WEB ATTACK DETECTION USING ML

A Project Report

Submitted by

Ruturaj Malavade	**111703037**
Heena Jamadar	**111703022**
Harshali Upadhye	**141803003**

in partial fulfilment for the award of the degree of

B.Tech Computer Engineering

Under the guidance of

Prof. Jagannath V. Aghav

College of Engineering, Pune

DEPARTMENT OF COMPUTER ENGINEERING AND INFORMATION TECHNOLOGY, COLLEGE OF ENGINEERING, PUNE-5

May, 2021

DEPARTMENT OF COMPUTER ENGINEERING AND INFORMATION TECHNOLOGY,

COLLEGE OF ENGINEERING, PUNE

CERTIFICATE

Certified that the project titled, “WEB ATTACK DETECTION USING ML” has been successfully completed by

Ruturaj Malavade	**111703037**
Heena Jamadar	**111703022**
Harshali Upadhye	**141803003**

and is approved for the partial fulfilment of the requirements for the degree of “B.Tech. Computer Engineering/ Information Technology”.

SIGNATURE	SIGNATURE
Mr. Deepak D. Kshirsagar	**Dr. Vahida Attar**
Internal Examiner	**Head**
Department of Computer Engineering	**Department of Computer Engineering**
and Information Technology,	**and Information Technology,**
College of Engineering Pune,	**College of Engineering Pune,**
Shivajinagar, Pune - 5.	**Shivajinagar, Pune - 5.**

Abstract

This project proposes a two-level IDS model that utilizes the UNSW-NB15 dataset, applying various ML feature reduction and classification methods to try and detect web attacks. We use a tool called Weka that simulates various ML algorithms to get the experimental values. By using various attribute evaluation algorithms such as InfoGain, Gain Ratio, Correlation, Symmetrical Uncertainty, Chi Squared, Relief F, we get the weights associated with the attributes of the training set. We then apply classifier functions such as tree-based classifiers, rule-based classifiers, and Bayes classifiers on the training set, and get the accuracy. We then select the classifiers that give the best accuracy, and perform performance analysis on the testing set by using these specific classifiers, by taking the sets of the attributes of the testing set according to various rules, such as taking threshold value of weights in steps of 0.1, taking the attributes in steps of 25 percent and 10 percent of all attributes, taking union-intersection of multiple attribute evaluation algorithms and running the analysis on those, and so on. This gives us the final accuracy for our first level, that means the best result which gives us the minimum time to build the model. BayesNet gives us the best time taken to build model, but Random Forest gives us the best accuracy. To improve this, we go to the second level of the proposed IDS. In the second level, we take the features selected from the first level i.e. by applying BayesNet classifier , and apply Random Forest classifier on these features. The accuracy values thus obtained are the final values. We also compare the model against traditional systems and existing ML classifiers.

Contents

List of Tables

List of Figures

List of Symbols

IG:	Info Gain attribute evaluator
GR:	Gain Ratio attribute evaluator
COR:	Correlation attribute evaluator
SU:	Symmetrical Uncertainty attribute evaluator
RF:	Relief F attribute evaluator
CS:	Chi Squared attribute evaluator
SCM:	Subset Combination Method

Chapter 1

Introduction

Web security is an important aspect of our everyday lives. Web security is important to prevent hackers and cyber-thieves from being able to access sensitive and/or private information. A web attack can also cause a company or business to suffer substantial financial losses, due to theft of financial and/or corporate information. Without a good web attack detection strategy, a business or website owner runs a risk of spreading of malware and attacks on other businesses, websites, or even networks.

Web attacks are attacks on network security; cyber-criminals utilize weaknesses or vulnerabilities in a network to get access to confidential data. There are many types of web attacks that exist, such as malware, which is any kind of malicious software (such as viruses, spyware or ransomware), fuzzers, in which an attacker tries to exploit flaws in system security by crashing it using large amounts of randomized data, analysis, an attack that penetrates web applications generally via email or web scripts, backdooring, which is bypassing of standard authentication methods, exploits, which take advantage of glitches or bugs in a system, Denial of Service (DoS), which prevents authorized requests [17] from accessing a device, worms, which is an attack that replicates itself over and over to spread to other devices, spoofing, in which an unknown source disguises itself as a trusted one, and so on.

A popular method of web attack detection is by using an Intrusion Detection System (IDS). An IDS [11] monitors a network or system for any malicious activity and raises alerts to detect any such activity. An IDS works by looking for signatures of known attacks, or activity that is different from normal activity, and sends alerts to a system administrator if it detects any such kind of abnormal activity.

IDSs are classified into 2 main categories, based on where detection takes place: host-based IDSs and network-based IDSs. A network-based IDS monitors traffic on all devices on a network, and detects malicious traffic, if any. It does so by scanning network packets at the host level, and logging any suspicious packets that might be present. Whereas, a host-based IDS runs on individual hosts in the network, and analyzes traffic and logs malicious behavior on the system on which it is installed. It uses a database of important objects of the system such as environment variables that it should monitor, and for each object, it creates a checksum that is stored in a database for comparison later.

IDSs can also be classified into 2 categories based on the detection method they use: signature-based IDSs, and anomaly-based IDSs. A signature-based IDS detects attacks by looking for specific patterns such as byte sequences in network traffic or known malicious instruction sequences in malware. It relies on a list of known indicators that indicate a system is compromised, such as malicious network behavior, email subject lines, malicious domain names, and so on. Whereas, an anomaly-based IDS monitors system activity and tries to classify it as normal or anomalous. It does so by creating a model of normal (valid or trustworthy) activity using machine learning (ML), and then compares new behavior in a system against this model.

A signature-based IDS suffers from a major drawback, namely that it is difficult for it to detect new attacks, as no prior information is available on said new attacks as compared to attacks that may have already had information on them. Since an anomaly-based IDS uses ML to create a model to compare against, and these models can be trained better, an anomaly-based IDS does not suffer from this issue.

A signature-based IDS also requires reprogramming for every new pattern that is to be detected in a system. However, it has a low false positive rate. While an anomaly-based IDS can suffer from false positives more, it does not need to be reprogrammed, because the ML model can learn the pattern of the new attack occurring. Hence, anomaly-based IDSs are used more compared to signature-based IDSs nowadays.

ML plays a major role in intrusion detection in modern times. This is because with recent developments, the amount of data that is handled becomes too large for classic methods to be usable. Further, classic methods are not able to actively learn data patterns and easily find web attacks as compared to ML-based intrusion detection methods, which are much more easily able to detect web attacks. ML finds use in intrusion detection as part of anomaly-based IDS methods, which use ML to build a model to compare potentially malicious behavior against.

Chapter 2

Literature Review

In the paper [1], a novel Hierarchical Intrusion Detection System (IDS) is proposed, combining three different classifiers: JRip algorithm, REP Tree and Forest PA, on decision tree and rules-based concepts. The proposed model uses three classifiers, where the first two methods take the dataset and classify its features as being either Normal or Attack, while the third method uses this classification along with the original dataset to classify even further.

The paper [2] aims to use ML techniques to identify rare cyber-attacks using the UNSW-NB15 dataset. It uses a hybrid feature selection method to identify the most important features for each type of attack in the dataset, of which there are nine: Exploits, Denial-Of-Service (DoS), Worms, Reconnaissance, Fuzzers, Generic, Analysis, Shellcode, and Backdoor. Overall, the most important features found were as follows: ct_srv_src (feature number 41), which occurred in 5 of the 9 attack types, service (feature number 14), which was found in 7 of the attack types (all except Exploit and Denial-Of-Service), sttl and dttl (features 10 and 11 respectively), which were found in 6 of the attack types, and dur (feature number 2), which was found to be in 8 of the attack types; all except worms. They used the Naïve Bayes classifier, and using the feature subsets found in the hybrid feature selection method, they were able to achieve a lower false acceptance rate as well as higher classification rate compared to other methods.

In the paper [3], an IDS is created using Deep Neural Network (DNN) for the Internet of Things. They used classification of intruded patterns as the basis for intrusion detection. They also found the performance of NSL and UNSW-NB15 datasets using the DNN.

The paper [4] presents an intrusion detection approach that uses a genetic algorithm. This paper uses a Support Vector Machine along with genetic algorithms, and introduced feature selection and attack detection methods that improve on the performance of IDSs. The feature selection method that uses the new fitness function measures TPR and FPR (true positive rate and false positive rate respectively), and selects features of the training data better than traditional methods. The experimentation and result analysis by using the UNSW-NB15 and KDD99 datasets are measured using metrics like classification accuracy, Receiver Operating Characteristic (ROC) curve, false positive rate, and true positive rate.

A new feature correlation technique, CorrCorr was introduced in [5].They combined CorrCorr with the state of the art addition-based correlation detection techniques and compared the results of the original features with most commonly used feature selection technique named PCA.

. The paper [6] proposes a network intrusion detection system using Extreme Gradient Boosting (XGBoost) on the UNSW-NB15 dataset. For analysis purposes, a subset of 23 features was taken, and the Random Forest classifier was applied to this subset. This gave an accuracy of 97.9060%.

In the paper [7], they did the study to find the significant features and compare them with findings of previous works done using KDD'99 and UNSW-NB15 datasets. They created two subsets and found the accuracy using machine learning techniques.

The paper [8] presented an attack procedure and its impact for each attack, execution of attacks wherever appropriate, as well as countermeasures that can be taken to prevent said attacks. To cope with threats like these, a Web developer must know about the risks in the system, as well as the impact they can have. Furthermore, the existing Web standards should be examined even more for potentially undiscovered vulnerabilities, so as to develop more advanced countermeasures against them.

Paper [9] proposes a SVM model to detect anomaly in the system.They have performed intrusion detection using SVM with binary-classification as well as multilabel-classification.

The paper [10] explores the XGBoost algorithm's application for feature selection in conjunction with multiple ML techniques, including Decision Trees, Support Vector Machines, k-Nearest Neighbors, Logistic Regression, and Artificial Neural Networks, in order to implement accurate IDSs. This paper uses multiclass and binary classification techniques on the UNSW-NB15 dataset. Further, applying an XGBoost-based feature selection algorithm on the dataset gave us 19 optimal features.

The paper [12] has proposed an integrated classification-based IDS and evaluates its performance on an existing dataset and a newly generated one. This paper evaluates the proposed model's performance on the UNSW-NB15 dataset, which covers more recent attacks (such as Denial-of-Service, Probe, Fuzzers, Generic, Exploits and so on) compared to another dataset, the KDD99 data set. It is observed that compared to older models that utilize Decision Trees, this model gives better values in terms of certain evaluation metrics. Further, this paper also generates a real-time data set at the NIT Patna CSE lab (RTNITP18), and it acts as the testing data set to evaluate the performance of the proposed model, which gives an accuracy of 83.8% on this dataset.

In the paper [13], an intelligent system is proposed which first performs feature ranking based on Information Gain (IG) and Correlation (COR). Then, feature reduction is performed by combining ranks obtained from both IG and COR, using a novel approach to identify features to be removed. The model is then trained and tested on the KDD99 dataset and gave outstanding results.

The paper [14] provides an overview of the KDD99 and UNSW-NB15 datasets' significant features for network IDSs. For analysis, it uses an ARM (Association Rule Mining) algorithm to generate the strongest features from both datasets. Analysis shows that that the KDD99 dataset gives more accuracy as compared to the UNSW-NB15 one.

In [15] paper, the analysis and the evaluation is done of the UNSW-NB15 data set. According to their study they found out Decision Tree gives the best efficiency compared to other techniques. The effficiency techniques using KDD99 dataset are better than that of UNSW NB-15 is what the have found.

A new Geometric Area Analysis (GAA) technique based on Beta Mixture Model (BMM) was introduced in [16]. PCA was used for selecting the highest variance data for both the selected original features and their principal components is discussed in their data preprocessing module.

In the paper [18], an ensemble-based multi-filter feature selection method is proposed that takes four filter methods, namely IG, Gain Ratio (GR), ReliefF, and Chi-Squared (CS), and puts together their analyzed outputs to achieve optimal selection. This method is then evaluated using the NSL-KDD dataset, and the evaluation produces a higher attack classification accuracy and attack detection rate in comparison with other classification methods.

Chapter 3

Research Gaps and Problem Statement

3.1 Research Gaps

In [3], using Deep Neural Network on NSL-KDD dataset they found out that the overall accuracy to detect intrusion is 91.5%, while DNN on UNSW NB-15 dataset gives an overall accuracy of 99.2%. Their method is almost successful to identify attacked behaviour.

In a paper using a new feature correlation technique named CorrCorr [5], the top 6 features selected using CorrCorr gave an accuracy of 93.22%.

In [6] they have efficiently reduced the 39 features of UNSW NB-15 Dataset to 23 features and applied the best intrusion detecting machine learning method, according to them, which is XGBoost machine leaarning technique. They got an accuracy of 75.88% to correctly identify the attack.

In [7] two subsets were examined having the best results. One subset is taken from

[14], which consists of the most frequently used features and another subset is created using their machine learning techinques. Working on UNSW NB-15 preprocessed dataset, They got accuracy of 75.6617% by running Random Forest Algorithm using Weka, on subset 1 while accuracy of 81.6175% by running Random Forest on subset 2.

In [9] a SVM based network intrusion detection system with log based scaling is been analysed on UNSW NB-15 data set. They were successful in getting 85.99% accuracy for binary-classification and 75.77% accuracy with multilabel-classification. Their approach mostly inspired us to get deep understanding of network intrusion detection and finding results with binary as well as multilabel classified UNSW NB-15 dataset.

In [10] filter-based method inspired from XGBoost technique is used for feature selection. Applied various machine learning methods on the reduced 19 features and got an accuracy of 90.85% as the hightest using Decision Tree algorithm.

In paper [12] an integrated classification based IDS is generated and used this model to examine the behaviour of UNSW NB-15 dataset. They used C5 to reduce the number of features to 13 and an accuracy to correctly identify the attack of 89.76%.

A method was proposed in [14] which used ARM technique for feature selection to 29 features giving an accuracy to correctly identify the attack of 78.06% using Naive Bayes machine learning algorithm.

In [15], the UNSW NB-15 dataset is analysed and evaluated to find out 85.56% accuracy to determine attack in the dataset for the 9 reduced features.

In [16], GAA-ADS technique is used and their method gave an overall accuracy of 91.80% with 15 reduced features.

3.2 Problem Statement

The problem statement was to develop IDS to detect web attacks with ML classifiers using feature selection.

The objectives we have are:

- To do Literature Review
- Perform analysis using various classifiers such as Tree-based, Rule-based and Bayes-based classifiers
- Detect web attacks using feature selection methods
- Test the system using the UNSW-NB15 dataset, and measure performance in terms of accuracy of the algorithm and time taken to build the model.

Chapter 4

Proposed Methodology / Solution

This section discusses the method, flow of working, dataset used for the development of this project. The section will also consist of a brief discussion about the requirements (hardware and software) for further development.

4.1 Dataset Used

We have chosen the UNSW-NB15 dataset for our project. This dataset was created by utilising an IXIA PerfectStorm tool to extract a hybrid of modern normal and contemporary attack activities of network traffic. A TCPDump tool was used to capture 100 GB of raw network traffic (PCAP files). Argus and Bro-IDS tools are used and 12 algorithms are developed to generate 49 features with the class label. The dataset contains 2,540,044 records, stored in 4 CSV files. Further, a part of this dataset is configured as a training set and a testing set. The training set contains 175,341 records, while the testing set contains 82,332 records from the different types, Attack and Normal.

This dataset contains records that are either of the Normal category (no attack), or the Attack category (attack). There are 9 types of attacks present in this dataset, namely, Fuzzers, Analysis, Backdoors, Denial-Of-Service (DoS), Exploits, Generic, Reconnaissance, Shellcode, and Worms.

4.2 Functional and Non-Functional Requirements

The functional and non-functional requirements for this project are as follows:

Functional Requirements:

- Detect whether a particular message sent over a network is a normal safe message or an attack message.

- Detect the specific type of attack a message might be if it is an attack message.

- Keep a record of events and incidents.

Non-functional Requirements:

- Accuracy

- Precision

- Effectiveness

- Low Response time

- Stability

- Reliability

- Fault Tolerance

4.3 Method Used

Initially, we take the training and testing datasets, and apply preprocessing steps on those datasets, using Python language. In the preprocessing stage, we use a LabelEncoder to convert all the alphanumeric values in each dataset into an encoded, purely numeric format. This is done because Weka cannot utilize alphabetical labels for column values. We also remove null-valued entries in this stage.

Next, we take the preprocessed training set, and run various classifiers on the entire training set. Here, we are trying to find which classifier gives us the best accuracy.

We then took the value of the same classifiers but on the testing set as a base case and then found three approaches that gave us maximum accuracy, better than the base case. Then we got the reduced features from these different approaches and the best one will be used for future work.

The 3 approaches we got were BayesNet, Random Forest and PART; while BayesNet gives us the best build time, Random Forest gives us the best accuracy. BayesNet reduces the features to the greatest extent, however, so we use BayesNet instead of other techniques. To improve this system, we take the reduced features from the best BayesNet approach (that reduces features to maximum extent), and apply these on the Random Forest multilabel classifier; this gives us the final accuracy.

We also compare the system against traditional systems and other ML classifiers.

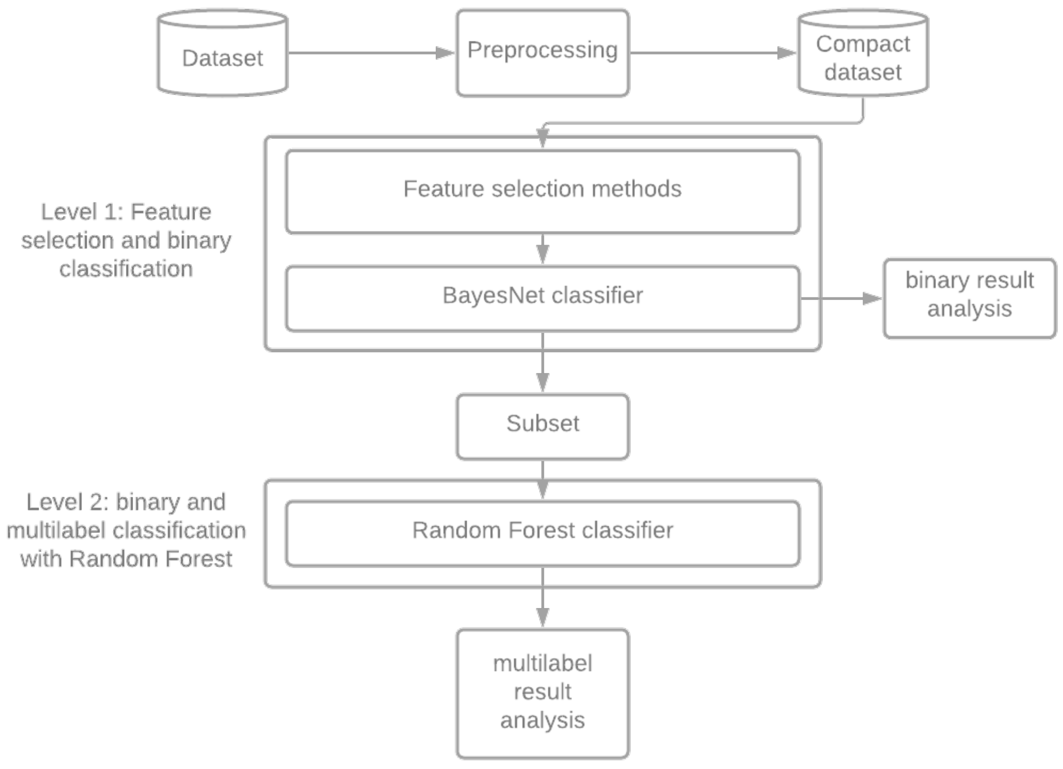

Dataset
Preprocessing
Compact dataset
Level 1: Feature selection and binary classification
Feature selection methods
BayesNet classifier
binary result analysis
Subset
Level 2: binary and multilabel classification with Random Forest
Random Forest classifier
multilabel result analysis

Figure 4.1: Proposed Methodology

Chapter 5

Experimental Setup

Hardware and software requirements are as follows:

5.1 Hardware Requirements

- RAM: 8 GB

- GPU, ideally

5.2 Software Requirements

- PC should be running Windows 10 OS

- Weka application installed

- Python 3 installed

Chapter 6

Results and Discussion

6.1 Level 1: Feature selection and binary classification

6.1.1 Approach 1 : Threshold

In this approach, we consider the weights in thresholds of jumps of 0.1. That is, we consider features with weights above 0, above 0.1, above 0.2 and so on. We apply these on the IG, GR, COR, SU and RF feature selection algorithms. The results for the same are shown below:

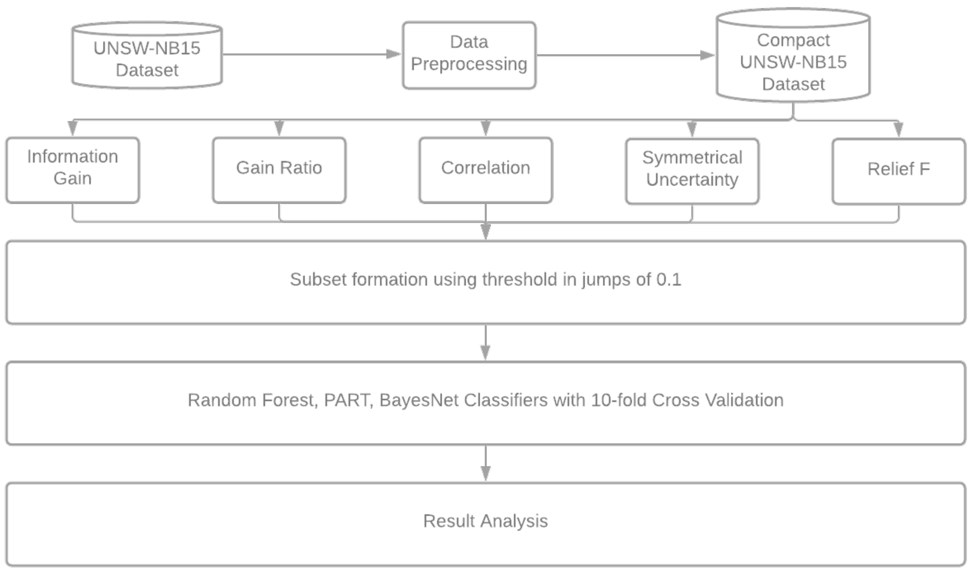

UNSW-NB15 Dataset
Data Preprocessing
Compact UNSW-NB15 Dataset
Information Gain
Gain Ratio
Correlation
Symmetrical Uncertainty
Relief F
Subset formation using threshold in jumps of 0.1
Random Forest, PART, BayesNet Classifiers with 10-fold Cross Validation
Result Analysis

Figure 6.1: Approach 1: Normal Threshold Approach

Method	**Threshold**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
PART	**0.1**	**31**	**97.1603**	**2.8397**	**54.46**
Random Forest	**0.1**	**31**	**97.6886**	**2.3114**	**38.71**
BayesNet	**0.4**	**11**	**87.7107**	**12.2893**	**0.85**

Table 6.1: IG, Threshold Approach

Method	**Threshold**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
PART	**0.1**	**17**	**93.1545**	**6.8455**	**15.46**
Random Forest	**0.1**	**17**	**93.8238**	**6.1762**	**52.36**
BayesNet	**0.1**	**17**	**84.5091**	**15.4909**	**1.27**

Table 6.2: GR, Threshold Approach

Method	**Threshold**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
PART	**0.1**	**21**	**96.0987**	**3.9013**	**44.83**
Random Forest	**0.1**	**21**	**97.1044**	**2.8956**	**45.69**
BayesNet	**0.5**	**2**	**80.3879**	**19.6121**	**0.12**

Table 6.3: COR, Threshold Approach

Method	**Threshold**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
PART	**0.1**	**25**	**93.2930**	**6.7070**	**31.12**
Random Forest	**0.1**	**25**	**94.1481**	**5.8519**	**62.73**
BayesNet	**0.1**	**25**	**83.5592**	**16.4408**	**2.23**

Table 6.4: SU, Threshold Approach

Method	**Threshold**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
PART	**0.1**	**2**	**75.4057**	**24.5943**	**2.84**
Random Forest	**0.1**	**2**	**75.4057**	**24.5943**	**0.09**
BayesNet	**0.1**	**2**	**75.4057**	**24.5943**	**0.45**

Table 6.5: RF, Threshold Approach

We also apply SCM on the BayesNet values, the results of which are shown below:

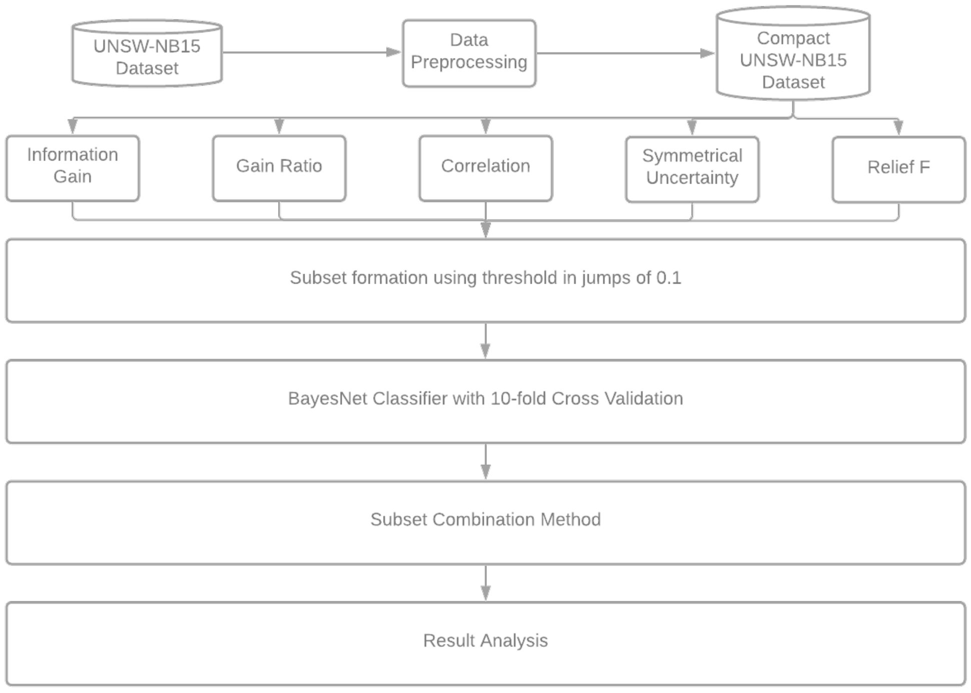

UNSW-NB15 Dataset
Data Preprocessing
Compact UNSW-NB15 Dataset
Information Gain
Gain Ratio
Correlation
Symmetrical Uncertainty
Relief F
Subset formation using threshold in jumps of 0.1
BayesNet Classifier with 10-fold Cross Validation
Subset Combination Method
Result Analysis

Figure 6.2: Approach 1: Threshold SCM

Method	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
SCM1	**26**	**85.1528**	**14.8472**	**3.37**
SCM2	**18**	**85.2815**	**14.7185**	**2.37**
SCM3	**7**	**88.1456**	**11.8544**	**0.52**
SCM4	**4**	**83.8556**	**16.1444**	**0.24**

Table 6.6: BayesNet SCM Threshold

6.1.2 Approach 2 : 4-Part Rank

In this approach, we consider features in ranked sections of 25%, that is, we consider 25%, 50%, 75% and all of the features. We apply these on the IG, GR, COR, SU, RF and CS feature selection algorithms. The results for the same are shown below:

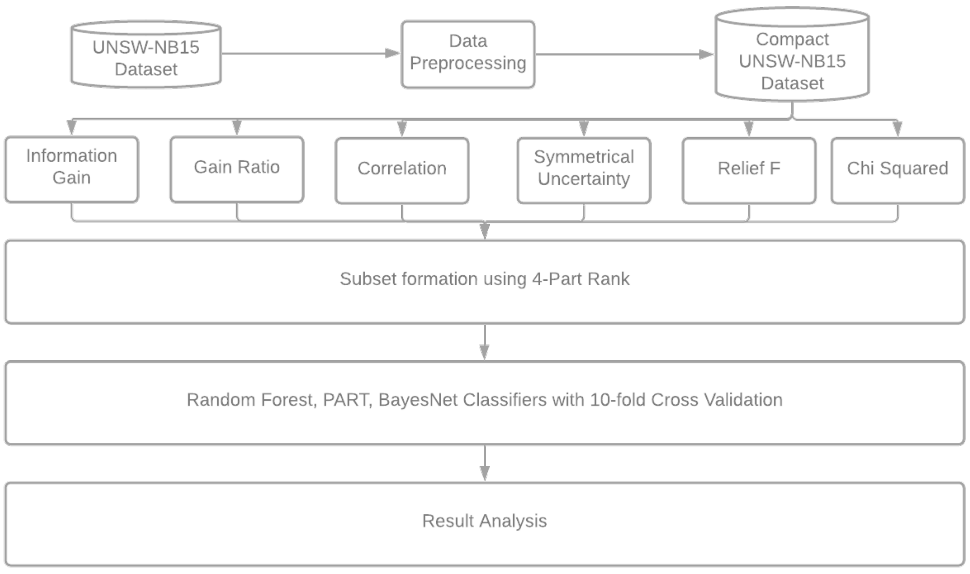

UNSW-NB15 Dataset
Data Preprocessing
Compact UNSW-NB15 Dataset
Information Gain
Gain Ratio
Correlation
Symmetrical Uncertainty
Relief F
Chi Squared
Subset formation using 4-Part Rank
Random Forest, PART, BayesNet Classifiers with 10-fold Cross Validation
Result Analysis

Figure 6.3: Approach 2: Normal 4-Part Rank

Method	Percentage of all features	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
Random Forest	**75**	**31**	**97.6886**	**2.3114**	**83.06**
PART	**75**	**31**	**97.1603**	**2.8397**	**76.11**
BayesNet	**25**	**10**	**87.7727**	**12.2273**	**0.99**

Table 6.7: IG, 4-Part Rank Approach

Method	Percentage of all features	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
Random Forest	**75**	**31**	**94.2404**	**5.7596**	**52.64**
PART	**75**	**31**	**93.5724**	**6.4276**	**64.82**
BayesNet	**50**	**21**	**84.7253**	**15.2747**	**2.12**

Table 6.8: GR, 4-Part Rank Approach

Method	Percentage of all features	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
Random Forest	**75**	**31**	**97.4214**	**2.5786**	**35.51**
BayesNet	**75**	**31**	**80.1244**	**19.8756**	**3.3**
PART	**75**	**31**	**96.5615**	**3.4385**	**130.32**

Table 6.9: COR, 4-Part Rank Approach

Method	Percentage of all features	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
Random Forest	**75**	**31**	**94.4202**	**5.5798**	**44.61**
PART	**75**	**31**	**93.1169**	**6.8831**	**182.63**
BayesNet	**50**	**21**	**84.6973**	**15.3027**	**1.84**

Table 6.10: SU, 4-Part Rank Approach

Method	**Percentage of all features**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
PART	**75**	**31**	**97.2757**	**2.7243**	**65.53**
BayesNet	**75**	**31**	**80.5896**	**19.4104**	**2.9**
Random Forest	**50**	**21**	**97.8089**	**2.1911**	**28.54**

Table 6.11: RF, 4-Part Rank Approach

Method	**Percentage of all features**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
Random Forest	**75**	**31**	**97.6886**	**2.3114**	**32.2**
PART	**75**	**31**	**96.1603**	**2.8397**	**99.75**
BayesNet	**25**	**10**	**87.7727**	**12.2273**	**1.04**

Table 6.12: CS, 4-part Rank Approach

In addition, we also used the SCM method to find the accuracy of the top 2, top 3, top 4, top 5 and all feature selection algorithms taken together; this was performed only on a tree-based approach, though. The results for this are shown below:

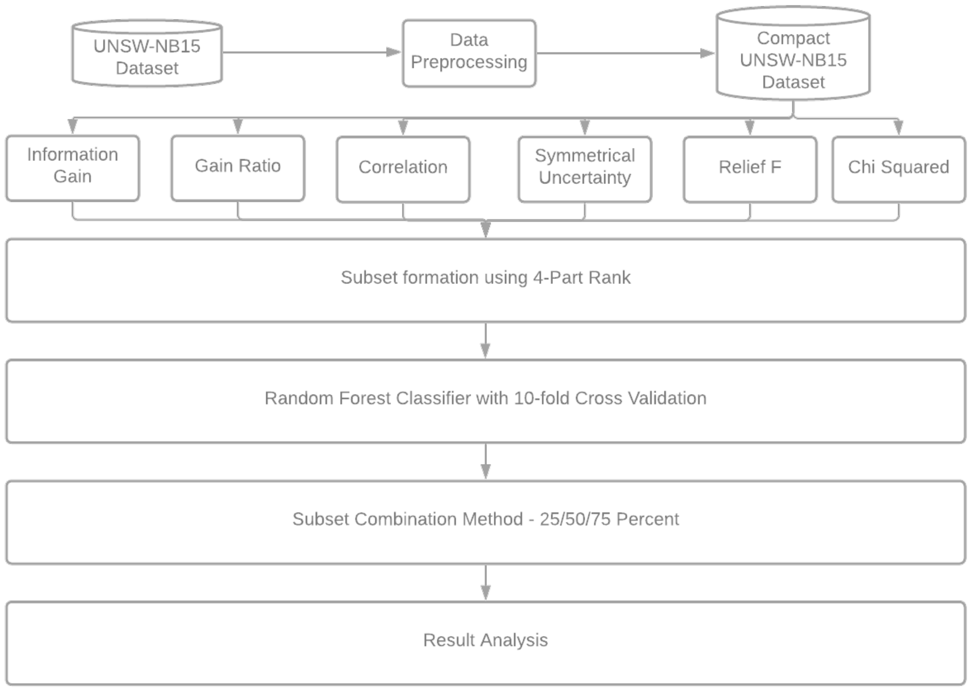

UNSW-NB15 Dataset
Data Preprocessing
Compact UNSW-NB15 Dataset
Information Gain
Gain Ratio
Correlation
Symmetrical Uncertainty
Relief F
Chi Squared
Subset formation using 4-Part Rank
Random Forest Classifier with 10-fold Cross Validation
Subset Combination Method - 25/50/75 Percent
Result Analysis

Figure 6.4: Approach 2: 4-Part Rank SCM

		Number of features	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
	Base case	**42**	**97.93**	**2.0745**	**59.06**
Top 2	**SCM1**	**10**	**93.49**	**6.509**	**26.41**
Top 3	**SCM1**	**16**	**94.86**	**5.1389**	**32.23**
Top 4	**SCM1**	**20**	**94.79**	**5.2143**	**30.6**
Top 5	**SCM1**	**24**	**94.90**	**5.1025**	**33.27**
Top 6	**SCM1**	**25**	**94.91**	**5.0916**	**33.62**

Table 6.13: 25 Percent SCM, 4-part Rank Approach

		Number of features	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
	Base case	**42**	**97.93**	**2.0745**	**59.06**
Top 2	**SCM1**	**26**	**97.79**	**2.2081**	**37.27**
Top 3	**SCM1**	**34**	**97.92**	**2.0757**	**36.91**
Top 4	**SCM1**	**35**	**97.92**	**2.0818**	**37.77**
Top 5	**SCM1**	**36**	**97.90**	**2.0952**	**39.84**
Top 6	**SCM1**	**36**	**97.90**	**2.0952**	**39.84**

Table 6.14: 50 Percent SCM, 4-part Rank Approach

		Number of features	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
	Base case	**42**	**97.93**	**2.0745**	**59.06**
Top 2	**SCM1**	**31**	**97.69**	**2.3114**	**33.34**
Top 3	**SCM1**	**39**	**97.89**	**2.111**	**37.7**
Top 4	**SCM2**	**36**	**97.88**	**2.117**	**39.4**
Top 5	**SCM1**	**40**	**97.88**	**2.117**	**41.05**
Top 6	**SCM1**	**41**	**97.89**	**2.1073**	**42.26**

Table 6.15: 75 Percent SCM, 4-part Rank Approach

6.1.3 Approach 3 : 10-Part Rank

In this approach, we consider features in ranked sections of 10%, that is, we consider 10%, 20%, 30% and so on upto all of the features. We apply these on the IG, GR, COR, SU, RF and CS feature selection algorithms. The results for the same are shown below:

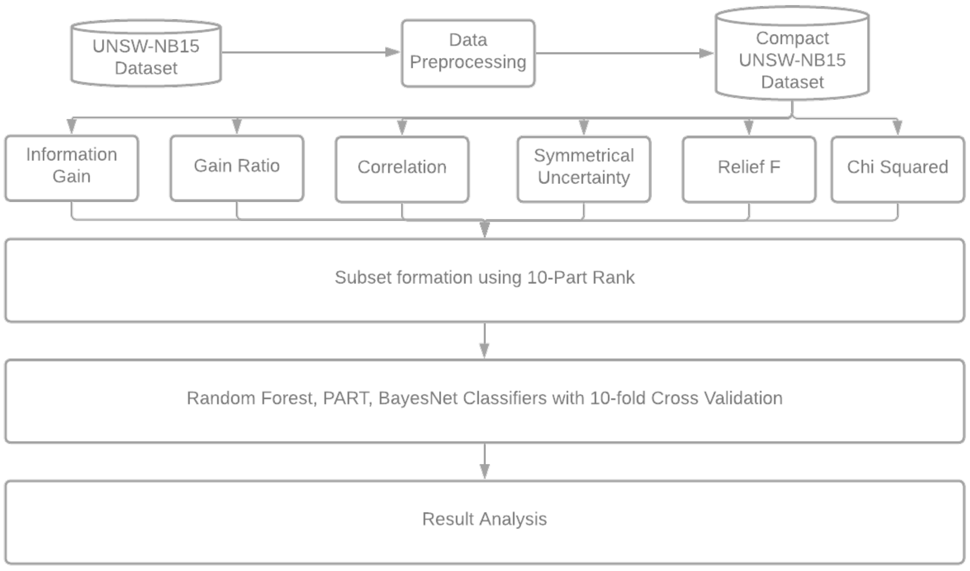

UNSW-NB15 Dataset
Data Preprocessing
Compact UNSW-NB15 Dataset
Information Gain
Gain Ratio
Correlation
Symmetrical Uncertainty
Relief F
Chi Squared
Subset formation using 10-Part Rank
Random Forest, PART, BayesNet Classifiers with 10-fold Cross Validation
Result Analysis

Figure 6.5: Approach 3: Normal 10-Part Rank

Method	**Percentage of all features**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
PART	**90**	**38**	**97.4129**	**2.5871**	**70.24**
Random Forest	**70**	**29**	**97.4481**	**2.5519**	**42.59**
BayesNet	**20**	**8**	**88.2609**	**11.7391**	**0.67**

Table 6.16: IG, 10-Part Rank Approach

Method	**Percentage of all features**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
Random Forest	**90**	**38**	**97.9182**	**2.0818**	**50.77**
PART	**90**	**38**	**97.4129**	**2.5871**	**70.24**
BayesNet	**50**	**21**	**84.7253**	**15.2747**	**2.12**

Table 6.17: GR, 10-Part Rank Approach

Method	Percentage of all features	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
PART	**90**	**38**	**97.4020**	**2.5980**	**68.76**
BayesNet	**90**	**38**	**81.4386**	**18.5614**	**3.64**
Random Forest	**50**	**21**	**97.1044**	**2.8956**	**39.5**

Table 6.18: COR, 10-Part Rank Approach

Method	Percentage of all features	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
PART	**90**	**38**	**97.4129**	**2.5871**	**92.52**
Random Forest	**80**	**34**	**97.5064**	**2.4936**	**52.2**
BayesNet	**40**	**17**	**86.3261**	**13.6739**	**1.75**

Table 6.19: SU, 10-Part Rank Approach

Method	Percentage of all features	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
PART	**90**	**37**	**97.2300**	**2.7700**	**73.81**
Random Forest	**40**	**17**	**97.9826**	**2.0174**	**25.27**
BayesNet	**20**	**8**	**82.3046**	**17.6954**	**0.54**

Table 6.20: RF, 10-Part Rank Approach

Method	Percentage of all features	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
PART	**90**	**38**	**97.4100**	**2.5900**	**70.01**
Random Forest	**70**	**29**	**97.4481**	**2.5519**	**34.78**
BayesNet	**20**	**8**	**88.1723**	**11.8277**	**0.7**

Table 6.21: CS, 10-part Rank Approach

Further, we take the best cases of 10-part rank from above, find the union and intersection of taking any 2 random feature selection algorithms at a time, and apply the classifiers on the same. We apply these on the IG, GR, COR, SU, RF and CS feature selection algorithms. This is termed the "Double Union Intersection" approach, the results of which are shown below:

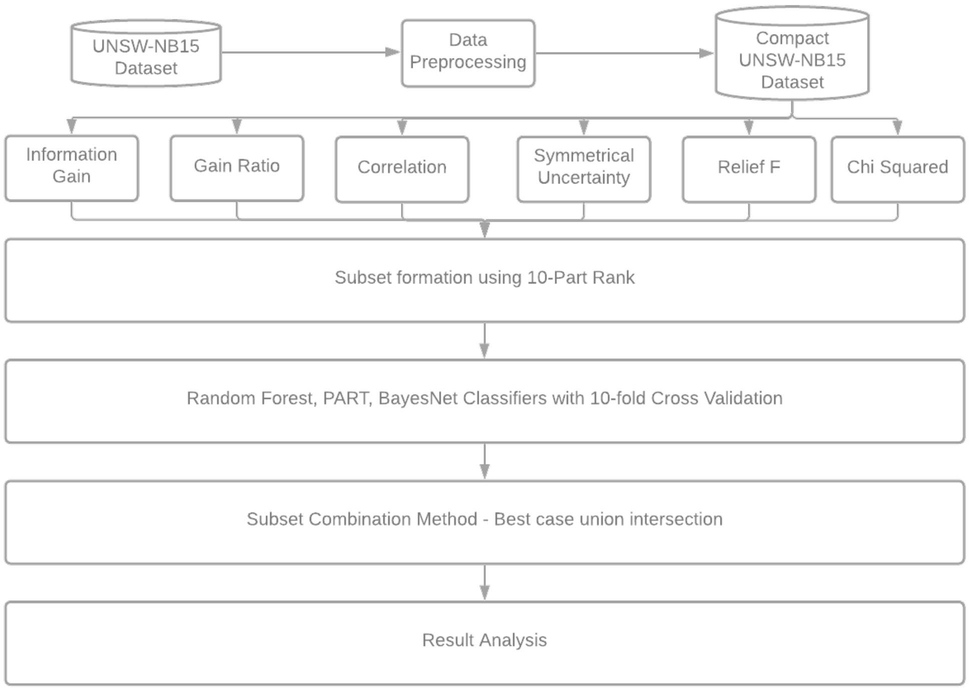

UNSW-NB15 Dataset
Data Preprocessing
Compact UNSW-NB15 Dataset
Information Gain
Gain Ratio
Correlation
Symmetrical Uncertainty
Relief F
Chi Squared
Subset formation using 10-Part Rank
Random Forest, PART, BayesNet Classifiers with 10-fold Cross Validation
Subset Combination Method - Best case union intersection
Result Analysis

Figure 6.6: Approach 3: 10 Part Rank SCM

Algorithm	**Combination of feature selection algorithms**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
PART	**IG Union GR**	**38**	**97.4129**	**2.5871**	**70.24**
Random Forest	**GR Intersection RF**	**17**	**97.9826**	**2.0174**	**25.51**
BayesNet	**IG Intersection SU**	**11**	**87.7107**	**12.2893**	**1.32**

Table 6.22: Double Union Intersection Approach

6.1.4 Approach 4 : Average Cases

In this approach, we find the average of all the weights of the feature selection algorithms' features, then consider only those features which have weight strictly greater than this value; this was performed only on tree-based and Bayes-based approaches. We apply these on the IG, GR, COR, SU, RF and CS feature selection algorithms. We call this the "Greater Than Average" approach, and the results for the same are shown below:

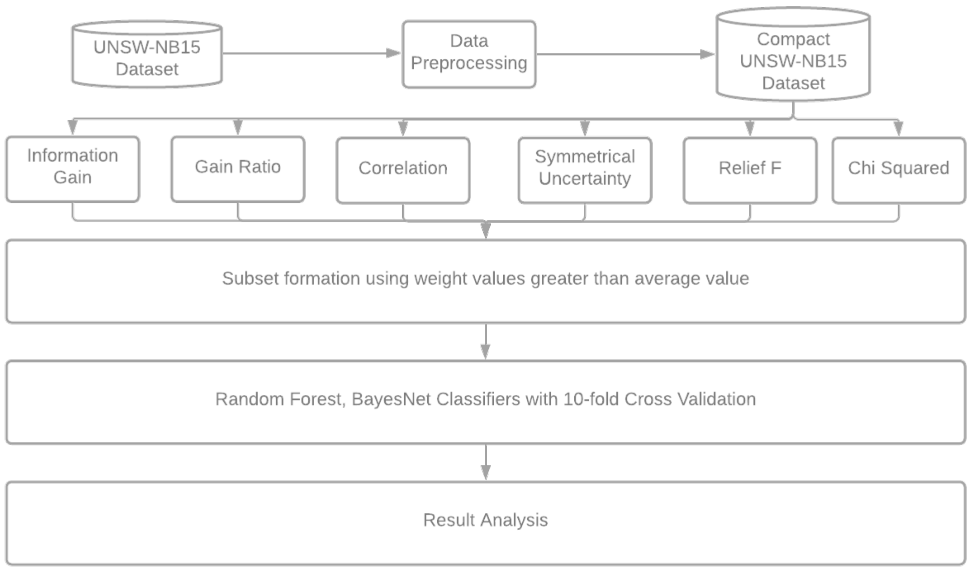

UNSW-NB15 Dataset
Data Preprocessing
Compact UNSW-NB15 Dataset
Information Gain
Gain Ratio
Correlation
Symmetrical Uncertainty
Relief F
Chi Squared
Subset formation using weight values greater than average value
Random Forest, BayesNet Classifiers with 10-fold Cross Validation
Result Analysis

Figure 6.7: Approach 4: Greater than Average

Method	Average Value	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
Chi Squared	**55458.53241**	**20**	**93.9793**	**6.0207**	**35.76**
Info Gain	**0.254492214**	**19**	**94.0157**	**5.9843**	**34.81**
Correlation	**0.176018905**	**19**	**97.0570**	**2.943**	**28.97**
Symmetrical Uncertainty	**0.141934476**	**18**	**93.9950**	**6.005**	**33.76**
Gain Ratio	**0.106205238**	**17**	**93.8238**	**6.1762**	**32.18**
Relief F	**0.018707844**	**15**	**97.8234**	**2.1766**	**20.16**

Table 6.23: Random Forest, Greater Than Average Approach

Method	Average Value	Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
Chi Squared	**55458.53241**	**20**	**84.7143**	**15.2857**	**2.37**
Info Gain	**0.254492214**	**19**	**84.9475**	**15.0525**	**1.92**
Correlation	**0.176018905**	**19**	**76.442**	**23.5583**	**3.37**
Symmetrical Uncertainty	**0.141934476**	**18**	**85.850**	**14.1500**	**2.25**
Gain Ratio	**0.106205238**	**17**	**84.5091**	**15.4909**	**1.96**
Relief F	**0.018707844**	**15**	**79.0810**	**20.9190**	**1.55**

Table 6.24: BayesNet, Greater Than Average Approach

Further, using the values obtained from the Greater Than Average approach, we use SCM to find accuracy of top 2, top 3, top 4, top 5 and all feature selection algorithms taken together. Called the ”Average SCM” approach, the results for them are shown below:

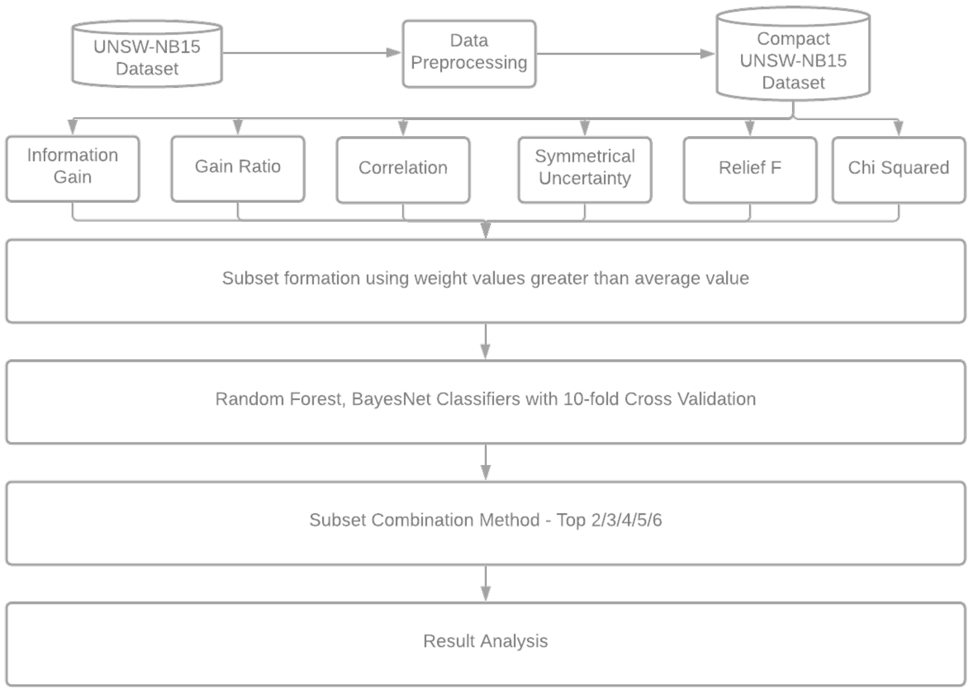

UNSW-NB15 Dataset
Data Preprocessing
Compact UNSW-NB15 Dataset
Information Gain
Gain Ratio
Correlation
Symmetrical Uncertainty
Relief F
Chi Squared
Subset formation using weight values greater than average value
Random Forest, BayesNet Classifiers with 10-fold Cross Validation
Subset Combination Method - Top 2/3/4/5/6
Result Analysis

Figure 6.8: Approach 4: Average SCM

Method	**SCM Case**	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
All	**SCM1**	**35**	**97.8684**	**2.1316**	**39.29**
	SCM2	**28**	**97.6850**	**2.315**	**32.02**
	SCM3	**18**	**93.9950**	**6.005**	**31.73**
	SCM4	**17**	**93.9574**	**6.0426**	**30.57**
	SCM5	**6**	**87.1581**	**12.8419**	**17.92**
	SCM6	**4**	**86.6820**	**13.318**	**16.45**
Top 5 - IG, COR, SU, RF, CS	**SCM1**	**35**	**97.8684**	**2.1316**	**39.29**
	SCM2	**27**	**97.7214**	**2.2786**	**30.38**
	SCM3	**17**	**93.9574**	**6.0426**	**30.57**
	SCM4	**8**	**93.0392**	**6.9608**	**21.67**
	SCM5	**4**	**86.6820**	**13.318**	**16.45**
Top 4 - IG, COR, SU, RF	**SCM1**	**34**	**97.8720**	**2.1280**	**36.94**
	SCM2	**24**	**97.7736**	**2.2264**	**32.73**
	SCM3	**8**	**92.6286**	**7.3714**	**31.59**
	SCM4	**4**	**86.6820**	**13.3180**	**18.21**
Top 3 - IG, COR, RF	**SCM1**	**34**	**97.8720**	**2.1280**	**38.71**
	SCM2	**15**	**97.6971**	**2.3029**	**21.67**
	SCM3	**4**	**86.6820**	**13.3180**	**18.21**
Top 2 - COR, RF	**Union**	**23**	**97.8927**	**2.1073**	**28.61**
	Intersection	**11**	**96.0635**	**3.9365**	**21.67**

Table 6.25: Average SCM Approach

6.1.5 Approach 5 : Top Absolute and Traditional Systems

We initially take the top 5, top 10, top 15, top 20 and top 25 features of each feature selection algorithm, in terms of their weight values. We then use SCM to find the accuracy of the top 2, top 3, top 4, top 5 and all algorithms for each case. The results for this "Top Absolute" method are shown below:

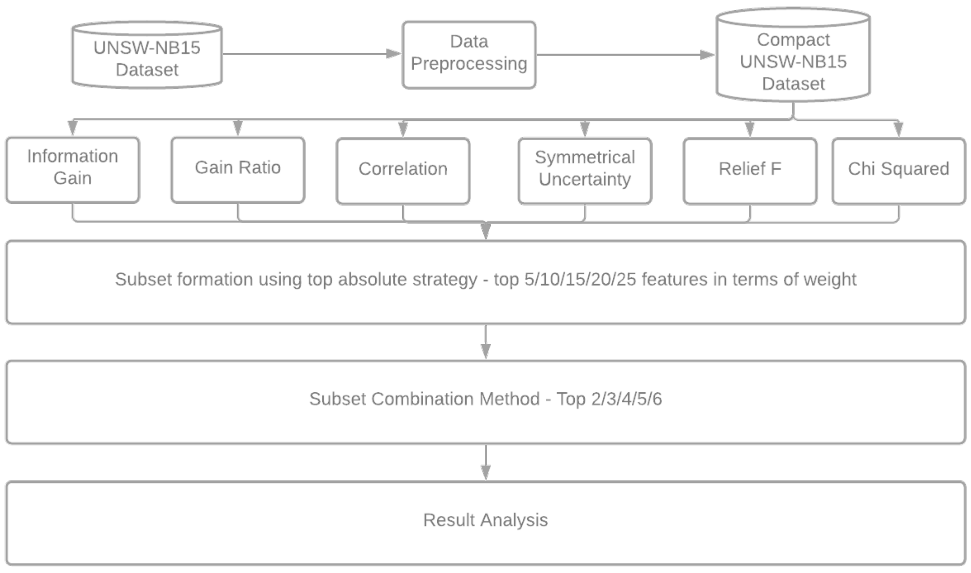

UNSW-NB15 Dataset
Data Preprocessing
Compact UNSW-NB15 Dataset
Information Gain
Gain Ratio
Correlation
Symmetrical Uncertainty
Relief F
Chi Squared
Subset formation using top absolute strategy - top 5/10/15/20/25 features in terms of weight
Subset Combination Method - Top 2/3/4/5/6
Result Analysis

Figure 6.9: Approach 5: Top Absolute

Top 5		**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
	Base case	**42**	**97.9255**	**2.0745**	**59.06**
Top 2	**SCM1**	**5**	**92.6663**	**7.3337**	**12.72**
	SCM2	**5**	**92.6663**	**7.3337**	**12.72**
Top 3	**SCM1**	**8**	**93.5153**	**6.4847**	**16.29**
	SCM2	**5**	**92.6663**	**7.3337**	**12.72**
	SCM3	**2**	**77.5774**	**22.4226**	**4.39**
Top 4	**SCM1**	**10**	**93.5019**	**6.4981**	**17.95**
	SCM2	**5**	**92.6663**	**7.3337**	**12.72**
	SCM3	**3**	**81.7422**	**18.2578**	**5.75**
	SCM4	**2**	**77.5774**	**22.4226**	**4.39**
Top 5	**SCM1**	**12**	**93.8724**	**6.1276**	**19.32**
	SCM2	**5**	**92.6663**	**7.3337**	**12.72**
	SCM3	**3**	**81.7422**	**18.2578**	**5.75**
	SCM4	**3**	**81.7422**	**18.2578**	**5.75**
	SCM5	**2**	**77.5774**	**22.4226**	**4.39**
All	**SCM1**	**13**	**94.1614**	**5.8386**	**22.18**
	SCM2	**7**	**92.8363**	**7.1637**	**14**
	SCM3	**3**	**81.7422**	**18.2578**	**5.75**
	SCM4	**3**	**81.7422**	**18.2578**	**5.75**
	SCM5	**3**	**81.7422**	**18.2578**	**5.75**
	SCM6	**1**	**76.6324**	**23.3676**	**2.28**

Table 6.26: Top 5, Top Absolute Approach

Top 10		**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
	Base case	**42**	**97.9255**	**2.0745**	**59.06**
Top 2	**SCM1**	**16**	**94.8611**	**5.1389**	**28.07**
	SCM2	**4**	**90.3209**	**9.6791**	**10.56**
Top 3	**SCM1**	**16**	**94.8611**	**5.1389**	**28.07**
	SCM2	**10**	**93.4910**	**6.509**	**24.18**
	SCM3	**4**	**90.3209**	**9.6791**	**10.56**
Top 4	**SCM1**	**23**	**94.9181**	**5.0819**	**31.77**
	SCM2	**10**	**93.4910**	**6.509**	**24.18**
	SCM3	**4**	**90.3209**	**9.6791**	**10.56**
	SCM4	**3**	**81.7422**	**18.2578**	**6.56**
Top 5	**SCM1**	**23**	**94.9181**	**5.0819**	**31.77**
	SCM2	**16**	**93.9465**	**6.0535**	**32.27**
	SCM3	**5**	**92.2970**	**7.703**	**10.56**
	SCM4	**3**	**81.7422**	**18.2578**	**6.56**
	SCM5	**3**	**81.7422**	**18.2578**	**6.56**
All	**SCM1**	**25**	**94.9084**	**5.0916**	**30.62**
	SCM2	**18**	**94.1493**	**5.8507**	**30.7**
	SCM3	**9**	**93.1145**	**6.8855**	**18.92**
	SCM4	**3**	**81.7422**	**18.2578**	**6.56**
	SCM5	**3**	**81.7422**	**18.2578**	**6.56**
	SCM6	**2**	**81.7349**	**18.2651**	**4.22**

Table 6.27: Top 10, Top Absolute Approach

Top 15		Number of features	Accuracy (%)	Incorrectly classified instances (%)	Time taken to build model (seconds)
	Base case	42	97.9255	2.0745	59.06
Top 2	SCM1	21	97.8514	2.1486	26.37
	SCM2	9	95.2351	4.7649	22.89
Top 3	SCM1	29	97.9097	2.0903	30.93
	SCM2	12	96.8372	3.1628	21.98
	SCM3	4	86.6820	13.318	17.37
Top 4	SCM1	28	97.8769	2.1231	29.29
	SCM2	19	96.7995	3.2005	27.34
	SCM3	7	92.9639	7.0361	18.87
	SCM4	4	86.6820	13.318	17.37
Top 5	SCM1	32	97.9133	2.0867	36.4
	SCM2	21	96.8226	3.1774	27.59
	SCM3	12	93.9720	6.028	22.21
	SCM4	6	87.1581	12.8419	18.2
	SCM5	4	86.6820	13.318	17.37
Top 6	SCM1	32	97.9133	2.0867	36.4
	SCM2	23	96.7874	3.2126	27.98
	SCM3	14	93.8748	6.1252	23.48
	SCM4	11	93.8189	6.1811	22.54
	SCM5	6	87.1581	12.8419	18.2
	SCM6	4	86.6820	13.318	17.37

Table 6.28: Top 15, Top Absolute Approach

Top 20		**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
	Base case	**42**	**97.9255**	**2.0745**	**59.06**
Top 2	**SCM1**	**24**	**97.8149**	**2.1851**	**29.49**
	SCM2	**16**	**96.3538**	**3.6462**	**28.38**
Top 3	**SCM1**	**34**	**97.9243**	**2.0757**	**39.05**
	SCM2	**19**	**97.5866**	**2.4134**	**29.16**
	SCM3	**7**	**87.4156**	**12.5844**	**18.89**
Top 4	**SCM1**	**36**	**97.9048**	**2.0952**	**36.98**
	SCM2	**29**	**97.6425**	**2.3575**	**30.3**
	SCM3	**10**	**93.3307**	**6.6693**	**21.09**
	SCM4	**5**	**87.1277**	**12.8723**	**17.01**
Top 5	**SCM1**	**36**	**97.9048**	**2.0952**	**36.98**
	SCM2	**30**	**97.6595**	**2.3405**	**31.59**
	SCM3	**20**	**93.9210**	**6.079**	**31.16**
	SCM4	**9**	**93.3379**	**6.6621**	**20.11**
	SCM5	**5**	**87.1277**	**12.8723**	**17.01**
	SCM1	**36**	**97.9048**	**2.0952**	**36.98**
	SCM2	**31**	**97.6850**	**2.315**	**31.1**
Top 6 multirow2***Top 2**	**SCM1**	**29**	**97.7773%**	**1830 (2.2227%)**	**33.25**
	SCM2	**21**	**96.9866%**	**2481 (3.0134%)**	**30.23**
Top 3	**SCM1**	**37**	**97.849%**	**1771 (2.151%)**	**36.51**
	SCM2	**25**	**97.5635%**	**2006 (2.4365%)**	**29.96**
	SCM3	**13**	**92.8448%**	**5891 (7.1552%)**	**23.51**
Top 4	**SCM1**	**37**	**97.849%**	**1771 (2.151%)**	**36.51**
	SCM2	**34**	**97.685%**	**1906 (2.315%)**	**36.45**
	SCM3	**17**	**93.7606%**	**5137 (6.2394%)**	**35.8**
	SCM4	**12**	**92.6189%**	**6077**	**25.52**

We also applied our respective classifiers against two traditional approaches - the Opeyemi approach, and the Aakash-Deep approach. The results for the same are listed below:

Method	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
PART	**10**	**97.3000**	**2.7000**	**50.72**
Random Forest	**10**	**93.7461**	**6.2539**	**24.49**
BayesNet	**10**	**86.9398**	**13.0617**	**0.78**

Table 6.30: [19] Traditional Approaches

Method	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
Random Forest	**28**	**97.5064**	**2.4936**	**32.88**
BayesNet	**28**	**83.0528**	**16.9472**	**2.73**
PART	**28**	**96.9672**	**3.0328**	**43.18**

Table 6.31: [20] Traditional Approaches

6.1.6 Level 1 Best Results

From all the above approaches, the best results are shown below:

Method	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
Random Forest	**17**	**97.9826**	**2.0174**	**25.27**
PART	**38**	**97.4129**	**2.5871**	**70.24**
BayesNet (Info Gain, Ranked)	**8**	**88.2609**	**11.7391**	**0.67**

Table 6.32: Best Results of Each Classifier

From the table, we see that Random Forest gives the best accuracy. However, the BayesNet method is able to reduce the features to the greatest extent (while keeping build time minimum). Hence, we choose to take the reduced features of BayesNet for further computation. To improve the accuracy further, we go to the 2nd level of the IDS.

6.2 Level 2: Binary and Multilabel Classification With Random Forest

6.2.1 Binary Classification Analysis

In the 2nd level, we take the reduced feature subset obtained from the Info Gain Ranked BayesNet feature selection method above, and apply Random Forest classifier on it. The results obtained are shown below:

Method	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
InfoGain (Ranked)	**8**	**93.4971**	**6.5029**	**22.21**

Table 6.33: Best Results of Random Forest, Level 2 Binary

6.2.2 Multilabel Classification Analysis

We also apply the classifier on multilabel case, using all 42 features as well as the 8 feature subset above. The results for the same are shown below:

Method	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
All Features	**42**	**86.9540 %**	**13.046**	**75.6**
InfoGain (Ranked)	**8**	**83.9078**	**16.0922**	**38.49**

Table 6.34: Best Results of Random Forest, Level 2 Multilabel

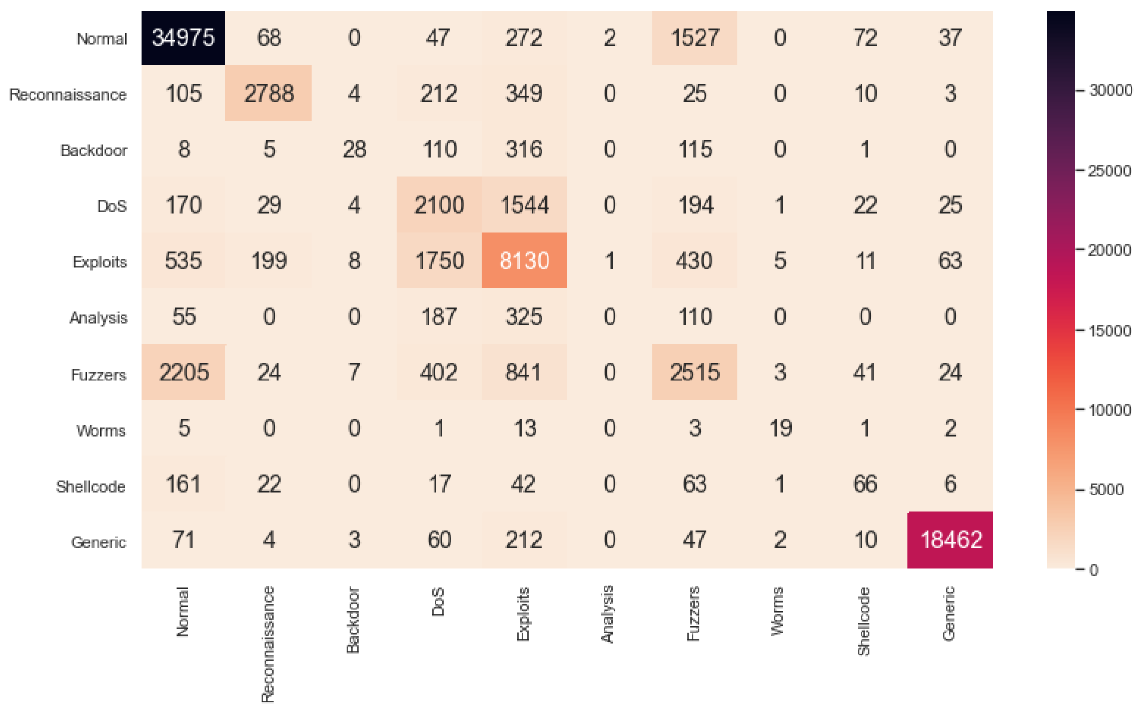
TP Rate	**FP Rate**	**Precision**	**Recall**	**F-Measure**	**MCC**	**ROC Area**	**PRC Area**	**Class**
0.945	**0.073**	**0.913**	**0.945**	**0.929**	**0.870**	**0.985**	**0.977**	**Normal**
0.797	**0.004**	**0.888**	**0.797**	**0.840**	**0.835**	**0.982**	**0.900**	**Reconnaissance**
0.048	**0.000**	**0.519**	**0.048**	**0.088**	**0.156**	**0.942**	**0.149**	**Backdoor**
0.514	**0.036**	**0.430**	**0.514**	**0.468**	**0.439**	**0.941**	**0.446**	**DoS**
0.730	**0.055**	**0.675**	**0.730**	**0.702**	**0.653**	**0.965**	**0.837**	**Exploits**
0.000	**0.000**	**0.000**	**0.000**	**0.000**	**NA**	**0.942**	**0.097**	**Analysis**
0.415	**0.033**	**0.500**	**0.415**	**0.454**	**0.417**	**0.910**	**0.500**	**Fuzzers**
0.432	**0.000**	**0.613**	**0.432**	**0.507**	**0.514**	**0.954**	**0.545**	**Worms**
0.175	**0.002**	**0.282**	**0.175**	**0.216**	**0.219**	**0.868**	**0.177**	**Shellcode**
0.978	**0.003**	**0.991**	**0.978**	**0.985**	**0.980**	**0.997**	**0.995**	**Generic**

Table 6.35: Random Forest InfoGain Multilabel 20% Detailed Accuracy

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.989	**0.031**	**0.963**	**0.989**	**0.976**	**0.955**	**0.998**	**0.997**	**Normal**
0.802	**0.003**	**0.927**	**0.802**	**0.860**	**0.857**	**0.960**	**0.896**	**Reconnaissance**
0.022	**0.000**	**0.394**	**0.022**	**0.042**	**0.092**	**0.877**	**0.085**	**Backdoor**
0.375	**0.031**	**0.384**	**0.375**	**0.380**	**0.348**	**0.939**	**0.393**	**DoS**
0.710	**0.063**	**0.638**	**0.710**	**0.672**	**0.619**	**0.967**	**0.838**	**Exploits**
0.080	**0.000**	**0.818**	**0.080**	**0.145**	**0.254**	**0.811**	**0.132**	**Analysis**
0.675	**0.025**	**0.682**	**0.675**	**0.679**	**0.653**	**0.957**	**0.822**	**Fuzzers**
0.091	**0.000**	**0.571**	**0.091**	**0.157**	**0.228**	**0.974**	**0.260**	**Worms**
0.484	**0.002**	**0.570**	**0.484**	**0.524**	**0.523**	**0.984**	**0.495**	**Shellcode**
0.976	**0.001**	**0.996**	**0.976**	**0.985**	**0.981**	**0.999**	**0.997**	**Generic**

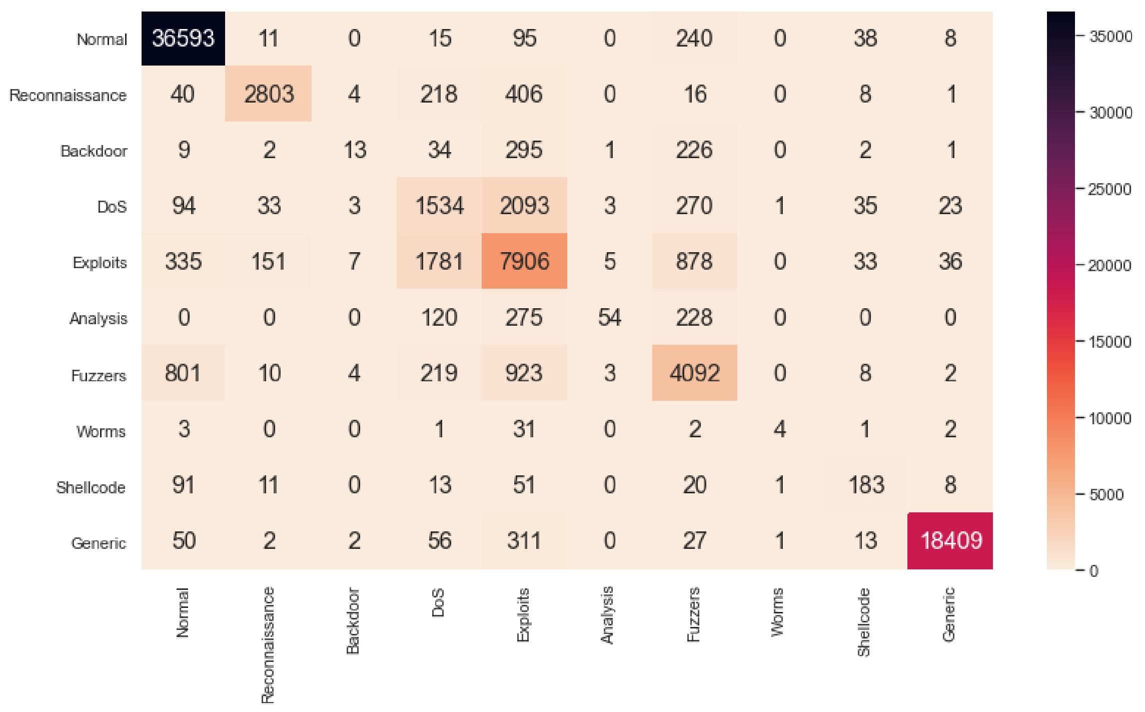
Table 6.36: Random Forest InfoGain Multilabel All Features Detailed Accuracy

6.2.3 Confusion Matrices



	Normal	Reconnaissance	Backdoor	DoS	Exploits	Analysis	Fuzzers	Worms	Shellcode	Generic
Normal	34975	68	0	47	272	2	1527	0	72	37
Reconnaissance	105	2788	4	212	349	0	25	0	10	3
Backdoor	8	5	28	110	316	0	115	0	1	0
DoS	170	29	4	2100	1544	0	194	1	22	25
Exploits	535	199	8	1750	8130	1	430	5	11	63
Analysis	55	0	0	187	325	0	110	0	0	0
Fuzzers	2205	24	7	402	841	0	2515	3	41	24
Worms	5	0	0	1	13	0	3	19	1	2
Shellcode	161	22	0	17	42	0	63	1	66	6
Generic	71	4	3	60	212	0	47	2	10	18462

Figure 6.10: Confusion Matrix Heatmap : Random Forest InfoGain Multilabel 20%



	Normal	Reconnaissance	Backdoor	DoS	Exploits	Analysis	Fuzzers	Worms	Shellcode	Generic
Normal	36593	11	0	15	95	0	240	0	38	8
Reconnaissance	40	2803	4	218	406	0	16	0	8	1
Backdoor	9	2	13	34	295	1	226	0	2	1
DoS	94	33	3	1534	2093	3	270	1	35	23
Exploits	335	151	7	1781	7906	5	878	0	33	36
Analysis	0	0	0	120	275	54	228	0	0	0
Fuzzers	801	10	4	219	923	3	4092	0	8	2
Worms	3	0	0	1	31	0	2	4	1	2
Shellcode	91	11	0	13	51	0	20	1	183	8
Generic	50	2	2	56	311	0	27	1	13	18409

Figure 6.11: Confusion Matrix Heatmap : Random Forest InfoGain Multilabel All Features

6.2.4 Comparison with Traditional Systems

Lastly, we compare the proposed 2 level IDS with traditional systems. The comparative results for the same are shown below:

Reference	**Number of features**	**Accuracy (%)**	**Incorrectly classified instances (%)**	**Time taken to build model (seconds)**
Proposed Model Best Result	**8**	**93.4971**	**6.5029**	**22.21**
[5]	**6**	**82.1394**	**17.8606**	**20.68**
[6]	**23**	**97.9060**	**2.0940**	**34.5**
[7]	**7**	**92.7999**	**7.2001**	**19.36**
[10]	**19**	**98.0506**	**1.9494**	**29.64**
[12]	**13**	**97.9680**	**2.0320**	**22.64**
[14]	**29**	**97.9461**	**2.0539**	**37.47**
[15]	**6**	**93.1655**	**6.8345**	**17.35**
[19]	**10**	**93.7461**	**6.2539**	**24.49**
[20]	**28**	**97.5064**	**2.4936**	**32.88**

Table 6.37: Traditional Methods Comparison

Chapter 7

Conclusion

This project proposes an Intrusion Detection System model which utilizes UNSW-NB15 dataset with a set of 45 features. It is chosen since it contains a recent set of attacks as compared to KDD'99 and NSL-KDD dataset. For the detection of web attacks, the project makes use of various Machine Learning feature reduction and classification methods. The proposed model evaluates the performance by using various classifiers such as BayesNet, Random Forest and PART along with the feature selection methods such as Information Gain, Gain Ratio, Correlation, Symmetrical Uncertainty, Chi Squared, and Relief F. Among the three classifiers when run on testing dataset, BayesNet gives the minimum build time as compared to others. While, Random Forest gives the best accuracy amongst three but with high build time.

Initially, we apply data preprocessing steps and reduce the 45 features of the raw dataset to 42 features of the compact dataset. Then, various approaches such as Threshold, 4-Part Rank, 10-Part Rank are applied. Thereafter, other approaches like Subset Combination Method is carried out on the earlier approaches, also approaches such as Average SCM, Greater than Average, Top Absolute are performed which reduced the features further to some extent. At the end, the smallest subset of features is obtained by reducing the dataset to a minimal 8 features, using BayesNet classifier.

The 8-feature subset from BayesNet when applied on a Random Forest binary classifier gives really good accuracy of 93.4971% with build time of 22.21 seconds, while when applied on a Random Forest multilabel classifier gives us an accuracy value of 83.9078% with a build time of 38.49 seconds. When Random Forest multilabel classifier is applied on all features, it gives us an accuracy of 86.9540% with a build time of 75.6 seconds.

Chapter 8

Research Paper Publication

The research paper titled "**Web Attack Detection Using Machine Learning**" has been submitted by Ruturaj Malavade, Heena Jamadar, Harshali Upadhye, Deepak Kshirsagar, Jagannath Aghav and is under review in the conference "**3RD INTERNATIONAL CONFERENCE ON DATA, ENGINEERING AND APPLICATIONS 2021 (IDEA 2k21)**".

The paper after acceptance will be published in **Springer Proceedings** and will be submitted to "**Lecture Notes in Electrical Engineering**" indexed in **SCOPUS** for publication.

Bibliography

[1] Ahmed Ahmim, Leandros Maglaras, Mohamed Amine Ferrag, Makhlouf Derdour, and Helge Janicke. A novel hierarchical intrusion detection system based on decision tree and rules-based models. In *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 228–233. IEEE, 2019.

[2] Sikha Bagui, Ezhil Kalaimannan, Subhash Bagui, Debarghya Nandi, and Anthony Pinto. Using machine learning techniques to identify rare cyber-attacks on the unsw-nb15 dataset. *Security and Privacy*, 2(6):e91, 2019.

[3] Sarika Choudhary and Nishtha Kesswani. Analysis of kdd-cup'99, nsl-kdd and unsw-nb15 datasets using deep learning in iot. *Procedia Computer Science*, 167:1561–1573, 2020.

[4] Hossein Gharaee and Hamid Hosseinvand. A new feature selection ids based on genetic algorithm and svm. In *2016 8th International Symposium on Telecommunications (IST)*, pages 139–144. IEEE, 2016.

[5] Florian Gottwalt, Elizabeth Chang, and Tharam Dillon. Corrcorr: A feature selection method for multivariate correlation network anomaly detection techniques. *Computers & Security*, 83:234–245, 2019.

[6] Anwar Husain, Ahmed Salem, Carol Jim, and George Dimitoglou. Development of an efficient network intrusion detection model using extreme gradient boosting

(xgboost) on the unsw-nb15 dataset. In *2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pages 1–7. IEEE, 2019.

[7] Tharmini Janarthanan and Shahrzad Zargari. Feature selection in unsw-nb15 and kddcup'99 datasets. In *2017 IEEE 26th international symposium on industrial electronics (ISIE)*, pages 1881–1886. IEEE, 2017.

[8] Meiko Jensen, Nils Gruschka, and Ralph Herkenhöner. A survey of attacks on web services. *Computer Science-Research and Development*, 24(4):185, 2009.

[9] Dishan Jing and Hai-Bao Chen. Svm based network intrusion detection for the unsw-nb15 dataset. In *2019 IEEE 13th International Conference on ASIC (ASICON)*, pages 1–4. IEEE, 2019.

[10] Sydney M Kasongo and Yanxia Sun. Performance analysis of intrusion detection systems using a feature selection method on the unsw-nb15 dataset. *Journal of Big Data*, 7(1):1–20, 2020.

[11] Deepak Kshirsagar and Sandeep Kumar. Identifying reduced features based on ig-threshold for dos attack detection using part. In *International Conference on Distributed Computing and Internet Technology*, pages 411–419. Springer, 2020.

[12] Vikash Kumar, Ditipriya Sinha, Ayan Kumar Das, Subhash Chandra Pandey, and Radha Tamal Goswami. An integrated rule based intrusion detection system: analysis on unsw-nb15 data set and the real time online dataset. *Cluster Computing*, 23(2):1397–1418, 2020.

[13] Ishfaq Manzoor, Neeraj Kumar, et al. A feature reduced intrusion detection system using ann classifier. *Expert Systems with Applications*, 88:249–257, 2017.

[14] Nour Moustafa and Jill Slay. The significant features of the unsw-nb15 and the kdd99 data sets for network intrusion detection systems. In *2015 4th international workshop on building analysis datasets and gathering experience returns for security (BADGERS)*, pages 25–31. IEEE, 2015.

[15] Nour Moustafa and Jill Slay. The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set. *Information Security Journal: A Global Perspective*, 25(1-3):18–31, 2016.

[16] Nour Moustafa, Jill Slay, and Gideon Creech. Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks. *IEEE Transactions on Big Data*, 5(4):481–494, 2017.

[17] Nour Moustafa, Benjamin Turnbull, and Kim-Kwang Raymond Choo. An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things. *IEEE Internet of Things Journal*, 6(3):4815–4830, 2018.

[18] Opeyemi Osanaiye, Haibin Cai, Kim-Kwang Raymond Choo, Ali Dehghantanha, Zheng Xu, and Mqhele Dlodlo. Ensemble-based multi-filter feature selection method for ddos detection in cloud computing. *EURASIP Journal on Wireless Communications and Networking*, 2016(1):1–10, 2016.

[19] Opeyemi Osanaiye, Haibin Cai, Kim-Kwang Raymond Choo, Ali Dehghantanha, Zheng Xu, and Mqhele E. Dlodlo. Ensemble-based multi-filter feature selection method for ddos detection in cloud computing. *EURASIP Journal on Wireless Communications and Networking*, 2016, 05 2016.

[20] Akashdeep Sharma, Ishfaq Manzoor, and Neeraj Kumar. A feature reduced intrusion detection system using ann classifier. *Expert Systems with Applications*, 88, 07 2017.