally, researchers could explore its integration into scalable healthcare systems, enabling real-time diagnostic support and enhancing its utility in clinical practice. Furthermore, user behavior analysis incorporated into the anomaly detection system may provide additional insights into security threats originating from insider threats or compromised user accounts. Addressing scalability challenges and optimizing the performance of anomaly detection algorithms can also be a part of future work, as well as integrating deep learning techniques such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs) to improve the detection accuracy of anomalies in structured workloads.

References

[1] Gümüşbaş, D., Yıldırım, T., Genovese, A. & Scotti, F., *A Comprehensive Survey of Databases and Deep Learning Methods for Cybersecurity and Intrusion Detection Systems*, IEEE Systems Journal, **15**(2), pp. 1717-1731, June 2021.

[2] Paul, P. & Aithal, P.S., *Database Security: An Overview and Analysis of Current Trend*, International Journal of Management, Technology, and Social Sciences (IJMTS), **4**(2), 53-58, 2019

[3] Li, S., Yin, Q., Li, G., Li, Q., Liu, Z. & Zhu, J., *Unsupervised Contextual Anomaly Detection for Database Systems.* International Conference on Management of Data (SIGMOD '22), Association for Computing Machinery, New York, NY, USA, pp.788-802, June 2022.

[4] Nassif, A.B., Talib, M.A., Nasir Q. & Dakalbab, F.M., *Machine Learning for Anomaly Detection: A Systematic Review*, in IEEE Access, **9**, pp. 78658-78700, 2021.

[5] Xu, H., Pang, G., Wang, Y. & Wang, Y., *Deep Isolation Forest for Anomaly Detection*. IEEE Transactions on Knowledge and Data Engineering, **35**(12), pp. 1-14, 2023.

[6] Moon, J., Noh, Y., Jung, S., Lee, J., Hwang, E., *Anomaly Detection using a Model-agnostic Meta-learning-based Variational Auto-encoder for Facility Management*, Journal of Building Engineering, **68**, 106099, 2023.

[7] Lakshmi, M.S., Rajavikram, G., Dattatreya, V., Jyothi, B.S., Patil, S. & Bhavsingh, M., *Evaluating the Isolation Forest Method for Anomaly Detection in Software-Defined Networking Security*, Journal of Electrical Systems, **19**(4), pp. 279-297, 2023.

[8] Elsaid, S.A. & Binbusayyis, A., *An Optimized Isolation Forest based Intrusion Detection System for Heterogeneous and Streaming Data in the Industrial Internet of Things (IIoT) Networks.* Discover Applied Sciences, **6**, 483, 2024.

[9] Lu, T., Wang, L. & Zhao, X., *Review of Anomaly Detection Algorithms for Data Streams.* Applied Sciences. **13**, 6353, 2023.

[10] Wang, B., Hua, Q., Zhang, H., Tan, X., Nan, Y., Chen, R. & Shu, X., *Research on Anomaly Detection and Real-Time Reliability Evaluation with the Log of Cloud Platform*, Alexandria Engineering Journal, **61**(9), pp. 7183-7193, 2022.

[11] Naserinia, V. & Beremark, M., *Anomaly Detection in a SQL Database: A Retrospective Investigation*, Student Thesis of Master's Programme in Network Forensics, Halmstad University, School of Information Technology, 2022.

[12] Chouliaras, S. & Sotiriadis, S., *Real-time Anomaly Detection of NoSQL Systems Based on Resource Usage Monitoring*, in IEEE Transactions on Industrial Informatics, **16**(9), pp. 6042-6049, Sept. 2020.

[13] Roh, J.-h., Lee, S.-H. & Kim, S., *Anomaly Detection of Access Patterns in Database,* 2015 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea (South), pp. 1112-1115, 2015

[14] Wang, T., Wei, J., Zhang, W., Zhong, H. & Huang, T., *Workload-Aware Anomaly Detection for Web Applications*, Journal of Systems and Software, **89**, pp.19-32, 2014

[15] Kamra, A., Terzi, E. & Bertino, E., *Detecting Anomalous Access Patterns in Relational Databases*. The VLDB Journal, **17**, pp.1063-1077, 2008.

[16] Akmeşe, Ö.F. *Diagnosing Diabetes with Machine Learning Techniques*. Hittite J Sci Eng., **9**(1), pp. 9-18, 2022.