**Matrik: Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer**
Vol. 25, No. 2, March 2026, pp. 263∼274
ISSN: 2476-9843, accredited by Kemenristekdikti, Decree No: 10/C/C3/DT.05.00/2025
DOI: 10.30812/matrik.v25i2.5760


# Enhancement of Supervised Learning Models for Intrusion Detection Through Mutual Information and Hyperparameter Tuning

**Deny Jollyta[1], Yoakhina Nicole Makaruku[2], Alyauma Hajjah[1], Yulvia Nora Marlim[1]**
[1]Institut Bisnis dan Teknologi Pelita Indonesia, Pekanbaru, Indonesia
[2]Institut Agama Kristen Negeri, Ambon, Indonesia

**Article Info**

***Article history:***

Received September 27, 2025
Revised November 14, 2025
Accepted January 27, 2026

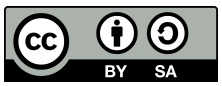
***Keywords:***

*Hyperparameter tuning;*
*Intrusion detection;*
*Mutual information;*
*Supervised learning.*

***ABSTRACT***

Enhancing the performance of supervised learning algorithms through feature and hyperparameter testing remains challenging for users, particularly when detecting computer network intrusions. There are opportunities to assess whether a supervised learning algorithm performs optimally, depending on the number of features and the choice of hyperparameters. The purpose of this research is to enhance the network intrusion detection performance of three supervised learning algorithms, namely Support Vector Machine (SVM), eXtreme Gradient Boosting, and Random Forest, by using the Mutual Information feature selection approach and hyperparameter tuning. Mutual Information measures the dependency of features on the target. Features with high values are the most informative. Hyperparameters are not learned from the data; they are set before training begins. Hyperparameters are selected in accordance with the requirements of the three algorithms via iterative training and testing on the NSL-KDD dataset. The dataset was split into 80:20, 70:30, and 60:40. The results showed that the fifteen features with the highest mutual information were identified and trained on the data using appropriate hyperparameters. By splitting the data in an 80:20 ratio, the accuracy of Support Vector Machine reached its maximum, increasing from 90% to 98%, whereas eXtreme Gradient Boosting and Random Forest reached their maximum, increasing from 97% and 98% to 100%, respectively. The study's findings advance our understanding of how algorithm performance depends on feature and hyperparameter selection.





***Corresponding Author:***

Deny Jollyta, +62 812-7585-546,
Faculty of Computer Science and Informatics,
Institut Bisnis dan Teknologi Pelita Indonesia, Pekanbaru, Indonesia,
Email: deny.jollyta@lecturer.pelitaindonesia.ac.id.

How to Cite:
D. Jollyta, Y. N. Makaruku, A. Hajjah, and Y. N. Marlim, "Enhancement of Supervised Learning Models for Intrusion Detection Through Mutual Information and Hyperparameter Tuning", *MATRIK: Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer*, Vol. 25, No. 2, pp. 263-274, March, 2026.

## 1. INTRODUCTION

One of the most important factors in the sustainable growth of nations, businesses, and organizations is the quick advancement of computer, communication, and network technology [1]. Given increased dependence, network availability and integrity must be maintained through enhanced security measures. Computer network security is increasingly important because computer networks are vital to many human operations. Given that computer networks are used for the majority of corporate processes and transactions, a strong security layer is required to safeguard the constant flow of data and information [2]. Cyber threats are becoming more sophisticated and increasingly difficult to detect in this era of digital connectivity. Cyberattacks can harm an organization's brand, interrupt services, and result in large financial losses. An efficient network intrusion detection system (NIDS) is therefore essential. Expert systems are the main component of traditional IDSs. Existing assault patterns are manually extracted from data records and kept in accordance with predetermined guidelines.

One promising approach to developing NIDS is to use machine learning techniques, particularly supervised learning. Prior studies have shown the enormous potential of supervised learning techniques in identifying different kinds of cyberattacks as seen in Figure 1 [3]. An NIDS often uses an interface in promiscuous mode to function as a sniffer, analyzing network traffic purely without the use of host-specific information (such as memory use, processing, and interfaces) [4]. This type of system usually operates with one or more sensors on the network and a monitoring station. Using a dataset with selected features and fixed hyperparameters, this study aims to assess how well supervised learning algorithms perform in identifying intrusions in computer networks. The algorithms that are employed are Support Vector Machine (SVM), eXtreme Gradient Boosting (XGBoost), and Random Forest (RF).

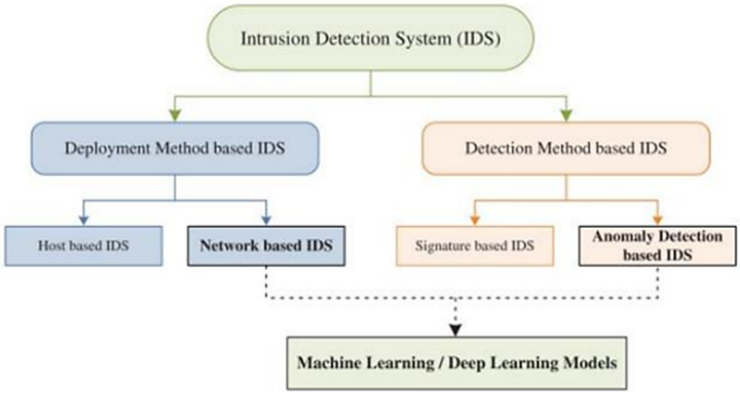



Figure 1. IDS classification

SVM has been applied to computer network intrusion detection in numerous studies. In study [5], the firework algorithm was used to choose parameters and feature subsets, which improved SVM accuracy by 2.99%. Other research [6], SVM was used to identify social media infiltration instances. Other research that addressed syn assaults using the XGBoost method [7]. The researcher employed different datasets, specifically the CICDDoS2019 dataset. The final model demonstrated that syn incursions may be accurately detected using XGBoost. Another popular method for identifying network issues is the RF algorithm. RF used radio frequency (RF) to create an online network intrusion detection system that has a 98% accuracy rate [8]. This online model effectively created trust that RF can identify intrusions more precisely in practical settings. Research by [9] further demonstrates the accuracy of RF in intrusion detection because RF performs admirably on CIC-IDS 2017 data. Additionally, because it concentrates on prominent features, the RF model can be further optimized by employing feature reduction techniques like the Principal Component Analysis (PCA) to identify network issues [10].

Some explanations of the previous research showed that the algorithm's performance can be impacted by the quantity of features in data. The classification technique employs feature selection, such as Mutual Information (MI) and hyperparameters, to optimize model formation. MI was chosen because the results of its feature selection had a more positive impact on improvement compared to others, such as p-test, chi-square, t-test, and handcrafted features [11]. In a study [12], MI achieved an SVM model accuracy of 65.63% by selecting 100 features with the highest MI values that had a good correlation with the target variable. Additionally, MI was utilized to choose the data features that were employed in the boosting model [13]. After choosing five features from the diabetes dataset with the greatest MI values, it was demonstrated that the XGBoost method generated a model accuracy of up to 71.86% [14]. In order to enhance the performance of the hybrid Random Forest model, MI chose up to 60% of the features in the birth pattern dataset [14]. Numerous studies have demonstrated that the model's performance is affected by the selected features.

Along with the application of hyperparameters. According to [15], hyperparameters are integral to machine learning, as they are essential for optimizing models produced by machine learning algorithms. Initially, hyperparameters are used in training. When a model overfits, it performs exceptionally well on the training data but fails to generalize to new data due to inaccurate hyperparameter selection [16]. Hyperparameters are determinable for every algorithm [17]. Hyperparameters can change the Random Forest model's accuracy by 0.15%, from 96.23% to 96.37%. The hyperparameters employed are max_depth (8), max_features (sqrt), n_estimators (240), min_sample_split (2), and min_sample_leaf (2). In addition, the accuracy of the model generated by the SVM algorithm using hyperparameters kernel type (linear, polynomial, sigmoid), C parameter (3, 10, 50), and degree parameter (1, 2, 3) that are applied to four different kinds of datasets was studied [18]. As it happens, the accuracy of the generated SVM models varies across datasets; some show an increase in accuracy, whereas others show a decrease. The model's accuracy on a single dataset was increased to 99% by tuning hyperparameters to specific values. However, using the same hyperparameters can also yield low accuracy, ranging from 54% to 73.63%, on a different dataset.

Based on earlier studies, the SVM, XGBoost, and RF algorithms' model accuracy was generally increased when MI and hyperparameter selection were applied to datasets other than network intrusion. This enables evaluation of how effectively the three tested algorithms identify network problems given specific characteristics and hyperparameters. The precision of the final model is anticipated to support user justifications for employing a supervised learning approach. Technically, this research has contributed to the selection of an appropriate supervised learning algorithm for identifying the associated computer network infiltration problem, while scientifically, it focuses on accuracy.

## 2. RESEARCH METHOD

### 2.1. SVM Algorithm

Information security has also used SVM for intrusion detection. Because of its strong generalization capabilities and capacity to overcome the curse of dimensionality, SVM has emerged as a preferred tool for anomalous intrusion detection [19]. Due to its ability to generalize effectively with kernel methods, even under conditions of limited training samples in high-dimensional spaces, SVM is also advantageous for employing structural risk minimization to achieve a global minimum of the true risk. Because real-time intrusion detection is crucial, one of the primary benefits of employing SVM for IDS is its speed. Because the classification complexity is independent of the feature space dimensionality, SVMs scale better and learn a wider range of patterns. When a new pattern emerges during classification, SVMs can also dynamically update the training patterns. Figure 2 illustrates how this works [20].

Figure 2 describes the classification workflow using a support vector machine, which begins by tuning the model parameters, followed by inputting the raw dataset into the system 19. Then, based on the best-tuned parameters, the data will be processed to identify the data point closest to the hyperplane and to calculate the margin between a sample and other sample types.

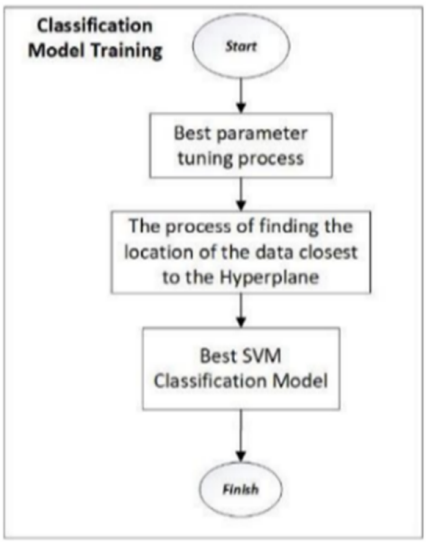



Figure 2. SVM algorithm

### 2.2. XGBoost Algorithm

In research [21], it was proven that XGBoost is the best supervised algorithm in detecting intrusions, with a very low log loss value on the validation set of 0.009, and a log loss for the test set of 0.011, after 5-fold cross-validation, and with a minimum training time of 30 minutes on such a huge dataset. According to research comparing different classification and regression methods for network detection, XGBoost is quick, effective, and very accurate [10]. Figure 3 is described XGBoost algorithm. XGBoost sequentially builds an ensemble of decision trees, with each new tree correcting errors from the previous one. It uses advanced optimization techniques and regularization methods that reduce overfitting and improve model performance [22]. In addition, the performance of XGBoost can be improved by combining grid search [23].

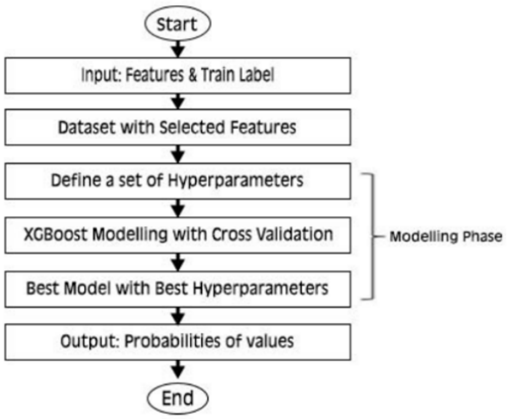



Figure 3. XGBoost algorithm

### 2.3. RF Algorithm

RF is a supervised learning algorithm for classification and regression [24]. The trees formed in an RF run in parallel. There is no interaction between these trees during tree construction. An illustration of the RF algorithm, as defined, is shown in Figure 4. The RF model begins by tuning the optimal RF parameters on the research data, then building a Decision Tree model and selecting a random subset of the training data via the Bootstrap step. The Decision Tree continues the next process; the results of the process will be selected, whether the node has a tree depth that has been fulfilled, and the node has produced the smallest gini index value [25]. If not, the Decision Tree is used to determine the next separator node; once it is satisfied, the model prediction error is calculated, and the RF process is complete. The RF technique is also capable of generating the optimum classification model for computer network intrusion detection. Using two distinct datasets—the eCICIDS-2017 dataset and the CICMalDroid2020 dataset—research by Iftikhar [26] and Mohammed Tarek [27] together showed that the RF method performed best in detecting network intrusions, achieving an accuracy of 99.8%.

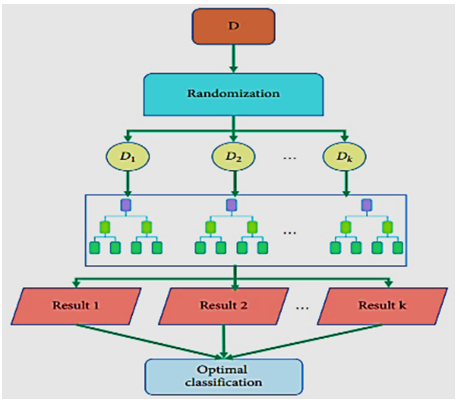



Figure 4. RF algorithm

### 2.4. Mutual Information (MI)

An effective selection technique that gauges the correlation between variables is MI [28]. In this study, the MI results indicate high mutual information, indicating that changes in feature values are strongly associated with changes in the target variable. These features provide significant information about the target and may be important predictors for models trained using SVM, XGBoost, and RF. In intrusion detection frameworks, applying MI can help identify attributes with high information value and relevance to attack classification. This aims to improve model performance.

### 2.5. Hyperparameter

Hyperparameter tuning is crucial for optimizing the performance and generalization of machine learning (ML) models. The process of determining the optimal hyperparameter configuration to achieve the best performance of an ML model is known as hyperparameter tuning [29]. This process is crucial because hyperparameters control the learning process and model structure, such as the learning rate, the number of neurons in an artificial neural network, or the kernel size in a support vector machine, which directly impact its performance. Hyperparameters are set before training and require careful selection. According to Justus [17], hyperparameter adjustment can enhance the generalization and performance of the model.

### 2.6. Research Approach

Preprocessing steps and model visualization for the three algorithms employed in this study are among the tasks required. The research approach in question is as Figure 5. Data processing begins with dataset preparation. The data used is the NSL-KDD dataset, an improved version of the KDD Cup 99 dataset, which includes categorical features. The NSL-KDD dataset was obtained from the Kaggle website and comprises 42 features. Preprocessing stages were implemented to ensure data validity, including feature selection based on mutual information (MI). According to research [27], this selection approach can rank the features by identifying the dependencies between pairs of features. Thus, the MI's ranking of the most significant features has been chosen. The MI identifies irrelevant features based on dependencies and sorts them by feature importance. Furthermore, the data set with the selected features was trained using the SVM, XGBoost, and RF algorithms with a splitting data of 80:20, 70:30, and 60:40. Hyperparameters were determined for each algorithm to ensure performance improvements. The training results in a model, which is then evaluated using a confusion matrix and the area under the receiver operating characteristic curve (AUC).

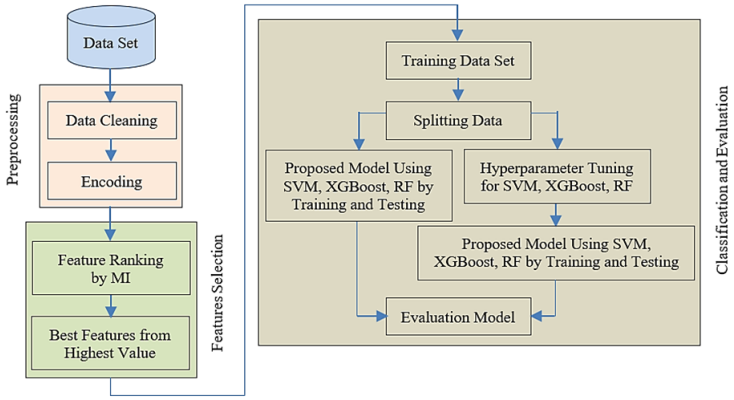



Figure 5. Research approach

### 2.7. Evaluation Model

The performance of the classification model in this research was evaluated using two methods: the confusion matrix and the Receiver Operating Characteristic (ROC) curve. A confusion matrix is often used in ML projects to see the performance of classification models in recognizing test data and sample data [29]. A confusion matrix provides detailed information about the model's errors and simplifies the analysis of model performance. Another performance evaluation model is the ROC. ROC is an ML operation that serves to evaluate the performance of classification models in binary and multiclass cases [30]. The ROC curve

generated during the ROC process represents the classification model's performance in distinguishing between object classes. The ROC curve connects coordinate points using "1-specificity" (false positive rate) as the x-axis and sensitivity as the y-axis for all boundary values measured from the test results [31].

## 3. RESULT AND ANALYSIS

### 3.1. Preprocessing Stages

A total of 125,600 data points were used in this research. To produce more accurate and reliable models and easily interpretable visualizations, the data must be cleansed to better identify patterns, trends, and anomalies. Data cleaning involves ensuring that data are free of missing or null values. Figure 6 shows that no data are missing for any feature. Data cleaning was also performed to identify any duplicate data.

```
[8]: duration                0
     protocol_type           0
     service                 0
     flag                    0
     src_bytes               0
     dst_bytes               0
     land                    0
     wrong_fragment          0
     urgent                  0
     hot                     0
     num_failed_logins       0
     logged_in               0
     num_compromised         0
     root_shell              0
     su_attempted            0
     num_root                0
     num_file_creations      0
     num_shells              0
     num_access_files        0
     num_outbound_cmds       0
     is_host_login           0
     is_guest_login          0
     count                   0
     srv_count               0
     serror_rate             0
     srv_serror_rate         0
     rerror_rate             0
     srv_rerror_rate         0
     same_srv_rate           0
     diff_srv_rate           0
     srv_diff_host_rate      0
     dst_host_count          0
     dst_host_srv_count      0
     dst_host_same_srv_rate  0
```

Figure 6. Data cleaning

Next preprocessing is encoding. The encoding process in this preprocessing process converts categorical data such as protocol_type, service, flag, and attack into numeric representations. This transformation allows the supervised learning algorithm to identify more complex patterns in the data and produce more accurate models. Figure 7 shows the encoding process in question using one-hot encoding.

```
cat_features = df.select_dtypes(include='object').columns
cat_features
```

```
Index(['protocol_type', 'service', 'flag', 'attack'], dtype='object')
```

+ Code + Markdown

```
from sklearn import preprocessing
le=preprocessing.LabelEncoder()
clm=['protocol_type', 'service', 'flag', 'attack']
for x in clm:
    df[x]=le.fit_transform(df[x])
```

Figure 7. Transforming categorical data into numerical data

### 3.2. Feature Selection

The primary objectives of feature selection are to enhance model performance, reduce model complexity, and facilitate interpretation of results. A feature selection technique based on mutual information was applied to identify the features that most contribute to attack detection. Features such as flag, logged_in, serror_rate, and srv_serror_rate showed strong correlations with attacks in the NSL-KDD dataset. A total of 42 features were used in the dataset. The fifteen features with the highest scores were selected based on the selection results, and they were subsequently processed to create the desired model. They are duration, protocol_type, service, flag, src_bytes, dst_bytes, wrong_fragment, hot, logged_in, num_compromised, count, srv_count, serror_rate, srv_serror_rate, and rerror_rate. The outcomes of feature selection utilizing mutual information are displayed in Figure 8.

| | |
|---|---|
| src_bytes | 0.566 |
| service | 0.468 |
| dst_bytes | 0.441 |
| flag | 0.369 |
| same_srv_rate | 0.366 |
| diff_srv_rate | 0.362 |
| dst_host_srv_count | 0.335 |
| dst_host_same_srv_count | 0.311 |
| logged_in | 0.291 |
| dst_host_serror_rate | 0.287 |
| dst_host_diff_srv_rate | 0.284 |
| dst_host_srv_serror_rate | 0.281 |
| serror_rate | 0.278 |
| srv_serror_rate | 0.269 |
| count | 0.265 |

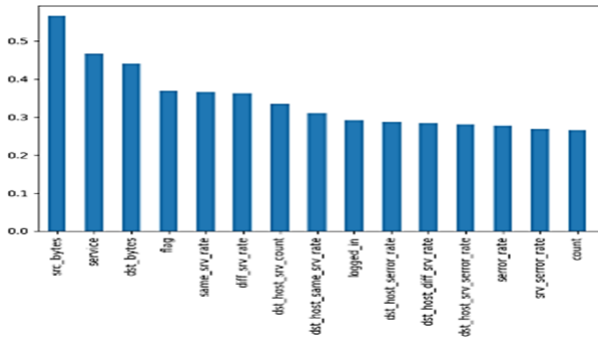



Figure 8. Feature selection results using mutual information

### 3.3. Training Data Set

A dataset was chosen for this study based on 15 features using MI. The following shows the training results with an 80:20 data split and trained both before and after hyperparameters. This was done to evaluate if the algorithm's ability to generate models had improved in any way. The accuracy of the models built by the SVM, XGBoost and RF algorithms before using hyperparameters was 0.90 (90%), 0.97 (97%), and 0.98 (98%), respectively. The dataset was then trained using the hyperparameters and values for each algorithm that were given. Each algorithm's hyperparameters are displayed in Figure 9.

SVM Algorithm

```
param_grid_svm = {
    'C': [0.1, 1, 10],
    'kernel': ['linear', 'rbf'],
    'gamma': ['scale', 'auto']
}
```

XGBoost Algorithm

```
param_grid = {
    "n_estimators": [50, 64, 100, 128],
    "max_depth": [2, 3, 4, 5, 6],
    "learning_rate": [0.01, 0.03, 0.05, 0.1],
    "subsample": [0.5, 0.8],
    "colsample_bytree": [0.5, 0.8]
}
```

RF Algorithm

```
param_grid_rf = {
    'n_estimators': [50, 100, 128],
    'max_depth': [2, 4, 6, None],
    'min_samples_split': [2, 5],
    'min_samples_leaf': [1, 2],
    'max_features': ['sqrt']
}
```

Figure 9. Algorithm hyperparameter

Figure 9 shows that each method has a distinct number of hyperparameters. SVM employs three hyperparameters, whereas XGBoost and RF use five. The purpose of these hyperparameters is to enhance the algorithm's ability to generate models using 15 specific dataset attributes. For each hyperparameter, a range of values is tested, and the hyperparameters that increase model accuracy are selected. Following dataset processing, the source code that achieves the optimal accuracy for each algorithm, using the specified hyperparameters, is shown in Figures 10, 11, and 12.

```
Fitting 5 folds for each of 12 candidates, totalling 60 fits
Best SVM Params: {'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}
Best SVM Score: 0.9785400628378321
```

Figure 10. Hyperparameter result of the SVM algorithm

```
Fitting 5 folds for each of 320 candidates, totalling 1600 fits
Best params: {'colsample_bytree': 0.8, 'learning_rate': 0.1, 'max_depth': 6, 'n_estimators': 128, 'subsample': 0.8}
Best score: 0.9982359282749724
```

Figure 11. Hyperparameter result of the XGBoost algorithm

```
Fitting 5 folds for each of 48 candidates, totalling 240 fits
Best RF Params: {'max_depth': None, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 128}
Best RF Score: 0.9984564389424954
```

Figure 12. Hyperparameter result of RF algorithm

Figure 10 shows that the training data are divided into 5 subsets (folds), with the SVM model trained on 4 subsets and tested on the remaining 1 subset. This process is repeated 5 times, so each subset is used as test data once. The SVM hyperparameters produce 12 candidates from the combinations shown in Figure 9, where C has 3 values (0.1, 1, and 10), kernel has 2 values (linear and RBF), and gamma has 2 values (scale and auto). All values are combined to produce 12 candidates, so the model is trained and tested 5-fold $\times$ 12 candidates = 60 times. Similar to SVM, XGBoost uses the same training data division and data training procedure. For XGBoost, the training data distribution and training process are the same as for SVM. However, the number of hyperparameters and their respective values are different, resulting in 320 candidates, so the model is trained and tested 5-fold $\times$ 320 candidates = 1600 times, as shown in Figure 11. This also applies to the RF algorithm. Figure 12 showed that the model is trained and tested 5 folds $\times$ 48 candidates = 240 times.

### 3.4. Model Evaluation

After the hyperparameter calculation procedure is completed, the model with the highest accuracy will be displayed for each algorithm. After that, the model is evaluated using an AUC curve and a confusion matrix. The confusion matrix and AUC curve for model evaluation are shown in Figures 13, 14, and 15. The SVM model's accuracy is 0.98 (98%), whereas the models produced by the XGBoost and RF algorithms achieve 1 (100%) accuracy, including precision, recall, and F1-score. When the model is trained and evaluated on both the training and test data, the outcomes remain the same. After tuning hyperparameters, the model's accuracy improves relative to its pre-tuning accuracy. This study demonstrates that the algorithm's performance is also influenced by the dataset's conditions, including features and hyperparameter settings. The accuracy of the final model is influenced by the quantity of features chosen during MI. The 15 features with the highest values were selected. To ensure that all three algorithms perform optimally, the use of appropriate hyperparameters further enhances performance.

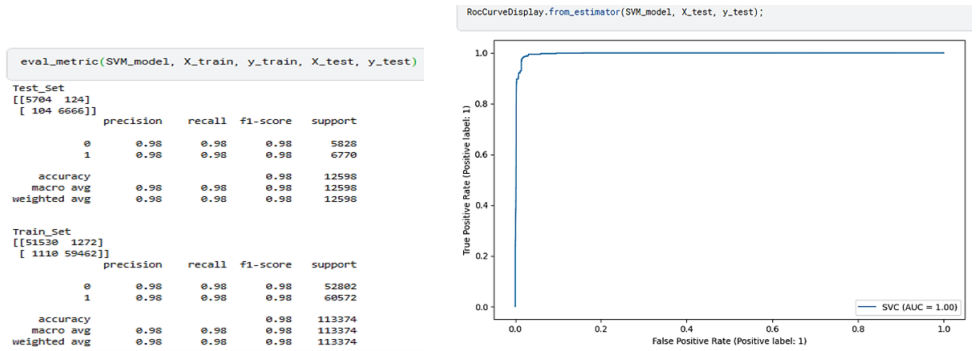



Figure 13. SVM evaluation results

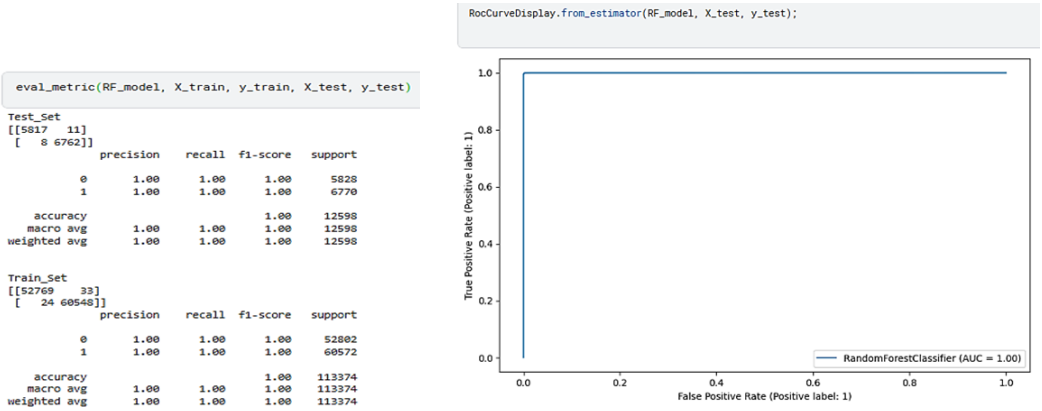



Figure 14. XGBoost evaluation result

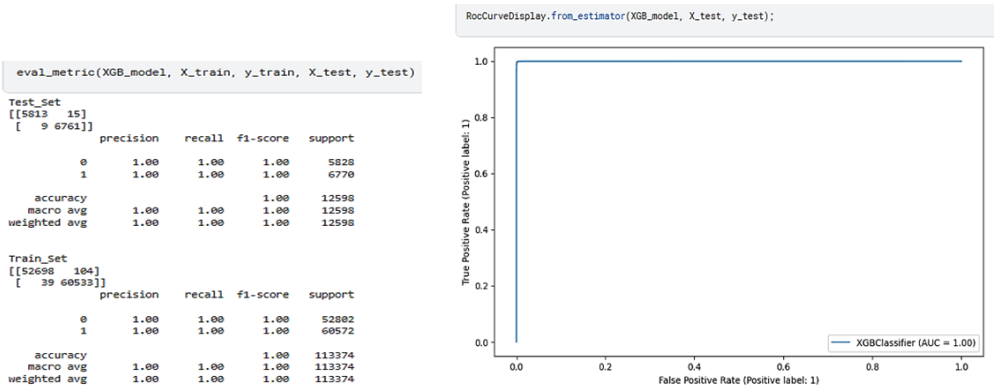



Figure 15. RF evaluation result

To evaluate the model's accuracy before and after hyperparameter tuning, the same dataset was additionally trained using 70:30 and 60:40 data splits. The evaluation results are shown in Tables 1 and 2. Table 1 contains the full accuracy results in percentage. Although the hyperparameters had not yet been established, Table 1 shows that the three algorithms effectively produced models. Every algorithm could generate a model with at least 90% accuracy. For instance, with an 80:20 data split, the RF algorithm's accuracy was close to 100%. Across all data splits, each model's accuracy increased when hyperparameters were tuned during training.

Table 1. Model's Accuracy Before Hyperparameter

| Splitting Data | SVM Algorithm | XGBoost Algorithm | RF Algorithm |
|---|---|---|---|
| 80:20:00 | 90% | 97% | 98% |
| 70:30:00 | 88% | 95% | 97% |
| 60:40:00 | 87% | 94% | 96% |

Table 2. Model's Accuracy Result After Hyperparameter

| Splitting Data | SVM Algorithm | XGBoost Algorithm | RF Algorithm |
|---|---|---|---|
| 80:20:00 | 98% | 100% | 100% |
| 70:30:00 | 98% | 98% | 99% |
| 60:40:00 | 97% | 97% | 98% |

Table 2 presents the model accuracy results after hyperparameter tuning. Table 2 indicates a notable rise in the SVM algorithm. For the SVM model, the hyperparameters C=10, kernel=rbf, and gamma=scale yielded the highest accuracy. Although SVM's accuracy increased with hyperparameter tuning, it remained lower than that of RF and XGBoost. This may be caused by several factors, including improper hyperparameter selection, the algorithm's capacity to handle features, and SVM's limitations when processing NSL-KDD categorical data. The models produced by the XGBoost and RF algorithms were accurate. This indicates that the models detected intrusions with a very high degree of accuracy overall. It also showed that the models had understood the underlying patterns in the data and were able to generalize them well.

## 4. CONCLUSION

This research has successfully answered the research problem in detecting network intrusions. It has been demonstrated that applying MI and selecting appropriate hyperparameters improves the accuracy of network intrusion detection. The use of MI helps select features relevant to intrusion conditions. This simplifies and accelerates the model generation process for all three algorithms. The use of hyperparameters can also prevent overfitting in algorithms that implement boosting techniques. The results show that the top-performing models were generated by the RF and XGBoost algorithms. These findings indicate that the three algorithms can be implemented to build an adaptive and accurate intrusion detection system. The research provides practical guidance for developing ML-based IDSs. Future research on data splitting could explore other datasets, such as real-time datasets, and apply techniques like transfer learning to improve model generalization.

## 5. ACKNOWLEDGEMENTS

The authors would like to express their deepest gratitude to all those who have contributed to the completion of this research. We want to express our deepest appreciation to our colleagues, for their unwavering support, review the training process, and invaluable feedback throughout the research. Our thanks also go to Institut Bisnis dan Teknologi Pelita Indonesia in Pekanbaru for providing resources and facilities that greatly assisted in conducting this research..

## 6. DECLARATIONS

AI USAGE STATEMENT

The authors acknowledge that Artificial Intelligence tools, including ChatGPT developed by OpenAI, were utilized to support language refinement, grammar correction, and paraphrasing in the manuscript preparation process. The authors confirm that all ideas, data interpretations, and conclusions are their own and not generated by the AI tool.

AUTHOR CONTIBUTION

The first author, Deny Jollyta, contributed to the idea, methodology, and training data. The second author, Yoakhina Nicole Makaruku, contributed to writing and references. The third author, Alyauma Hajjah, contributed to writing and analysis. The fourth author, Yulvia Nora Marlim, contributed to writing and model creation.

FUNDING STATEMENT

Thank you to Institut Bisnis dan Teknologi Pelita Indonesia and Institut Agama Kristen Negeri (IAKN) Ambon for providing funding for this research.

COMPETING INTEREST

The authors declare that there are no competing interests associated with the research presented in this manuscript.

## REFERENCES

[1] W. Tang and Y. Liu, "University mobile employment network information system in the internet age," *Journal of Physics: Conference Series*, vol. 1881, no. 2, p. 022095, Apr. 2021, https://doi.org/10.1088/1742-6596/1881/2/022095.

[2] S. Rysbekov, A. Aitbanov, Z. Abdiakhmetova, and A. Kartbayev, "Advancing network security: A comparative research of machine learning techniques for intrusion detection," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 15, no. 2, p. 2271, Apr. 2025, https://doi.org/10.11591/ijece.v15i2.pp2271-2281.

[3] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, Jan. 2021, https://doi.org/10.1002/ett.4150.

[4] M. Kaif, P. P, and L. V, "A study on network intrusion detection system," *International Journal For Multidisciplinary Research*, vol. 6, no. 3, p. 20214, Jun. 2024, https://doi.org/10.36948/ijfmr.2024.v06i03.20214.

[5] Y. Zhang, "Fwa-svm network intrusion identification technology for network security," *IEEE Access*, vol. 13, pp. 18 579–18 593, January, 2025, https://doi.org/10.1109/ACCESS.2025.3532619.

[6] K. M. Abuali, L. Nissirat, and A. Al-Samawi, "Intrusion detection techniques in social media cloud:review and future directions," *Wireless Communications and Mobile Computing*, vol. 2023, pp. 1–25, Apr. 2023, https://doi.org/10.1155/2023/6687023.

[7] K. A. Binsaeed and A. M. Hafez, "Enhancing intrusion detection systems with xgboost feature selection and deep learning approaches," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 5, 2023, https://doi.org/10.14569/IJACSA.2023.01405112.

[8] H. S. Neto, W. S. Lacerda, and R. V. Francozo, "Random forests for online intrusion detection in computer networks," *Journal of Computer Science*, vol. 17, no. 10, pp. 905–914, Oct. 2021, https://doi.org/10.3844/jcssp.2021.905.914.

[9] P. V. Chavan and N. V. Alone, "Optimizing intrusion detection with random forest:a high-accuracy approach using cic-ids 2017," *International Journal of Computer Applications*, vol. 187, no. 3, pp. 17–22, May 2025, https://doi.org/10.5120/ijca2025924816.

[10] S. A. Ajagbe, J. B. Awotunde, and H. Florez, "Intrusion detection:a comparison study of machine learning models using unbalanced dataset," *SN Computer Science*, vol. 5, no. 8, p. 1028, Nov. 2024, https://doi.org/10.1007/s42979-024-03369-0.

[11] A. Dhindsa, S. Bhatia, S. Agrawal, and B. S. Sohi, "An improvised machine learning model based on mutual information feature selection approach for microbes classification," *Entropy*, vol. 23, no. 2, p. 257, Feb. 2021, https://doi.org/10.3390/e23020257.

[12] M. Hassan and N. Kaabouch, "Impact of feature selection techniques on the performance of machine learning models for depression detection using eeg data," *Applied Sciences*, vol. 14, no. 22, p. 10532, Nov. 2024, https://doi.org/10.3390/app142210532.

[13] A. Alsahaf, N. Petkov, V. Shenoy, and G. Azzopardi, "A framework for feature selection through boosting," *Expert Systems with Applications*, vol. 187, p. 115895, Jan. 2022, https://doi.org/10.1016/j.eswa.2021.115895.

[14] L. Ragha and H. S. Deshpande, "A hybrid random forest-based feature selection model using mutual information and f-score for preterm birth classification," *International Journal of Medical Engineering and Informatics*, vol. 15, no. 1, p. 1, 2023, https://doi.org/10.1504/IJMEI.2023.10051207.

[15] C. Arnold, L. Biedebach, A. Kupfer, and M. Neunhoeffer, "The role of hyperparameters in machine learning models and how to tune them," *Political Science Research and Methods*, vol. 12, no. 4, pp. 841–848, Oct. 2024, https://doi.org/10.1017/psrm.2023.61.

[16] M. A. K. Raiaan, S. Sakib, N. M. Fahad, A. A. Mamun, M. A. Rahman, S. Shatabda, and M. S. H. Mukta, "A systematic review of hyperparameter optimization techniques in convolutional neural networks," *Decision Analytics Journal*, vol. 11, p. 100470, Jun. 2024, https://doi.org/10.1016/j.dajour.2024.100470.

[17] J. A Ilemobayo, O. Durodola, O. Alade, O. J Awotunde, A. T Olanrewaju, O. Falana, A. Ogungbire, A. Osinuga, D. Ogunbiyi, A. Ifeanyi, I. E Odezuligbo, and O. E Edu, "Hyperparameter tuning in machine learning:a comprehensive review," *Journal of Engineering Research and Reports*, vol. 26, no. 6, pp. 388–395, Jun. 2024, https://doi.org/10.9734/jerr/2024/v26i61188.

[18] H. Tariq, M. Majeed, and M. Ahmad, "Optimizing svm performance through combinatorial hyperparameter tuning and model selection," *International Journal Bioautomation*, vol. 29, no. 2, pp. 117–144, Jun. 2025, https://doi.org/10.7546/ijba.2025.29.2.000981.

[19] "Machine learning-based network anomaly detection:design."

[20] M. Das Nath and T. Bhattasali, "Anomaly Detection Using Machine Learning Approaches," *Azerbaijan Journal of High Performance Computing*, vol. 3, no. 2, pp. 196–206, Dec. 2020, https://doi.org/10.32010/26166127.2020.3.2.196.206.

[21] T. A. Deepak, "Xgboost classification based network intrusion detection system for big data using pysparkling water," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 1, pp. 377–382, Feb. 2020, https://doi.org/10.30534/ijatcse/2020/55912020.

[22] Z. Arif Ali, Z. H. Abduljabbar, H. A. Tahir, A. Bibo Sallow, and S. M. Almufti, "extreme gradient boosting algorithm with machine learning: A review," *Academic Journal of Nawroz University*, vol. 12, no. 2, pp. 320–334, May 2023, https://doi.org/10.25007/ajnu.v12n2a1612.

[23] E. Ismanto, J. Al Amien, and V. Vitriani, "A comparison of enhanced ensemble learning techniques for internet of things network attack detection," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 23, no. 3, pp. 543–556, Jun. 2024, https://doi.org/10.30812/matrik.v23i3.3885.

[24] W. Li, "Optimization and application of random forest algorithm for applied mathematics specialty," *Security and Communication Networks*, vol. 2022, pp. 1–9, May 2022, https://doi.org/10.1155/2022/1131994.

[25] M. Savargiv, B. Masoumi, and M. R. Keyvanpour, "A new random forest algorithm based on learning automata," *Computational Intelligence and Neuroscience*, vol. 2021, no. 1, p. 5572781, Jan. 2021, https://doi.org/10.1155/2021/5572781.

[26] I. Ahmad and H. S. A. Qahtani, "A comparative analysis of gradient boosting, random forest and deep neural networks in intrusion detection system," *ARPN Journal of Engineering and Applied Sciences*, vol. 8, no. 12, pp. 1392–1402, aug 2023, https://doi.org/10.59018/0623177.

[27] M. T. Abdelaziz, A. Radwan, H. Mamdouh, A. S. Saad, A. S. Abuzaid, A. A. AbdElhakeem, S. Zakzouk, K. Moussa, and M. S. Darweesh, "Enhancing network threat detection with random forest-based nids and permutation feature importance," *Journal of Network and Systems Management*, vol. 33, no. 1, p. 2, Jan. 2025, https://doi.org/10.1007/s10922-024-09874-0.

[28] C. V. Priscilla and D. P. Prabha, "A two-phase feature selection technique using mutual information and XGB-RFE for credit card fraud detection," *International Journal of Advanced Technology and Engineering Exploration*, vol. 8, no. 85, pp. 1656–1668, Dec. 2021, https://doi.org/10.19101/IJATEE.2021.874615.

[29] F. Aghamohammadi and F. Shakeri, "The critical role of hyperparameter tuning in machine learning: A focus on the svd method for matrix completion," *International Journal of Computer Applications*, vol. 187, no. 24, pp. 1–6, Jul. 2025, https://doi.org/10.5120/ijca2025925371.

[30] F. S. Nahm, "Receiver operating characteristic curve: Overview and practical use for clinicians," *Korean Journal of Anesthesiology*, vol. 75, no. 1, pp. 25–36, Feb. 2022, https://doi.org/10.4097/kja.21209.

[31] S. K. Corbaci ouglu and G. Aksel, "Receiver operating characteristic curve analysis in diagnostic accuracy studies: A guide to interpreting the area under the curve value," *Turkish Journal of Emergency Medicine*, vol. 23, no. 4, pp. 195–198, Oct. 2023, https://doi.org/10.4103/tjem.tjem_182_23.