JIKO (Jurnal Informatika dan Komputer) Accredited KEMDIKTISAINTEK, No.0173/C3/DT.05.00/2025
Vol. 9, No. 2, August 2026, pp. 145-153 p-ISSN: 2614-8897
DOI: 10.33387/jiko.v9i2.11875 e-ISSN: 2656-1948

# ANALYSIS ON MACHINE LEARNING MODELS ROBUSTNESS AGAINST NOISE AND CONCEPT DRIFT IN THE CICIDS2017 DATASET

**Azhar Bintang Pramudyanto[1], Chyntia Raras Ajeng Widiawati[2], M. Syaiful Amin[3]**

[1,2,3]Faculty of Computer Science, Amikom Purwokerto University, Purwokerto, 53127, Indonesia
Email: [1]azharbintang0@gmail.com, [2]chyntiaraw@amikompurwokerto.ac.id,
[3]syaifulamin@amikompurwokerto.ac.id

(Received: 11 May 2026, Revised: 26 May 2026, Accepted: 8 July 2026)

**Abstract**

Intrusion Detection Systems (IDS) based on machine learning face significant challenges in real-world deployment due to noise in network data and concept drift caused by evolving attack patterns. This study analyzes the robustness of three machine learning models Logistic Regression, Decision Tree, and Random Forest against noise and concept drift using the CICIDS2017 dataset. The experimental design includes baseline testing, noise robustness testing with three intensity levels (5%, 10%, 20%), and concept drift evaluation through temporal data splitting. Performance is measured using accuracy, precision, recall, and F1-score metrics, with robustness score calculated as a weighted average (40% baseline, 30% noise, 30% drift). Results show Logistic Regression achieves the highest robustness score (60.07%) due to excellent noise tolerance (1.09% F1-score degradation), followed by Random Forest (59.65%) and Decision Tree (57.60%). However, all models are categorized as NOT ROBUST against concept drift with degradation ranging from 29.64% to 30.34%, indicating sudden drift between Thursday and Friday data. The findings reveal that noise robustness and concept drift robustness are independent characteristics that do not correlate. Logistic Regression is recommended for practical deployment due to its optimal combination of robustness, interpretability, and computational efficiency. This research contributes to understanding model stability under non-ideal data conditions and emphasizes the necessity of implementing adaptive mechanisms such as periodic retraining and drift detection in operational IDS architecture.

**Keywords**: Intrusion Detection System, Machine Learning, Robustness Analysis, Concept Drift, CICIDS2017

*This is an open access article under the CC BY license.*

**Corresponding Author: Azhar Bintang Pramudyanto*

## 1. INTRODUCTION

Network security is becoming increasingly critical as complex cyber activities rise. In 2023, BSSN recorded 403,990,813 cyberattacks, and the 2024 attack on the National Data Center demonstrates that traditional intrusion detection systems are not robust enough [1]. Machine learning-based Intrusion Detection Systems (IDS) have been developed to detect new attack patterns, but most research focuses only on accuracy under ideal conditions, even though real-world network data often contains noise and experiences concept drift [2].

Concept drift refers to the phenomenon in which the statistical properties of the target variable (attack classification) change over time, causing the relationship between input features and output labels to evolve. In the context of network intrusion detection, concept drift manifests when attack patterns, traffic characteristics, or the distribution of attack types shift between the training and deployment periods. This can occur through several mechanisms, including sudden drift (abrupt changes, such as the emergence of new attack campaigns), gradual drift (the slow evolution of existing attack techniques), incremental drift (continuous small changes over an extended period), or recurring concepts (cyclical patterns that reappear) [3]. Unlike noise, which represents random perturbations in feature values, concept drift is a systematic change in the underlying data distribution that fundamentally challenges the model's ability to maintain performance on data differing from its training distribution. For static machine learning models, concept drift is highly problematic because the learned decision boundary becomes suboptimal or even invalid when the data distribution shifts, requiring adaptive mechanisms such as periodic retraining or online learning to maintain operational effectiveness [4].

To address these threats, Intrusion Detection Systems (IDS) were developed as the primary mechanism for detecting suspicious activity on computer networks. With the continuous advancement of technology [5], Machine Learning (ML) based IDS have become widely adopted due to their ability to recognize new patterns of network attacks [6]. Machine Learning algorithms such as Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF) have become models frequently used in classifying network attack traffic because they are considered effective at automatically identifying attacks without the need for explicit rules, as is the case with signature-based systems [7].

However, most research focuses on improving model accuracy on ideal and static data. In reality, network data often undergoes changes in distribution over time. In addition to being variable, data often contains noise, whether due to recording errors, anomalies, or inconsistent traffic variation [8]. This can lead to a significant decline in model performance. In dynamic network environments, the presence of noise and the phenomenon of concept drift are unavoidable and, in fact, pose the primary challenges for the real-world deployment of machine learning models [9], [10].

The study by Catillo et al. (2023) tested the models' resilience to adversarial attacks on the CICIDS2017 dataset and found that Decision Trees are vulnerable to input manipulation, while Deep Autoencoders are more robust. However, the study did not examine the models' resilience to natural noise and concept drift, which are common in real-world environments. Furthermore, studies systematically comparing the robustness of Logistic Regression, Decision Trees, and Random Forests against a combination of these two disturbances on the CICIDS2017 dataset remain limited [11].

Based on the literature review conducted, there are several research gaps that need to be addressed. First, the study by Catillo et al. (2023) focuses on model robustness against adversarial attacks, which involve intentional input manipulation, but has not examined model robustness against natural noise and concept drift, which are more common in the daily operation of networks. Second, most research on IDS still emphasizes improving classification accuracy without considering the model's robustness when facing changes in data distribution over time. Third, research systematically comparing the resilience of Logistic Regression, Decision Tree, and Random Forest models against a combination of noise and concept drift on the CICIDS2017 dataset remains limited. In fact, the three models have different characteristics Logistic Regression is interpretable, Decision Trees can handle non-linear data, and Random Forests are more resistant to overfitting [12], [13], [14].

This study aims to analyze and evaluate the robustness of Logistic Regression, Decision Tree, and Random Forest models against noise and concept drift in the CICIDS2017 dataset, in order to determine which model is the most stable and reliable for detecting network attacks under dynamic and non-ideal data conditions.

## 2. RESEARCH METHOD

This study employs an experimental approach to analyze the robustness of machine learning models against noise and concept drift on the CICIDS2017 dataset. The research design includes a comparison of three classification models Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF) tested across three scenarios a baseline test (without interference), a noise test with three intensity levels (5%, 10%, and 20%), and a concept drift test using a temporal data partitioning scheme. The experiments were designed to measure the decline in model performance using the metrics accuracy, precision, recall, and F1-score to determine the model most robust to changes in data characteristics. The research process begins with data collection, followed by data processing, model development, temporal data splitting, robustness experiments, and finally the compilation of results along with analysis and visualization, as shown in Figure 1.

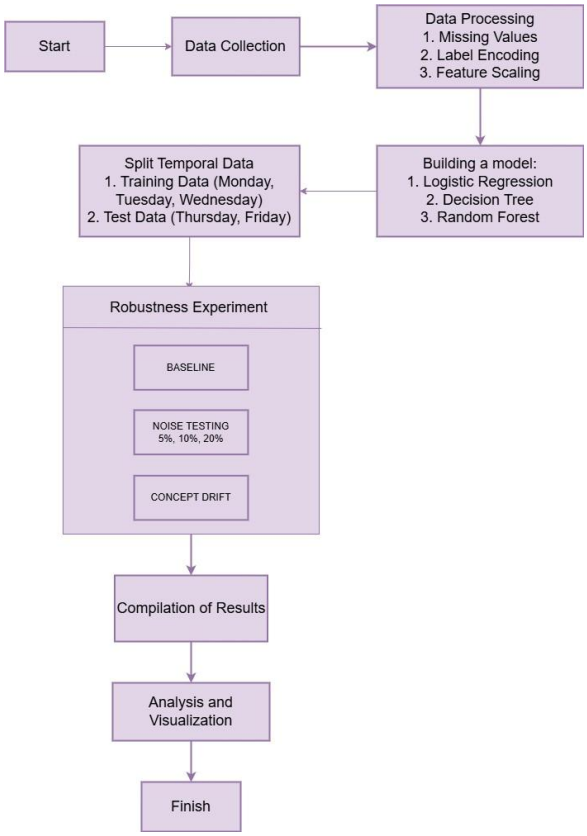


Figure 1. Research Process

This study began with data collection and initial exploration to understand the dataset structure, followed by preprocessing that included handling missing values using median imputation, encoding categorical labels into numerical values, and feature

standardization using StandardScaler. The dataset was then temporally split, with data from Monday through Wednesday used as training data and data from Thursday through Friday as test data to simulate real-world conditions where the model must adapt to constantly changing data.

Three models were built and tested, starting with Logistic Regression, Decision Tree, and Random Forest. Each model underwent three experimental scenarios sequentially baseline (clean test data), noise testing (adding 5%, 10%, and 20% Gaussian noise), and concept drift testing (performance evaluation on Thursdays and Fridays).

The results of all experiments were compiled into accuracy, precision, recall, and F1-score metrics, then visualized through baseline performance graphs, noise impact graphs, daily concept drift analysis, and confusion matrices. The final stage is a robustness analysis, in which each model is assigned a Robustness Score based on the average F1-score across the three scenarios the percentage decrease in performance is calculated, and the models are then ranked from the most robust to the most vulnerable to serve as the basis for the research conclusions.

### 2.1 Data Collection

This study uses the CICIDS2017 public dataset, which was downloaded from Kaggle and developed by the Canadian Institute for Cybersecurity. The dataset contains 2,830,743 samples with 79 numerical features and one label column it is available in CSV format and organized by collection date. The labels include the BENIGN class and 14 types of attacks. Initial data exploration was conducted to understand the structure and nature of the dataset by examining the dataset size, the first few rows, column data type information, and descriptive statistics (mean, standard deviation, minimum, and maximum values).

### 2.2 Data Preprocessing

Data preprocessing involves three main steps to transform raw data into high-quality data ready for processing by the model. First, missing values are handled using the median imputation strategy, as missing values can potentially cause errors during model training or result in a biased model. Second, label encoding converts categorical labels (e.g., 'BENIGN', 'DDoS', 'PortScan') into a numerical format that can be processed by machine learning algorithms, given that most ML models can only handle numerical data. Third, feature scaling uses StandardScaler to standardize numerical features so they have a mean of 0 and a standard deviation of 1, which is crucial for distance-based models such as Logistic Regression.

### 2.3 Split Data

The dataset was split in this study using a temporal split scheme specifically designed to test the model's robustness against two major challenges noise and concept drift. This approach draws on research Catillo et al., (2023) that considers the temporal aspect of the data, where earlier data is used for model training, while newer data is used for testing this represents a real-world scenario where an Intrusion Detection System (IDS) model trained on historical data must function on network traffic data that is constantly changing. Specifically, the data splitting scheme used consists of datasets from Monday, Tuesday, and Wednesday as training data, and datasets from Thursday and Friday as test data.

### 2.4 Building a Model

#### 2.4.1 Logistic Regression (LR)

Logistic Regression uses the sigmoid function to generate probability values. It is often used as a baseline in IDS research because it is simple and easy to interpret, but it has limitations in handling complex nonlinear patterns [15], [16], [17].

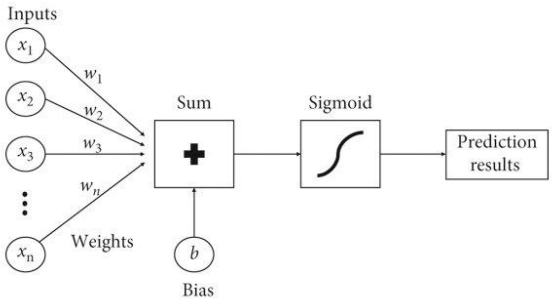


Figure 2. Logistic Regression Flowchart

#### 2.4.2 Decision Tree (DT)

Decision Tree constructs a decision tree structure based on the most informative features. It can handle non-linear data and categorical variables, but is prone to overfitting without pruning and is less stable with small variations in the data [18], [19].

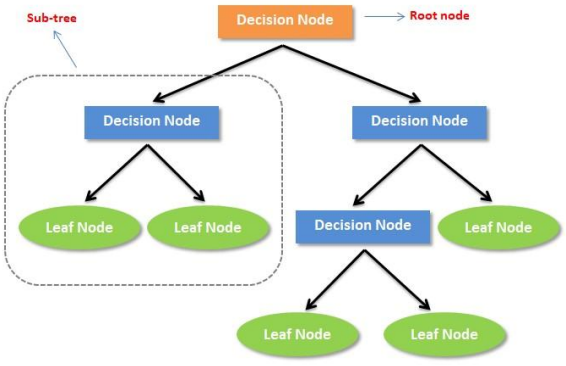


Figure 3. Decision Tree Algorithm

#### 2.4.3 Random Forest (RF)

Random Forest is a Decision Tree-based ensemble method that builds many trees randomly and then combines the results through voting. It effectively reduces overfitting, is consistent on high-dimensional data, and has proven accurate for attack classification in IDS [20], [21].

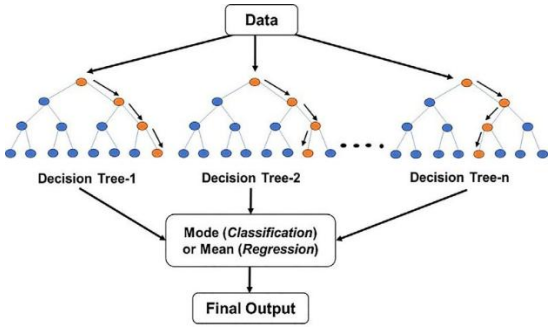


Figure 4. Random Forest Algorithm

### 2.5 Eksperimen Robustness

Each model was tested across three scenarios through an iterative loop starting with a baseline test on clean, unnoised test data this serves as a performance benchmark. Noise testing involved adding Gaussian noise to the test data at three intensity levels 5%, 10%, and 20% of the feature's standard deviation. Noise is injected randomly across all numerical features of the test data [22]. Concept Drift Testing, measured by the decline in baseline performance on Thursday versus Friday, illustrates the temporal distribution shift between test days [23].

To evaluate model performance across all scenarios, four standard classification metrics were used Accuracy, Precision, Recall, and F1-Score. These metrics are derived from the confusion matrix components True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) and are calculated as follows [24].

Accuracy measures the proportion of correct predictions out of the total data:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

Precision measures how accurately the model predicts positive classes, i.e., out of all positive predictions, how many are actually positive:

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2)$$

Recall (or Sensitivity) measures the model's ability to detect all actual positive samples:

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

F1-Score is the harmonic mean of Precision and Recall, providing a balance between the two this metric is particularly relevant when there is class imbalance in the dataset:

$$\text{F1-score} = 2 \text{ X } \frac{Precision\ X\ Recall}{Precision+Recall} \quad (4)$$

Collectively, these four metrics provide a comprehensive picture of each model's robustness under clean, noisy, and concept-shift conditions.

### 2.6 Robustness Evaluation and Scoring

The weighting in the Robustness Score formula is based on the hierarchical importance of evaluation criteria in operational IDS deployment. The baseline F1-score receives the highest weight (40%) because fundamental performance is a key prerequisite a model with poor baseline accuracy cannot be relied upon regardless of its robustness to disturbances. Equal weights (30% each) for the noise and drift components reflect their comparable importance in real-world scenarios noise represents the persistent challenge of imperfect measurements and environmental variations, while drift represents the systematic challenge of attack patterns that evolve over time. This weighting scheme follows a multi-criteria evaluation framework commonly used in machine learning robustness assessments [25], where primary performance metrics are given higher weights than secondary robustness indicators. The 40-30-30 distribution ensures that models are not selected solely based on baseline performance while providing balanced consideration of both types of operational challenges. The evaluation metrics used in each scenario include accuracy, precision, recall, and F1-score from the confusion matrix [26], [24]. To compare the robustness of the models, a Robustness Score is calculated as the average F1-score across the three scenarios:

$$\text{Robustness Score: } \frac{F1\ baseline+F1\ noise+F1\ drift}{3} \quad (5)$$

Where F1noise is the average F1-score across three noise levels. Additionally, the percentage decrease in performance due to noise and concept drift relative to the baseline is calculated as an indicator of degradation.

## 3. RESULT AND DISCUSSION

The results of the research are based on a logical sequence to form a story. The contents show facts/data. Can use Tables and Numbers but do not repeat the same data in pictures, tables, and text. To further clarify the description, can use subtitles.

Discussion is the basic explanation, relationship and generalization shown by the results. The description answers a research question. If there are any dubious results then show them objectively.

### 3.1. Data Collection

The CICIDS2017 dataset was successfully imported into the Google Colab environment, consisting of a total of 2,830,743 data samples with 79 feature columns and 1 label column. The label column classifies each record as normal traffic (BENIGN) or one of 14 types of cyberattacks.

Table 1 Attack Class Distribution

| No | Traffic Type | Amount | Percentage |
|---|---|---|---|
| 1 | *BENIGN* | 2.273.097 | 80.30% |
| 2 | *DoS Hulk* | 231.073 | 8.16% |
| 3 | *DoS GoldenEye* | 10.293 | 0.36% |
| 4 | *DoS Slowloris* | 5.796 | 0.20% |
| 5 | *DoS SlowHTTPTest* | 5.499 | 0.19% |
| 6 | *DDoS* | 128.027 | 4.52% |
| 7 | *Brute Force-FTP* | 7.938 | 0.28% |
| 8 | *Brute Force-SSH* | 5.897 | 0.21% |
| 9 | *Web Attack– XSS* | 652 | 0.02% |

| | | | |
|---|---|---|---|
| 10 | *Web Attack– SQL Injection* | 21 | 0.001% |
| 11 | *Web Attack – Brute Force* | 1.507 | 0.05% |
| 12 | *Infiltration* | 36 | 0.001% |
| 13 | *Botnet* | 1.966 | 0.07% |
| 14 | *PortScan* | 158.930 | 5.62% |
| 15 | *Heartbleed* | 11 | 0.0004% |

Table attack class distribution shows a highly extreme imbalance, where the BENIGN class dominates with 2,273,097 samples (80.30%) of the total data. The DoS Hulk attack ranks second with 231,073 samples (8.16%), followed by PortScan with 158,930 samples (5.62%) and DDoS with 128,027 samples (4.52%). Conversely, some attack classes have very small representations, such as Web Attack SQL Injection with only 21 samples (0.001%), Infiltration with 36 samples (0.001%), and Heartbleed with 11 samples (0.0004%). This extreme imbalance reflects real-world network conditions, but at the same time poses a challenge for classification models because the minority classes have very few examples to learn from.

### 3.2. Data Preprocessing

The preprocessing was conducted in three sequential stages. First, an examination of missing values revealed that only one column, Flow Bytes/s, contained 1,358 missing values out of a total of 2,830,743 samples (0.05%). Given this very small proportion, row deletion was chosen as the handling strategy to avoid potential bias that could be caused by imputation in complex network traffic data. As a result, the dataset was reduced from 2,830,743 rows to 2,829,385 rows, while data integrity and representativeness were maintained.

Second, label encoding was performed to convert 15 categorical labels into numerical codes in the range of 0 to 14**.** This step was necessary because all the machine learning algorithms used can only process numerical data directly. Third, feature standardization was performed using StandardScaler, which applies a z-score transformation to each feature. Before standardization, the data distribution was highly skewed, with a mean of 1,353,172.12 and a standard deviation of 10,881,864.09, reflecting the heterogeneity of feature scales in the dataset. After standardization, the mean approached 0 and the standard deviation became 0.947331**.** The standard deviation value, which is slightly below the ideal value of 1.0, is due to the prior handling of outliers. Nevertheless, outliers with extreme z-scores are retained because they can represent attack patterns with unique characteristics.

### 3.3. Split Dataset

The CICIDS2017 dataset is distributed using a temporal split approach to evaluate the model's ability to handle concept drift in data over time. The temporal split approach was chosen because it is more realistic for real-world intrusion detection systems, where models are trained on historical data and are then expected to identify attacks on new data collected from different time periods.

```
======================================================
DATASET SPLIT SUMMARY (TEMPORAL SPLIT)
======================================================
✓ Training set (Monday-Wednesday) :  1,668,530 samples (59.0%)
  ✓ Test set Thursday             :    458,968 samples (16.2%)
  ✓ Test set Friday               :    701,887 samples (24.8%)
  ✓ Total test set                :  1,160,855 samples (41.0%)
======================================================
```

Figure 5. Temporal Data Split

Figure 5 shows the results of the temporal data split from Monday to Wednesday combined as training data with a total of 1,668,530 samples (59%) of the entire dataset). Data from Thursday is used as the first test set with 458,968 samples (16.2%), covering Web Attack and Infiltration. Friday's data is used as the second test set with 701,887 samples (24.8%), covering PortScan and DDoS attacks. Combining the testing data resulted in a total of 1,160,855 samples (41%). The differences in attack types between Thursday and Friday inherently create a concept drift condition, which is one of the testing objectives in this study.

### 3.4. Model Initialization and Training Experiment Robustness

These three models are initialized with parameters that ensure reproducibility, as detailed in the following table:

Table 2. Model Configuration Parameters

| Model | Parameter | Justification |
|---|---|---|
| Logistic Regression | max_iter (1000) | Ensuring convergence on large datasets |
| | Random_state (42) | Reproducibility of results |
| | n_jobs (-1) | Parallel processing |
| Decision Tree | solver | Efficient for multi-class problems |
| | max_iter (20) | Preventing excessive overfitting while preserving the model's capacity |
| | Random_state (42) | Reproducibility of results |
| | Criterion | Standard splitting criteria |
| | min_samples_split | The minimum number of samples required for a split node |

| | | |
|---|---|---|
| Random Forest | n_estimators (100) | The balance between performance and computational cost |
| | Max depth (20) | Consistent with a fair comparison |
| | Random_state (42) | Reproducibility of results |
| | n_jobs (-1) | Parallel construction |
| | Bootstrap | Random sampling with replacement |

All models were configured with parameters balancing performance, computational efficiency, and reproducibility. The max_depth constraint (20) for tree-based models serves as regularization to prevent complete memorization of training data, while n_jobs=-1 enables parallel processing to reduce training time. The random_state parameter ensures experimental reproducibility across multiple runs.

Training was performed sequentially using the combined Monday–Wednesday data. Logistic Regression achieved a training accuracy of 98.86% in 373.08 seconds. Decision Tree achieved a training accuracy of 99.97%. Random Forest achieved a training accuracy of 99.98%. The very high training accuracy of Decision Tree and Random Forest indicates that both models have sufficient capacity to capture nearly all patterns in the training data, but this potentially leads to overfitting, which will become apparent during the testing phase.

### 3.5. Eksperiment Robustness Testing

Robustness testing experiments are designed to evaluate the resilience of machine learning models in the face of various suboptimal data conditions, which frequently occur in the operational environment of real-world intrusion detection systems.

Table 3. Baseline Performance Testing Comparison

| Metrik | Logistic Regression | Decision Tree | Random Forest |
|---|---|---|---|
| Accuracy | 74.38% | 74.73% | 74.90% |
| Precision | 59.86% | 60.44% | 59.19% |
| Recall | 74.38% | 74.73% | 74.90% |
| F1-Score | 66.34% | 66.83% | 66.12% |

Baseline testing using clean, uncontaminated test data showed relatively similar performance among the three models. Random Forest achieved the highest accuracy at 74.90%, followed by Decision Tree at 74.73%, and Logistic Regression at 74.38%, with a maximum difference of only 0.52%. However, for the F1-score metric, Decision Tree actually outperformed the others at 66.83%, followed by Logistic Regression at 66.34%, and Random Forest at 66.12%. This is due to the trade-off between precision and recall Decision Tree has the highest precision (60.44%), reflecting better ability to avoid false positives, while Random Forest has the highest recall but the lowest precision. The consistently lower precision values compared to recall (59-60% and 74-75%) across all models indicate that all three models tend to produce more false positives than false negatives under normal data conditions. Similar baseline performance across models suggests that the features in the CICIDS2017 dataset can be captured well even by linear models, meaning that increasing algorithmic complexity does not result in a proportional increase in accuracy.

Noise testing was conducted by injecting Gaussian noise into the test data at three intensity levels 5%, 10%, and 20%. All three models exhibited very different responses to this interference.

Table 4. Random Forest Model Noise Test Results

| Noise level | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| 5% | 74.92% | 56.96% | 74.92% | 64.71% |
| 10% | 74.96% | 56.79% | 74.92% | 64.60% |
| 20% | 74.92% | 56.61% | 74.92% | 64.49% |

Random Forest demonstrated remarkable resilience, with accuracy actually increasing slightly from 74.90% (baseline) to 74.92% at 5% noise, 74.96% at 10% noise, and back to 74.92% at 20% noise. However, its F1-score experienced a more noticeable decline, from 66.12% to 64.49% at 20% noise, due to a decrease in precision, indicating that the model became slightly more lenient in predicting attack classes.

Table 5. Logistic Regression Model Noise Test Results

| *Noise level* | *Accuracy* | *Precision* | *Recall* | *F1-score* |
|---|---|---|---|---|
| 5% | 73.64% | 60.23% | 73.64% | 66.26% |
| 10% | 71.73% | 60.61% | 71.73% | 65.70% |
| 20% | 67.49% | 60.49% | 67.49% | 63.78% |

Logistic Regression showed moderate and progressive degradation, with accuracy dropping from 74.38% to 67.49% at 20% noise (a decrease of 6.89%) and the F1-score dropping from 66.34% to 63.78% (a decrease of 2.56%). Interestingly, precision actually increased slightly as the noise level rose, indicating that the model became more conservative in predicting attacks, thereby reducing false positives but increasing false negatives.

The robustness of the logistic regression model to noise can be explained from several theoretical perspectives. First, from a geometric perspective, logistic regression forms a linear decision boundary (hyperplane) in the feature space that separates different classes. Gaussian noise with a mean of zero causes data points to shift randomly around their original positions, but these disturbances tend to cancel each other out statistically because positive and negative shifts occur with equal probability. Since the

decision boundary is smooth and continuous rather than rigid, small disturbances tend not to push samples across the threshold unless the sample is already close to the margin. Second, prior feature standardization using StandardScaler creates a normalized feature space where all dimensions contribute equally to the decision function. This normalization acts as an implicit regularization mechanism that makes the model inherently more stable against small-scale variations. Third, Logistic Regression optimizes a convex loss function during training, which inherently promotes solutions that are less sensitive to variations in individual data points. Unlike Decision Trees, which create rectangular regions parallel to the axes with sharp thresholds, the probabilistic nature of Logistic Regression allows for gradual transitions between classes, providing buffer zones that absorb disturbances caused by noise. The observed increase in precision from 59.86% to 60.49% as noise levels rise further confirms this theoretical explanation when noise shifts some attack samples away from the decision boundary, the model becomes more conservative, reducing false positives while maintaining overall stability. This behavior stands in stark contrast to the rigid, threshold-based decisions of a Decision Tree, where even minor feature perturbations can trigger a categorical change in the predicted class by crossing the split threshold.

Table 6. Decision Tree Model Noise Test Results

| *Noise level* | *Accuracy* | *Precision* | *Recall* | *F1-score* |
|---|---|---|---|---|
| 5% | 57.80% | 55.38% | 57.80% | 56.56% |
| 10% | 56,62% | 55.61% | 56.62% | 55.60% |
| 20% | 56.38% | 54.57% | 56.38% | 55.46% |

The Decision Tree model exhibits the most significant vulnerability, with a drastic drop in accuracy even at the lowest noise levels from 74.73% to 57.80% at just 5% noise (a 16.93% decrease). Further increases in noise result in relatively minor additional degradation 56.62% at 10% noise and 56.38% at 20% noise indicating a saturation effect following the initial drastic drop. This vulnerability can be explained by the axis-aligned rectangular decision regions of the Decision Tree small perturbations in feature values can push samples past the decision threshold into a different leaf node with a different predicted class.

Based on the average F1-score degradation across all noise levels, the order of resilience from best to worst is Logistic Regression (average degradation of 1.09%), Random Forest (1.52%), and Decision Tree (10.96%). The dramatic difference between the Decision Tree and the other two models confirms that a single Decision Tree is not suitable for environments with uncertainty in the input data.

Concept drift testing was conducted by evaluating the models' performance separately on Thursday and Friday data. The results showed starkly contrasting patterns.

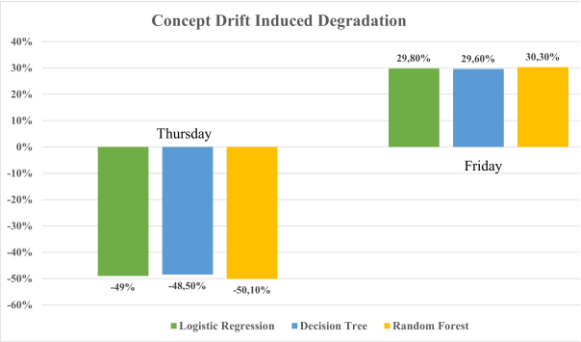


Figure 6. Concept Drift Induced Degradation

Based on the results of the Concept Drift-Induced Degradation analysis of the data from Thursday, all three models achieved near-perfect F1-scores Random Forest 99.26%, Decision Tree 99.24%, and Logistic Regression 98.83%. This exceptional performance indicates that the characteristics of the Thursday data are very similar to the training data, allowing the models to classify with near-perfect accuracy.

However, on Friday's data, all three models experienced a drastic degradation, with F1-scores dropping to the 46–47% range Decision Tree 47.02%, Logistic Regression 46.54%, and Random Forest 46.06%. The very small differences between the models (less than 1%) indicate that the change in data distribution on Friday is fundamental to the data itself, not specific to a particular algorithm. This reinforces the hypothesis of a systematic sudden drift in the Friday data, likely involving a change in the dominant attack type, class distribution, or network traffic characteristics that are fundamentally different from the training period. Based on the degradation of the F1-score relative to the baseline, the concept drift degradation for all three models exceeded the 20% threshold, so all three were categorized as not robust to concept drift Decision Tree 29.64%, Logistic Regression 29.85%, and Random Forest 30.34%.

Based on the results of the Concept Drift Induced Degradation analysis, the shift between the data from Thursday and Friday exhibits characteristics of a sudden shift, rather than a gradual or incremental one. This classification is supported by three key indicators first, a dramatic shift in performance from nearly perfect F1 scores on Thursday (98.83%–99.26%) to significantly lower scores on Friday (46.06%–47.02%) without an intermediate transition second, an identical degradation pattern across all three models despite their differing algorithmic foundations, indicating that the drift is data-centric rather than model-specific and finally, the consistency of Friday's performance across models, suggesting convergence toward a fundamentally different data distribution. The sudden shift was likely caused by a combination of factors inherent to the structure of the CICIDS2017 dataset Thursday's data primarily contained Web Attack and Infiltration patterns very similar to the

Monday–Wednesday training distribution, while Friday introduced PortScan and DDoS attacks with vastly different traffic characteristics, feature distributions, and class proportions. The sudden nature of this drift makes it very challenging for static models to handle, as they lack the opportunity for gradual adaptation that would be possible with incremental or recurring drift patterns.

### 3.6. Comprehensive Robustness Analysis

The robustness score is calculated as the weighted average F1-score of three components baseline (weight 40%), average noise (weight 30%), and Friday drift (weight 30%)

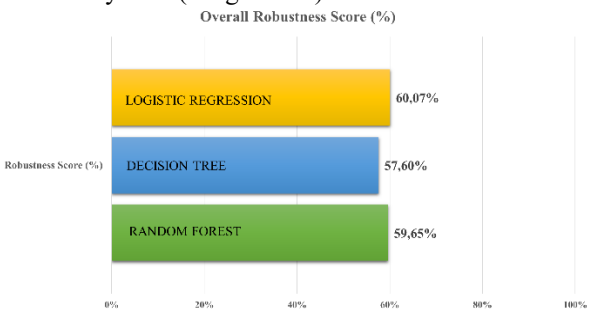


Figure 7. Overall Robustness Score

Logistic Regression achieved the highest robustness score of 60.07%, driven primarily by its best resilience to noise (average noise F1 = 65.25%). Random Forest ranked second with 59.65%, just 0.42% behind Logistic Regression, with an average noise F1 of 64.60%. Decision Tree ranked last with 57.60% although it had the highest baseline F1-score (66.83%) and the highest drift F1-score (47.02%), its critical weakness in noise robustness (average noise F1 = 55.87%) could not be compensated for by its advantages in the other two components.

These findings confirm that robustness to noise and robustness to concept drift are two independent characteristics. The Random Forest model that performs best in handling noise actually exhibits the highest degradation in concept drift, while the Decision Tree model that is most vulnerable to noise demonstrates slightly better resistance to concept drift. For practical implementation, both Logistic Regression and Random Forest are worth considering as IDS models that are more reliable overall compared to Decision Trees. However, the results showing that none of the models are robust against concept drift emphasize that algorithm selection alone is insufficient. Additional mechanisms such as periodic retraining, drift detection, or online learning need to be integral components of the IDS system architecture in dynamic network environments.

## 4. CONCLUSION

This study analyzed the robustness of Logistic Regression, Decision Tree, and Random Forest models against noise and concept drift on the CICIDS2017 dataset. Logistic Regression achieved the highest robustness score (60.07%) due to excellent noise tolerance (1.09% F1-score degradation), followed by Random Forest (59.65%) and Decision Tree (57.60%). While Random Forest demonstrated exceptional noise resistance (1.63% degradation), it exhibited the highest concept drift degradation (30.34%). Decision Tree, despite having the best baseline F1-score (66.83%), showed critical vulnerability to noise with 10.96% degradation. All models were categorized as non-robust to concept drift (29.64%-30.34% degradation), confirming that noise robustness and drift robustness are independent characteristics.

The findings reveal that sudden drift between Thursday and Friday data represents a fundamental, data-centric challenge rather than an algorithm-specific limitation. For practical IDS deployment, Logistic Regression is recommended as the optimal choice, offering the best combination of robustness, interpretability, and computational efficiency. Random Forest serves as a viable alternative for noise-intensive environments, while Decision Tree is not recommended for production deployment despite its strong baseline performance.

Study limitations include single-dataset validation and the use of only Gaussian noise. Future research should validate findings on additional datasets (NSL-KDD, UNSW-NB15, CIC-IDS2018), implement adaptive learning mechanisms for drift mitigation, explore diverse noise types, and investigate periodic retraining strategies with optimal frequencies to maintain IDS reliability in dynamic operational environments.

## 5. REFERENCE

[1] Imanuel Toding Bua and Nur Isdah Idris, "Analisis Kebijakan Keamanan Siber di Indonesia: Studi Kasus Kebocoran Data Nasional pada Tahun 2024," *Desentralisasi J. Hukum, Kebijak. Publik, dan Pemerintah.*, vol. 2, no. 2, pp. 100–114, May 2025, doi: 10.62383/desentralisasi.v2i2.653.

[2] D. M. Manias, A. Chouman, and A. Shami, "Model Drift in Dynamic Networks," *IEEE Commun. Mag.*, vol. 61, no. 10, pp. 78–84, Oct. 2023, doi: 10.1109/MCOM.003.2200306.

[3] Z. Liao, "A Dynamic Intrusion Detection System Integrating Concept Drift Detection and Incremental Learning," vol. 3, no. 1, pp. 20–26, 2026.

[4] S. B. Mannapur, "Understanding Data Drift and Concept Drift in Machine Learning Systems," 2025.

[5] J. Smithie, "Intrusion detection in cybersecurity," *Adv. Eng. Innov.*, vol. 2, no. 1, pp. 13–16, Oct. 2023, doi: 10.54254/2977-3903/2/2023014.

[6] R. Morzelona and R. R. Mirajkar, "Implementation and Evaluation of Intrusion Detection Systems using Machine Learning Classifiers on Network Traffic Data," *Res. J.*

*Comput. Syst. Eng.*, vol. 4, no. 2, pp. 103–116, Dec. 2023, doi: 10.52710/rjcse.81.

[7] I. Maltseva, Y. Chernysh, and Y. Protsyuk, "Development of algorithms for early detection of cyberattacks on networks using machine learning," *Commun. Informatiz. cybersecurity Syst. Technol.*, vol. 1, no. 6, pp. 105–115, Dec. 2024, doi: 10.58254/viti.6.2024.08.105.

[8] E. Mousavipour and A. Dimanchev, "Adaptive Anomaly Detection in Evolving Network Environments".

[9] E. Victor, L. Barboza, P. Ricardo, and L. De Almeida, "Challenges on Classifying Data Streams with Concept Drift," no. September 2022, pp. 126–132, 2024.

[10] W. Yang and J. Guo, *A Concept Drift Detection Approach Based on Jensen-Shannon Divergence for Network Traffic Classification*, vol. 1, no. 1. Association for Computing Machinery. doi: 10.1145/3573942.3573979.

[11] M. Catillo, A. Del Vecchio, A. Pecchia, and U. Villano, "A Case Study with CICIDS2017 on the Robustness of Machine Learning against Adversarial Attacks in Intrusion Detection," *ACM Int. Conf. Proceeding Ser.*, 2023, doi: 10.1145/3600160.3605031.

[12] P. Neirz, H. Allende, and C. Saavedra, "Attribute Relevance Score: A Novel Measure for Identifying Attribute Importance," *Algorithms*, vol. 17, no. 11, p. 518, Nov. 2024, doi: 10.3390/a17110518.

[13] S. Gaïffas, I. Merad, and Y. Yu, "WildWood: A New Random Forest Algorithm," *IEEE Trans. Inf. Theory*, vol. 69, no. 10, pp. 6586–6604, Oct. 2023, doi: 10.1109/TIT.2023.3287432.

[14] F. Acito, "Logistic Regression," in *Predictive Analytics with KNIME*, Cham: Springer Nature Switzerland, 2023, pp. 125–167. doi: 10.1007/978-3-031-45630-5_7.

[15] O. Graham and J. Lloris, "Comparative Study of Supervised Learning Algorithms for Intrusion Detection with a Focus on Logistic Regression." May 07, 2025. doi: 10.20944/preprints202505.0420.v1.

[16] Koushik Paul, Sayandeep Paik, Siddhartha Kuri, Soumyadip Majumder, and Avijit Kumar Chaudhuri, "A Novel Intrusion Detection System Using Multiple Linear Regression," *Int. J. Eng. Technol. Manag. Sci.*, vol. 7, no. 2, pp. 75–86, 2023, doi: 10.46647/ijetms.2023.v07i02.010.

[17] R. Kimanzi, P. Kimanga, D. Cherori, and P. K. Gikunda, "Deep Learning Algorithms Used in Intrusion Detection Systems -- A Review," 2024, [Online]. Available: http://arxiv.org/abs/2402.17020

[18] E. Gilmore, V. Estivill-Castro, and R. Hexel, "More Interpretable Decision Trees," 2021, pp. 280–292. doi: 10.1007/978-3-030-86271-8_24.

[19] Galuh Nurvinda K, "Mengenal Algoritma Terpopuler dalam Data Science 2024," *DQLAB*, 2024. https://dqlab.id/mengenal-algoritma-terpopuler-dalam-data-science-2024#:~:text=3. Algoritma Decision Tree Decision Tree adalah,dan hasilnya disajikan dalam bentuk pohon keputusan.

[20] A. A. A. Hadi and A. M. Hadi, "Improving cybersecurity with random forest algorithm-based big data intrusion detection system: A performance analysis," 2024, p. 040012. doi: 10.1063/5.0191707.

[21] A. T Devi, L. Tukaram, and R. Purad, "Enhancing Intrusion Detection with Principal Component Analysis and Random Forest," *Int. J. Sci. Eng. Res.*, pp. 10–13, Sep. 2025, doi: 10.70729/SE25904191409.

[22] O. H. Ram, N. Gorea, A. Reif, L. Bonasera, and S. Augustin, "The Role of Noisy Data in Improving CNN Robustness for Image Classification," vol. 13606, pp. 1–16, 2025, doi: 10.1117/12.3063563.

[23] S. Greco, B. Vacchetti, D. Apiletti, and T. Cerquitelli, "Unsupervised Concept Drift Detection from Deep Learning Representations in Real-time," pp. 1–23, 2025.

[24] D. Krstinić, A. K. Skelin, I. Slapničar, and M. Braović, "Multi-Label Confusion Tensor," *IEEE Access*, vol. 12, pp. 9860–9870, 2024, doi: 10.1109/ACCESS.2024.3353050.

[25] H. Sayadi, Z. He, T. Miari, and M. Aliasgari, "Redefining Trust : Assessing Reliability of Machine Learning Algorithms in Intrusion Detection Systems," *IEEE*, pp. 1–5, 2024.

[26] S. Sathyanarayanan, "Confusion Matrix-Based Performance Evaluation Metrics," *African J. Biomed. Res.*, pp. 4023–4031, Nov. 2024, doi: 10.53555/AJBR.v27i4S.4345.