# Performance Comparison of Feature Selection Methods for Machine Learning Models on DDoS Attack Dataset

Nicole Kayambu
*Department of Electrical & Computer Engineering*
*University of Texas- Rio Grande Valley*
Edinburg, USA
nicole.kayambu01@utrgv.edu

Sanjeev Kumar
*Department of Electrical & Computer Engineering*
*University of Texas- Rio Grande Valley*
Edinburg, USA
sj.kumar@utrgv.edu

2025 IEEE World AI IoT Congress (AIIoT) | 979-8-3315-2508-8/25/$31.00 ©2025 IEEE | DOI: 10.1109/AIIoT65859.2025.11105263

*Abstract*—Distributed Denial of Service (DDoS) attacks have posed a major threat to the stability and security of computer networks and Internet. Detection and mitigation of DDoS attacks remain challenging due to the methods in which such attacks are launched. Machine Learning (ML) and Artificial intelligence (AI) are powerful tools that can be used to develop effective Intrusion Detection Systems (IDS) for the purpose of detecting and mitigating DDoS attacks. ML and AI models, however, require different features to be observed to create an efficient model. In this study, four different feature selection methods were used to determine the efficiency of selected features and to determine the effect they have on model accuracy-based on DDoS attack data from the CIC-DDoS 2019 dataset. The feature selection methods explored in this paper were Extra Trees Regressor, Decision Tree Regressor, Mutual Information, and Analysis of Variance. For this study, 200,000 samples of attack data were used. To address the data imbalance of this dataset, Synthetic Minority Oversampling Technique (SMOTE) was used in combination with Edited Nearest Neighbors (ENN). The model used to test the different feature selection methods is the Random Forest (RF) scheme, which is common among ML DDoS detection and mitigation applications. The model was evaluated using metrics such as precision, recall, F1-score, balanced accuracy, and the AUC score for each of the feature importance and selection methods. In this paper, the performance of these models was obtained and analyzed alongside their run times. Analysis of results showed a 75% decrease in testing times when using the top 5 features to train the RF model with SMOTE+ENN.

***Keywords—DDoS attack, Machine Learning, Feature selection***

## I. INTRODUCTION

Distributed Denial of Service (DDoS) attacks are a critical threat to network infrastructure [1]. These attacks target computer systems by exhausting its resources to make it crash. DDoS attacks can severely decrease the availability of the services offered by the targeted systems.

Machine learning and deep learning applications to help detect and mitigate these attacks have been studied in recent years, using a variety of algorithms. A popular application is in intrusion detection systems, which are used to detect potentially harmful network traffic to ensure the safety of a network [2].

This paper offers an analysis of the effects of the feature selection methods and number of features on the performance of the Machine learning models to allow for a methodical approach to feature selection for DDoS datasets used.

This paper is structured as follows: Section II, which contains related works, Section III, which explains the proposed methodology of this study, Section IV, discussing the experiment and an analysis of the results, Section V, which contains the conclusion of this study, and Section VI, which adds future works of this study.

## II. RELATED WORK

Some of previously published research work related to detection of DDoS attacks using machine learning schemes and feature selection methods are described below.

Deb et al. [2] provides analysis of the performance of different machine learning algorithms when using different data balancing techniques. The dataset used for this paper, the CIC-DDoS 2019 dataset, was a highly imbalanced dataset, which required some sort of balancing technique to resolve. In this paper [2], three machine learning algorithms were used, namely, Random Forest

(RF), Naïve Bayes (NB), and K-Nearest Neighbors (KNN), and four different balancing techniques were used which were SMOTE, ADASYN, SMOTE + Tomek Link, and SMOTE + ENN. It was concluded that the Random Forest along with SMOTE + ENN had the superior performance. However, for preprocessing data, it was not clear how or what specific feature selection method was used. In this paper [2], it seemed like some educated technical insights were used to select features for modeling purposes.

Ning et al. [3] provide different feature engineering methods in machine learning and deep learning for network intrusion detection systems. This paper [3] used the NSL-KDD dataset for feature engineering to evaluate the effectiveness of these methods on different machine learning and deep learning algorithms. The different feature engineering methods used were Mutual Information, Entropy, Chi-Square, and Analysis of Variance (ANOVA), and the different machine learning and deep learning methods used were Random Forest, Support Vector Machines, Recurrent Neural Networks, and deep learning with Multi-Layer Perceptron. This study found that machine learning algorithms relied more on feature engineering methods than deep learning algorithms. Of the machine learning algorithms used in this study, Random Forest performed the best. DDoS attacks using machine learning and feature selection methods. No specific data balancing technique was discussed in this paper.

Saha et al. [4] explains the usage of different feature selection methods to find an optimal feature subset. This study had used UNSW-NB15 dataset to classify DDoS attacks using machine learning and deep learning models. This study had also used the ensemble feature selection technique, Majority Voting, to combine the different feature selection method results to extract an optimal feature set. This study used 15 different feature selection methods and had found that the ensemble feature set based classification models had higher accuracy, lower false positive rate, and better execution time than individual feature set based models. DDoS attacks using machine learning and feature selection methods. The dataset used in this paper was mentioned to be balanced.

Prima et al. [5] compares different machine learning models for classification of DDoS attacks. This paper had used a DDoS dataset from Kaggle, which contained data extracted from public intrusion detection system datasets. This paper studied 14 different models and found that Multi-Layer Perceptron, K-Nearest Neighbors, Gradient Boosting Classifier, Extra Trees Classifier, Support Vector Classifier, XGBoost Classifier, and Random Forest had achieved 1.0 in all evaluation metrics, which includes accuracy, precision, recall, and F1-Score. This paper had not mentioned any feature selection method when discussing preprocessing of the data. DDoS attacks using machine learning and feature selection methods. This paper mentioned no specific balancing method for the data.

Arora et al. [6] uses the CIC-DDoS 2019 dataset to compare different machine learning and deep learning algorithms to detect DDoS attacks. The different machine learning and deep learning algorithms used in this paper are Gated Recurrent Unit, Recurrent Neural Network, Random Forest, K-Nearest Neighbors, Long Short-Term Memory, and Linear Regression. Of the machine learning algorithms used in this study, Random Forest had performed slightly lower than the Gated Recurrent Unit and Recurrent Neural Network at 99% on accuracy, precision, recall, and F1-score, compared to the 99.99% or 100% that the Recurrent Neural Network and Gated Recurrent Unit achieved. This study had also not mentioned any specific feature selection method being used. No data balancing technique was mentioned in this paper.

Essa and Bhaya [7] studied majority voting and averaging techniques are used to select features to be used for different machine learning algorithms to achieve best performance for each classifier. This paper had used the InSDN dataset as their use case was for Software Defined Networks. This paper had used a combination of different feature selection methods of all types, wrapper-based, embedded methods, and filter-based methods. Then they are passed to three feature selection methods, sequential forward selection, Extra Trees Classifier, and Mutual Information, to extract the smallest number of features and test the different algorithms with those features. The different machine learning algorithms tested are Support Vector Machines, eXtreme Gradient Boosting, Multilayer Perceptron, Decision Tree, Naïve Bayes, Random Forest, and K-Nearest Neighbors. After the feature selection process, of the original 77 features, only 12 were chosen for testing. This study found that Random Forest had the best performance for protecting these networks. This paper had mentioned no imbalance in the dataset.

In the discussed literature, Random Forest was a common machine learning algorithm used that had achieved high performance, while no specific balancing techniques were used, or had used data that was already balanced. In this study, we plan to use Random Forest in combination with Synthetic Minority Oversampling Technique (SMOTE) with Edited Nearest Neighbors (ENN) to evaluate different feature selection methods for DDoS attack data.

## III. Proposed Methodology

In this research, the CIC-DDoS 2019 dataset was used. The data used had to first be preprocessed before feature selection methods could be applied. Once features were selected, they could be used for data balancing and then training the model to obtain performance metrics. The model itself will be used for binary classification and the precision, recall, F1 score, balanced accuracy, and AUC score will be obtained for evaluation purposes.

### A. Dataset

The dataset used in this paper is the CIC 2019 DDoS. This dataset is developed by the Canadian Institute for Cybersecurity from the University of New Brunswick. This dataset is divided into two categories, reflection-based DDoS attacks, and exploitation-based attacks [8]. From this dataset, only a subset of 200,000 samples was used. Benign data in this dataset is labeled using '0' while the attack data is labeled using '1'. This distribution of the dataset is shown Table 1, which highlights the data imbalance.

Table 1: Data Distribution (CIC 2019 DDoS Dataset)

| Data Type | Number of Samples |
|---|---|
| Attack Data | 199653 |
| Benign Data | 347 |
| Total | 200000 |

### B. Data Preprocessing

The dataset is stored in a CSV file and contains 88 different features. To preprocess this dataset, several steps were taken utilizing Python libraries such as Pandas, NumPy, and Scikit-learn.

- Removing Null and Infinite Values: Columns containing missing, null, or infinite values were removed. After removal, a total of 82 features remaining.
- Removing Columns of Zero Values: Columns containing only '0' values were also removed from the dataset. This left 76 features.
- Removing Columns with Unneeded String Information: Columns of unneeded string information such as IP addresses, timestamp, and flow ID, were removed from the dataset. This reduced the number of features to 70.
- Feature Scaling: To handle the variation in the data of each feature, feature scaling was used. The data in this dataset does not follow any normal distribution, so min-max scaling was performed using equation (1).

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (1)$$

- Label Encoding: The target feature was encoded using the LabelEncoder function from the Scikit-learn library for binary classification.

### C. Data Balancing

The data was balanced using the Synthetic Minority Oversampling Technique (SMOTE) in combination with Edited Nearest Neighbors (ENN), which is a well performing method to balance data [2]. This allows for oversampling of the minority class to be done using SMOTE and noisier samples that were generated to be removed using ENN, increasing the difference in the class boundaries to enhance performance of the model. More information on how this method works is found in Batista et al. [9].

### D. Model

The machine learning algorithm used in this study was Random Forest (RF) classifier. RF is a popular ML algorithm used in DDoS applications. RF is an ensemble of decision trees which are used to classify entries by evaluating the entry by all the trees in the forest, and the final classification of the entry is determined by what the majority of the trees evaluated the entry as [10]. Randomization is used in the generation of trees to allow for diversity among the decision trees [10].

### E. Feature Selection Methods

In this study, four different feature selection methods were used to select features to train the models and evaluate performance. Feature selection functions are used from the Python Scikit-learn library. These features selection methods were used as they are commonly used in other similar DDoS attack datasets [3][4]. Listed below are the different feature selection methods.

- Extra Trees Regressor: Extra Trees Regressor is a tree based ensemble method used for regression problems. It consists of strongly randomizing the attribute and cut off points when splitting tree nodes. It is used to improve predictive accuracy and control overfitting. In this case it is used to rank the importance of different features in the dataset to allow for more significant features to be selected [11].
- Decision Tree Regressor: Decision Tree Regressor is a tree based supervised learning method which predicts the value of a target variable by learning decision rules learned from

the given features. In this case, it is used to rank the importance of different features in the dataset to be able to select more significant features [10][12][13].

- Mutual Information: Mutual information is a method which measures the dependency between variables. These dependencies are based on entropy estimations from K Nearest Neighbors. In this study, MI is used to rank features that the target depends on more to be used to train the model [14].
- Analysis of Variance: Analysis of Variance (ANOVA) is a method which checks if the means of two or more groups are significantly different from each other. In this case it is used to check which features are significantly different to each other and still be significant to classifying the given data [15].

### F. Evaluation Metrics

The evaluation metrics obtained are commonly used to evaluate the performance of models. The purpose of collecting these metrics is to analyze the performance of the different features selected on the model to determine which feature selection method offers the best performance. The evaluation metrics are calculated using the collected confusion matrix. The confusion matrix contains the following elements.

- True Positive (TP): Attack classified correctly as attack.
- True Negative (TN): Benign classified correctly as benign.
- False Positive (FP): Benign classified incorrectly as attack.
- False Negative (FN): Attack classified incorrectly as benign.

Using the confusion matrix, the evaluation metrics calculated are precision, recall, F1 score, balanced accuracy, and AUC score which are listed in the following equations.

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

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

$$F1\text{-}Score = \frac{2xrecallxprecision}{recall + precision} \quad (4)$$

$$Balanced\ Accuracy = \frac{Sensitivity + Specificity}{2} \quad (5)$$

$$AUC\ Score = \int_0^1 ROC(x)dx \quad (6)$$

In which, ROC is the receiver operating characteristic graph, which is obtained by plotting the true positive rate against the false positive rate.

## IV. Experiment and Result Analysis

### A. Hardware Utilized for Experiment

The hardware utilized for feature selection, training, and testing the models generated for each of the feature selection methods was a computer which contained an Intel Core Ultra 7 155H, a 3.80GHz processor. This is vital to consider when discussing timing as the speed of the computer's processor will also affect these outcomes.

### B. Selected Features Using Each Method

For each of the feature selection methods, the top 20, 15, 10, and 5 features were chosen and used to train the model to evaluate the effect the number of features had on model performance. The top 20 features are shown in Table 2 in order of significance. The top 15, 10, and 5 features are similarly chosen from Table 2. Of the 76 original features available after data preprocessing, only 46 different features overall were chosen by the different feature selection methods as significant. Understanding these different features is important to consider when reviewing the mechanisms of a DDoS attack. Context of these features and their representations allows for consideration of how the different types of features play a role in the determination of a given traffic being counted as attack or not [16].

Table 2: Top 20 Features Selected by Each Selection Method

| No. | Extra Trees Regressor | Decision Tree Regressor | Mutual Information | ANOVA |
|---|---|---|---|---|
| 1 | Inbound | Inbound | Packet Length Std | Inbound |
| 2 | URG Flag Count | Source Port | Init_Win_bytes_forward | ACK Flag Count |
| 3 | SYN Flag Count | Destination Port | Packet Length Mean | Bwd Packet Length Min |
| 4 | Destination Port | Bwd Header Length | ACK Flag Count | Protocol |
| 5 | Min Packet Length | Init_Win_bytes_forward | Max Packet Length | URG Flag Count |
| 6 | Fwd Packet Length Min | Protocol | Flow Duration | Bwd Packet Length Mean |

| | | | | |
|---|---|---|---|---|
| 7 | Source Port | Flow Duration | Fwd Packet Length Max | Avg Bwd Segment Size |
| 8 | min_seg_size_forward | Total Fwd Packets | Destination Port | CWE Flag Count |
| 9 | Init_Win_bytes_forward | Total Backward Packets | Fwd Header Length | Packet Length Std |
| 10 | Bwd IAT Min | Total Length of Fwd Packets | Bwd Packets/s | Max Packet Length |
| 11 | Fwd Header Length | Total Length of Bwd Packets | Inbound | Bwd Packet Length Max |
| 12 | Fwd Packets/s | Fwd Packet Length Max | Packet Length Variance | act_data_pkt_fwd |
| 13 | Active Min | Fwd Packet Length Min | Avg Fwd Segment Size | Fwd PSH Flags |
| 14 | Avg Fwd Segment Size | Fwd Packet Length Mean | Fwd Packet/s | RST Flag Count |
| 15 | Fwd IAT Total | Fwd Packet Length Std | Avg Bwd Segment Size | Fwd Packet Length Std |
| 16 | Fwd Header Length | Bwd Packet Length Max | Total Length of Fwd Packets | Bwd Packet Length Std |
| 17 | Idle Mean | Bwd Packet Length Min | Bwd Packet Length Max | Packet Length Variance |
| 18 | Bwd Header Length | Bwd Packet Length Mean | Flow IAT Std | Fwd Packet Length Max |
| 19 | Flow IAT Max | Bwd Packet Length Std | Fwd Packet Length Mean | Total Length of Fwd Packets |
| 20 | Total Backward Packets | Flow IAT Mean | Average Packet Size | Subflow Fwd Bytes |

Table 3: Performance Evaluation using RF model for Top 20 Features

| Selection Method | F1-Score | Balanced Accuracy | AUC Score |
|---|---|---|---|
| Extra Trees Regressor | 0.9963728 ± 0.00052 | 0.9956588 ± 0.00006 | 0.9956588 ± 0.00006 |
| Decision Tree Regressor | 0.9971060 ± 0.00129 | 0.9962318 ± 0.00077 | 0.9962318 ± 0.00077 |
| Mutual Information | 0.9917504 ± 0.00143 | 0.9956424 ± 0.00005 | 0.9878214 ± 0.01748 |
| ANOVA | 0.9782478 ± 0.00064 | 0.9970266 ± 0.00004 | 0.9970266 ± 0.00004 |

### *C. Performance Analysis of Feature Selection Methods*

The feature selection methods were evaluated by collecting different evaluation metrics from each of the models. The dataset was split using 10-fold cross validation. Each of the models was run five times for each of the different number of the top features (20, 15, 10, 5) which are obtained from Table 2, evaluation metrics were averaged, and the standard deviation was obtained. The features obtained using the Decision Tree Regressor had achieved the best performance among most the models despite the change in the number of features used. Extra Trees Regressor outperformed Decision Tree Regressor features but only in the case of 10 features. Features selected using ANOVA showed the worst performance among all the feature selection methods tested. Performance was evaluated using the RF model created and with SMOTE + ENN for data balancing. Overall, Extra Trees Regressor and Decision Tree Regressor had similar performance, performing slightly better than both Mutual Information and ANOVA. When looking into the specific features selected, it can be considered that Extra Trees Regressor had better performance as the features selected by this method are used more commonly in traditional DDoS attack detection. Specifically noting the different features that record flag values such as URG and SYN flag counts. This is valuable as it can be considered alongside more traditional detection methods and newer methodologies. Other features of note are the Active Min and Idle Mean features, which are defined as the minimum time a flow was active before becoming idle, and the meantime a flow was idle before becoming active [16]. These can be significant when analyzing how these features may relate to different DDoS attacks. For example, SYN Flood attacks require a large number of SYN connections in a short amount of time to successfully deny the service of a specific device. This reveals that the features selected, and the number of features, have some effect on the performance of the model. However, the effects that the feature selection methods had on the model performance were minimal.

Table 4: Performance Evaluation using RF model for Top 15 Features

| Selection Method | F1-Score | Balanced Accuracy | AUC Score |
|---|---|---|---|
| Extra Trees Regressor | 0.9962394 ± 0.00074 | 0.9956582 ± 0.00005 | 0.9956582 ± 0.00005 |
| Decision Tree Regressor | 0.9968280 ± 0.00124 | 0.9962308 ± 0.00077 | 0.9962308 ± 0.00077 |
| Mutual Information | 0.9931576 ± 0.00103 | 0.9956474 ± 0.00005 | 0.9956474 ± 0.00005 |
| ANOVA | 0.9778870 ± 0.00087 | 0.9952808 ± 0.00164 | 0.9952808 ± 0.00164 |

Table 5: Performance Evaluation using RF model for Top 10 Features

| Selection Method | F1-Score | Balanced Accuracy | AUC Score |
|---|---|---|---|
| Extra Trees Regressor | 0.9978378 ± 0.00073 | 0.9979792 ± 0.00081 | 0.9979792 ± 0.00081 |
| Decision Tree Regressor | 0.9976720 ± 0.00057 | 0.9968126 ± 0.00062 | 0.9968126 ± 0.00062 |
| Mutual Information | 0.9892448 ± 0.00129 | 0.9950712 ± 0.00131 | 0.9950712 ± 0.00131 |
| ANOVA | 0.9781594 ± 0.00088 | 0.9955922 ± 0.00005 | 0.9955922 ± 0.00005 |

Table 6: Performance Evaluation using RF model for Top 5 Features

| Selection Method | F1-Score | Balanced Accuracy | AUC Score |
|---|---|---|---|
| Extra Trees Regressor | 0.9940208 ± 0.00074 | 0.9956504 ± 0.00005 | 0.9956504 ± 0.00005 |
| Decision Tree Regressor | 0.9985512 ± 0.00099 | 0.9985434 ± 0.00002 | 0.9985434 ± 0.00002 |
| Mutual Information | 0.9851796 ± 0.00135 | 0.9959202 ± 0.00066 | 0.9959202 ± 0.00066 |
| ANOVA | 0.9698096 ± 0.00289 | 0.9883232 ± 0.00519 | 0.9883232 ± 0.00519 |

As shown in tables 3-6, the F1-score and AUC score were both above 96% for all feature selection methods, with most being above 99%. This tells us that these commonly chosen features, even with the variations of these features chosen by the selection methods, had minimal effect on the model's performance.

Table 7: Accuracy and Timing for Training and Testing with Top 20 Features

| Selection Method | Train Accuracy | Test Accuracy | Training Time (s) | Test Time (s) |
|---|---|---|---|---|
| Extra Trees Regressor | 0.999999 | 0.999975 | 186.63 | 0.60 |
| Decision Tree Regressor | 1.000000 | 0.999985 | 193.63 | 0.63 |
| Mutual Information | 0.999978 | 0.99995 | 185.45 | 0.63 |
| ANOVA | 0.999848 | 0.99984 | 96.91 | 0.55 |

Table 8: Accuracy and Timing for Training and Testing with Top 15 Features

| Selection Method | Train Accuracy | Test Accuracy | Training Time (s) | Test Time (s) |
|---|---|---|---|---|
| Extra Trees Regressor | 0.999999 | 0.999975 | 145.09 | 0.59 |
| Decision Tree Regressor | 1.000000 | 0.999975 | 165.58 | 0.61 |
| Mutual Information | 0.999983 | 0.99995 | 159.59 | 0.62 |
| ANOVA | 0.99985 | 0.999845 | 74.16 | 0.56 |

Table 9: Accuracy and Timing for Training and Testing with Top 10 Features

| Selection Method | Train Accuracy | Test Accuracy | Training Time (s) | Test Time (s) |
|---|---|---|---|---|
| Extra Trees Regressor | 1.000000 | 0.99998 | 129.41 | 0.58 |
| Decision Tree Regressor | 1.000000 | 0.999985 | 156.04 | 0.56 |

| | | | | |
|---|---|---|---|---|
| Mutual Information | 0.999954 | 0.99993 | 168.20 | 0.61 |
| ANOVA | 0.999851 | 0.999845 | 69.81 | 0.59 |

Table 10: Accuracy and Timing for Training and Testing with Top 5 Features

| Selection Method | Train Accuracy | Test Accuracy | Training Time (s) | Test Time (s) |
|---|---|---|---|---|
| Extra Trees Regressor | 0.999981 | 0.99996 | 80.68 | 0.22 |
| Decision Tree Regressor | 0.999999 | 0.999985 | 97.26 | 0.16 |
| Mutual Information | 0.999917 | 0.999905 | 87.87 | 0.22 |
| ANOVA | 0.999786 | 0.999785 | 36.30 | 0.14 |

Comparison of the training and testing times, shown in Tables 7-10, also reveals little difference in performance of each of the models trained with the selected features. It did, however, show some change in training and testing time based on the number of features. For example, when using 10 or more features, the testing time (in seconds) was reduced from between 0.58s and 0.63s to between 0.14s and 0.22s (for top 5 features). This is crucial to consider, while the difference between the F1 & AUC score performance of the 5 feature models compared to the 10 or more feature models is not significantly different (Tables 11-14), however, there is a significant difference in the Testing times. Reduced training times would allow for faster detection when these methods are implemented for real time detection.

*D. Analysis of Testing Time*

Further analysis of the F1-score, AUC score, training time, and testing time was conducted. Since the performance of the top 20 features, top 15 features, and top 10 features models are similar, we will focus on the difference between the top 10 features models and the top 5 features models for analyzing the reduction of the training and testing times.

Table 11: Percent Reduction of Timings for Extra Trees Regressor

| Number of Features | F1-Score | AUC Score | Training Time (s) | Testing Time (s) |
|---|---|---|---|---|
| 10 Features | 0.9978378 | 0.9979792 | 129.40744 | 0.582006 |
| 5 Features | 0.9940208 | 0.9956504 | 80.684119 | 0.215705 |
| Percent Reduction (%) | 0.3825271 | 0.233351557 | 37.65109719 | 62.93766731 |

Table 12: Percent Reduction of Timings for Decision Tree Regressor

| Number of Features | F1-Score | AUC Score | Training Time (s) | Testing Time (s) |
|---|---|---|---|---|
| 10 Features | 0.997672 | 0.9968126 | 156.037184 | 0.564004 |
| 5 Features | 0.9985512 | 0.9985434 | 97.257095 | 0.155914 |
| Percent Reduction (%) | -0.0881251 | -0.1736334 | 37.67056511 | 72.35586982 |

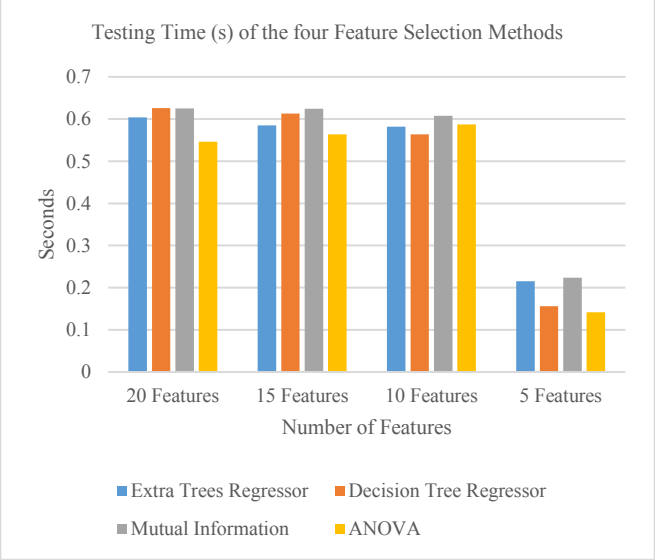
Table 13: Percent Reduction of Timings for Mutual Information

| Number of Features | F1-Score | AUC Score | Training Time (s) | Testing Time (s) |
|---|---|---|---|---|
| 10 Features | 0.9892448 | 0.9950712 | 168.203714 | 0.608011 |
| 5 Features | 0.9851796 | 0.9959202 | 87.870325 | 0.223836 |
| Percent Reduction (%) | 0.4109397 | -0.0853205 | 47.75958098 | 63.18553447 |

Table 14: Percent Reduction of Timings for ANOVA

| Number of Features | F1-Score | AUC Score | Training Time (s) | Testing Time (s) |
|---|---|---|---|---|
| 10 Features | 0.9781594 | 0.9955922 | 69.813866 | 0.587045 |
| 5 Features | 0.9698096 | 0.9883232 | 36.29617 | 0.141741 |
| Percent Reduction (%) | 0.85362365 | 0.73011821 | 48.01008442 | 75.85517294 |

As shown in Tables 11-14, the percent increase and reduction in the F1-score and AUC score are less than 1% for all feature selection methods. What differs, however, is the reduction of the training and testing time. For Extra Trees Regressor models, we see a reduction in the training time by 37.65% and testing time of 62.93%. The Decision Tree Regressor models had a similar reduction in training time of 37.67%, and testing time reduction of 72.35%. The Mutual Information models had a training time reduction of 47.76% and a testing time reduction of 63.19%. Lastly, ANOVA produced the highest reductions of training and testing time, with training time reduced by 48.01%, and testing time reduced by 75.86%. These results show that each of the methods had a significant reduction in timings, when comparing the 10 feature models and the 5 feature models. This tells us that reducing the features selected for training and testing models to 5 features can reduce the training times up to 48% and testing times up to 75%, while having minimal decreases in performance, regardless of which feature selection method is utilized. Fig. 1 illustrates the differences between the testing times for the 20, 15, 10, and 5 feature models, highlighting the reduction in the testing time between the 5 feature models and the models with higher feature counts.




## V. Conclusion

The purpose of this study was to observe the effects of feature selection methods, and the number of features used on model performance for the detection and mitigation of DDoS attacks using the CIC-DDoS 2019 dataset. The four different feature selection methods chosen for this task are Extra Trees Regressor, Decision Tree Regressor, Mutual Information, and ANOVA. The performance of these features was determined using Random Forest and the imbalance in the dataset was resolved using SMOTE + ENN for binary classification based on previous works. Lastly, performance metrics such as precision, recall, F1 score, balanced accuracy and AUC score were collected to evaluate the performance of each of the models for comparison. In conclusion, for each of the numbers of features selected the models performed similarly, with accuracies of 96% or higher for all models. What was found to differ was that among all the number of features selected, there was a noticeable decrease in the testing time, up to 75% decrease, for the models trained using 5 features, despite the different feature selection methods applied. This can be significant when considering how many features to use when training models for real time applications when using the CIC-DDoS 2019 dataset.

## Acknowledgment

Authors will like to thank Dipok Deb, Vincent Agbenyeavu, Hansapani Rodrigo for useful discussions on AI/ML tools and techniques.

This research, in part, was supported by NSF Award number 2334389; and supported in part by the US DHS Science and Technology Summer Research Team program ORISE, ORAU under DOE contract DE-SC0014664.

All opinions expressed in this article are that of authors' and do not necessarily reflect the policies and views of NSF, DHS, DOE, ORAU/ORISE.

## References

[1] S. Kumar and Einar Petana, "Mitigation of TCP-SYN Attacks with Microsoft's Windows XP Service Pack2 (SP2) Software," Apr. 2008, doi: https://doi.org/10.1109/icn.2008.77.

[2] D. Deb, H. Rodrigo, and S. Kumar, "Performance Analysis of Machine Learning Algorithms on Imbalanced DDoS Attack Dataset," May 2024, doi: https://doi.org/10.1109/aiiot61789.2024.10579021.

[3] S. Ning, K. Nguyen, S. Bagchi, and Y. Park, "The Study of Feature Engineering in Machine Learning and Deep Learning for Network Intrusion Detection Systems," *2024 Silicon Valley Cybersecurity Conference (SVCC)*, vol. 2, pp. 1–5, Jun. 2024, doi: https://doi.org/10.1109/svcc61185.2024.10637359.

[4] S. Saha, A. T. Priyoti, A. Sharma, and A. Haque, "Towards an Optimal Feature Selection Method for AI-Based DDoS Detection System," *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*, Jan. 2022, doi: https://doi.org/10.1109/ccnc49033.2022.9700569.

[5] F. Prima, L. Dylan, and A. Agung, "Comparison of Machine Learning Models for Classification of DDoS Attacks," pp. 1–6, Oct. 2023, doi: https://doi.org/10.1109/icoris60118.2023.10352232.

[6] S. Arora, P. Khare, and S. Gupta, "AI-Driven DDoS Mitigation at the Edge: Leveraging Machine Learning for Real-Time Threat Detection and Response," pp. 1–7, Jul. 2024, doi: https://doi.org/10.1109/icdsns62112.2024.10690930.

[7] A. Al and W. S. Bhaya, "Detection of DDoS Attacks in Software-Defined Networks Based on Majority Voting-Average for Feature Selection and Machine Learning Approaches," vol. 81, pp. 1–6, Feb. 2023, doi: https://doi.org/10.1109/aca57612.2023.10346864.

[8] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy," *2019 International Carnahan Conference on Security Technology (ICCST)*, Oct. 2019, doi: https://doi.org/10.1109/ccst.2019.8888419.

[9] G. Batista, R. C. Prati, and M. C. Monard. "A study of the behavior of several methods for balancing machine learning training data," ACM Sigkdd Explorations Newsletter 6 (1), 20-29, 2004.

[10] L. Breiman, "Random Forests", Machine Learning, 45(1), 5-32, 2001.

[11] P. Geurts, D. Ernst., and L. Wehenkel, "Extremely randomized trees", Machine Learning, 63(1), 3-42, 2006.

[12] L. Breiman, J. Friedman, R. Olshen, and C. Stone, "Classification and Regression Trees", Wadsworth, Belmont, CA, 1984.

[13] T. Hastie, R. Tibshirani and J. Friedman. "Elements of Statistical Learning", Springer, 2009.

[14] Strehl, Alexander, and Joydeep Ghosh (2002). "Cluster ensembles - a knowledge reuse framework for combining multiple partitions". Journal of Machine Learning Research 3: 583-617. doi:10.1162/153244303321897735.

[15] R. Lowry, "Concepts and Applications of Inferential Statistics", Chapter 14, 2014.

[16] CanadianInstituteForCybersecurity, "CICFlowMeter/ReadMe.txt at master · CanadianInstituteForCybersecurity/CICFlowMeter," *GitHub*, 2020. https://github.com/CanadianInstituteForCybersecurity/CICFlowMeter/blob/master/ReadMe.txt