Chapter 6

INFS-MICC: An Incremental Feature Selection Method

6.1 Introduction

In the field of network intrusion detection, feature selection is a crucial step. To achieve better classification performance, selection of an optimal subset of features can help significantly. The learning model becomes needlessly complex and challenging to comprehend when irrelevant and redundant features are used. A subset of features from an initial collection of features in machine learning are chosen in feature selection [181]. Both supervised and unsupervised feature selection techniques have been put forth over the years. By enhancing model performance, enabling improved data visualization and comprehension, and better use of computational resources, feature selection strategies improve the learning process [128].
A quality feature selection method selects a subset of relevant features i.e. those features which distinguishingly characterize the target class. The distinction between the target classes is made possible by feature relevance, without which the feature does not contribute to the prediction of the target class for a given test instance. At the same time, the features should operate independently of one another i.e. redundancy among the features should be less. Duplicate or dependent features do not provide any new knowledge to the learning process and may reduce model performance. As a result, the candidate features in a chosen subset should be independent of one another.

6.1.1 Incremental Data

With whirlwind developments in information technology all around, there is rapid increase in data being generated. Generation and collection of data is always an ongoing process. Data that is relevant today for a system may not be relevant after a few years. On the other hand, it may so happen that all the data for an already functioning system may not be available at one go. Such systems are dynamic systems and such data in the literature are termed as incomplete data [229]. At different points in time, data for such a system may inhibit dynamic varying characteristics. This means that, for any object the features that characterize the object may vary dynamically with time. Such added-in data are called the incremental data.

6.1.2 Necessity of Incremental Feature Selection

In dynamic real world applications, data collection and generation may be an ongoing process. A traditional (non-incremental) feature selection method is typically computationally expensive for such dynamic data because, whenever new data arrives all the computations are again done from scratch. This issue necessitates the need of an incremental approach to feature selection so as to speed up the feature selection process in dynamic data.

6.1.3 Related Work

6.1.3.1 HTTP Flooding Attacks

Tremendous amount of research efforts have been made to detect HTTP flooding attacks at the earliest with minimum false alarms. In the literature, researchers have categorized defense mechanisms for detecting HTTP flooding attacks based on disparate ideas. Zargar et al. [105] categorize the mechanisms based on the deployment site. Destination based mechanisms deploy their defenses at the victim end, i.e., at the Web server end, and Hybrid mechanisms are deployed on both the client and the server. Praseed et al. [106] propose a taxonomy where the detection mechanisms are classified according to request dynamics (traffic estimation, request statistics like entropy based measures), and request semantics (request composition and request sequence). Singh and De [107] use a multi layer perceptron (MLP) to detect HTTP flooding attacks with features such as HTTP requests count, number of the IP addresses. For learning and adjusting the weights of the perceptrons genetic algorithm is used. The main advantage of this method is that it provides a high detection accuracy and a low false positive rate. Zhao et al. [108] try to differentiate between flash crowd and application layer DDoS attacks. Two measures based on entropy namely, EUPI (Entropy of URL per IP) and EIPU (Entropy of IP per URL) are used. The main idea behind using such measures is that normal requests vary in size, speed and intent and as such have a high entropy value. Whereas attack packets are more similar to one another and thus have low entropy value. Wen et al. [111] propose a traffic estimation based defense mechanism which focuses on request dynamics. If the request rate is above an expected threshold at a particular point of time it means something abnormal (either an attack or flash crowd) is going on. Kalman filter is used as a measure for this purpose. Additionally, source IP distribution is used to actually identify an attack flood. In [230] Yatagai et al. proposes an HTTP-GET flood attack detection method where the underlying idea is to analyse the page access behavior based on two detection methods. First method finds the sequence in which the pages are browsed by a user and the second measures the correlation between browsing time of a page and its information size. The downside to the first method is its low detection accuracy, however it prioritizes acitivities of normal clients, meaning normal clients will not be wrongly barred from their usual activity. On the other hand, the second method promises high detection accuracy but may misclassify a normal client. Dhanapal and Nithyanandam [231] proposes an OpenStack based testbed framework which detects HTTP-Flooding attacks in the cloud computing platform. According to the authors detection of such attacks in the cloud is difficult because of the existence of numerous potential attack paths. Their method is highly accurate in detecting low-rate attacks in the early stages. Similar, techniques for protecting cloud computing platforms are also proposed in [232] and [233]. Mohammadi et al. [234] propose HTTPScout, a security module which helps detect and mitigate flooding attacks using machine learning and Software Defined Networks (SDN). The proposed module continuously observes the incoming HTTP traffic flows. If any particular flow is sensed to be malicious, its source is blocked at the edge switch. This way the valuable network resources are safeguarded from the adversaries. A similar detection method is also proposed in [235], where the authors consider both transport layer and application layer attacks in a modular SDN-based architecture. For detection both machine learning and deep learning models are employed. On the other hand, the Mininet emulator[1] and Open Network Operating System[2] (ONOS) SDN controller is also deployed for implementation in a simulated environment. In the recent years, several such methods to detect application layer DDoS attacks in Software Defined Networks (SDN) have gained popularity [236],[237][238].

6.1.3.2 Feature Selection

The most typical presumption according to many is monotonicity, which is of the notion that adding more features would be beneficial for a learning system perform better [239][240]. Researchers in machine learning and knowledge discovery who are interested in enhancing algorithm performance have over the years given feature selection a lot of interest. Since many learning algorithms may fail or take an excessive amount of time to run before data is reduced, feature selection is a very crucial step in pre-processing when dealing with enormous data. Feature selection methods in the literature are mainly categorized into: Filter, Wrapper and Embedded methods.
In order to give an ordered list of feature ranks, filter methods use statistical measures such as information gain, correlation, and mutual information [127, 187, 188]. These ranking systems aid in highlighting the characteristics that are crucial. Prior to executing the classification task, irrelevant features are filtered out and eliminated because their presence does not help improve the performance of a machine learning algorithm. To maximize the advantages of competitive ranking, numerous filter methods have been utilized in conjunction with population-based heuristic search methodologies [189–192]. Feature-feature and feature-class mutual information are used in the widely used MIFS (Mutual Information Feature Selection) method to choose a feature subset that maximizes classification accuracy [193].
Two categories of wrapper feature subset selection methods that are often used in the literature are sequential selection algorithms and heuristic search algorithms [129, 130, 188, 202, 203]. To choose feature subset, Maldonado and Weber [204] suggest a sequential backward selection wrapper approach utilizing Support Vector Machines (SVMs). Using cosine similarity and SVMs, Gang and Jin [205] choose relevant and independent features.
On the other hand, Hsu et al. [206] provide a hybrid feature selection method, where the filter methods assist in effectively finding the candidate features and the

[1]`http://mininet.org/`
[2]`https://opennetworking.org/onos/`

wrappers are in charge of delivering the subset of features that ensures the best possible classification accuracy. In order to produce distinct lists of ranked features, ensemble feature selection methods like [207] rely on base feature selection algorithms. The final ranked list of features is created by combining these individually ranked features based on a score. In conjunction with four search algorithms, including ensemble forward and backward sequential selection, hill-climbing, and genetic search, Tsymbal et al. [208] examine diversity metrics to assess diversity in the ensemble feature selection methods.

6.1.4 Motivation

In terms of sample sizes and dimensions, the amount of data that is currently available has significantly increased across all fields, including network security, bioinformatics, text classification, and computer vision, to mention a few. Despite the fact that a lot of data is produced, not all of it is of high quality to be used in predictive data analysis. At the same time it is important to note that, data may be continuously arriving at regular intervals. For such cases the learning models need to be trained from scratch which is again computationally expensive and inefficient. To offer valuable insights to predictive modeling, machine learning algorithms need relevant, independent, intelligible, meaningful, and recent data. In light of this, feature selection is essentially an important pre-processing step. It aids a learning model in simplifying the learning process so that it can acquire essential and vital knowledge for predicting tasks. Although a good number of feature selection methods have been introduced, none of these are free from limitations, particularly in the context of network security which demands for best possible accuracy at low cost. All the mentioned reasons have collectively motivated for the development of an incremental feature selection technique which focuses on selecting relevant and irredundant features to achieve best predictive performance with minimum false alarms and at the same time to avoid processing the whole data from scratch when new data instances arrive at regular intervals.

6.1.5 Contribution

The main contribution reported in this chapter is an Incremental Feature Selection method based on Mutual Information and Correlation (INFS-MICC). It is incremental in the sense that it can handle growing data incrementally and can

Table 6.1: Symbol Table for the Proposed Method (INFS-MICC)

Symbol	Symbol Meaning	Symbol	Symbol Meaning
D	Dataset	F_2^{old}	Contains highly relevant features from $F'_{D_{old}}$
s	No. of samples in D	F_3^{new}	Contains highly relevant features from $F'_{D_{new}}$
d	Dimension of the dataset D		
T_m	Time	F^D	Combination of F_1^{common}, F_2^{old} and F_3^{new}
T_n	Time	I	Mutual Information
D_{old}	Data arriving at time T_m	M	Random variable
D_{new}	Data arriving at time T_n	N	Random variable
F	Original feature set of D	m	Marginal probability distribution of M
$F'_{D_{old}}$	Feature subset containing relevant and irredundant features from D_{old}	n	Marginal probability distribution of N
$F'_{D_{new}}$	Feature subset containing relevant and irredundant features from D_{new}	p(m,n)	Joint probability distribution of M and N
F_1^{common}	Contains common features from $F'_{D_{old}}$ and $F'_{D_{new}}$	C_k	Target class

identify a subset of highly relevant and irredundant feature subset without processing the whole data from scratch, thereby saving computational resources. The proposed method is used to detect HTTP-flooding attacks with high accuracy at low cost. Three well-known HTTP-Flooding datasets have been used to establish the effectiveness of INFS-MICC.

Table 6.1 depicts the symbols and notations used to describe the proposed method.

6.2 Problem Definition

Let's assume a dataset D containing s samples. D is partitioned into D_{old} and D_{new}. D_{old} is the data that has already arrived at time T_m and D_{new} is the incremental batch of data newly arrived at time T_n. Dataset D is characterized by the feature set $F = \{f_1, f_2, f_3, ...f_d\}$, where d is the dimension of the dataset. Next task is to find, $F'_{D_{old}}$ which is the feature subset containing relevant (high feature-class mutual information) and irredundant (low feature-feature correlation) features selected from D_{old}. Now, for the given data increment D_{new}, the problem is to identify the optimal subset of features, F' for the whole dataset i.e. $D_{old} \cup D_{new}$ i.e D which ensures the best possible accuracy for D at low cost.

6.3 Background

This section presents the background of our method. It exploits the power of mutual information and correlation measures to design the proposed feature selection method.

6.3.1 Mutual Information for Feature Selection

Mutual Information is important for feature selection since it helps determine how pertinently a specific feature (or characteristic) is related to the target class. In other words, it enables the assessment of a feature's predictive value for a given class. Because it will know more about the target, a feature, such as f_i, that has a higher mutual information score with the target class than another feature, say f_j will be more valuable in predicting the target class.
Let's suppose that there are two random variables named M and N. *Mutual Information* is the amount of information that M knows about N. This can be expressed mathematically as in Equation 6.1, where m and n are the marginal probability distributions for M and N respectively.

$$I(M;N) = \sum_{m,n} p(m,n) \log \frac{p(m,n)}{p(m)p(n)} \tag{6.1}$$

The joint probability distribution function for the random variables M and N is actually expressed as $p(m,n)$ in Equation 5.1, while $p(m)$ and $p(n)$ denote the marginal probability distributions for M and N. It is important to note that the Mutual Information between M and N is said to be zero if they are statistically independent.

6.3.2 Correlation for Feature Selection

To determine the relationship between two features, such as f_i and f_j, in feature selection, the statistical measure of correlation is utilized. Two attributes can have a positive, negative, or even a zero correlation value depending on how they are related to one another. If they are closely linked, including just one of them in the feature set is sufficient. On the other side, having both features would make the feature set superfluous.Therefore, the goal is to select a subset of features from the original feature set with the least amount of overlap between them.
The linear relationship between two entities, let's say M and N, is described by Pearson's correlation coefficient as shown in Equation 6.2. The value of the correlation coefficient ranges from -1 to +1. High negative correlation is represented by a number of -1, and high positive correlation is represented +1. Additionally, a value of 0 denotes a lack of association between the two entities. For the proposed method, the correlation coefficient's absolute value is taken into account because, of interested is the relationship's strength not its direction of positive or negative.

$$Corr - Coeff = \frac{\sum (m_i - \bar{m})(n_i - \bar{n})}{\sqrt{\sum (m_i - \bar{n})^2 \sum (n_i - \bar{n})^2}} \tag{6.2}$$

Since, the aim is to identify an optimal subset of features which are highly relevant and irredundant. To achieve highly relevant features for a given class, mutual information is exploited whereas for irredundant feature selection pearson's correlation measure is applied.

6.4 INFS-MICC: Proposed Method

In this section, the proposed method named INFS-MICC is elaborated in length. The goal is to select an optimal subset of ranked list of features to detect HTTP-Flooding attacks with high accuracy at low cost. To provide a final ranked list of features, our method combines mutual information and correlation measurements. The main attraction of our method is its ability to handle incremental data while selecting the subset of features to ensure best possible classification accuracy. It avoids redoing the entire computation from scratch to gain new knowledge. It makes use of the previous results obtained from the original data along with new results from the added-in data. The proposed method is depicted in Figure 6.1.
The feature sets F_1^{common}, F_2^{old}, F_3^{new}, and F^D described in Figure 6.1 are explained

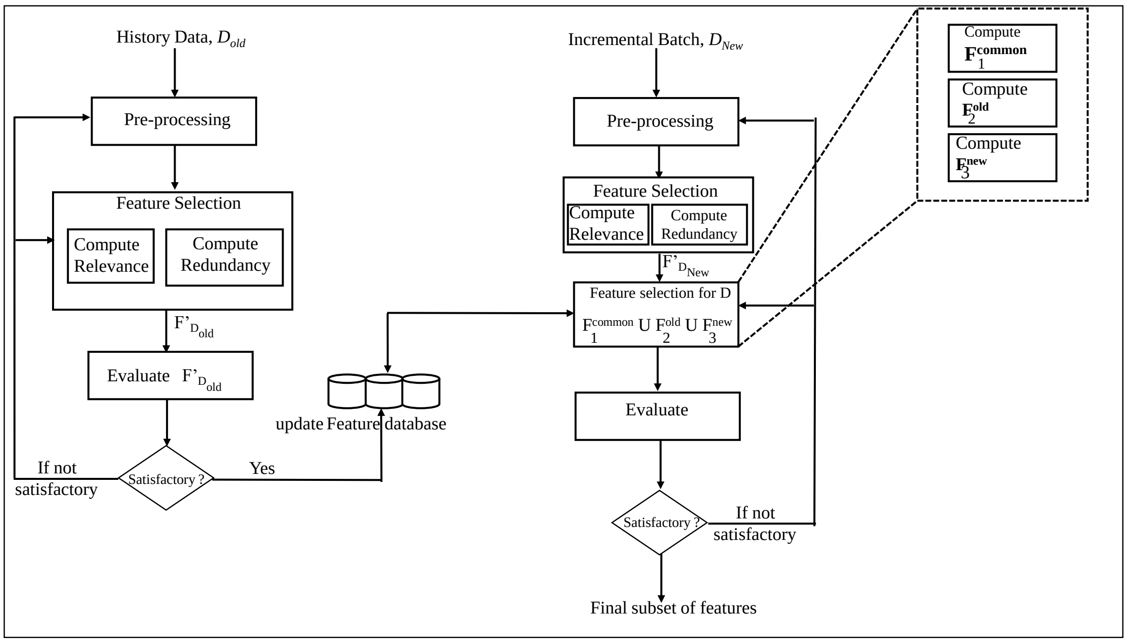


Figure 6.1: Proposed Framework for INFS-MICC

mathematically in Equation 6.3, 6.4, 6.5 and 6.6.

$$F_1^{common} = F'_{D_{new}} \cap F'_{D_{old}} \tag{6.3}$$

$$F_2^{old} = \{f_i \in F'_{D_{old}} | rank(f_i) \geq \alpha\} \tag{6.4}$$

$$F_3^{new} = \{f_j \in F'_{D_{new}} | rank(f_j) \geq \alpha\} \tag{6.5}$$

$$F^D = F_1^{common} \cup F_2^{old} \cup F_3^{new} \tag{6.6}$$

The preprocessing tasks are carried out after data is added-in or generated. Primarily, three tasks are performed. First, the missing values if any are estimated, by averaging the column values. Second, features which have zero variance are eliminated because they do not contribute in the decision-making during prediction. Third, to bring the values to a uniform range of 0 to 1, min-max normalization technique is used.

Next, the proposed feature selection method is applied which is designed to handle incremental data or added-in data. The feature selection process is mainly based on computing the relevance of the features in terms of mutual information and computing the independence among the features in terms of correlation.

6.4.1 Relevance and Independent Feature Subset Finding

INFS-MICC is an incremental feature selection method which is based on Mutual information and correlation. It assumes that data arrives in batches. For simplicity, two batches of data D_{old} (already arrived) and D_{new} (now arrives) are considered. First, it is assumed that for D_{old}, the feature subset $F^{'}_{D_{old}}$ is selected using the score described in Equation 5.4. Over time, as incremental batch of data D_{new} becomes available, the respective feature subset $F^{'}_{D_{new}}$ is also identified using the same approach. It is important here to note that, data batch D_{new} may acquire some new features over time. So, $F^{'}_{D_{new}}$ may contain some features which were not previously present in $F^{'}_{D_{old}}$ and it may or may not be relevant for D_{old}. To address this issue, the feature subset F^D is calculated. F^D is a combination of three feature subsets namely F_1^{common}, F_2^{old} and F_2^{new} as shown in Equation 6.6. F_1^{common} contains the common features from $F^{'}_{D_{new}}$ and $F^{'}_{D_{old}}$ as shown in Equation 6.3. F_2^{old} contains the highly relevant (i.e. features with $rank \geq \alpha$, a user defined threshold) features from $F^{'}_{D_{old}}$ as described in Equation 6.4. Similarly, F_3^{new} contains the highly relevant features from $F^{'}_{D_{new}}$ as shown in Equation 6.5. The feature set F^D is now used to evaluate the new batch data, D_{new}. If performance is found to be satisfactory, then a complete scan and evaluation of D_{old} can be avoided. Following definitions provide the theoretical basis of the proposed method.

Definition 6.1. (Feature Relevance) For any feature f_i, its relevance is defined in terms of mutual information to the target class C_k. Higher the mutual information score for f_i, higher is its relevance to C_k.

Definition 6.2. (Highly Relevant) A feature f_i is said to be highly relevant iff any of the following two cases holds.
Case 1: $f_i \in F^{'}_{D_{new}}$, and relevance(f_i,C_k) $> \alpha$ in D_{new} and $f_i \notin F^{'}_{D_{old}}$,
where $0 < \alpha < 1$, a user defined threshold.
Case 2: $f_i \in F^{'}_{D_{old}}$, and relevance(f_i,C_k) $> \alpha$ in D_{old} and $f_i \notin F^{'}_{D_{new}}$,
where $0 < \alpha < 1$, is a user defined threshold.

Definition 6.3. (Independence of a Feature) The independence of a feature f_i is defined in terms of average correlation with respect to all other features in feature set F. Feature f_i has high independence if its average correlation score with all other features is low.

Proposition 6.1. A feature $f_i \in F'$ is relevant and irredundant if and only if $f_i \in F'_{D_{new}}$ or $f_i \in F'_{D_{old}}$.

Proof. Suppose $f_i \in F'$, however, $f_i \notin F'_{D_{new}}$ or $f_i \notin F'_{D_{old}}$. A feature f_i is included in F' only when it is highly relevant (Definition 6.2) and independent of other features. Now, a feature f_i can be highly relevant for D ($D_{new} \cup D_{old}$) only when -

- $f_i \in F'_{D_{new}}$ and $f_i \in F'_{D_{old}}$, or,
- f_i is highly relevant for either D_{new} or D_{old}.

Hence, f_i must be relevant and irredundant. ∎

Proposition 6.2. The feature subset selected by INFS-MICC is optimal.

Proof. Assume that feature subset F^D selected by INFS-MICC is not optimal. In other words, there is possibility of inclusion or exclusion of feature(s) in F^D. However, a feature f_i is included in F^D only when any of the following condition is true:
Condition 1: $f_i \in F'_{D_{old}}$ and $f_i \in F'_{D_{New}}$
Condition 2: f_i is assigned higher rank for either D_{new} or D_{old}.
Further, the feature subset created considering the conditions 1 and 2 undergoes a recursive feature elimination process to find the optimal subset of selected features so that any exclusion or inclusion of feature(s) leads to deterioration of performance. Hence, the assumption does not hold and hence the proof. ∎

6.5 Experimental Results

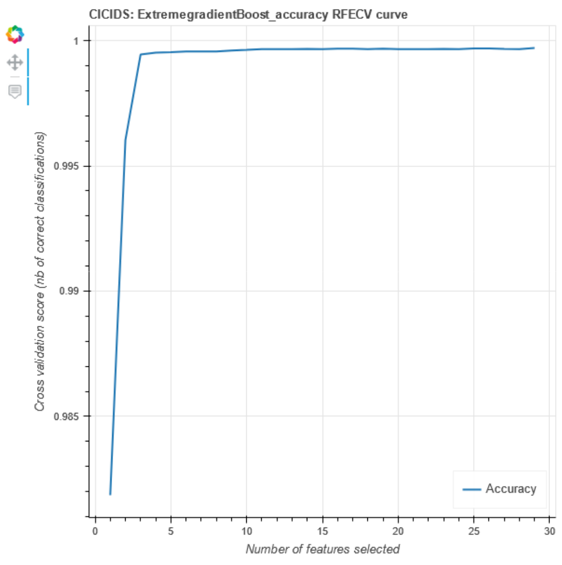
This section presents the experimental results. For evaluation purposes, five different ensemble learners namely, Adaboost, Gradient Boosting, Extreme Gradient Boosting, Random Forest and Extra Trees are used. For each of these learners, the results are evaluated in terms of both Accuracy and F1-score. Along with these two measures, the number of optimal features are also presented. For this, Recursive Feature Elimination (RFE) is used in a cross validation setting. In Table 6.2 the datasets used for evaluating the proposed method is presented.

[3]https://www.kaggle.com/datasets/jacobvs/ddos-attack-network-logs?resource=download

Table 6.2: Datasets Used

Dataset name	No. of instances	No. of features	No. of classes
HTTP Flood [3]	21,60,668	28	2
UNSW[137]	25,40,044	49	2
CICIDS[241]	28,30,540	80	2

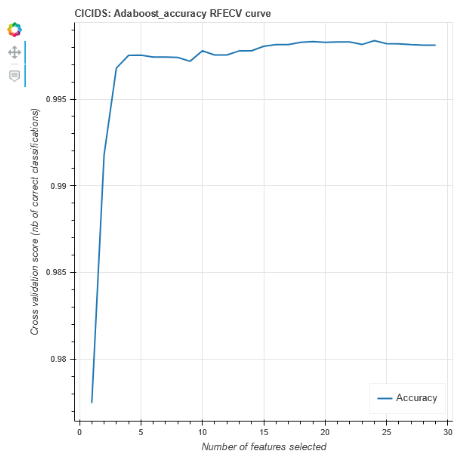
In case of the CICIDS dataset, the highest accuracy of 99.8% is obtained by Extreme Gradient Boosting classifier with 3 features as shown in Figure 6.2. All the other classifiers i.e AdaBoost, Gradient Boosting, Random Forest and Extra Trees show similar performance in terms of accuracy as shown in Figures 6.3, 6.4, 6.5 and 6.6 respectively. However, to achieve that performance the classifiers require 4 (for Adaboost), 4 (for Gradient Boosting) and 10 (for both Random forest and Extra Trees) which is higher than the number of features required by Extreme Gradient Boosting classifier. Hence, in this case, it is concluded that the optimal performance is given by Extreme Gradient Boosting classifier with 3 features. For the same dataset, F1-scores are illustrated in Figure 6.7 for Gradient Boosting, Figure 6.8 for Adaboost, Figure 6.9 for XGBoost, Figure 6.10 for Random Forest, and Figure 6.11 for Extra Trees classifier.



Highest accuracy: 99.8%
No. of selected features: 3

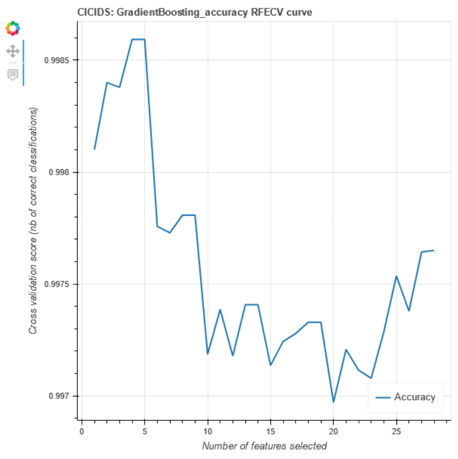
Figure 6.2: RFE with XGBoost Classifier for CICIDS Dataset (Accuracy)

In case of the UNSW dataset, the highest accuracy of 99.7% is obtained by Gradient Boosting classifier with 5 features as shown in Figure 6.12. All the other classifiers i.e AdaBoost, Extreme Gradient Boosting, Random Forest and Extra



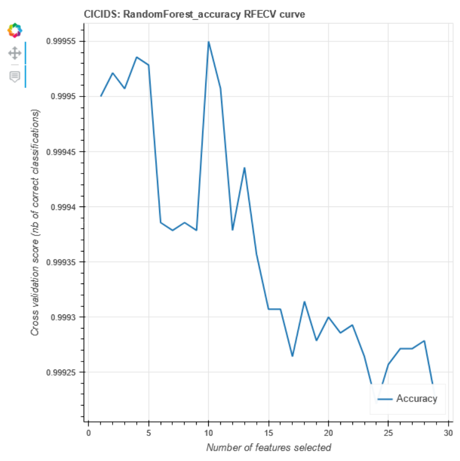
Highest accuracy: 99.8%
No. of selected features: 4

Figure 6.3: RFE with Adaboost Classifier for CICIDS Dataset (Accuracy)



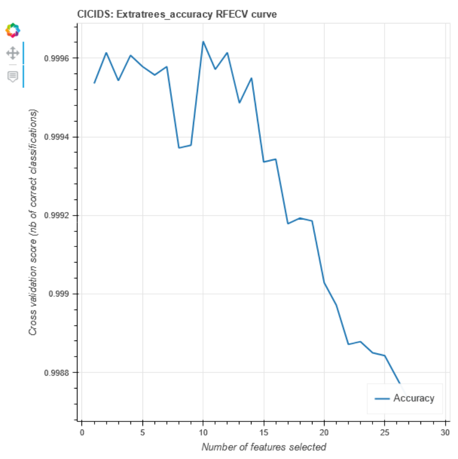
Highest accuracy: 99.8%
No. of selected features: 4

Figure 6.4: RFE with Gradient Boosting Classifier for CICIDS Dataset (Accuracy)



Highest accuracy: 99.9%
No. of selected features: 10

Figure 6.5: RFE with Random Forest Classifier for CICIDS Dataset (Accuracy)

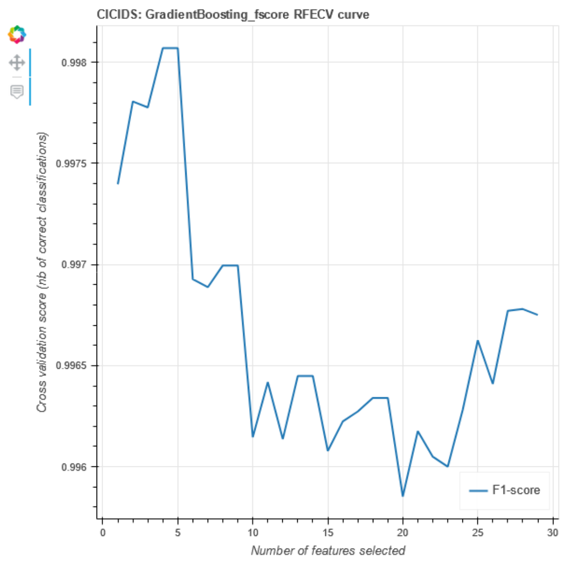


Highest accuracy: 99.9%
No. of selected features: 10

Figure 6.6: RFE with Extra Trees Classifier for CICIDS Dataset (Accuracy)

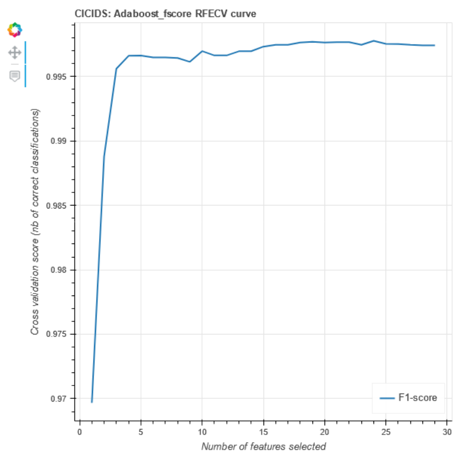
Trees show similar performance in terms of accuracy as shown in Figures 6.13, 6.14, 6.15 and 6.16 respectively. However, to achieve that performance the classifiers require 7 (for Adaboost), 13 (for both Extreme Gradient Boosting and Random forest) and 8 (for Extra Trees) which is higher than the number of features required by Gradient Boosting. Hence, in this case, it is concluded that the optimal performance is given by Gradient Boosting classifier with 5 features. For the same dataset, F1-scores are illustrated in Figure 6.17 for Gradient Boosting, Figure 6.18 for Adaboost, Figure 6.19 for XGBoost, Figure 6.20 for Random Forest, and Figure 6.21 for Extra Trees classifier.

On the other hand, for the HTTP Flood dataset, Extreme Gradient Boosting



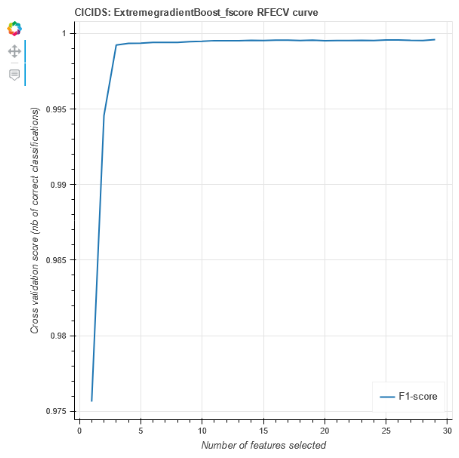
Highest f1-score: 99.8%
No. of selected features: 4

Figure 6.7: RFE with Gradient Boosting Classifier for CICIDS Dataset (F1-score)



Highest f1-score: 99.7%
No. of selected features: 4

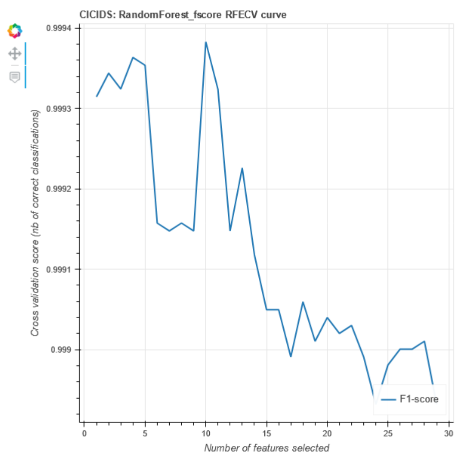
Figure 6.8: RFE with Adaboost Classifier for CICIDS Dataset (F1-score)



Highest f1-score: 99.8%
No. of selected features: 3

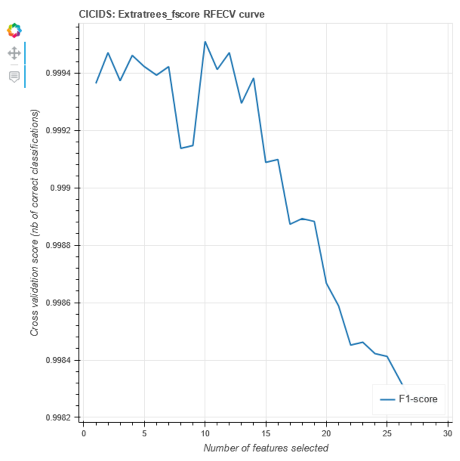
Figure 6.9: RFE with XGBoost Classifier for CICIDS Dataset (F1-score)

achieves the highest accuracy with 99.9% with 2 features only as shown in Figure 6.22. All other learners give similar performance with highest accuracy 99.9% as shown in Figures 6.23 (for Adaboost), 6.24 (for Gradient Boosting), 6.25 (for Random Forest), 6.26 (for Extra Trees). However, to achieve that performance the learners i.e. Adaboost, Gradient Boosting, Random Forest and Extra Trees require 3, 4, 3 and 4 features respectively which is higher than the number of features required by Extreme Gradient Boosting classifier. Hence, in this case, it is concluded that the optimal performance is given by Extreme Gradient Boosting



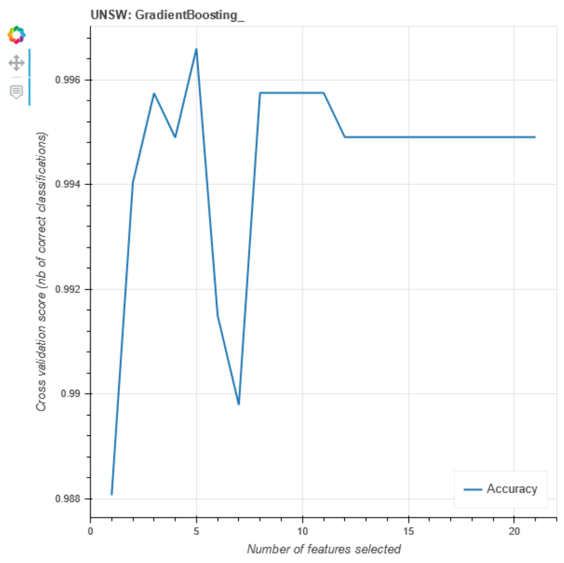
Highest accuracy: 99.9%
No. of selected features: 10

Figure 6.10: RFE with Random Forest Classifier for CICIDS Dataset (F1-score)



Highest f1-score: 99.9%
No. of selected features: 10

Figure 6.11: RFE with Extra Trees Classifier for CICIDS Dataset (F1-score)

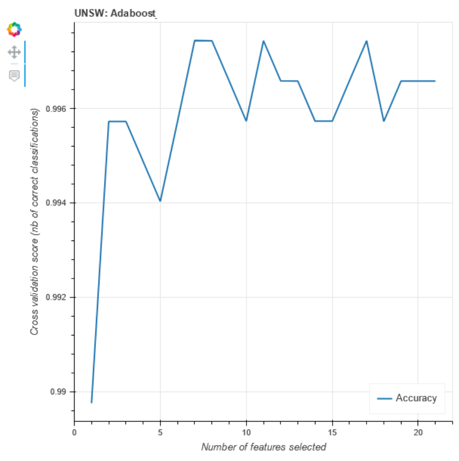


Highest accuracy: 99.7%
No. of selected features: 5

Figure 6.12: RFE with Gradient Boosting Classifier for UNSW Dataset (Accuracy)

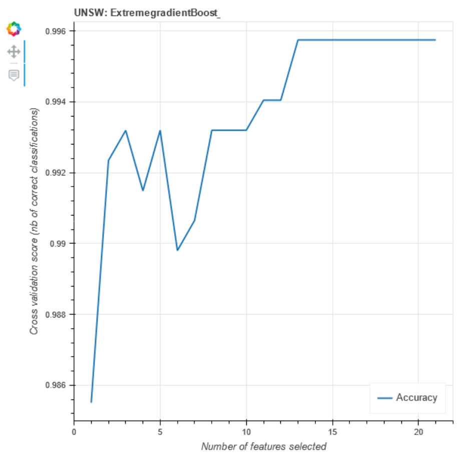
classifier with 2 features. For the same dataset, F1-scores are illustrated in Figure 6.27 for Gradient Boosting, Figure 6.28 for Adaboost, Figure 6.29 for XGBoost, Figure 6.30 for Random Forest, and Figure 6.31 for Extra Trees classifier.

Table 6.3 shows the top 10 ranked features for each of the HTTP Flooding dataset considered as given by the proposed incremental feature selection method. The columns *feature name* and *index number* gives the name of the feature and the index number in the dataset.



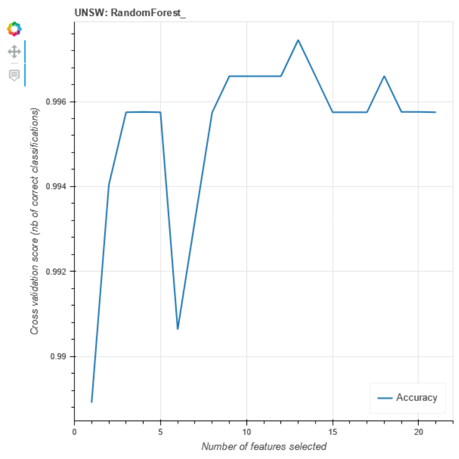
Highest accuracy: 99.7%
No. of selected features: 7

Figure 6.13: RFE with Adaboost Classifier for UNSW Dataset (Accuracy)



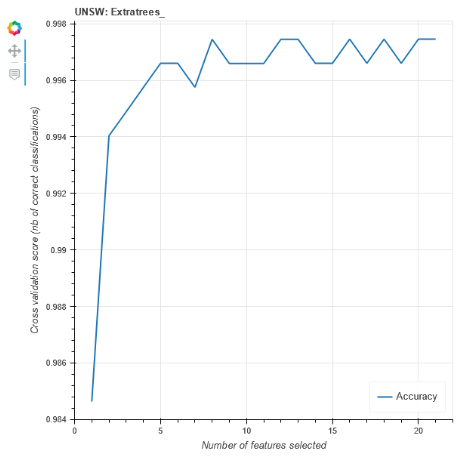
Highest accuracy: 99.6%
No. of selected features: 13

Figure 6.14: RFE with XGBoost Classifier for UNSW Dataset (Accuracy)



Highest accuracy: 99.7%
No. of selected features: 13

Figure 6.15: RFE with Random Forest Classifier for UNSW Dataset (Accuracy)

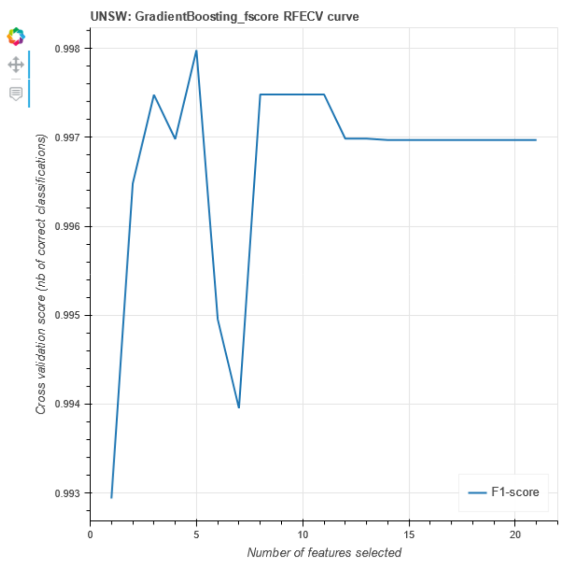


Highest accuracy: 99.7%
No. of selected features: 8

Figure 6.16: RFE with Extra Trees Classifier for UNSW Dataset (Accuracy)

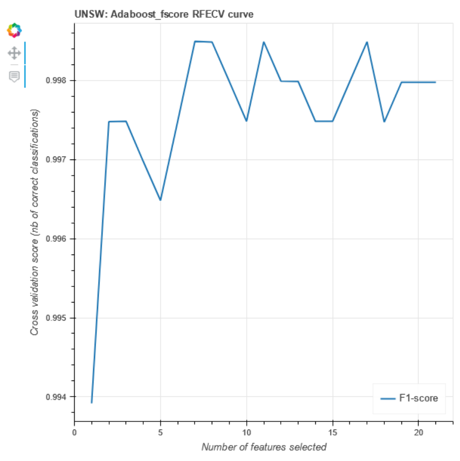
6.5.1 Comparison with other Feature Selection Methods

The proposed incremental feature selection method which is designed to detect HTTP Flood attacks in the application layer is compared against seven state-of-the-art feature selection methods such as MIFS [193], CMIM [200], and mRMR [228], DISR [242], JMI, Gini index and ANOVA feature selection. Figure 6.32, 6.33 and 6.34 illustrates the comparative analysis of INFS-MICC against its counter parts. For comparative analysis, F1-score is used as an evaluation metric as it is a better measure than accuracy in case of unbalanced datasets. For the CICIDS dataset,



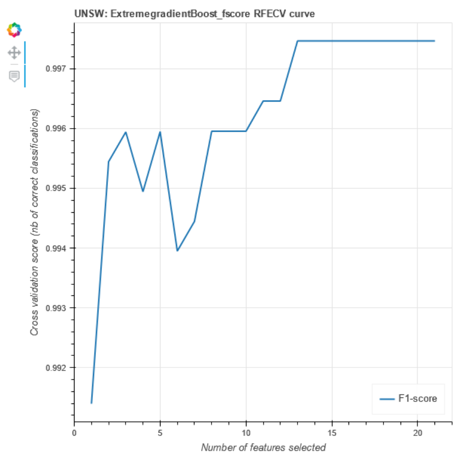
Highest f1-score: 99.8%
No. of selected features: 5

Figure 6.17: RFE with Gradient Boosting Classifier for UNSW Dataset (F1-score)



Highest f1-score: 99.8%
No. of selected features: 7

Figure 6.18: RFE with Adaboost Classifier for UNSW Dataset (F1-score)



Highest f1-score: 99.7%
No. of selected features: 13

Figure 6.19: RFE with XGBoost Classifier for UNSW Dataset (F1-score)

since the proposed method obtained highest accuracy with 3 features, the F1-score comparison is also done considering 3 features only for each feature selection method. Similarly, for UNSW and HTTP-Flood dataset highest accuracies are obtained with 5 and 2 features respectively. Hence, comparison for these two datasets are done considering the said number of features for each feature selection method. From the comparative analysis, it is seen that the proposed method gives on par or better performance compared to the other feature selection methods.

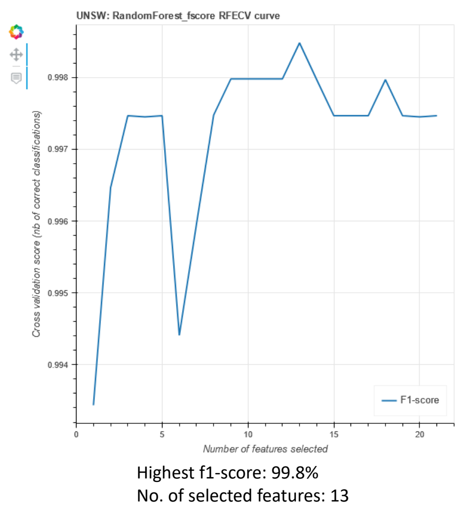


Figure 6.20: RFE with Random Forest Classifier for UNSW Dataset (F1-score)

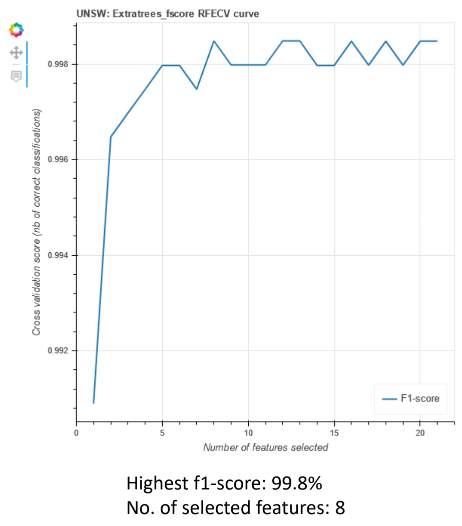


Figure 6.21: RFE with Extra Trees Classifier for UNSW Dataset (F1-score)

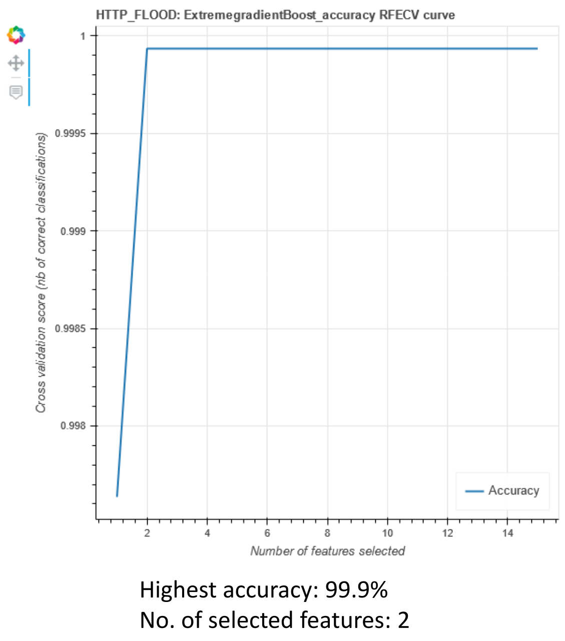


Figure 6.22: RFE with XGBoost Classifier for HTTP Flood Dataset (Accuracy)

6.5.2 Discussion

In this chapter, an incremental feature selection method called INFS-MICC is proposed to detect HTTP-Flooding attacks. The proposed method aids to identify a final ranked list of feature subset which consists of highly relevant and independent features. For computing the relevance, feature-class mutual information is considered and for the independence among features, the feature-feature correlation is computed. The main highlight of our method is that it is incremental in nature,

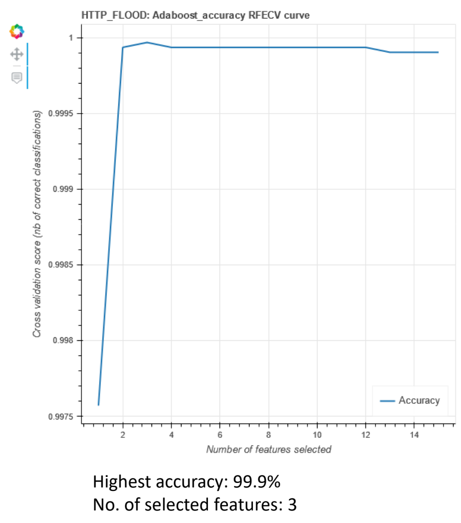


Figure 6.23: RFE with Adaboost Classifier for HTTP Flood Dataset (Accuracy)

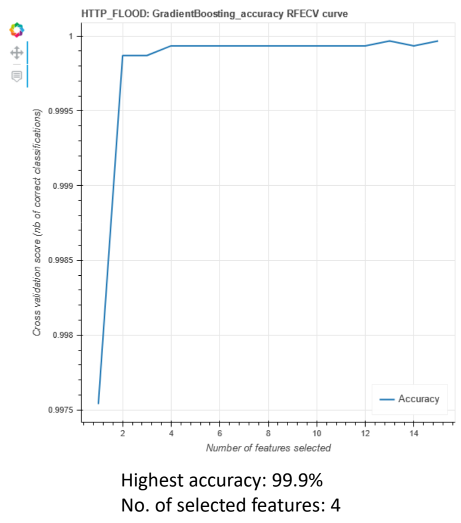


Figure 6.24: RFE with Gradient Boosting Classifier for HTTP Flood Dataset (Accuracy)

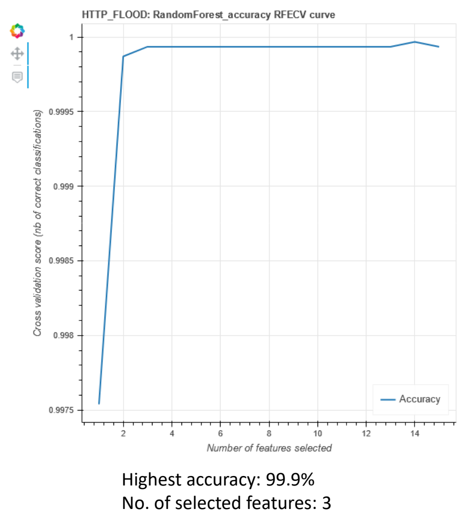


Figure 6.25: RFE with Random Forest Classifier for HTTP Flood Dataset (Accuracy)

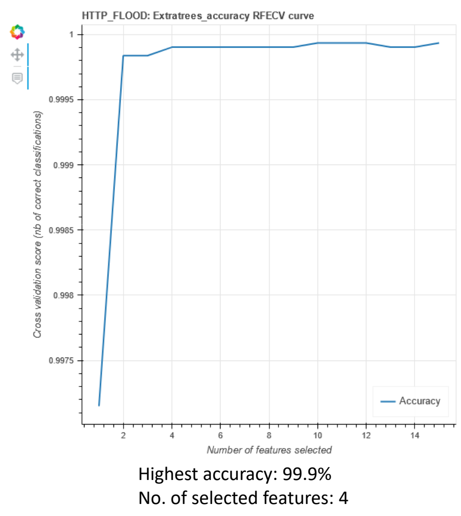
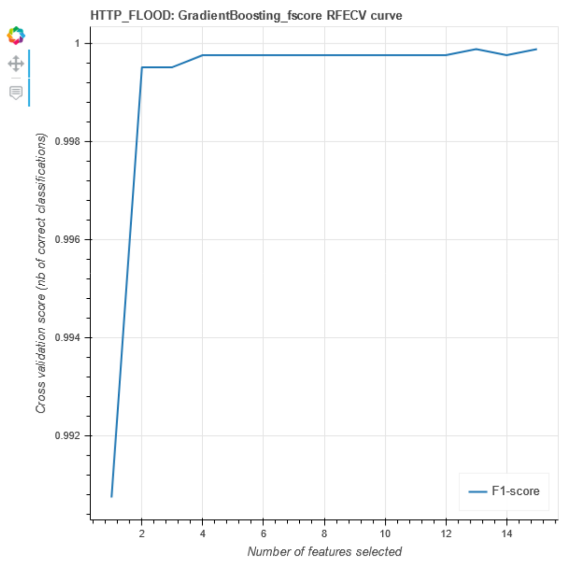


Figure 6.26: RFE with Extra Trees Classifier for HTTP Flood Dataset (Accuracy)

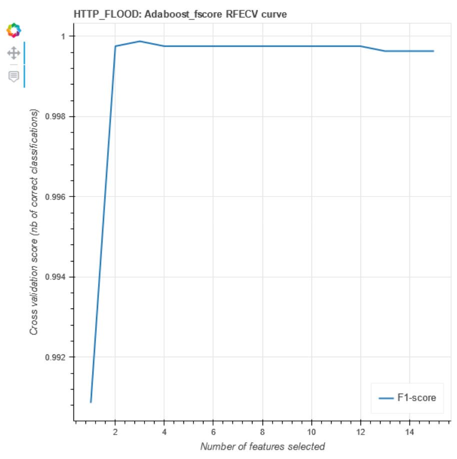
as it can handle added-in data which avoids re-computation of the whole dataset. The proposed method is evaluated with three HTTP-Flooding datasets and five ensemble predictors using two evaluation metrics namely Accuracy and F1-score. For two out of the three datasets Extreme Gradient Boosting gives optimal performance whereas for one of the dataset Gradient Boosting classifier performs the best.

The next chapter introduces an ensemble feature selection method named FSRA for the detection of attacks in Critical Infrastructure. It uses three base feature selectors and aggregates their individual ranks using a proposed score.



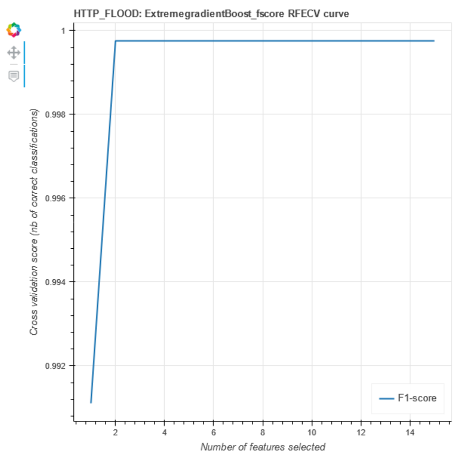
Highest f1-score: 99.9%
No. of selected features: 4

Figure 6.27: RFE with Gradient Boosting Classifier for HTTP Flood Dataset (F1-score)



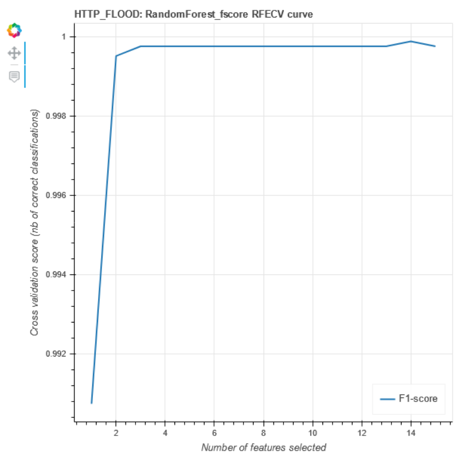
Highest f1-score: 99.9%
No. of selected features: 3

Figure 6.28: RFE with Adaboost Classifier for HTTP Flood Dataset (F1-score)



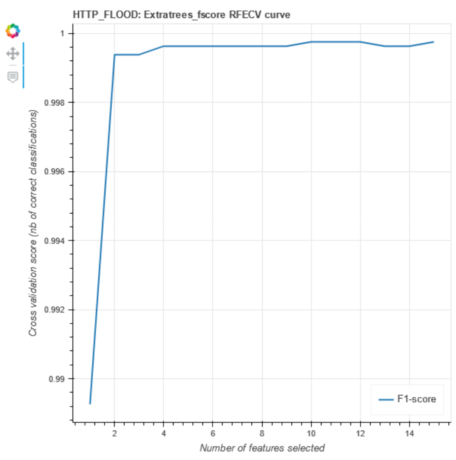
Highest f1-score: 99.9%
No. of selected features: 2

Figure 6.29: RFE with XGBoost Classifier for HTTP Flood Dataset (F1-score)



Highest f1-score: 99.9%
No. of selected features: 3

Figure 6.30: RFE with Random Forest Classifier for HTTP Flood Dataset (F1-score)



Highest f1-score: 99.9%
No. of selected features: 4

Figure 6.31: RFE with Extra Trees Classifier for HTTP Flood Dataset (F1-score)

Table 6.3: Top 10 Ranked Features of the HTTP Flooding Datasets

Top 10 features (ranked) of HTTP Flooding Datasets						
	UNSW		CICIDS		HTTP-FLOOD	
	Feature Name	Index Number	Feature Name	Index Number	Feature Name	Index Number
1	dport	3	Flow Bytes/s	14	SEQ_NUMBER	8
2	daddr	2	Bwd Packets/s	37	NODE_NAME_FROM	11
3	drate	15	Init_Win_bytes_backward	67	FIRST_PKT_SENT	24
4	AR_P_Proto_P_SrcIP	22	Destination Port	0	BYTE_RATE	18
5	N_IN_Conn_P_SrcIP	25	Init_Win_bytes_forward	66	LAST_PKT_RESEVED	25
6	srate	14	Packet Length Mean	40	PKT_DELAY	21
7	dur	8	Average Packet Size	52	FID	7
8	TnP_Per_Dport	21	Max Packet Length	39	NODE_NAME_TO	12
9	bytes	5	Avg Bwd Segment Size	54	NUMBER_OF_BYTE	10
10	TnBPSrcIP	16	Bwd Packet Length Mean	12	UTILIZATION	20

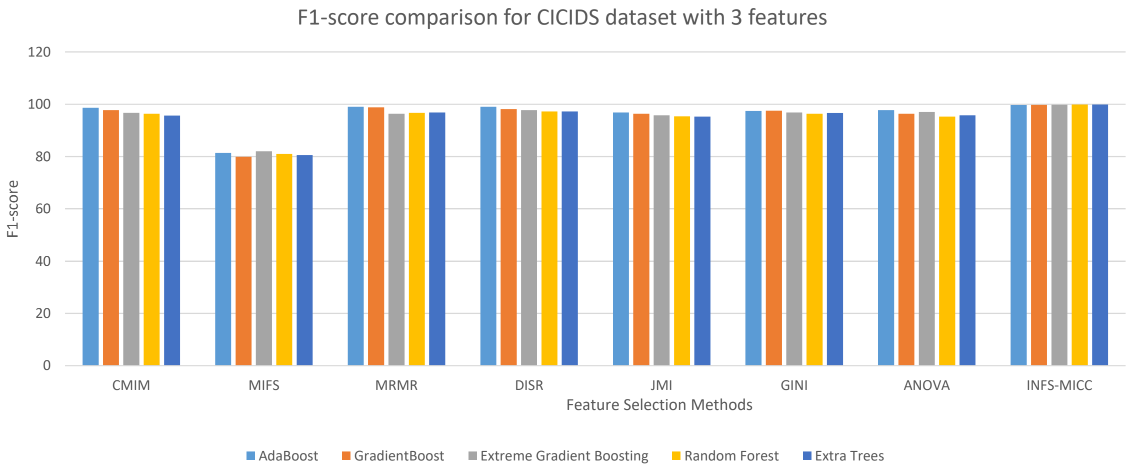


Figure 6.32: F1-score Comparison for CICIDS Dataset

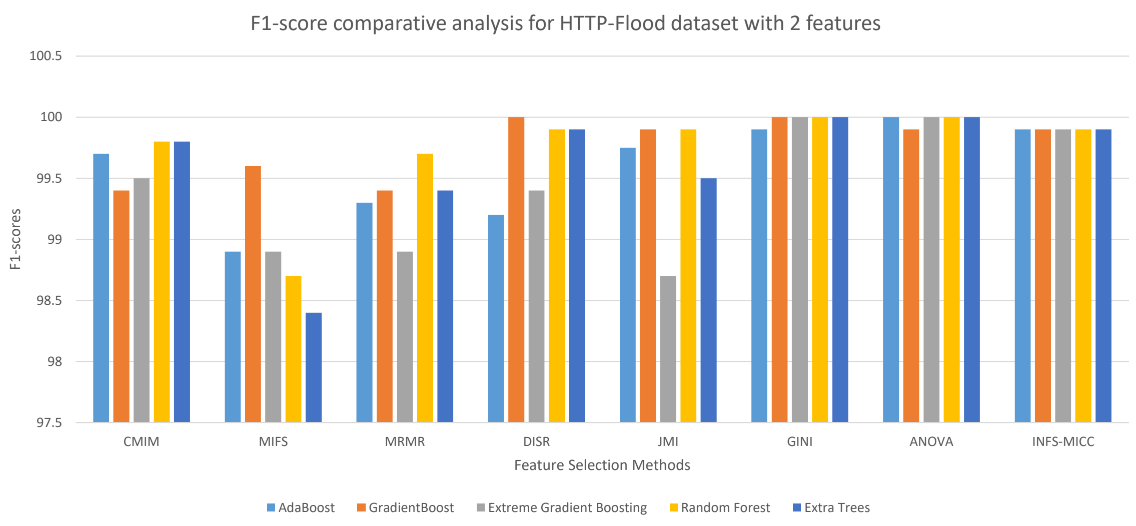


Figure 6.33: F1-score Comparison for HTTP-Flood Dataset

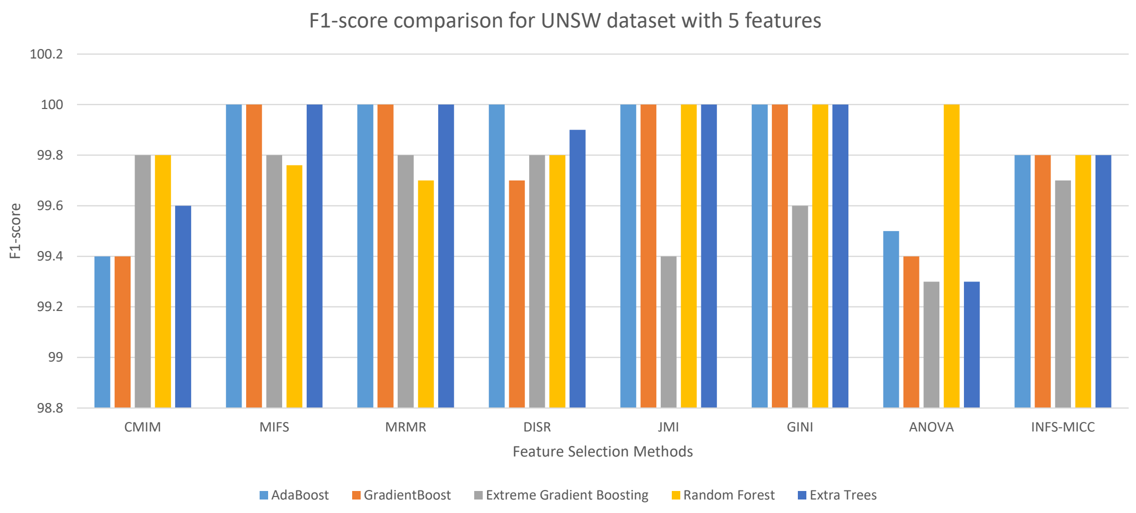


Figure 6.34: F1-score Comparison for UNSW Dataset