Viability Assessment about Applicability of Machine Learning in Cloud Anomaly Detection

K. Vani[1], Dr.S. Britto Ramesh Kumar[2]

[1]Research Scholar, Department of Computer Science, St. Joseph's College (Autonomous),Affiliated to Bharathidasan University, Tiruchirappalli, Tamil Nadu, India, Email: dr.vanikarthikeyan@gmail.com
[2]Assistant Professor, Department of Computer Science, St. Joseph's College (Autonomous), Affiliated to Bharathidasan University, Tiruchirappalli, Tamil Nadu, India, Email: brittork@gmail.com

Received: 14.07.2024 Revised: 20.08.2024 Accepted: 25.09.2024

ABSTRACT
The rapid growth of cloud computing has introduced more opportunities as well as significant challenges in ensuring the reliability and performance of cloud-based systems.Detecting unusual activities such as performance drops, unexpected high resource usage, or security threats, is crucial to avoid disruptions. This work named as "Viability Assessment about Applicability of Machine Learning in Cloud Anomaly Detection (VAAMLCAD)" looks into whether machine learning methods can help to identify and predict anomalies in cloud environment. Different machine learning models in particular Random Forest (RF), Decision Tree (DT), Support Vector Machine (SVM) K-Nearest Neighbor, and K-Means method to see how well aforementioned methods work in identifying anomalies in dynamic cloud environments. A dedicated testbed is constructed with state-of-the-art development frameworks and evaluation tools to measure the performance such as Accuracy Precision, Sensitivity, Specificity, F-Score, and Average Processing Time for the benchmark datasets KDD-Cup and UNSW-NB15. Rank wise discussions based on the performance of the compared methods are vividly elucidated in this work.

Keywords: Anomaly detection, Cloud computing environment, Machine Learning, Random Forest, Decision Tree, Support Vector Machine, K-Nearest Neighbor, K-Means

1. INTRODUCTION

Cloud computing environment is a system where users can access computing resources such as storage, servers, and applications over the internet. Instead of owning physical hardware, consumers can rent and use these ready-to-use resources, and can benefit through pay-per-use concept [1]. This makes the cloud computing environment into flexible and cost-effective one.There are different types of cloud environments in practice in particular,Public clouds [2], which are shared by multiple organizations and are available over the internet, Private clouds [3], which are dedicated to one organization, offering more control and security, and Hybrid clouds those combine both public and private clouds, allowing data and applications to be shared between them [4]. Multi-cloud environments [5] use multiple cloud services from different providers.Cloud computing environments are widely used for data storage, application development and deployment, huge dataset interpretations, and for running machine learning models. They help businesses and individuals to use the latest technology without investing much in physical infrastructures [6].Cloud computing also enhances collaboration between the teams to work on shared projects and access resources Globally. In addition, it offers automatic software updates and maintenance, that reduces the burden of the users significantly to manage and secure their systems [7].
An anomaly in a cloud computing environment refers to any unusual or unexpected behavior in the cloud environment that deviates from usual patterns. This can include sudden spikes in resource usage, such as CPU or memory, performance slowdowns, security threats like unauthorized access, or chaotic configuration changes. These anomalies can disrupt cloud services severely [8], thus it is essential to detect and confront them in a proper way to ensure the reliability, performance, and security of cloud systems. There are severaltechniques, including machine learning,and statistical analysis, are employed to monitor and identify irregularities in data accessing patterns [9]. These methods analyze large datasets to detect anomalies that deviate from the expected regular behavior, which may indicate potential issues or security threats. Machine learning modelscan learn from historical data and continuously adapt to new patterns, making them highly effective in identifying subtle and evolving anomalies. By automating the detection process, these technologies enhance the accuracy and efficiency of monitoring systems [10],

The VAAMLCAD work is intended to analyze the performance metrics of RF, DT, SVM, KNN, and K-Means algorithms in detecting cloud environment anomalies and classifying them into major attack categories, such as DoS, Probe, U2R, R2L, as well as non-attack anomalies, which are labeled as Unclassified Cloud Anomaly (UCA).

2. Existing Method

There are five renowned machine learning techniques specifically RF, DT, SVM, KNN, and K-Means are explored here for the relevance in cloud environment anomaly detection.

2.1. Random Forest (RF)

Random Forest is a powerful machine learning algorithm that belongs to the family of ensemble methods. It is primarily used for classification and regression tasks, making it one of the most preferred algorithms in the data science constellation. The fundamental principle of Random Forest is to create a 'forest' of several decision trees, each trained on anarbitrary subset of the data. This technique promotes to mitigate the risk of overfitting [11], which is a pervasive problem in traditional decision tree models. By aggregating the predictions from multiple trees, Random Forest improves the accuracy and robustness, making it suitable for various applications, including finance, healthcare, marketing, and anomaly detection in this case [12].

The construction of a Random Forest encompasses two main processes namely bootstrap sampling and feature selection. First, the algorithm uses bootstrap sampling, where it arbitrarily selects subsets of the training data with replacement to build individual decision trees [13]. This means some data points may be included in multiple trees, while others may not be included at all. Then, for each split in a decision tree, a random subset of features is selected. This randomness helps ensure that the trees in the forest are in a diversified manner, which is crucial for improving the performance of the model. During the prediction phase, the Random Forest aggregates the outputs of all treestypically using majority voting for classification or averaging for regressionto generate a final prediction [14].

One of the significant advantages of Random Forest is its robustness to overfitting, especially when dealing with larger datasets. While individual decision trees can easily overfit the training data, the ensemble approach of Random Forest helps reduce these inaccuracies. Additionally, Random Forest can handle both categorical and numerical data, making it highly adaptable to various types of datasets. It also provides an inherent measure of feature importance, allowing users to interpret which features significantly influence predictions. This feature is particularly beneficial for exploratory data analysis and feature selection.Its ability to handle complex datasets with high dimensionality makes it a valuable tool in any data scientist's toolkit.

While comparing Decision Trees, Random Forest models can become computationally intensive, especially with large datasets and numerous trees, that leads to longer training times [15]. The algorithm also tends to be less effective when the dataset has a high number of irrelevant features, as it can introduce noise into the model.

The architecture of random forest is illustrated in Figure 1.

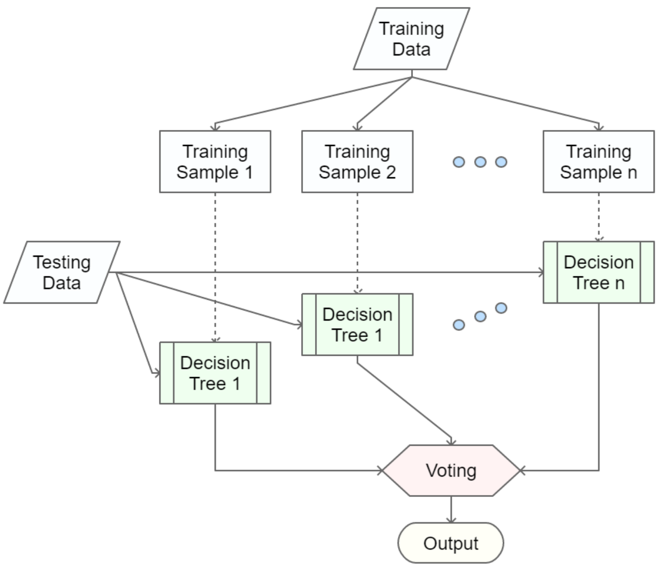

Figure 1: Random Forest Architecture

2.2. Decision Tree (DT)

A Decision Tree is one of the most prominent machine learning algorithms used for classification and regression tasks. DT works by splitting a dataset into smaller subsets based on the values of the features. The structure is like a flowchart, with internal nodes representing decisions branches representing the possible outcomes of those decisions, and leaf nodes representing the final prediction or classification [16]. Decision Trees help in making systematic progressive decisions by breaking down complex problems into simpler and manageable segments.

The tree starts with a root node that represents the whole dataset. This root node splits the data into subsets based on the feature that best dissects the data, optimizing for criteria like Gini impurity or information gain. The process of splitting and branching continues recursively until the algorithm reaches a break point, such as when all data points in a subset belong to the same class, or when further splits don't refinement of the model [17].One of the main advantages of Decision Trees is their modesty and interpretability. They replicate human decision-making processes, making them easy to understand. In addition, Decision Trees can handle both numerical and categorical data devoid of the need for normalization or scaling. Though decision trees can be vulnerable to overfitting, in particular when they grow excessively complex by splitting too deeply. Techniques like pruning or setting depth limits are used to prevent the overfitting problem of decision trees [18].

Decision Trees are also sensitive to small changes in the data, which can result in utterly different tree structures. Regardless of these limitations, Decision Trees remain a foundational tool in machine learning and serve as the building blocks for more sophisticated aggregation ofdifferent methods such as Random Forests and Gradient Boosting Machines, to improve stability and accuracy.

A typical Decision Tree model representation is provided in Figure 2.

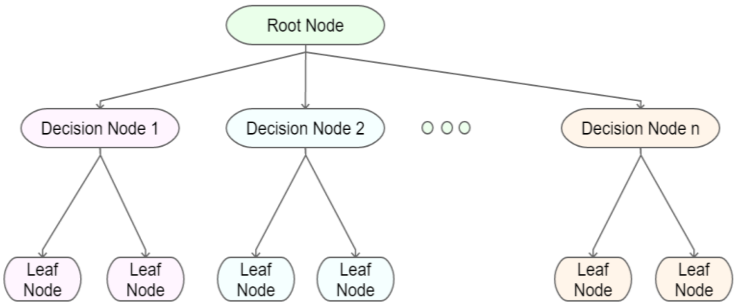

Figure 2: Decision Tree Model

2.3. Support Vector Machine (SVM)

Support Vector Machine is a supervised machine learning algorithm which is widely used for classification and regression purposes. The primary strength of SVM is its ability to find the optimal hyperplane to separates different classes within a dataset. A hyperplane serves as the decision threshold, while support vectors are the data points proximate to the hyperplane [19]. These support vectors are essential for defining the position and orientation of the hyperplane, and they directly impact the decision-making process of the model.

The margin in SVM is the distance intervening the hyperplane and the nearest support vectors from either class, and the goal of the method is to maximize this margin to enhance the generalization capabilities of the model. SVM utilizes the kernel trick to handle non-linear relationships in the data, that transforms the original feature space into a higher-dimensional space using multiple kernel functions. There are several common kernels such as linear, polynomial, and radial basis function (RBF), each allowing SVM to tackle both linear and non-linear classification challenges in an effective manner.

SVM offers several advantages, including effectiveness in high-dimensional spaces and robustness against overfitting, particularly when the appropriate kernel and proper regularization techniques are employed. However, there are some limitations such as increased training time with large datasets and the complexity of selecting the appropriate kernel and tuning parameters. Moreover, SVM also struggles with noisy data where class overlap is pivotal[20].

Collectively, Support Vector Machines are powerful tools in machine learning, making them suitable for several applications, including finance, bioinformatics, and image recognition. Their ability to handle complex datasets and deliver accurate results has established them as a prominent choice in the field. Thus, SVM is included in this work to detect anomalies in cloud computing environments.

A typical SVM Architecture is portrayed in Figure 3.

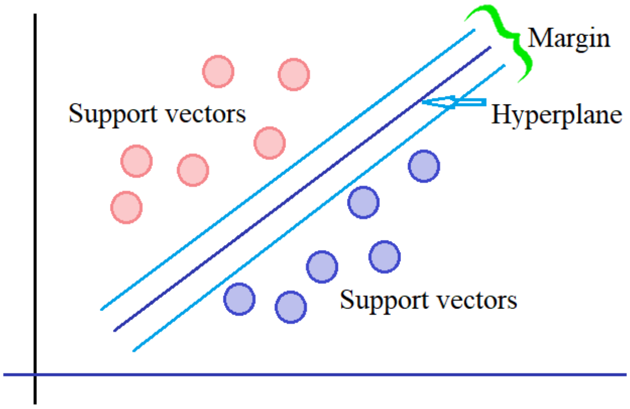

Figure 3: Support Vector Machine

2.4. K-Nearest Neighbor (KNN)

KNN is a supervised machine learning algorithm which is used for both classification and regression purposes. It works by classifying new data points according to their similarity with existing data in the training set. The algorithm quantifies the distance between new data point with all other points, to select the k nearest neighbors [21]. For classification tasks, KNN assigns the data point to the most common class among its neighbors, while in regression, it predicts the value by calculating the average of the values of the nearest neighbors.

KNN is a lazy learner, that means it doesnot create an explicit model during training phase. Instead, it preserves the entire dataset and performs calculations exclusivelyduring the prediction process. KNN is a non-parametric, so it refrains from assuming any specific distribution for the data, which allows flexibility when dealing with different types of datasets with various features sets.While KNN is simple and easy to understand, it has some limitations. It can be computationally expensive, especially with large datasets, since it must compute distances for every individual prediction [22]. In addition, KNN algorithm is sensitive to irrelevant features, and performance can vary based on the value of k. KNN is effective for smaller datasets and is widely used in applications such as image recognition, recommendation systems, and other tasks where data proximity plays a crucial role in decision-making.

KNN serves as a powerful tool for anomaly detection in cloud environments, leveraging its simplicity and effectiveness to monitor and maintain system integrity. KNN offers various advantages, including simplified implementation and interpretability.Thus, KNN is nominated in this work to explore the performance of cloud environment anomaly detection [23].The diagrammatic representation of KNN algorithm is displayed in Figure 4.

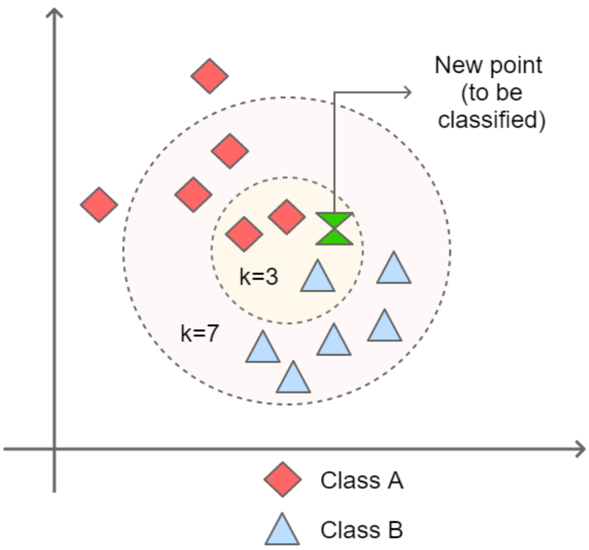

Figure 4: KNN Model

2.5. K-Means

K-Means is a commonly used unsupervised machine learning algorithm designed for clustering tasks. The primary objective of K-Means is to segment a dataset into k distinct clusters, with each data point assigned to the cluster represented by the nearest mean – represented as centroid in classification context. It is widely used in market segmentation to tailor marketing strategies based on customer behavior and demographics [24]. KNN also clusters similar documents in natural language processing, enhancing search and retrieval systems. In anomaly detection, K-Means identifies outliers, which is valuable for fraud detection in finance and performance monitoring in IT. Additionally, it improves recommendation systems by analyzing user-item interactions and providing personalized suggestions. In social network analysis, it identifies communities based on user interactions, while in healthcare, it clusters patient data for personalized medicine.

The algorithm begins by randomly selecting k initial centroids from the dataset. In the assignment step, each data point is assigned to the nearest centroid based on a distance metric, typically Euclidean distance to form k clusters. As a subsequent update step, the centroids are recalculated by computing the average of the data points within each cluster, and the process iterates until convergence, where centroids stabilize and assignments are no longer changing significantly.K-Means is characterized by its simplicity and efficiency, making it easy to implement and suitable for large datasets [25]. K-Means has some limitations such as sensitivity to the initial choice of centroids and the assumption that clusters are spherical and equally sized. In addition, users must specify the number of clusters k in advance, which can be challenging if the optimal number is ambiguous.

K-Means is effective for various applications, such as segmenting customers based on behavior, reducing the number of colors in images, and clustering similar documents. Overall, K-Means remains a powerful tool in unsupervised learning, offering valuable insights through its clustering capabilities. A diagrammatic representation is provided in Figure 5.

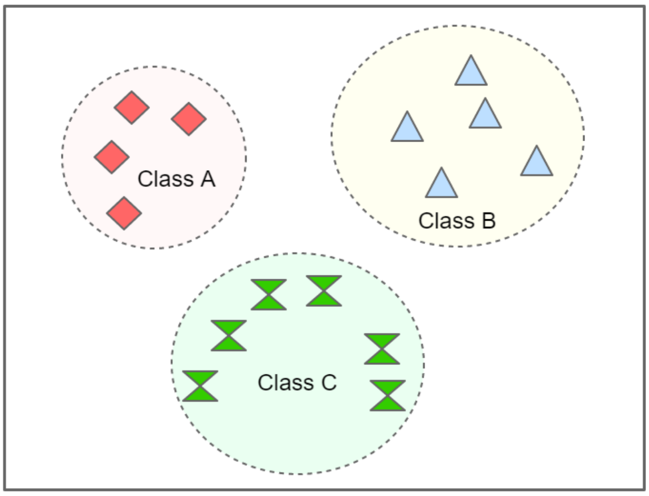

Figure 5: K-Means Clusters

3. Experimental Setup

A unified testbed is essential to compare different machine learning algorithms in anomaly detection for a handful of reasons such as Standardization, Reproducibility, Benchmarking, Variable configurability, Evaluation efficiency, and to maintain Equality among the compared methods.

A unified testbed ensures that all ML algorithms are evaluated using the same data, metrics, and experimental conditions. This standardization eliminates inconsistencies that could arise from differences in datasets or evaluation criteria, providing a fair and consistent basis for comparison. It will be easier to attribute performance differences to the algorithms themselves rather than external factors while using a standard experimental setup. It will be also easier to maintain reproducibility in anomaly detection research, as it allows experiments to be replicated with identical conditions, ensuring the reliability of performance outcomes. It also serves as a benchmarking platform where various ML models can be evaluated and ranked based on consistent metrics, offering a clear comparison of their strengths. Additionally, it controls important variables, such as noise or class imbalance, to ensure that the comparison focuses on the algorithms themselves rather than external factors. This streamlined setup enables efficient evaluation and helps test models under realistic conditions, simulating the challenges they would face in real-world scenarios.

A computer with i7 8th generation processor backed by 16GB memory, and 1TB NVMe M2 storage is used to write the source codes and to evaluate the performance of the compared methods. A dedicated User Interface (UI) is designed to load the datasets, communicate with a simulator, execute the compared methods sequentially, collect the experimental log results from the simulator, and to generate the graphs. Visual Studio IDE is used to develop the UI and C+ 20.0 is used to code the compared algorithms. Benchmark datasets in specific KDD-Cup and UNSW-NB15 are used in the evaluation process. The industrial standard simulator OPNET is used to evaluate the performance of the discussed methods. The experimental setup structural diagram is provided in Figure 6.

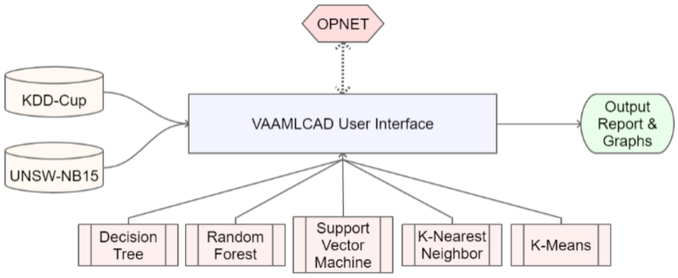

Figure 6: Experimental Setup

4. RESULTS AND DISCUSSIONS

During the evaluation process, elementary parameters such as True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) are measured for different anomaly classes for every 10% of data of KDD-Cup, and UNSW-NB15 dataset individually. Based on the readings, the benchmark anomaly detection performance metrics such as Accuracy, Precision, Sensitivity, Specificity, and F-Score values are computed. Average anomaly detection time is also measured during the entire evaluation process for the nominated methods. The observed readings are tabulated in this section along with the comparison graphs.

4.1. Accuracy

Anomaly detection accuracy in cloud computing is crucial for ensuring security, performance, and cost-efficiency. It minimizes false positives and false negatives, preventing unnecessary interventions and undetected issues that could lead to service disruptions or security breaches. By optimizing resource usage and detecting performance bottlenecks early, it helps maintain high availability and improved user experience. Accuracy is calculated by the formula Accuracy $= \frac{TP + TN}{TP + TN + FP + FN}$

Measured Accuracy of the methods for KDD-Cup, and UNSW-NB15 datasets are listed in Table 1 and Table 2 Respectively.

Table 1: Accuracy (KDD-Cup Dataset)

Accuracy (%) [KDD-Cup]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	96.28799	94.159	95.786	96.467	95.380997
20	96.327	94.14	95.811	96.475	95.464989
30	96.30299	93.99699	95.752	96.393	95.331009
40	96.347	94.06999	95.85201	96.48799	95.363998
50	96.33299	94.246	95.842	96.424	95.434998
60	96.332	94.302	95.898	96.476	95.447006
70	96.319	94.19801	95.92	96.44401	95.310005
80	96.328	94.25301	95.827	96.40399	95.463997
90	96.341	94.023	95.748	96.427	95.423996
100	96.33501	94.18099	95.862	96.42701	95.358994

Based on the observed results, the rank-wise performance on the KDD-Cup dataset, KNNemerges as the top-performing model, with accuracy ranging from 96.393% to 96.48799%, slightly outperforming Random Forest. RF follows closely in second place, maintaining strong and stable accuracy between 96.28799% and 96.347% across all data percentages. SVM ranks third, with accuracy ranging from 95.748% to 95.920%, showing solid performance but slightly below KNN and RF. In fourth place is K-Means, which achieves accuracy between 95.310005% and 95.464989%, consistently lower than the top two models. DT ranks last, with accuracy ranging from 93.99699% to 94.302%, making it the weakest performer among the models tested.

Table 2: Accuracy (UNSW-NB15 DATASET)

Accuracy(%) [UNSW-NB15]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	93.136	90.9	92.519	92.899	92.310997
20	92.80901	90.56	92.878	92.88001	92.378006
30	92.92799	90.791	92.39101	92.92101	92.219994
40	93.176	90.92999	92.481	92.815	92.385994
50	93.071	90.548	92.44	92.788	92.228989
60	93.103	90.828	92.384	93.035	92.402
70	93.06599	90.93001	92.481	92.79299	92.365005
80	92.89201	91.039	92.605	92.965	92.416
90	93.051	91.045	92.385	92.979	92.324997
100	93.12	90.95799	92.484	92.85899	92.353996

As per the observed results, the rank-wise performance on the UNSW-NB15 dataset, RF ranks first, consistently achieving the highest accuracy, ranging from 92.80901% to 93.176%.KNNfollows in second place, with accuracy between 92.788% and 93.035%, closely trailing RF. In third place is SVM, with accuracy ranging from 92.384% to 92.878%, performing well but slightly below KNN and RF. K-Means ranks fourth, with accuracy between 92.219994% and 92.416%, maintaining lower but consistent accuracy. DT ranks last, with accuracy from 90.548% to 91.045%, making it the weakest model in the comparison.
The comparison graphs of Accuracy score for KDD-Cup, and UNSW-NB15 datasets are given in Figure 7 and 8 respectively.

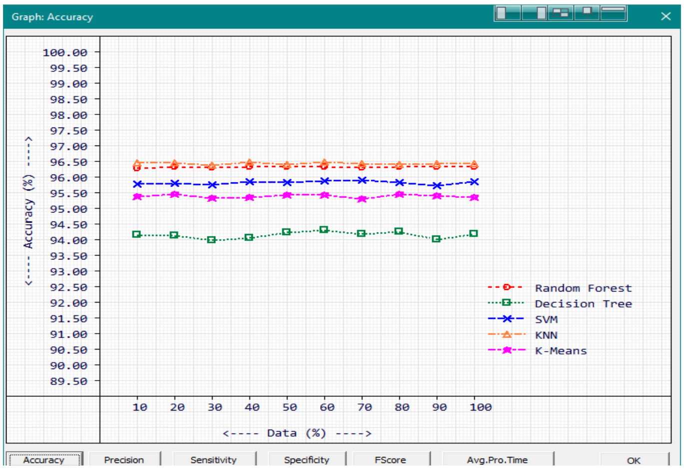

Figure 7: Accuracy (KDD-Cup Dataset)

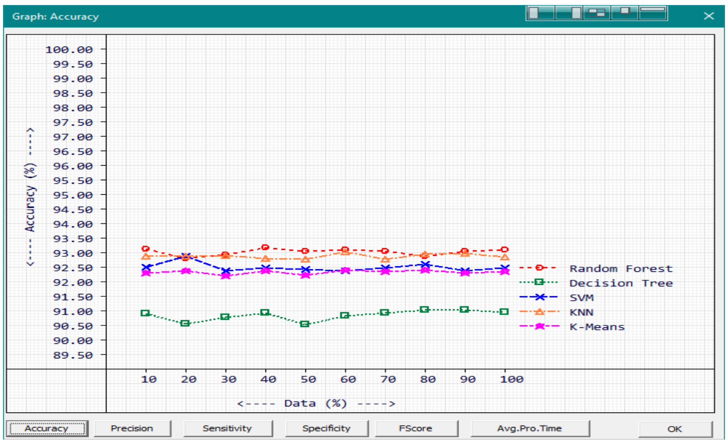

Figure 8: Accuracy (UNSW-NB15 dataset)

In both the KDD-Cup and UNSW-NB15 datasets, RF and KNN consistently outperformed the other models, achieving the highest accuracy rates. SVM followed closely, showing competitive but slightly lower performance, while K-Means and DT followed behind. Overall, RF and KNN proved to be the most reliable models across both datasets for accurate anomaly detection.

4.2.Precision

Precision is one of the crucial factors in cloud anomaly detection because it directly impacts the efficiency of the system, resource management, and security. High precision ensures that detected anomalies are actually significant issues, minimizing false positives. False positives in cloud environments can lead to unnecessary interventions, such as triggering costly recovery processes or reallocating resources when no real threat exists. This wastes computational power and increases operational costs. In addition, precision is critical for maintaining trust in automated anomaly detection systems. If a system constantly flags benign activities as anomalies, administrators may begin to ignore alerts, potentially missing actual critical issues. This can compromise the security and performance of cloud systems. Therefore, high precision allows cloud services to respond only to genuine threats or irregularities, optimizing resource use, reducing downtime, and improving overall system reliability. Precision is calculated using the formula $\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$.. Observed precision values for KDD-Cup and UNSW-NB15 datasets are provided in Table 3, and 4 respectively.

Table 3: Precision (KDD-Cup dataset)

Precision (%) [KDD-Cup]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	96.07	94.76601	95.894	96.334	95.853996
20	96.196	94.632	95.794	96.29601	96.036003
30	96.106	94.556	95.82201	96.162	95.841995
40	96.14	94.74199	95.888	96.34599	95.917999
50	96.164	94.87399	95.91	96.25	95.954002
60	96.17799	94.992	95.918	96.37199	95.949997
70	96.15199	94.872	95.994	96.284	95.860001
80	96.174	94.946	95.83199	96.21	95.939995
90	96.174	94.658	95.718	96.302	95.852005
100	96.15801	94.77999	95.896	96.26601	95.93

Table 4: Precision (UNSW-NB15 dataset)

Precision(%) [UNSW-NB15]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	92.60799	91.21999	92.468	93.18	92.491997
20	92.388	91.09599	93.08801	93.006	92.774002
30	92.426	91.034	92.472	93.11	92.348007
40	92.67201	91.14	92.626	92.818	92.573997
50	92.788	90.744	92.65601	92.98399	92.496002
60	92.754	91.264	92.594	93.062	92.339996
70	92.732	91.332	92.788	92.898	92.477997
80	92.438	91.316	92.65199	93.084	92.552002
90	92.584	91.43199	92.536	92.98	92.584
100	92.838	91.308	92.746	92.90199	92.603996

Based on precision-based rank-wise performance on the KDD-Cup dataset, KNN ranks first, achieving the highest precision values between 96.162% and 96.37199%, demonstrating its strong capability in accurately identifying true positives. RFfollows closely in second place, with precision ranging from 96.07% to 96.196%, also showing high reliability in minimizing false positives. In third place is SVM, with precision values between 95.718% and 95.918%, maintaining solid performance but slightly lower than KNN and RF. K-Means ranks fourth, with precision between 95.841995% and 96.036003%, consistently lower than the top three models. DT ranks last, with precision ranging from 94.556% to 94.992%, making it the least reliable model for minimizing false positives in this dataset.

Based on precision-based rank-wise performance on the UNSW-NB15 dataset, KNN ranks first, achieving the highest precision values between 92.60799% and 93.18%, showcasing its effectiveness in accurately identifying true positives. RF follows closely in second place, with precision ranging from 92.388% to 92.838%, demonstrating strong reliability in minimizing false positives.The comparison graphs of precision scores for KDD-Cup, and UNSW-NB15 datasets are plotted in Figure 9, and 10 in order.

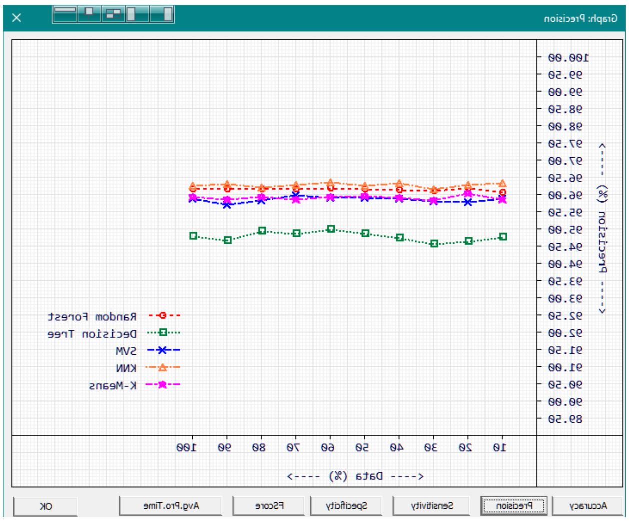

Figure 9: Precision (KDD-Cup dataset)

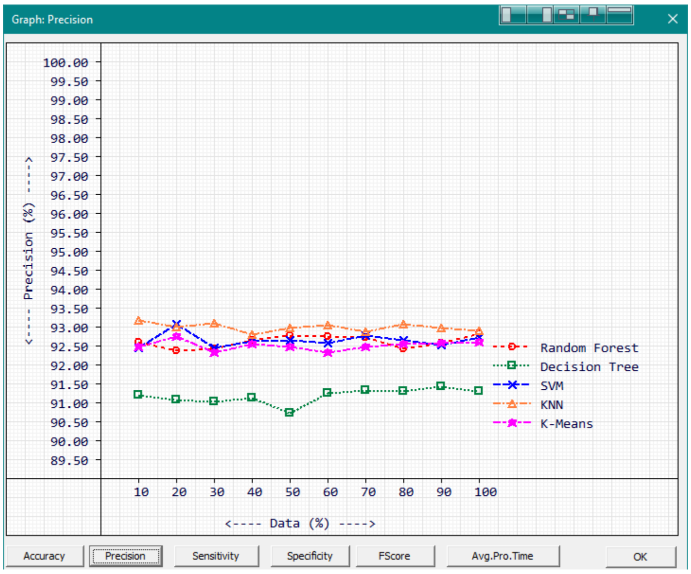

Figure 10: Precision (UNSW-NB15 dataset)

4.3. Sensitivity

Sensitivity, also known as recall or True Positive Rate, is crucial in cloud anomaly detection because it measures the ability of the system to correctly identify true positive cases among all actual anomalies. High sensitivity ensures that most genuine threats are detected, which is vital for maintaining security and performance in cloud environments. Missing out on actual anomalies can lead to severe consequences, including data breaches, service disruptions, or operational inefficiencies.In a cloud setting, where resources are shared and can be vulnerable to various attacks or faults, timely detection of anomalies is essential to mitigate risks. If a detection system has low sensitivity, it may overlook significant security threats or performance issues, allowing them to escalate and potentially cause extensive damage. Therefore, ensuring high sensitivity helps organizations maintain a robust security and a reliable infrastructure, safeguarding both data and user trust. Sensitivity is calculated using the formula $\text{Sensitivity} = \frac{\text{TP}}{\text{TP}+\text{FN}}$.

The sensitivity scores of the compared methods for KDD-Cup, and UNSW-NB15 datasets are given in Table 5, and 6 in sequence.

Table 5: Sensitivity (KDD-Cup dataset)

Sensitivity (%) [KDDCup]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	96.49328	93.62907	95.68647	96.59345	94.955734
20	96.44968	93.71376	95.82646	96.64582	94.953079
30	96.48663	93.51475	95.68687	96.6114	94.874252
40	96.54201	93.48601	95.81899	96.62405	94.871445
50	96.4931	93.70354	95.77749	96.58931	94.968895
60	96.47837	93.70207	95.87975	96.57666	94.996315
70	96.47564	93.61726	95.85158	96.59581	94.819748
80	96.47258	93.65509	95.82142	96.58907	95.034378
90	96.50028	93.47191	95.77269	96.54839	95.039566
100	96.49973	93.65746	95.8293	96.57977	94.847717

Table 6: Sensitivity (UNSWW-NB115 dataset)

Sensitivity(%) [UNSW-NB15]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	93.59672	90.64452	92.56963	92.66118	92.1521
20	93.17195	90.1296	92.70126	92.77561	92.043839
30	93.36218	90.59098	92.3245	92.7622	92.110489
40	93.61725	90.76654	92.35832	92.81828	92.224243
50	93.31329	90.39233	92.26644	92.62978	91.999939
60	93.40718	90.47714	92.20454	93.01881	92.455292
70	93.35405	90.60658	92.21738	92.71344	92.264832
80	93.28636	90.8062	92.56751	92.86305	92.295319
90	93.45223	90.7253	92.25655	92.98363	92.103645
100	93.36708	90.68548	92.25791	92.83181	92.140121

The corresponding comparison graphs are provided in Figure 11, and 12.

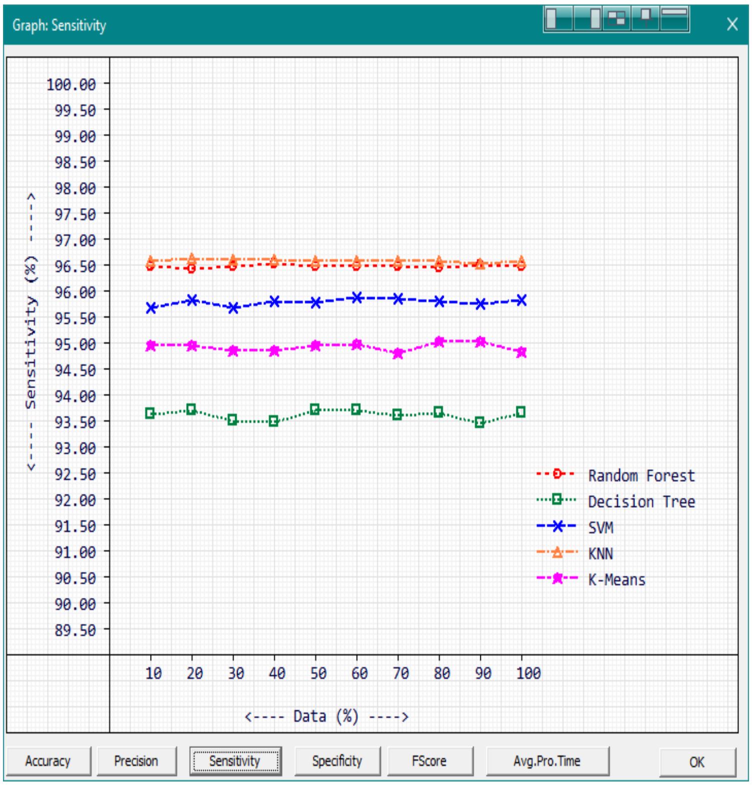

Figure 11: Sensitivity (KDD-Cup dataset)

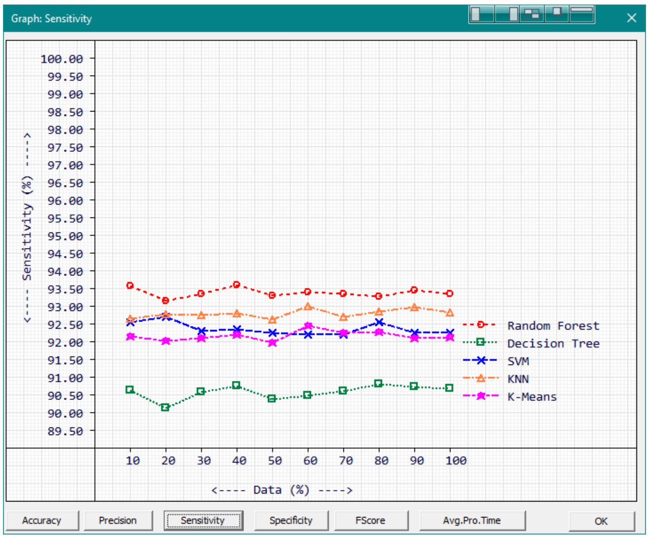

Figure 12: Sensitivity (UNSW-NB15 dataset)

Based on rank-wise performance for sensitivity on the KDD-Cup dataset, KNN ranks first, achieving the highest sensitivity values ranging from 96.59345% to 96.64582%, indicating its exceptional ability to identify true positive cases. RF follows closely in second place, with sensitivity ranging from 96.4931% to 96.54201%, showcasing strong performance in anomaly detection. SVM ranks third, with sensitivity values between 95.68647% and 95.87975%, demonstrating solid capability but slightly lower than KNN and RF. K-Means comes in fourth, achieving sensitivity between 94.874252% and 95.034378%, consistently lower than the top three models.

In the sensitivity evaluation of the UNSW-NB15 dataset, RF leads the rankings with sensitivity scores between 93.17195% and 93.61725%, highlighting its strong capability to accurately detect true positives. Following closely in second place is KNN, which achieves sensitivity values from 92.66118% to 93.01881%, reflecting its reliable performance in identifying anomalies. SVM is in the third rank with sensitivity figures ranging from 92.20454% to 92.70126%, showing decent performance, though it falls slightly short compared to RF and KNN. K-Means occupies the fourth position, obtaining sensitivity values between 91.999939% and 92.455292%, consistently lower than the top three models.

4.4. Specificity

Specificity is important in cloud anomaly detection since it helps the system to correctly identify normal behavior and avoid mistakenly flagging regular activities as anomalies. High specificity reduces false positives, which are unnecessary alerts, and prevents cloud administrators from being overwhelmed by irrelevant notifications. In a busy cloud environment, frequent false alarms can lead to wasted resources, increased costs, and service disruptions. By improving specificity, the detection system becomes more reliable and efficient, allowing administrators to focus on real threats and keeping the cloud environment running smoothly. Specificity is computed using the formula. Specificity $= \frac{TN}{FP+TN}$. The Specificity scores of compared methods for KDD-Cup, and UNSW-NB15 datasetsare provided in Table 7, and in Table 8. Similarly associated graphs are provided in Figure 13 and 14.

Table 7: Specificity (KDD-Cup dataset)

Specificity (%) [KDD-Cup]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	96.0848	94.7047	95.88738	96.34187	95.814713
20	96.2052	94.58022	95.79604	96.30622	95.9897
30	96.12161	94.49641	95.81886	96.17766	95.797592
40	96.15457	94.67403	95.88677	96.35432	95.868301
50	96.17471	94.80737	95.90694	96.261	95.911263
60	96.18754	94.91976	95.91731	96.37685	95.907318
70	96.16451	94.79823	95.9899	96.29392	95.811874
80	96.18461	94.86934	95.83278	96.22204	95.902878
90	96.18361	94.59116	95.72444	96.30788	95.815231
100	96.17199	94.7206	95.8953	96.27605	95.882385

Table 8: specificity (UNSW-NB15 dataset)

Specificity(%) [UNSW-NB15]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	92.68544	91.1611	92.4701	93.14094	92.474915
20	92.45404	91.00446	93.05644	92.98769	92.719887
30	92.50581	91.00219	92.45847	93.08264	92.330307
40	92.74544	91.097	92.60484	92.81691	92.551071
50	92.83453	90.70573	92.6181	92.95046	92.464874
60	92.80314	91.18588	92.56584	93.05381	92.350174
70	92.78529	91.26647	92.7499	92.87705	92.467758
80	92.50872	91.28252	92.64398	93.06797	92.539795
90	92.66332	91.37217	92.5154	92.97701	92.550667
100	92.8773	91.23849	92.71478	92.89405	92.572495

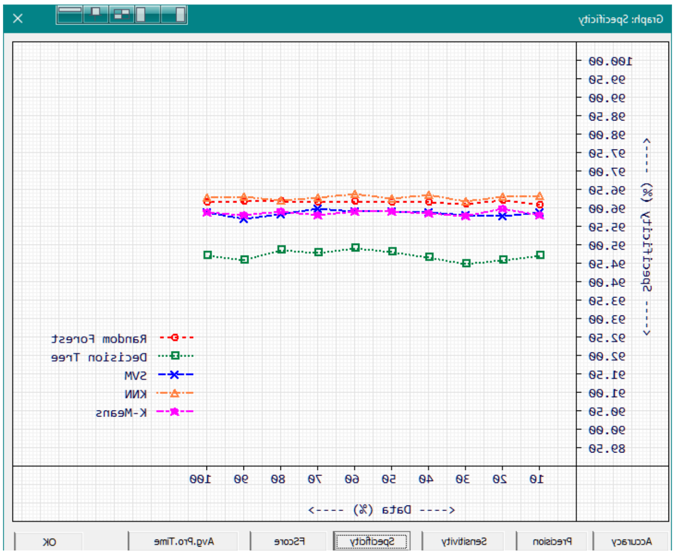

Figure 13: Specificity (KDD-Cup dataset)

In terms of specificity for the KDD-Cup dataset, KNN performs the best, with values ranging from 96.18% to 96.38%, showing its strong ability to accurately identify normal behavior. RF follows closely in second place, with specificity values between 96.08% and 96.21%, reflecting its reliable performance in distinguishing normal operations from anomalies. K-Means ranks third, with values between 95.80% and 95.99%, slightly lower but still effective. SVM comes in fourth, with specificity values ranging from 95.72% to 95.92%, indicating good performance but lower than the top three models. DT demonstrates specificity values between 94.50% and 94.92%, suggesting that it may be slightly less effective at distinguishing normal cases compared to some of the other models in this dataset.

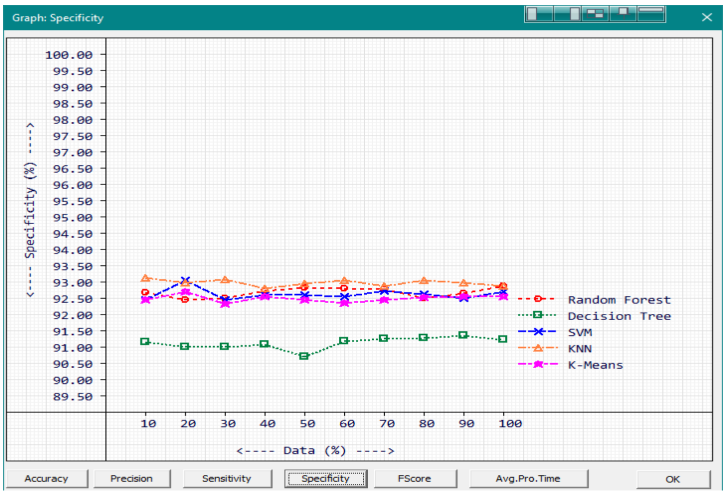

Figure 14: Specificity (UNSW-NB15 dataset)

While using UNSW-NB15 dataset, KNN demonstrates the highest specificity, ranging from 92.82% to 93.14%, reflecting its strong ability to correctly identify normal behavior. RF also performs reliably, with values between 92.45% and 92.88%, showcasing its consistent effectiveness. SVM shows solid performance, with specificity values ranging from 92.46% to 93.06%. K-Means follows closely, maintaining good results with values from 92.33% to 92.72%. DT method managed to get the specificity score values between 90.70% and 91.28%.
In both the KDD-Cup and UNSW-NB15 datasets, K-Nearest Neighbors (KNN) regularly attains the highest specificity, which shows its strong ability to accurately identify normal behavior in various situations. Random Forest (RF) also shows solid performance in both datasets, coming in second place. This indicates that RF is dependable when it comes to telling the difference between normal data and anomalies.

4.5. F-Score

The F-Score, which combines precision and recall into a single metric, is crucial in cloud anomaly detection for several reasons. First, it provides a balanced view of a model's performance by considering both false positives and false negatives. This balance is particularly important in cloud environments, where misclassifying normal activities as anomalies (false positives) can lead to unnecessary alerts and wasted resources, while failing to detect actual anomalies (false negatives) can result in security breaches or system failures. F-Score is computed using the formula $\mathrm{FScore} = 2 \times \frac{(\mathrm{Recall}\ \times \mathrm{Precision}\ )}{(\mathrm{Recall}\ + \mathrm{Precision}\ )}$. Calculated F-Score values based on the observations for KDD-Cup, and UNS-NB15 datasets are provided in Table 9 and 10, as well as the graphs are presented as Figure 15 and 16 proportionally.

Table 9: F-Score (KDD-Cup dataset)

F-Score (%) [KDD-Cup]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	96.2811	94.19342	95.78975	96.46338	95.402664
20	96.32262	94.16923	95.8101	96.47038	95.491226
30	96.29572	94.03089	95.75398	96.3859	95.355553
40	96.34033	94.10878	95.85304	96.48441	95.391647
50	96.32808	94.28382	95.84364	96.41904	95.458809
60	96.32768	94.34234	95.89861	96.47394	95.470642
70	96.31326	94.23949	95.92241	96.43945	95.336845
80	96.32298	94.29567	95.82666	96.39873	95.484795
90	96.33666	94.06049	95.74504	96.42461	95.444
100	96.32842	94.21454	95.8625	96.42241	95.385719

Table 10: F-Score (UNSWW-NB15)

F-Score(%) [UNSW-NB15]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	93.09959	90.93071	92.51835	92.91948	92.320732
20	92.77778	90.60897	92.89417	92.88991	92.40686
30	92.89079	90.80946	92.39799	92.93534	92.229027
40	93.14171	90.95227	92.49184	92.81676	92.398239
50	93.04906	90.56759	92.45998	92.80574	92.246162
60	93.07941	90.86863	92.39864	93.03971	92.39724
70	93.04101	90.96578	92.5013	92.8045	92.370728
80	92.85913	91.05819	92.60935	92.97324	92.422829
90	93.01438	91.07661	92.39575	92.9811	92.342667
100	93.10135	90.99451	92.50069	92.86481	92.370834

Based on the F-Score performance for the KDD-Cup dataset, RF consistently outperforms the other algorithms, achieving the highest F-Score values that range from 96.2811% to 96.34033%. This indicates its strong ability to balance precision and recall effectively. Following RF, KNN ranks second, with F-Scores between 96.3859% and 96.46338%, demonstrating solid performance in accurately detecting anomalies while maintaining a good rate of correct normal behavior identification.SVM comes in third place, with F-Scores ranging from 95.75398% to 95.89861%, showing its effectiveness but slightly trailing behind RF and KNN. K-Means has consistent performance as well, with F-Scores between

95.355553% and 95.491226%, though it falls short compared to the top three methods. Overall, the results highlight Random Forest as the most reliable choice for anomaly detection in this dataset, with KNN as a strong alternative.

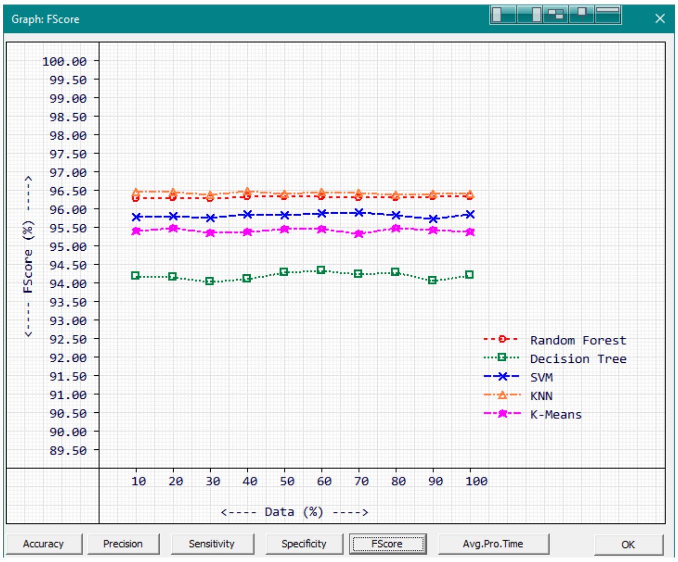

Figure 15: F-Score (KDD-Cup dataset)

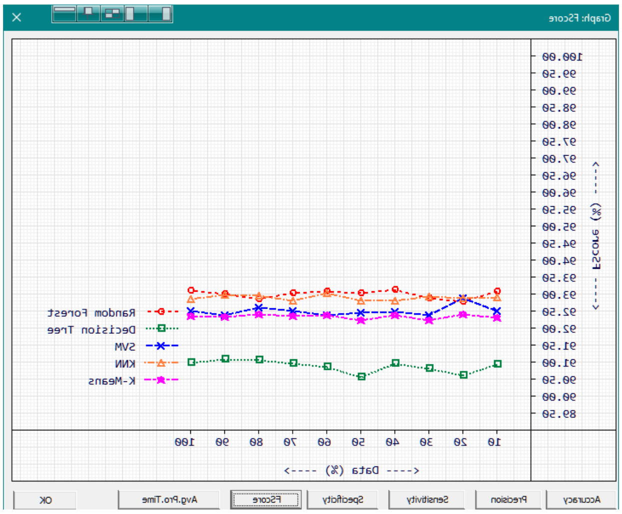

Figure 16: F-Score (UNSW-NB15 dataset)

In the F-Score performance evaluation for the UNSW-NB15 dataset, RF demonstrates the highest effectiveness, with F-Scores ranging from 92.77778% to 93.14171%. This shows its strong ability to balance precision and recall in identifying anomalies. Following closely is KNN, which achieves F-Scores between 92.8045% and 92.97324%, highlighting its solid performance in correctly detecting both anomalies and normal behavior.SVM ranks third, with F-Scores ranging from 92.39799% to 92.89417%, indicating its competence in anomaly detection, although it falls slightly behind RF and KNN. K-Means presents a consistent but lower performance with F-Scores between 92.229027% and 92.40686%, suggesting that while it is functional, it does not perform as effectively as the top three methods. Overall, Random Forest emerges as the leading choice for anomaly detection in the UNSW-NB15 dataset, closely followed by KNN.

4.6. Average Anomaly Detection Time

Average anomaly detection time one of the very important factors in cloud anomaly detection since it enables real-time responses to potential disruptions or attacks, ensuring the reliability and availability of cloud services. Efficient detection minimizes delays, maintaining optimal system performance and enhancing user satisfaction. Additionally, lower detection times facilitate better resource management and scalability while reducing costs associated with service downtime, data loss, or security breaches. By prioritizing quicker detection, organizations can adopt a proactive approach to addressing threats, ultimately enhancing security and operational resilience. Overall, minimizing detection time is essential for maintaining efficient and secure cloud environments. Average anomaly detection time is calculated using the formula

$$\text{Average anomaly detection time} = \frac{\text{Total time for anomaly detection}}{\text{Number of Anomalies detected}}.$$

Measured anomaly detection times for KDD-Cup, and UNSW-NB15 datasets are listed in Table 11 and in Table 12, as well as the comparison graphs for the same are providedin Figure 17, and Figure 18 respectively.

Table 11: Average Detection Time (KDD-Cup dataset)

Average Detection Time(mS) [KDD-Cup]					
Data	**RF**	**DT**	SVM	**KNN**	**K-Means**
10	678	571	636	568	725
20	660	574	646	593	742
30	673	573	641	577	712
40	681	574	631	584	722
50	697	566	635	596	734
60	672	580	631	593	737
70	690	576	640	595	719
80	694	572	650	595	727
90	688	567	620	600	718
100	669	570	649	587	721

Table 12: Average Detection Time (UNSW-NB15dataset)

Average Detection Time (mS) [UNSW-NB15]					
Data	**RF**	**DT**	**SVM**	**KNN**	**K-Means**
10	935	797	894	838	983
20	932	827	890	840	992
30	922	824	906	844	998
40	919	822	874	819	978
50	924	819	902	848	995
60	927	821	901	816	983
70	936	826	904	827	980
80	929	806	896	838	994
90	932	829	894	821	993
100	938	810	914	821	966

Based on the average detection time in milliseconds (mS) for the KDD-Cup dataset, KNN demonstrates the fastest performance overall, with values consistently around 568 to 600 mS across various data points. DT follows closely behind, maintaining a competitive average detection time ranging from 566 to 580 mS, indicating its efficiency in detecting anomalies. RF has an average detection time between 660 to 697 mS, showcasing its reliability but with slightly longer processing times than KNN and DT. SVM exhibits average times from 620 to 646 mS, showing good performance but still slower than the top-performing methods. Lastly, K-Means has the longest detection times, ranging from 712 to 742 mS, indicating it may require more time to identify anomalies compared to the other methods. This analysis highlights the varying detection times across different algorithms, emphasizing the trade-off between speed and detection capability in cloud anomaly detection.

For the UNSW-NB15 dataset, the average detection time in milliseconds (mS) reveals that DT has the best performance, with detection times ranging from 797 mS to 827 mS, making it the fastest algorithm for detecting anomalies in this dataset. Following closely is the KNN algorithm, with average times from 816 mS to 844 mS, indicating its efficiency as well. RF has slightly longer detection times, averaging between 919 mS and 938 mS, but still performs adequately. The SVM method shows average detection times ranging from 874 mS to 914 mS, suggesting a good performance but at a slower pace compared to DT and KNN. Lastly, K-Means consistently records the longest detection times, ranging from 966 mS to 998 mS, indicating that it may be less optimal for real-time anomaly detection scenarios. This summary illustrates the trade-offs in detection speed among various algorithms used for anomaly detection in cloud environments.

In summary, the performance of various anomaly detection algorithms in terms of average detection time reveals distinct strengths across the KDD-Cup and UNSW-NB15 datasets. In the KDD-Cup dataset, KNN emerges as the fastest method, followed closely by RF and SVM, indicating that these algorithms can effectively balance speed and accuracy. Conversely, in the UNSW-NB15 dataset, DT demonstrates the quickest detection times, making it the most efficient choice for real-time applications, while K-Means shows the longest detection times, which may limit its practical use in time-sensitive scenarios. Overall, the choice of algorithm for cloud anomaly detection should consider not only accuracy and precision but also the average detection time, which is crucial for timely responses to potential threats.

5. CONCLUSION

The performance of nominated anomaly detection methods across the key parameters such as accuracy, precision, sensitivity, specificity, F-Score, and average detection time—offers a comprehensive view of their effectiveness in both the KDD-Cup and UNSW-NB15 datasets. Random Forest (RF) consistently outperformed other methods, achieving the highest accuracy in both datasets, underscoring its robustness in identifying both normal and anomalous instances. K-Nearest Neighbors (KNN) also demonstrated strong accuracy, closely following RF, indicating its reliability. In terms of precision, both RF and KNN led the pack, showcasing their effectiveness in minimizing false positives, which is crucial in cloud environments to avoid unnecessary resource consumption. KNN excelled in sensitivity, effectively identifying true positives across both datasets, while RF also performed well, ensuring critical issues are recognized promptly. KNN consistently achieved the highest specificity, proving its strength in accurately identifying normal behavior, with RF following closely. The F-Score metrics revealed that both RF and KNN maintain a strong balance between precision and recall, achieving top scores that reflect their ability to detect anomalies while minimizing false positives. When it comes to average detection time, KNN was the fastest in the KDD-Cup dataset, indicating its efficiency in processing data, while Decision Tree (DT) showed the quickest detection times in the UNSW-NB15 dataset. But the advantage of using DT is diminished due to the subpar performance in all other benchmark parameters excluding average detection time. Thus it is understood that RF and KNN exhibited superior performance across most parameters, the choice of an anomaly detection method should consider the specific application requirements, including the trade-off between speed and accuracy in real-time environments.

Conflict of Interest

There is no conflict of interest between the authors in terms of conducted evaluations and conclusions.

Code availability

Complete source code are available online, and download link will be provided based on E-Mail request to the authors.

REFERENCES

[1] Liagkou, V., Fragiadakis, G., Filiopoulou, E. et al. Assessing the Complexity of Cloud Pricing Policies: A Comparative Market Analysis. J Grid Computing 22, 65 (2024). https://doi.org/10.1007/s10723-024-09780-4

[2] N. Santos, B. Ghita and G. L. Masala, "Medical Systems Data Security and Biometric Authentication in Public Cloud Servers," in IEEE Transactions on Emerging Topics in Computing, vol. 12, no. 2, pp. 572-582, April-June 2024, https://doi.org/10.1109/TETC.2023.3271957

[3] Vassilev, V. et al. (2023). Data Platform and Urban Data Services on Private Cloud. In: Senjyu, T., So-In, C., Joshi, A. (eds) Smart Trends in Computing and Communications. SmartCom 2023. Lecture Notes in Networks and Systems, vol 650. Springer, Singapore. https://doi.org/10.1007/978-981-99-0838-7_23

[4] Rahim, Robbi. "Quantum Computing in Communication Engineering: Potential and Practical Implementation." Progress in Electronics and Communication Engineering 1.1 (2024): 26-31.

[5] Zaixing Sun, Hejiao Huang, Zhikai Li, ChonglinGu, RuitaoXie, Bin Qian, Efficient, economical and energy-saving multi-workflow scheduling in hybrid cloud, Expert Systems with Applications, Volume 228, 2023, 120401, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2023.120401

[6] Alonso, J., Orue-Echevarria, L., Casola, V. et al. Understanding the challenges and novel architectural models of multi-cloud native applications – a systematic literature review. J Cloud Comp 12, 6 (2023). https://doi.org/10.1186/s13677-022-00367-6

[7] BhavanaGodavarthi, NirmalajyothiNarisetty, KalpanaGudikandhula, R. Muthukumaran, Dhiraj Kapila, J.V.N. Ramesh, Cloud computing enabled business model innovation, The Journal of High Technology Management Research, Volume 34, Issue 2, 2023, 100469, ISSN 1047-8310, https://doi.org/10.1016/j.hitech.2023.100469
[8] Zähl, P.M., Theis, S., Wolf, M.R., Köhler, K. (2023). Teamwork in Software Development and What Personality Has to Do with It - An Overview. In: Chen, J.Y.C., Fragomeni, G. (eds) Virtual, Augmented and Mixed Reality. HCII 2023. Lecture Notes in Computer Science, vol 14027. Springer, Cham. https://doi.org/10.1007/978-3-031-35634-6_10
[9] Mitropoulou, K., Kokkinos, P., Soumplis, P. et al. Anomaly Detection in Cloud Computing using Knowledge Graph Embedding and Machine Learning Mechanisms. J Grid Computing 22, 6 (2024). https://doi.org/10.1007/s10723-023-09727-1
[10] Yunkang Cao, Xiaohao Xu, Weiming Shen, Complementary pseudo multimodal feature for point cloud anomaly detection, Pattern Recognition, Volume 156, 2024, 110761, ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2024.110761
[11] Xu, H., Sun, Z., Cao, Y. et al. A data-driven approach for intrusion and anomaly detection using automated machine learning for the Internet of Things. Soft Comput 27, 14469–14481 (2023). https://doi.org/10.1007/s00500-023-09037-4
[12] Arthur Hoarau, Arnaud Martin, Jean-Christophe Dubois, Yolande Le Gall, Evidential Random Forests, Expert Systems with Applications, Volume 230, 2023, 120652, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2023.120652
[13] Jajarmi, Amin, Samaneh Sadat Sajjadi, and Ahamad Hajipour. "Steam generator identification using piecewise affine model." Results in Nonlinear Analysis 2.4 (2019): 149-159.
[14] Mohsen Bayat Pour, Jonas Niklewski, Amir Naghibi, Eva Frühwald Hansson, Robust probabilistic modelling of mould growth in building envelopes using random forests machine learning algorithm, Building and Environment, Volume 243, 2023, 110703, ISSN 0360-1323, https://doi.org/10.1016/j.buildenv.2023.110703
[15] Daniel Borup, Bent Jesper Christensen, Nicolaj Søndergaard Mühlbach, Mikkel Slot Nielsen, Targeting predictors in random forest regression, International Journal of Forecasting, Volume 39, Issue 2, 2023, Pages 841-868, ISSN 0169-2070, https://doi.org/10.1016/j.ijforecast.2022.02.010
[16] Mafarja, M., Thaher, T., Al-Betar, M.A. et al. Classification framework for faulty-software using enhanced exploratory whale optimizer-based feature selection scheme and random forest ensemble learning. ApplIntell 53, 18715–18757 (2023). https://doi.org/10.1007/s10489-022-04427-x
[17] Ali YA, Awwad EM, Al-Razgan M, Maarouf A. Hyperparameter Search for Machine Learning Algorithms for Optimizing the Computational Complexity. Processes. 2023; 11(2):349. https://doi.org/10.3390/pr11020349
[18] James, G., Witten, D., Hastie, T., Tibshirani, R., Taylor, J. (2023). Tree-Based Methods. In: An Introduction to Statistical Learning. Springer Texts in Statistics. Springer, Cham. https://doi.org/10.1007/978-3-031-38747-0_8
[19] Z. Azam, M. M. Islam and M. N. Huda, "Comparative Analysis of Intrusion Detection Systems and Machine Learning-Based Model Analysis Through Decision Tree," in IEEE Access, vol. 11, pp. 80348-80391, 2023, https://doi.org/10.1109/ACCESS.2023.3296444
[20] Arifuzzaman M, Hasan MR, Toma TJ, Hassan SB, Paul AK. An Advanced Decision Tree-Based Deep Neural Network in Nonlinear Data Classification. Technologies. 2023; 11(1):24. https://doi.org/10.3390/technologies11010024
[21] Atin Roy, Subrata Chakraborty, Support vector machine in structural reliability analysis: A review, Reliability Engineering & System Safety, Volume 233, 2023, 109126, ISSN 0951-8320, https://doi.org/10.1016/j.ress.2023.109126
[22] Quan, Z., Pu, L. An improved accurate classification method for online education resources based on support vector machine (SVM): Algorithm and experiment. Educ Inf Technol 28, 8097–8111 (2023). https://doi.org/10.1007/s10639-022-11514-6
[23] Ukey N, Yang Z, Li B, Zhang G, Hu Y, Zhang W. Survey on Exact kNN Queries over High-Dimensional Data Space. Sensors. 2023; 23(2):629. https://doi.org/10.3390/s23020629
[24] SubhrangshuAdhikary, Saikat Banerjee, Introduction to Distributed Nearest Hash: On Further Optimizing Cloud Based Distributed kNN Variant, Procedia Computer Science, Volume 218, 2023, Pages 1571-1580, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2023.01.135
[25] Xin, R., Liu, H., Chen, P. et al. Robust and accurate performance anomaly detection and prediction for cloud applications: a novel ensemble learning-based framework. J Cloud Comp 12, 7 (2023). https://doi.org/10.1186/s13677-022-00383-6

[26] S. M. Miraftabzadeh, C. G. Colombo, M. Longo and F. Foiadelli, "K-Means and Alternative Clustering Methods in Modern Power Systems," in IEEE Access, vol. 11, pp. 119596-119633, 2023, https://doi.org/10.1109/ACCESS.2023.3327640

[27] Li, M., Frank, E. &Pfahringer, B. Large scale K-means clustering using GPUs. Data Min Knowl Disc 37, 67–109 (2023). https://doi.org/10.1007/s10618-022-00869-6