STREAMLINING HTTP FLOODING ATTACK DETECTION THROUGH INCREMENTAL FEATURE SELECTION

Upasana Sarmah **Parthajit Borah**
D. K. Bhattacharyya
Department of Computer Science & Engineering, Tezpur University, Tezpur, Assam
{upatink, parthajit, dkb}@tezu.ernet.in

ABSTRACT

Applications over the Web primarily rely on the HTTP protocol to transmit web pages to and from systems. There are a variety of application layer protocols, but among all, HTTP is the most targeted because of its versatility and ease of integration with online services. The attackers leverage the fact that by default no detection system blocks any HTTP traffic. Thus, by exploiting such characteristics of the protocol, attacks are launched against web applications. HTTP flooding attacks are one such attack in the application layer of the OSI model. In this paper, a method for the detection of such an attack is proposed. The heart of the detection method is an incremental feature subset selection method based on mutual information and correlation. INFS-MICC helps in identifying a subset of highly relevant and independent feature subset so as to detect HTTP Flooding attacks with best possible classification performance in near-real time.

***K*eywords** HTTP Flooding · Incremental · Feature · Feature Selection.

1 Introduction

HTTP flooding attacks on the application layer of the OSI model are a kind of DDoS attack where the services offered by a Web server are brought down by an attacker. These kinds of attacks consume less bandwidth as the attacker floods the server with legitimate HTTP requests. After a certain point of time upon flooding, the server is overwhelmed and cannot respond to legitimate requests of its user base. The attacker customizes the requests according to the target web application and the ultimate aim is to exhaust the server's limited resources. HTTP protocol is the most targeted protocol due to its versatility and ease of use with a variety of Web applications [1]. It is important to note here, that an attacker may employ a botnet army to launch a coordinated attack to bring down the services.
Compared to others, there is far less difference between a legitimate HTTP request and an attack request, which is why these attacks are hard to detect [2]. This is because the underlying Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) connections for both an attack request and a normal HTTP request are the same. The only difference between the two is the intent with which a request is made. Additionally, as seen from real instances of application layer DDoS attacks[1] it is difficult to perceive the starting and ending point of an attack. Depending on the scale of the attack, the disruptions caused by the attack often live longer than the attack itself, and this is the reason why it is hard to estimate the actual length of an attack.
The selection of appropriate characteristics or features plays an important role in identifying actions that can lead to an attack. All features may not be equally informative; some may be redundant or irrelevant and thus play no role in the detection process [3]. In fact, a subset of highly informative features should be selected for good performance. On the other hand, in the case of a dynamic real-world application all the data may not be available at one time. Such cases necessitate the use of incremental learning, so as to avoid learning from scratch each time data is available [4]. Thus, a defense mechanism that incrementally selects relevant and irredundant features will definitely aid in solving the problem of HTTP flooding attacks in the application layer.

[1] https://www.enisa.europa.eu/publications/enisa-threat-landscape-for-dos-attacks

1.1 Attack Statistics and Defense Systems in Applications

The Cisco Annual Report[2] outlined how DDoS attacks have increased over the years (2018-2023). Figure 1 shows the number of DDoS attacks during the same period. In 2023 alone, a total of 15.4 million DDoS attacks were recorded. According to Kaspersky's press release[3], the gaming and gambling industry faced the highest number of DDoS threats

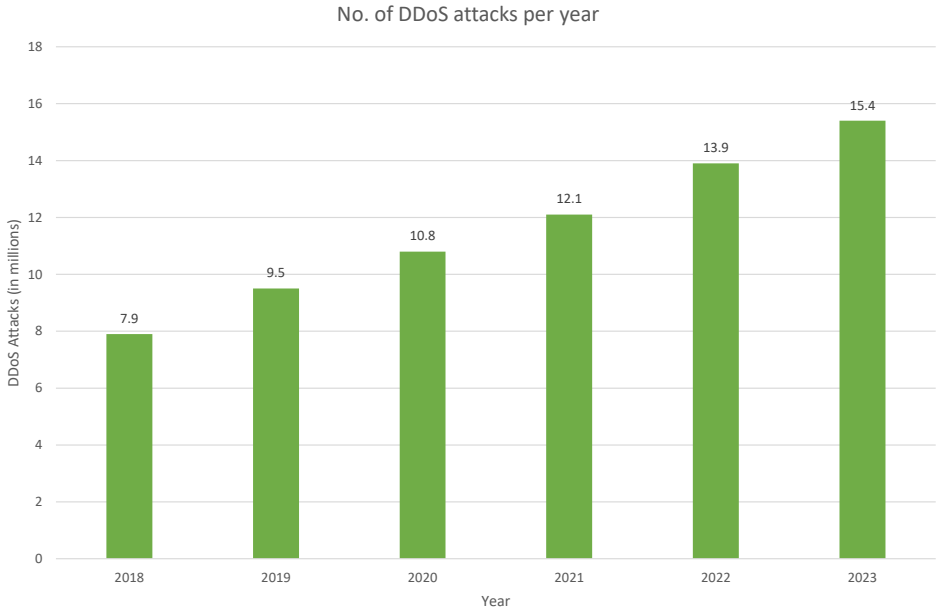


Figure 1: DDoS Attacks Over the Years

between 2022-2023. This is because nearly 40% of the world's population forms the gaming community, which is why it becomes an attractive target for the attackers. To make things more interesting, it is found that Fridays recorded the busiest day of attacks with approximately 15.36%, while the lowest are recorded on Thursdays with 12.99% [4].

From a security perspective, the last few years have been very eventful with several high-profile data breaches, threats, and attacks against web applications[5]. Web application attacks are possible due to existing vulnerabilities that put both end users and businesses at risk[5]. An effective defense approach is essential to detect and secure a web application in a timely way. Typically, a defense system may consist of four modules- i) Detection module which tries to analyze data for specific attack occurrence, ii) A prevention module which tries to prevent to attack from occurring (may be at source end or the victim end), iii) A mitigation module which tries to employ mitigation techniques (such as blocking some IPs) after attack occurrence, and iv) A tolerance module which tries to provide services to the legitimate users even when an attack is occurring. In addition, a defense approach may follow a centralized approach or even a distributed approach. When developing a defense approach for the detection of attacks in web applications, the following important points should be noted.

1. Most web applications are in themselves heavily complicated because of the technologies used, hence the design of the defense approach should be such that it does not further add to the application's complexity.
2. The operations of legitimate users should not be harmed by the deployment of a defense approach.
3. A defense approach should be scalable and robust.

1.2 Motivation

In terms of sample sizes and dimensions, the amount of data that is currently available has increased significantly. 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 [6]. At

[2] https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[3] https://www.kaspersky.com/about/press-releases/2023_kaspersky-reports-growth-in-gamer-cyberattacks-in-2023

[4] https://www.getastra.com/blog/security-audit/ddos-attack-statistics/

[5] https://www.enisa.europa.eu/publications/enisa-threat-landscape-2023

Table 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

the same time, 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[7]. 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[8]. It aids a learning model in simplifying the learning process so that it can acquire essential and vital knowledge for prediction tasks. All the mentioned reasons have collectively motivated for the development of an incremental feature selection technique which focuses on selecting relevant and irredundant (independent) features to achieve best predictive performance. The incremental nature of the method helps to avoid processing the whole data from scratch when new data instances arrive at regular intervals.

1.3 Contribution

The primary contribution of this paper is a detection method for HTTP flooding attacks in the application layer. The detection method is based on an incremental feature selection method called INFS-MICC. It is incremental in the sense that it can handle incoming data incrementally and can identify a subset of highly relevant and irredundant feature subset without processing the whole data from scratch. This process is beneficial because, in a way it has the ability to memorize thereby saving valuable time and computational resources. Table 1 gives the symbols used to describe the proposed method. The effectiveness of the proposed method is established in terms of high accuracy for three well-known HTTP-Flooding datasets.

2 Related Work

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. [9] 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. [10] 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 [11] 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. [12] 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. [13] 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 [14] 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 [15] 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 [16] and [17]. Mohammadi et al. [18] 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 [19], 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[6] and Open Network Operating System[7] (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 [20],[21][22]. 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 [23][3]. 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 [24][25][26]. 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 [27] [28][29] [30]. 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 [31].
Two categories of wrapper feature subset selection methods that are often used in the literature are sequential selection algorithms and heuristic search algorithms [32][26][33][34][35]. To choose feature subset, Maldonado and Weber [36] suggest a sequential backward selection wrapper approach utilizing Support Vector Machines (SVMs). Using cosine similarity and SVMs, Gang and Jin [37] choose relevant and independent features. On the other hand, Hsu et al. [38] provide a hybrid feature selection method, where the filter methods assist in effectively finding the candidate features and the 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 [39] 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. [40] examine diversity metrics to assess diversity in the ensemble feature selection methods.

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.

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. A feature, say f_i, gives more information on the target if it has a higher mutual information score with the target class compared to another feature, say f_j. So, f_i is more valuable in predicting the target class.

[6] `http://mininet.org/`

[7] `https://opennetworking.org/onos/`

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 1, where *m* and *n* are the marginal probability distributions for *M* and *N* respectively.

$$I(M;N) = \sum_{m,n} p(m,n) \log \frac{p(m,n)}{p(m)p(n)} \tag{1}$$

The joint probability distribution function for the random variables *M* and *N* is actually expressed as $p(m,n)$ in Equation 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.

3.2 Correlation for Feature Selection

In feature selection, to determine the relationship between two features, such as f_i and f_j, the statistical measure of correlation is used. Two attributes can have a positive, negative, or zero correlation value depending on how they are related to each other. 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 2. The value of the correlation coefficient ranges from -1 to +1. A high negative correlation is represented by -1, and a 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 absolute value of the correlation coefficient is taken into account because the relationship's strength is of interest and not its direction of positive or negative.

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

The aim is to identify an optimal subset of features that are both highly relevant and non-redundant. Mutual information is used to ensure relevance, while Pearson's correlation measure ensures irredundancy.

4 Problem Definition

Let's assume a dataset D containing s samples. D_{old} comprises of samples that arrived at time T_m and D_{new} is the incremental batch of data newly arrived at time T_n ($T_m < T_n$). 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^D for the whole dataset i.e. $D_{old} \cup D_{new}$ i.e D, ensuring best possible accuracy and without starting the computation from scratch.

5 Proposed Work

The proposed method, INFS-MICC selects a ranked optimal subset of features to detect HTTP-Flooding attacks. 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 re-doing 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 2.
The feature sets $F_1^{common}, F_2^{old}, F_3^{new}$, and F^D described in Figure 2 are explained mathematically in Equation 3, 4, 5 and 6.

$$F_1^{common} = F'_{D_{new}} \cap F'_{D_{old}} \tag{3}$$

$$F_2^{old} = \{f_i \in F'_{D_{old}} | rank(f_i) \geq \alpha\} \tag{4}$$

$$F_3^{new} = \{f_j \in F'_{D_{new}} | rank(f_j) \geq \alpha\} \tag{5}$$

$$F^D = F_1^{common} \cup F_2^{old} \cup F_3^{new} \tag{6}$$

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

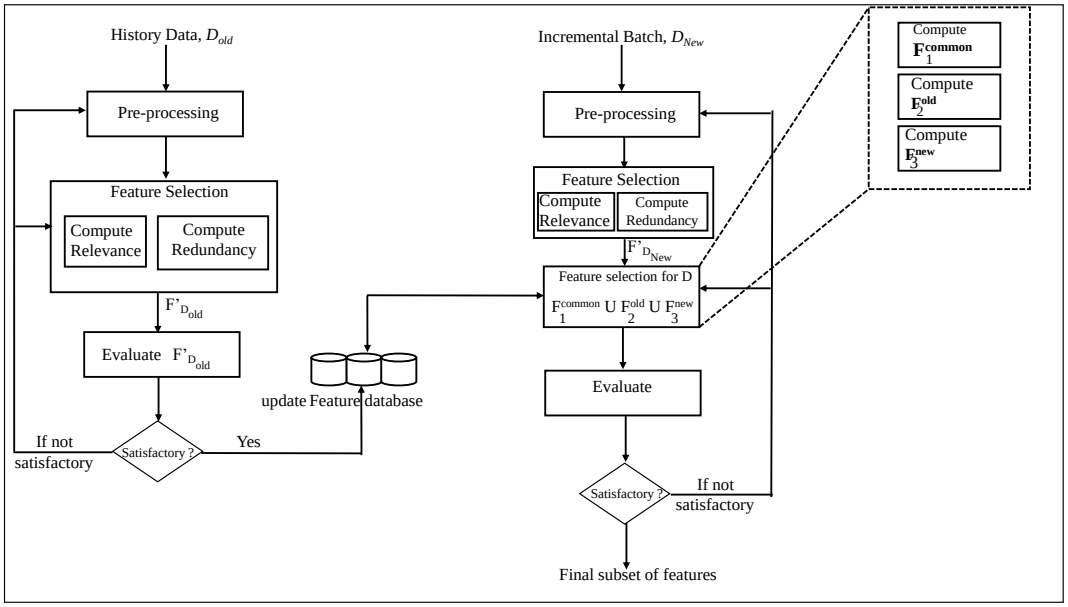


Figure 2: Proposed Framework for INFS-MICC

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 redundacny among the features in terms of correlation.

5.1 Specifics of INFS-MICC

To assess the strength of the relationship between features correlation is used, specifically Pearson's correlation coefficient. Higher strength indicates that the features are highly correlated. The proposed method uses Pearson's correlation coefficient because of three main reasons: $i)$ It measures the relationship between two continuous variables (raw values of the variables), unlike other correlation measures such as Spearman Rank Correlation which takes into consideration the ranks of the data or Kendall's Tau correlation measure which considers the ordinal association between two variables [41], $ii)$ It is a widely accepted standard measure and hence is not influenced by the scales of the continuous valued features [42], $iii)$ Pearson's correlation coefficient is simple and fast in terms of computational complexity ($\mathcal{O}(n)$) compared to Spearman's coefficient ($\mathcal{O}(n \log n)$) [43]. Additionally, as already mentioned correlation measures (in this case Pearson's correlation measure) are sensitive to outliers. This drawback is overcome with a two fold solution. First, mutual information (as it is insensitive to outliers [44]) is introduced to the proposed score. Second, to negate out the outlier effects of a feature say f_j when calculating the correlation with feature f_i, the average correlation of f_i is introduced and subtraction of the two entities is performed as shown in Equation 7. This ensures that the outlier values of f_j will not influence the values of feature f_i.

5.2 Relevance and Independent Feature Subset Finding

INFS-MICC is an incremental feature selection method, which 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 7.

$$MICC\text{-}UD(f_i) = \frac{Relevance\_score(f_i, C)}{max_{i \neq j}(|avg\_Corr|(f_i) - Corr(f_i, f_j))} \tag{7}$$

The relevance score and average correlation (avg_Corr) mentioned in Equation 7 is calculated as shown below in Equation 8 and Equation 9.

$$Relevance\_score(f_i, C) = MutualInformation(f_i, C) \tag{8}$$

$$avg\_Corr(f_i) = \frac{\sum_{\substack{j=1 \\ i \neq j}}^{d} (Corr(f_i, f_j))}{d} \tag{9}$$

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. F_1^{common} contains the common features from $F'_{D_{new}}$ and $F'_{D_{old}}$ as shown in Equation 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 4. Similarly, F_3^{new} contains the highly relevant features from $F'_{D_{new}}$ as shown in Equation 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.

5.3 Optimal Feature Subset Identification using Recursive Feature Elimination

After obtaining the ranked list of features, next the optimal feature subset needs to be identified. Optimal subset of features mean adding features to this subset does not increase the classifier performance and at the same time, removing any feature from the subset deteriorates the classifier performance. Here, the optimal feature subset is obtained by recursively eliminating the features from the ranked list. The recursive feature elimination step works primarily with five predictors for making the final predictions.
Following definitions provide the theoretical basis of the proposed method.

Definition 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 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 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 1. *A feature $f_i \in F^D$ is relevant and irredundant if and only if $f_i \in F'_{D_{new}}$ or $f_i \in F'_{D_{old}}$.*

Proof. Suppose $f_i \in F^D$, however, $f_i \notin F'_{D_{new}}$ or $f_i \notin F'_{D_{old}}$. A feature f_i is included in F^D only when it is highly relevant (Definition 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 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 Experiments and 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 2 the datasets used for evaluating the proposed method is presented.

Table 2: Datasets Used

Dataset name	No. of instances	No. of features	No. of classes
HTTP Flood [8]	21,60,668	28	2
UNSW[45]	25,40,044	49	2
CICIDS[46]	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 3. 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 4, 5, 6 and 7 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 8 for Gradient Boosting, Figure 9 for Adaboost, Figure 10 for XGBoost, Figure 11 for Random Forest, and Figure 12 for Extra Trees classifier.

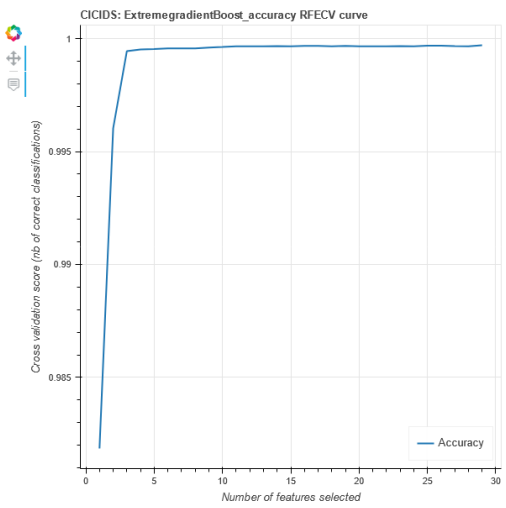


Figure 3: RFE with XGBoost Classifier for CICIDS Dataset (Accuracy)

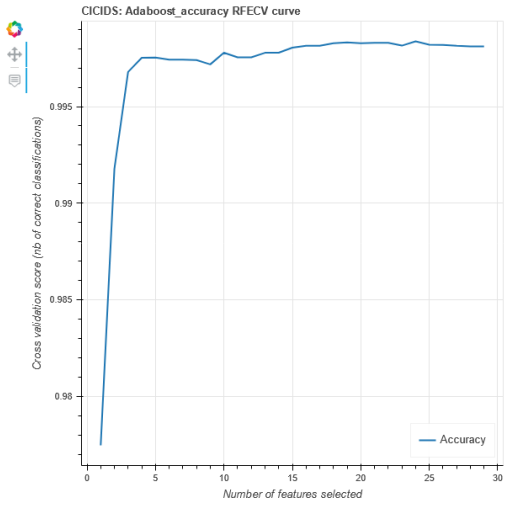
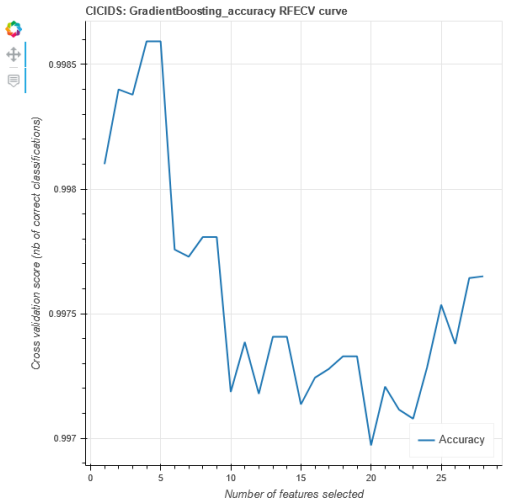


Figure 4: RFE with Adaboost Classifier for CICIDS Dataset (Accuracy)

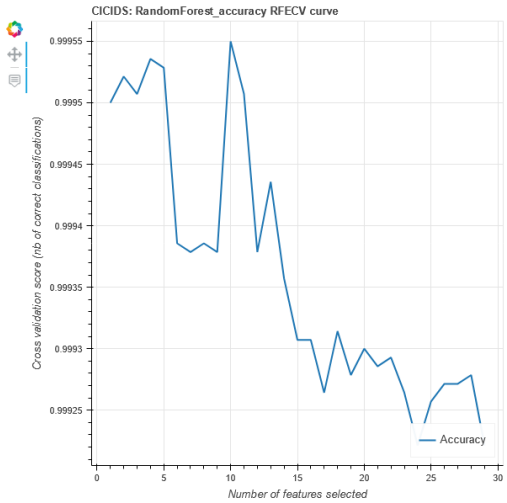
In case of the UNSW dataset, the highest accuracy of 99.7% is obtained by Gradient Boosting classifier with 5 features as shown in Figure 13. All the other classifiers i.e AdaBoost, Extreme Gradient Boosting, Random Forest and Extra Trees show similar performance in terms of accuracy as shown in Figures 14, 15, 16 and 17 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

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



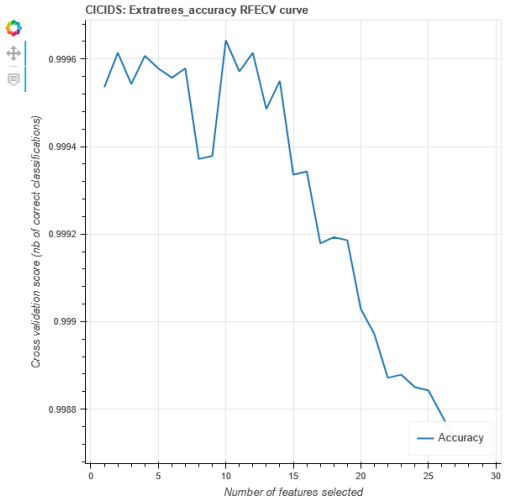
Highest accuracy: 99.8%
No. of selected features: 4

Figure 5: RFE with Gradient Boosting Classifier for CICIDS Dataset (Accuracy)



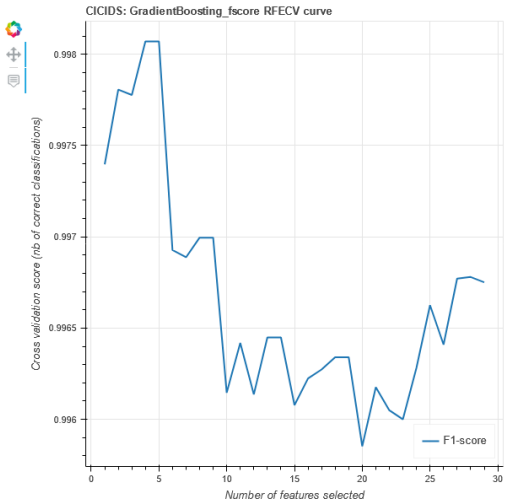
Highest accuracy: 99.9%
No. of selected features: 10

Figure 6: RFE with Random Forest Classifier for CICIDS Dataset (Accuracy)



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

Figure 7: RFE with Extra Trees Classifier for CICIDS Dataset (Accuracy)

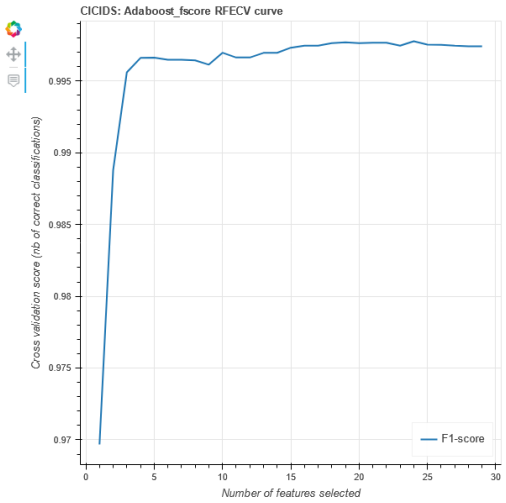


Highest f1-score: 99.8%
No. of selected features: 4

Figure 8: RFE with Gradient Boosting Classifier for CICIDS Dataset (F1-score)

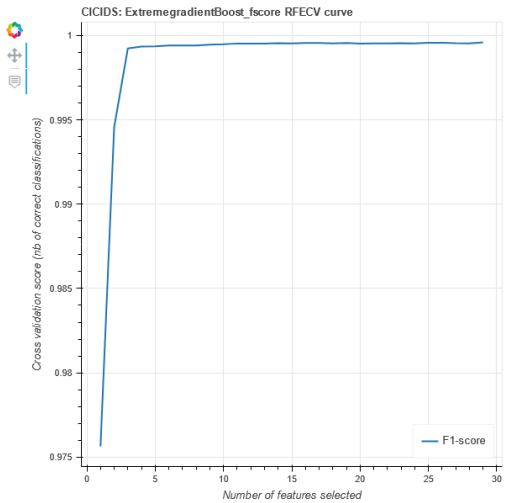
the same dataset, F1-scores are illustrated in Figure 18 for Gradient Boosting, Figure 19 for Adaboost, Figure 20 for XGBoost, Figure 21 for Random Forest, and Figure 22 for Extra Trees classifier.

On the other hand, for the HTTP Flood dataset, Extreme Gradient Boosting achieves the highest accuracy with 99.9% with 2 features only as shown in Figure 23. All other learners give similar performance with highest accuracy 99.9% as shown in Figures 24 (for Adaboost), Figure 25 (for Gradient Boosting), Figure 26 (for Random Forest), Figure 27 (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 classifier with 2 features. For the same dataset, F1-scores are illustrated in Figure 28 for Gradient Boosting, Figure 29 for Adaboost, Figure 30 for XGBoost, Figure 31 for Random Forest, and Figure 32 for Extra Trees classifier.



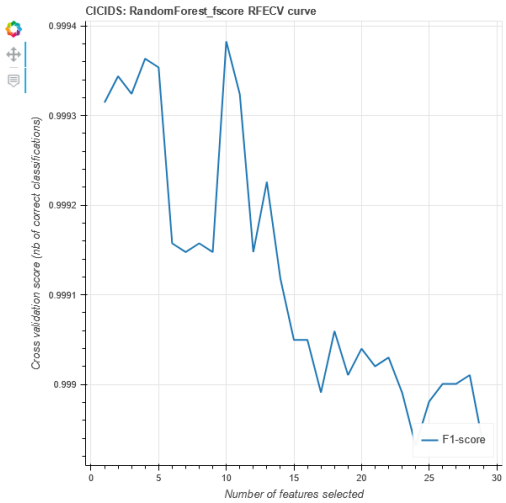
Highest f1-score: 99.7%
No. of selected features: 4

Figure 9: RFE with Adaboost Classifier for CICIDS Dataset (F1-score)



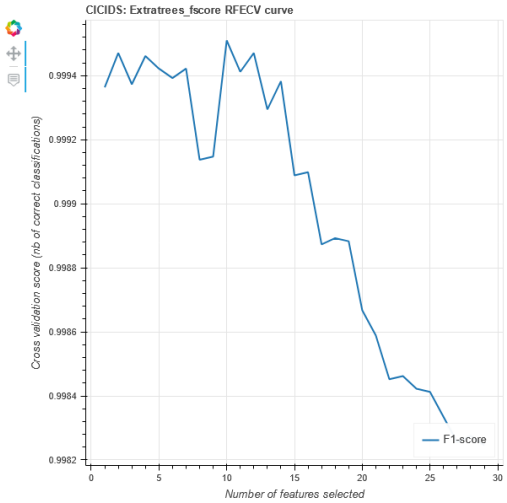
Highest f1-score: 99.8%
No. of selected features: 3

Figure 10: RFE with XGBoost Classifier for CICIDS Dataset (F1-score)



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

Figure 11: RFE with Random Forest Classifier for CICIDS Dataset (F1-score)



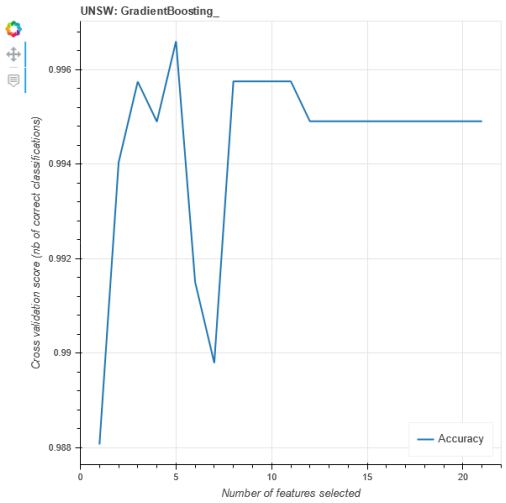
Highest f1-score: 99.9%
No. of selected features: 10

Figure 12: RFE with Extra Trees Classifier for CICIDS Dataset (F1-score)

Table 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.

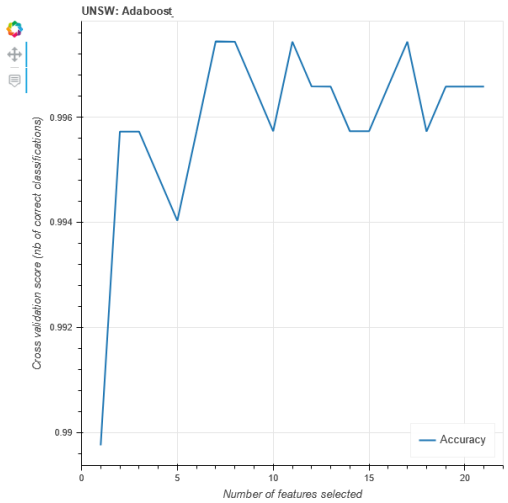
6.1 Comparison with other Feature Selection Methods

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 [31], CMIM [47], and mRMR [48], DISR [49], JMI, Gini index and ANOVA feature selection. Figure **??**, **??** and **??** 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, since the proposed method obtained highest accuracy with 3 features, the F1-score comparison is also done considering 3 features only for each



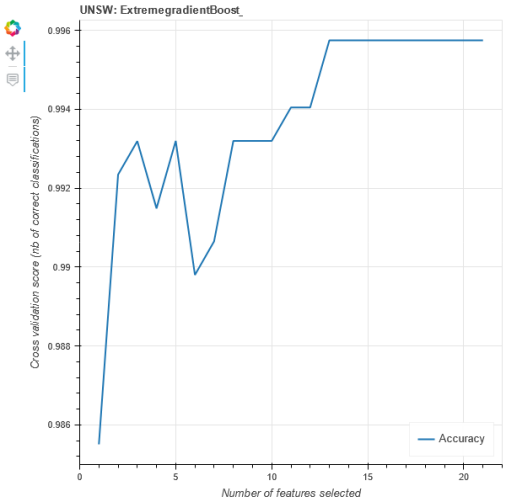
Highest accuracy: 99.7%
No. of selected features: 5

Figure 13: RFE with Gradient Boosting Classifier for UNSW Dataset (Accuracy)



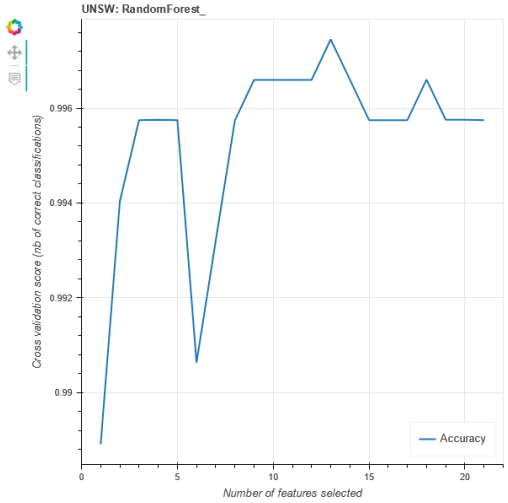
Highest accuracy: 99.7%
No. of selected features: 7

Figure 14: RFE with Adaboost Classifier for UNSW Dataset (Accuracy)



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

Figure 15: RFE with XGBoost Classifier for UNSW Dataset (Accuracy)



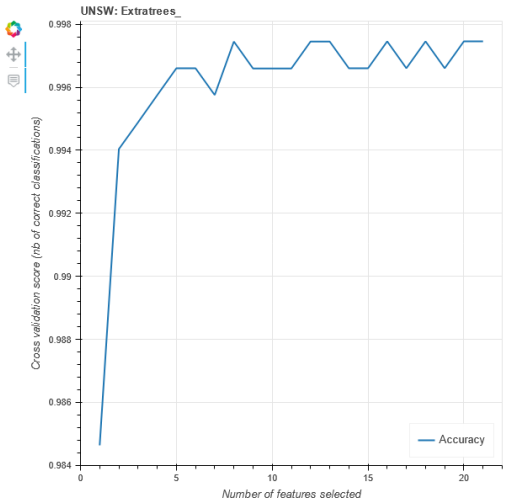
Highest accuracy: 99.7%
No. of selected features: 13

Figure 16: RFE with Random Forest Classifier for UNSW Dataset (Accuracy)

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.

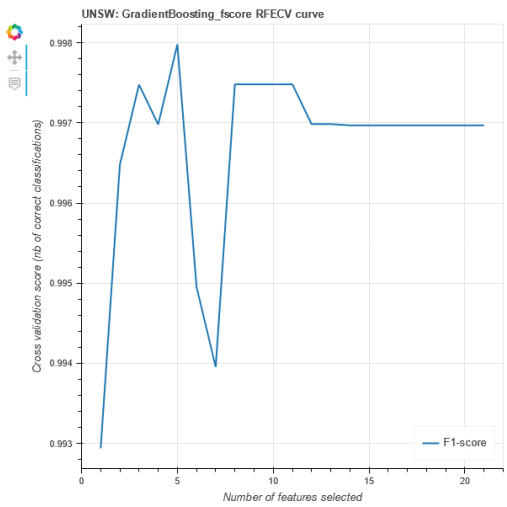
7 Conclusion

INFS-MICC is an incremental feature selection method, 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 irredundant features.



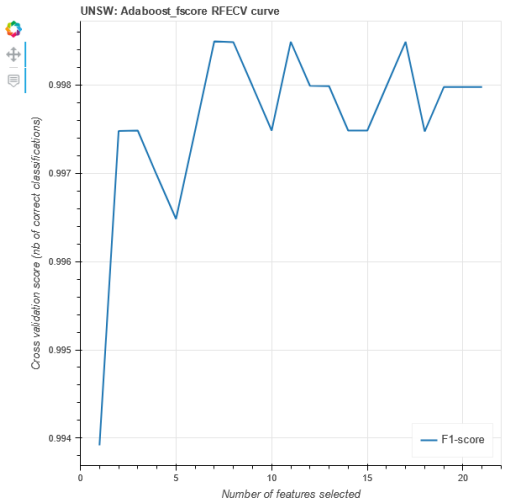
Highest accuracy: 99.7%
No. of selected features: 8

Figure 17: RFE with Extra Trees Classifier for UNSW Dataset (Accuracy)



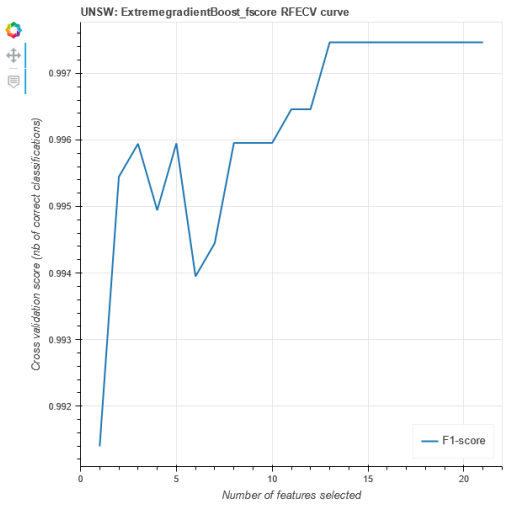
Highest f1-score: 99.8%
No. of selected features: 5

Figure 18: RFE with Gradient Boosting Classifier for UNSW Dataset (F1-score)



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

Figure 19: RFE with Adaboost Classifier for UNSW Dataset (F1-score)



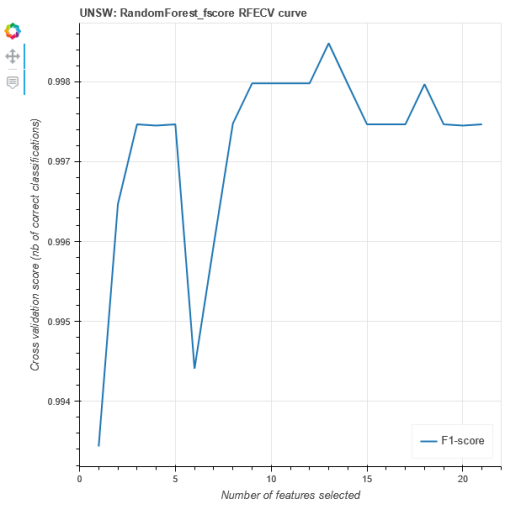
Highest f1-score: 99.7%
No. of selected features: 13

Figure 20: RFE with XGBoost Classifier for UNSW Dataset (F1-score)

For computing the relevance, feature-class mutual information is considered and for the irredundancy among features, the feature-feature correlation is computed. The main highlight of our method is that it is incremental in nature, 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.

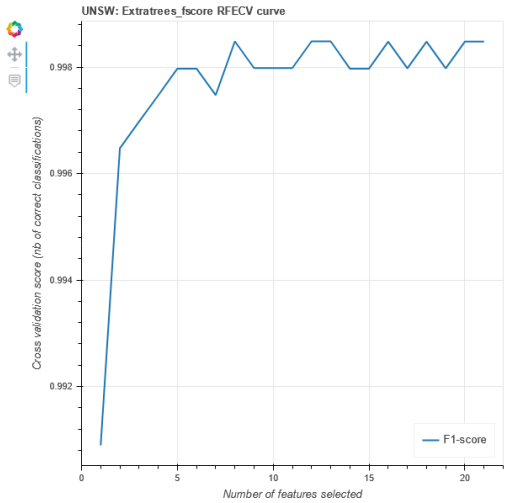
References

[1] Abhishek Singh, Baibhav Singh, and Hirosh Joseph. Vulnerability Analysis for HTTP. In *Vulnerability Analysis and Defense for the Internet*, pages 79–110. Springer, 2008.



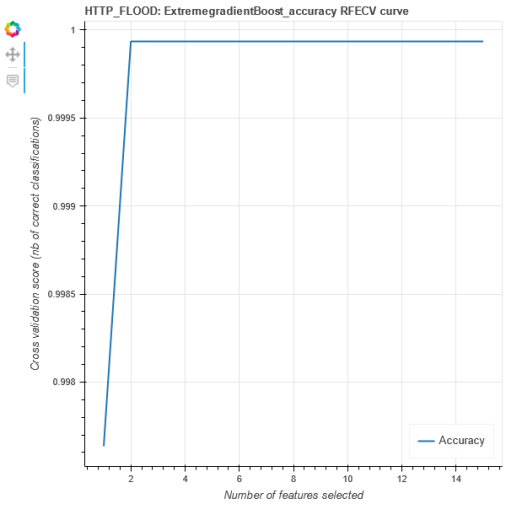
Highest f1-score: 99.8%
No. of selected features: 13

Figure 21: RFE with Random Forest Classifier for UNSW Dataset (F1-score)



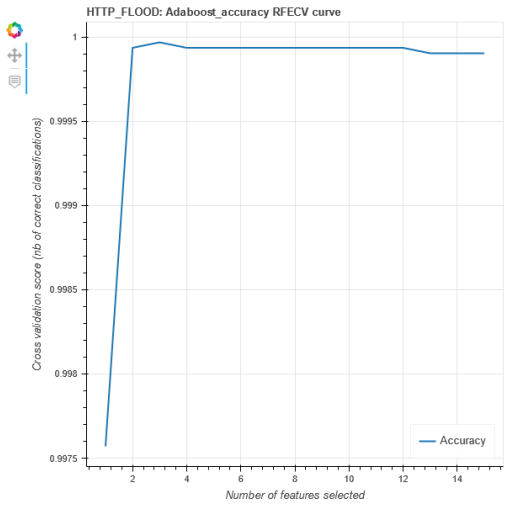
Highest f1-score: 99.8%
No. of selected features: 8

Figure 22: RFE with Extra Trees Classifier for UNSW Dataset (F1-score)



Highest accuracy: 99.9%
No. of selected features: 2

Figure 23: RFE with XGBoost Classifier for HTTP Flood Dataset (Accuracy)



Highest accuracy: 99.9%
No. of selected features: 3

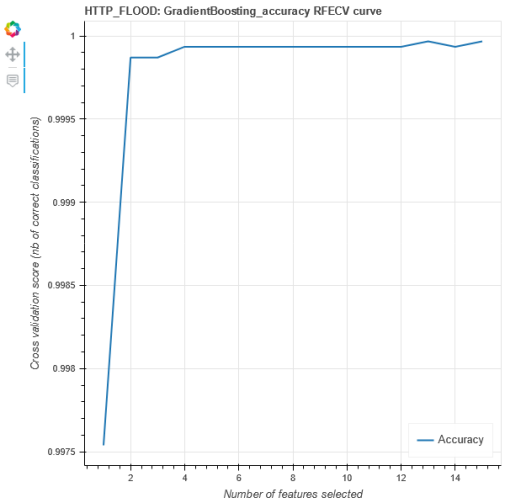
Figure 24: RFE with Adaboost Classifier for HTTP Flood Dataset (Accuracy)

[2] Indraneel Sreeram and Venkata Praveen Kumar Vuppala. HTTP flood Attack Detection in Application Layer using Machine Learning Metrics and Bio Inspired Bat Algorithm. *Applied computing and informatics*, 15(1):59–66, 2019.

[3] Pat Langley et al. Selection of Relevant Features in Machine Learning. In *Proceedings of the AAAI Fall symposium on relevance*, volume 184, pages 245–271. California, 1994.

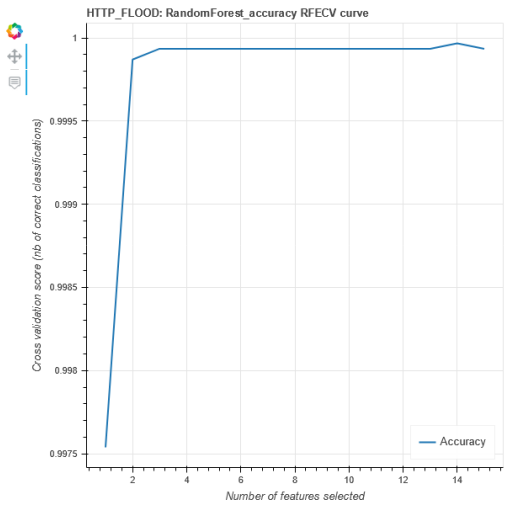
[4] Yong Luo, Liancheng Yin, Wenchao Bai, and Keming Mao. An Appraisal of Incremental Learning Methods. *Entropy*, 22(11):1190, 2020.

[5] Marthony Taguinod, Adam Doupé, Ziming Zhao, and Gail-Joon Ahn. Toward a Moving Target Defense for Web Applications. In *2015 IEEE international conference on information reuse and integration*, pages 510–517. IEEE, 2015.



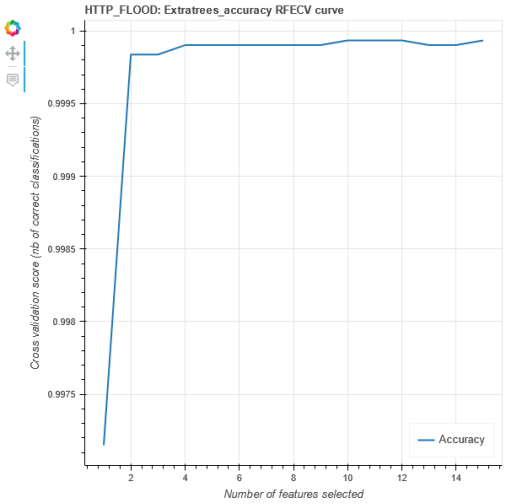
Highest accuracy: 99.9%
No. of selected features: 4

Figure 25: RFE with Gradient Boosting Classifier for HTTP Flood Dataset (Accuracy)



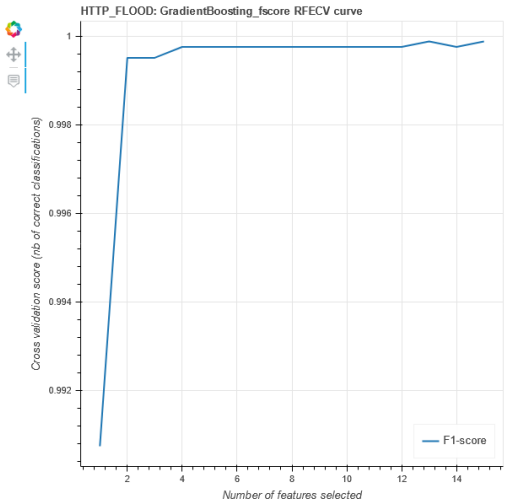
Highest accuracy: 99.9%
No. of selected features: 3

Figure 26: RFE with Random Forest Classifier for HTTP Flood Dataset (Accuracy)



Highest accuracy: 99.9%
No. of selected features: 4

Figure 27: RFE with Extra Trees Classifier for HTTP Flood Dataset (Accuracy)



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

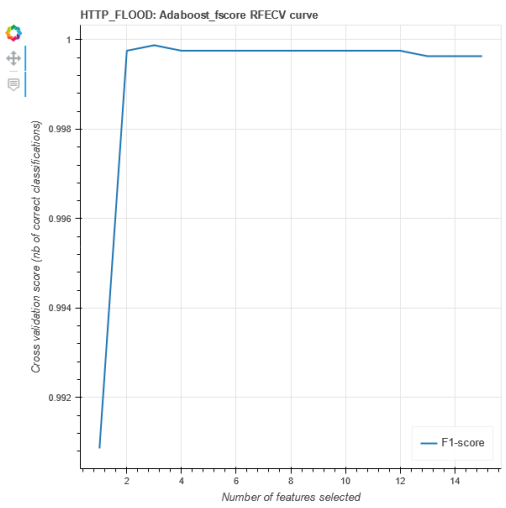
Figure 28: RFE with Gradient Boosting Classifier for HTTP Flood Dataset (F1-score)

[6] Abhinav Jain, Hima Patel, Lokesh Nagalapatti, Nitin Gupta, Sameep Mehta, Shanmukha Guttula, Shashank Mujumdar, Shazia Afzal, Ruhi Sharma Mittal, and Vitobha Munigala. Overview and Importance of Data Quality for Machine Learning Tasks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3561–3562, 2020.

[7] Christophe Giraud-Carrier. A Note on the Utility of Incremental Learning. *Ai Communications*, 13(4):215–223, 2000.

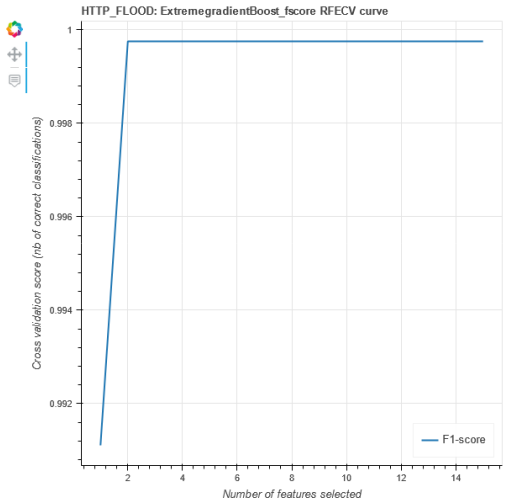
[8] Pradip Dhal and Chandrashekhar Azad. A Comprehensive Survey on Feature Selection in the Various Fields of Machine Learning. *Applied Intelligence*, 52(4):4543–4581, 2022.

[9] Saman Taghavi Zargar, James Joshi, and David Tipper. A Survey of Defense Mechanisms against Distributed Denial of Service (DDoS) Flooding Attacks. *IEEE Communications Surveys & Tutorials*, 15(4):2046–2069, 2013.



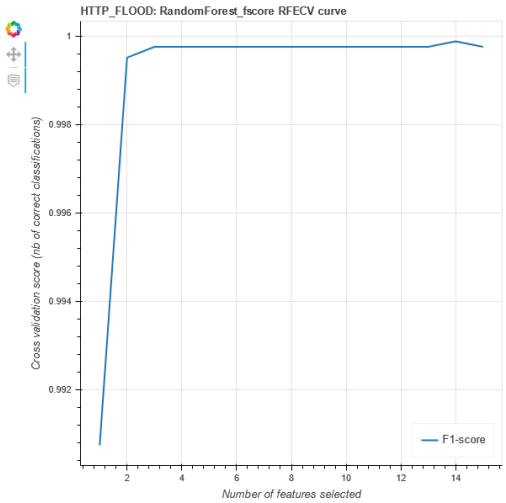
Highest f1-score: 99.9%
No. of selected features: 3

Figure 29: RFE with Adaboost Classifier for HTTP Flood Dataset (F1-score)



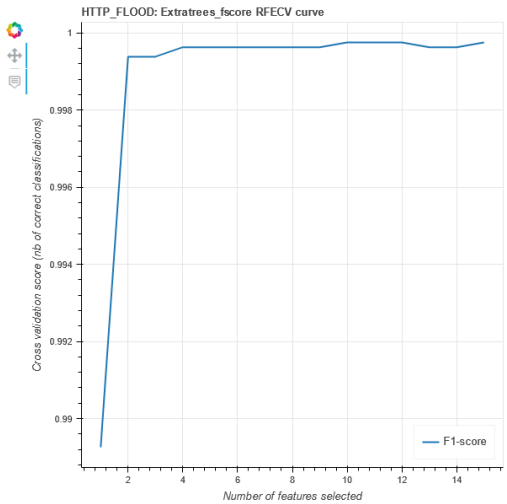
Highest f1-score: 99.9%
No. of selected features: 2

Figure 30: RFE with XGBoost Classifier for HTTP Flood Dataset (F1-score)



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

Figure 31: RFE with Random Forest Classifier for HTTP Flood Dataset (F1-score)



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

Figure 32: RFE with Extra Trees Classifier for HTTP Flood Dataset (F1-score)

[10] Amit Praseed and P Santhi Thilagam. DDoS Attacks at the Application Layer: Challenges and Research Perspectives for Safeguarding Web Applications. *IEEE Communications Surveys & Tutorials*, 2018.

[11] Khundrakpam Johnson Singh and Tanmay De. MLP-GA based algorithm to detect Application layer DDoS Attack. *Journal of Information Security and Applications*, 36:145–153, 2017.

[12] Yuntao Zhao, Wenbo Zhang, Yongxin Feng, and Bo Yu. A Classification Detection Algorithm Based on Joint Entropy Vector against Application-layer DDoS Attack. *Security and Communication Networks*, 2018, 2018.

[13] Sheng Wen, Weijia Jia, Wei Zhou, Wanlei Zhou, and Chuan Xu. CALD: Surviving various Application-layer DDoS Attacks that mimic Flash Crowd. In *4th international conference on Network and System Security (NSS)*, pages 247–254. IEEE, 2010.

[14] Takeshi Yatagai, Takamasa Isohara, and Iwao Sasase. Detection of HTTP-GET Flood Attack based on Analysis of

Table 3: Top 10 Ranked Features of the 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

Page Access Behavior. In *2007 IEEE Pacific rim conference on communications, computers and signal processing*, pages 232–235. IEEE, 2007.

[15] A Dhanapal and P Nithyanandam. The Slow HTTP DDOS Attacks: Detection, Mitigation and Prevention in the Cloud Environment. *Scalable Comput. Pract. Exp.*, 20(4):669–685, 2019.

[16] A Dhanapal and P Nithyanandam. An OpenStack based Cloud Testbed Framework for Evaluating HTTP Flooding Attacks. *Wireless Networks*, 27(8):5491–5501, 2021.

[17] Hala Albaroodi, Selvakumar Manickam, and Mohammed Anbar. A Proposed Framework for Outsourcing and Secure Encrypted Data on OpenStack Object Storage (Swift). *Journal of Computer Science*, 11(3):590, 2015.

[18] Reza Mohammadi, Chhagan Lal, and Mauro Conti. HTTPScout: A Machine Learning based Countermeasure for HTTP Flood Attacks in SDN. *International Journal of Information Security*, 22(2):367–379, 2023.

[19] Noe Marcelo Yungaicela-Naula, Cesar Vargas-Rosales, and Jesus Arturo Perez-Diaz. SDN-based Architecture for Transport and Application Layer DDoS Attack Detection by using Machine and Deep Learning. *IEEE Access*, 9:108495–108512, 2021.

[20] Ahmad Zainudin, Love Allen Chijioke Ahakonye, Rubina Akter, Dong-Seong Kim, and Jae-Min Lee. An Efficient Hybrid DNN for DDoS Detection and Classification in Software Defined IIoT Networks. *IEEE Internet of Things Journal*, 2022.

[21] Noe M Yungaicela-Naula, Cesar Vargas-Rosales, Jesús Arturo Pérez-Díaz, and Diego Fernando Carrera. A Flexible SDN-based Framework for Slow-rate DDoS Attack Mitigation by using Deep Reinforcement Learning. *Journal of network and computer applications*, 205:103444, 2022.

[22] Omerah Yousuf and Roohie Naaz Mir. DDoS Attack Detection in Internet of Things using Recurrent Neural Network. *Computers and Electrical Engineering*, 101:108034, 2022.

[23] George H John, Ron Kohavi, and Karl Pfleger. Irrelevant Features and the Subset Selection Problem. In *Machine learning proceedings 1994*, pages 121–129. Elsevier, 1994.

[24] Isabelle Guyon and André Elisseeff. An Introduction to Variable and Feature Selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[25] Cosmin Lazar, Jonatan Taminau, Stijn Meganck, David Steenhoff, Alain Coletta, Colin Molter, Virginie de Schaetzen, Robin Duque, Hugues Bersini, and Ann Nowe. A Survey on Filter Techniques for Feature Selection in Gene Expression Microarray Analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9(4):1106–1119, 2012.

[26] Alan Jović, Karla Brkić, and Nikola Bogunović. A Review of Feature Selection Methods with Applications. In *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1200–1205. Ieee, 2015.

[27] Ali Muhammad Usman, Umi Kalsom Yusof, and Syibrah Naim. Filter-based Multi-objective Feature Selection using NSGA III and Cuckoo Optimization Algorithm. *IEEE Access*, 8:76333–76356, 2020.

[28] Yingying Zhu, Junwei Liang, Jianyong Chen, and Zhong Ming. An Improved NSGA-III Algorithm for Feature Selection used in Intrusion Detection. *Knowledge-Based Systems*, 116:74–85, 2017.

[29] Nazrul Hoque, Dhruba K Bhattacharyya, and Jugal K Kalita. MIFS-ND: A Mutual Information-based Feature Selection Method. *Expert Systems with Applications*, 41(14):6371–6385, 2014.

[30] Martin Binder, Julia Moosbauer, Janek Thomas, and Bernd Bischl. Multi-objective Hyperparameter Tuning and Feature Selection using Filter Ensembles. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pages 471–479, 2020.

[31] Roberto Battiti. Using Mutual Information for Selecting Features in Supervised Neural Net Learning. *IEEE Transactions on neural networks*, 5(4):537–550, 1994.

[32] Naoual El Aboudi and Laila Benhlima. Review on Wrapper Feature Selection Approaches. In *2016 International Conference on Engineering & MIS (ICEMIS)*, pages 1–5. IEEE, 2016.

[33] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[34] Jianyu Miao and Lingfeng Niu. A Survey on Feature Selection. *Procedia Computer Science*, 91:919–926, 2016.

[35] Ron Kohavi and George H John. Wrappers for Feature Subset Selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[36] Sebastián Maldonado and Richard Weber. A Wrapper Method for Feature Selection using Support Vector Machines. *Information Sciences*, 179(13):2208–2217, 2009.

[37] Gang Chen and Jin Chen. A Novel Wrapper Method for Feature Selection and its Applications. *Neurocomputing*, 159:219–226, 2015.

[38] Hui-Huang Hsu, Cheng-Wei Hsieh, and Ming-Da Lu. Hybrid Feature Selection by Combining Filters and Wrappers. *Expert Systems with Applications*, 38(7):8144–8150, 2011.

[39] Upasana Sarmah and DK Bhattacharyya. Cost-Effective Detection of Cyber Physical System Attacks. In *Advances in Machine Learning for Big Data Analysis*, pages 33–69. Springer, 2022.

[40] Alexey Tsymbal, Mykola Pechenizkiy, and Pádraig Cunningham. Diversity in Search Strategies for Ensemble Feature Selection. *Information Fusion*, 6(1):83–98, 2005.

[41] Carolin A Rickert, Manuel Henkel, and Oliver Lieleg. An Efficiency-driven, Correlation-based Feature Elimination Strategy for Small Datasets. *APL Machine Learning*, 1(1), 2023.

[42] Hae-Young Kim. Statistical Notes for Clinical Researchers: Covariance and Correlation. *Restorative dentistry & endodontics*, 43(1), 2018.

[43] Pablo A Jaskowiak, Ricardo JGB Campello, Thiago F Covoes, and Eduardo R Hruschka. A Comparative Study on the use of Correlation Coefficients for Redundant Feature Elimination. In *2010 Eleventh Brazilian Symposium on Neural Networks*, pages 13–18. IEEE, 2010.

[44] Carlos D Correa and Peter Lindstrom. The Mutual Information Diagram for Uncertainty Visualization. *International Journal for Uncertainty Quantification*, 3(3), 2013.

[45] Nour Moustafa and Jill Slay. UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set). In *2015 military communications and information systems conference (MilCIS)*, pages 1–6. IEEE, 2015.

[46] Hossein Hadian Jazi, Hugo Gonzalez, Natalia Stakhanova, and Ali A Ghorbani. Detecting HTTP-based Application Layer DoS Attacks on Web Servers in the presence of Sampling. *Computer Networks*, 121:25–36, 2017.

[47] François Fleuret. Fast Binary Feature Selection with Conditional Mutual Information. *Journal of Machine learning research*, 5(9), 2004.

[48] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature Selection based on Mutual Information Criteria of Max-dependency, Max-relevance, and Min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, 27(8):1226–1238, 2005.

[49] Gavin Brown, Adam Pocock, Ming-Jie Zhao, and Mikel Luján. Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection. *The journal of machine learning research*, 13(1):27–66, 2012.