EVALUATING OVERSAMPLING TECHNIQUES FOR NETWORK INTRUSION DETECTION DATA

CHAN JIA LIN

UNIVERSITI TUNKU ABDUL RAHMAN

EVALUATING OVERSAMPLING TECHNIQUES FOR NETWORK INTRUSION DETECTION DATA

CHAN JIA LIN

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Science (Hons.) Software Engineering

Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

MAY 2022

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature	:	vicrace
Name	:	Chan Jia Lin
ID No.	:	1902879
Date	:	21/4/2022

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"EVALUATING OVERSAMPLING TECHNIQUES FOR NETWORK INTRUSION DETECTION DATA"** was prepared by **CHAN JIA LIN** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (Honours) Software Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature : 

Supervisor : Dr. Khor Kok Chin

Date : 21/4/2022

ABSTRACT

In this digital era, the amount of information being exchanged over the networks has increased exponentially due to technological advancement. Thus, cyber-attacks have increased in tandem with the exponential expansion of digitalisation worldwide. As a result, implementing an IDS is one of the approaches to overcome the security problem in the network. Many network intrusion data sets are introduced and used as a benchmark to train predictive models and evaluate the IDS. However, the unbalanced class distribution in network intrusion data sets has become a significant challenge in building classification models, leading to low intrusion detection rates (DR). This research identified four unbalanced network intrusion detection data sets: UNSW-NB15, NSL KDD, CICIDS2017, and CICDDOS 2019, with low detection rates in minority attack classes. Five oversampling techniques: ROS, SMOTE, Borderline SMOTE (BSMOTE), ADASYN and K-Mean SMOTE (KMSMOTE), were then applied to the minority attack classes in the datasets. Eventually, models, i.e., Gaussian Bayes, Logistic Regression and Decision Tree, were built using the data sets, and the model performance was compared. According to the analysis, each data set has a different oversampling method that outperforms. KMSMOTE outperforms in UNSW-NB15, ROS excels in NSL KDD, and SMOTE outperforms in CICIDS 2017 and CICDDOS 2019, while SMOTE has the highest number of top-performing occurrences among all data sets. In general, oversampling can increase the detection rate (DR) for the minority attack classes, the DR increment ranging from 11.93 % in CICDDOS 2019 to a maximum of 20.02 % in NSL KDD.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

ADA	ADA Boost
ADASYN	Adaptive Synthetic Sampling Technique
ANN	Artificial Neural Network
ARM	Association Rule Mining
AUC	Area Under the Curve
CFS	Correlation-based Feature Selection
CIC	Canadian Institution of Cyber Security
CNBC	Consumer News and Business Channel
CNN	Convolutional Neural Network
CRISP-DM	Cross-industry standard process for Data Mining
CSARF	Cost sensitive Adaptive Random Forest
CVS	Comma-separated value
DDoS	Distributed Denial of Service
DNN	Deep Neural Network
DNS	Domain Name System
DoS	Denial of Service
DR	Detection Rate
DT	Decision Tree
EML	Ensemble Machine Learning
FAR	False Alarm Rate
FN	False Negative
FP	False Positive
FPR	False Positive Rate
FTP	File Transfer Protocol
GB	Gaussian Bayes
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IMAP	Internet Message Access Protocol
IP	Internet Protocol

IR	Imbalanced Ratio
ISCX	Installation Support Center of Expertise
KNN	K Nearest Neighbour
LDA	Linear Discriminant Analysis
LIBSVM	Library for Support Vector Machine
LR	Logistic Regression
MCC	Mathews Correlation Coefficient
MLP	Multilayer Perceptron
MLP	Multilayer Perceptron
MWMOTE	Majority Weighted Over Sampling Technique
POP3	Post Office Protocol 3
R2L	Remote to Local
ROC	Receiving Operating Characteristic Curve
ROS	Random Over Sampling
ROTE	Random Over-Sampling Technique
RTT	Round Trip Time
SMOTE	Synthetic Minority Over-Sampling Technique
SMTP	Simple Mail Transfer Protocol
SOM	Self Organisation Map Artificial Neural Network
SQL	Structured Query Language
SSH	Secure Shell
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TN	True Negative
TP	True Positive
U2R	User to Root
UDP	User Datagram Protocol
XSS	Cross Site Scripting

A LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

In this digital era, the growth of the internet and technologies provide convenience for users to exchange information with limited constraints. It has increased the amount of information being exchanged over the networks exponentially. The enormous increase in the utilisation of the network and internet has posed a risk to internet security. This is because not everyone utilises it for functional purposes but to exploit it for their benefit. Hence, cybersecurity has become a major concern, and IDS is a method to overcome the security issue by monitoring and analysing the data to recognise intruders in the computer networks (Aziz and Ahmad, 2021). Network intrusion data sets such as UNSW-NB15, CICIDS 2017, ISCX IDS 2012, and kyoto2006 are benchmarks to train predictive models and evaluate the IDS (Hindy et al., 2020)

Most network intrusion data sets are highly unbalanced in class distribution, one ubiquitous challenge in machine learning. A highly unbalanced class distribution is defined by a rare-to-dominant ratio starting from 100:1 and above (Leevy and Khoshgoftaar, 2020). In those observed data sets, the size of benign traffic outweighs malicious traffic. The low presentation of minority class(es) might cause the classifiers to fail to identify the attack types or the target effectively, leading to a low detection rate for the attack traffic. Therefore, few approaches have been introduced to mitigate the unbalanced class problem.

The algorithm-level approach is to modify the existing algorithm to mitigate the bias toward majority classes. The data-level approach modifies the data to balance the class distributions for standard learning algorithms such as oversampling and under sampling methods (Krawczyk, 2016). Oversampling methods are the data-level approach that increases the size of minority class(es) until the data set reaches a balance distribution. The oversampling method has the problem of overfitting, where the data are duplicated directly from the minority data. To overcome this issue, a few advanced oversampling techniques such as SMOTE, Borderline-SMOTE, and ADASYN were introduced.

This research shall evaluate a few over-sampling methods on network intrusions data sets that are suffered from unbalanced class distribution to find out which techniques generally perform well for the data sets.

1.1 Problem Statement

This section discussed the challenging faced in most intrusion detection systems (IDS). Throughout the investigation, unbalanced class distribution always a concern in most network intrusion data sets for IDS evaluation. It is because the accuracy of a classifier will be biased towards the majority classes, causing performance to be biased. The classifiers may achieve high overall accuracy, but the detection rate for the informative minority attack classes remains low.

1.1.1 The unbalanced class distribution in data sets

A sufficient sample in one data set is vital for learning classifiers to produce a high accuracy predictive model in machine learning. However, unbalanced class distribution is a compelling and inherent problem in the network intrusion data sets. Typically, the malicious network traffic represents a small proportion of all network traffic, yet this minority data is important for the health of the network. CICIDS 2017, for instance, contains 2.8 million data, with only 19.7 % of the total number are attack traffic. From a granular view, CICIDS 2017 also has an unbalanced class distribution among the attack traffic (Leevy and Khoshgoftaar, 2020). This unbalanced distribution will cause inappropriate inductive bias and inefficiency in extracting minority class(es) features. Eventually, the accuracy and probability measures implemented by learning classifiers will skew due to the unbalanced data set. Therefore, over-sampling techniques shall be implemented to mitigate the unbalanced class problem in the network intrusion data set.

1.1.2 Low detection rates for intrusions caused by unbalanced data sets

The purpose of implementing data mining techniques in the network intrusion data set is to obtain a satisfactory detection rate for intrusions to secure the network. Nevertheless, the unbalanced class distribution problem on attack class(es) has caused a low detection rate on the intrusion class. This will have the classifier recognise benign traffic as it is and might also recognise attack traffic as benign.

Typically, in multiclassification, if a certain attack type consists of a smaller subset among all the attack traffics, these minority attack traffic may be recognised as benign traffic due to insufficient samples in the data sets for reference. When a network traffic data set is used for intrusion detection, correctly identifying the minority attack traffic is more important than identifying the majority benign traffic.

1.2 Research Question

The following are the questions that this research is aims to address:

1. Which network intrusion data sets suffered from an unbalanced class distribution problem?
2. Which oversampling technique can detect the intrusion effectively for the chosen data sets?

1.3 Project Objectives

The following are the research objectives:

1. To identify network intrusion data sets suffered from the unbalanced class distribution problem with low detection rates for minority classes.
2. To implement oversampling techniques and evaluate which technique generally improves the detection rates for the minority classes of the selected data sets.

1.4 Research Approach

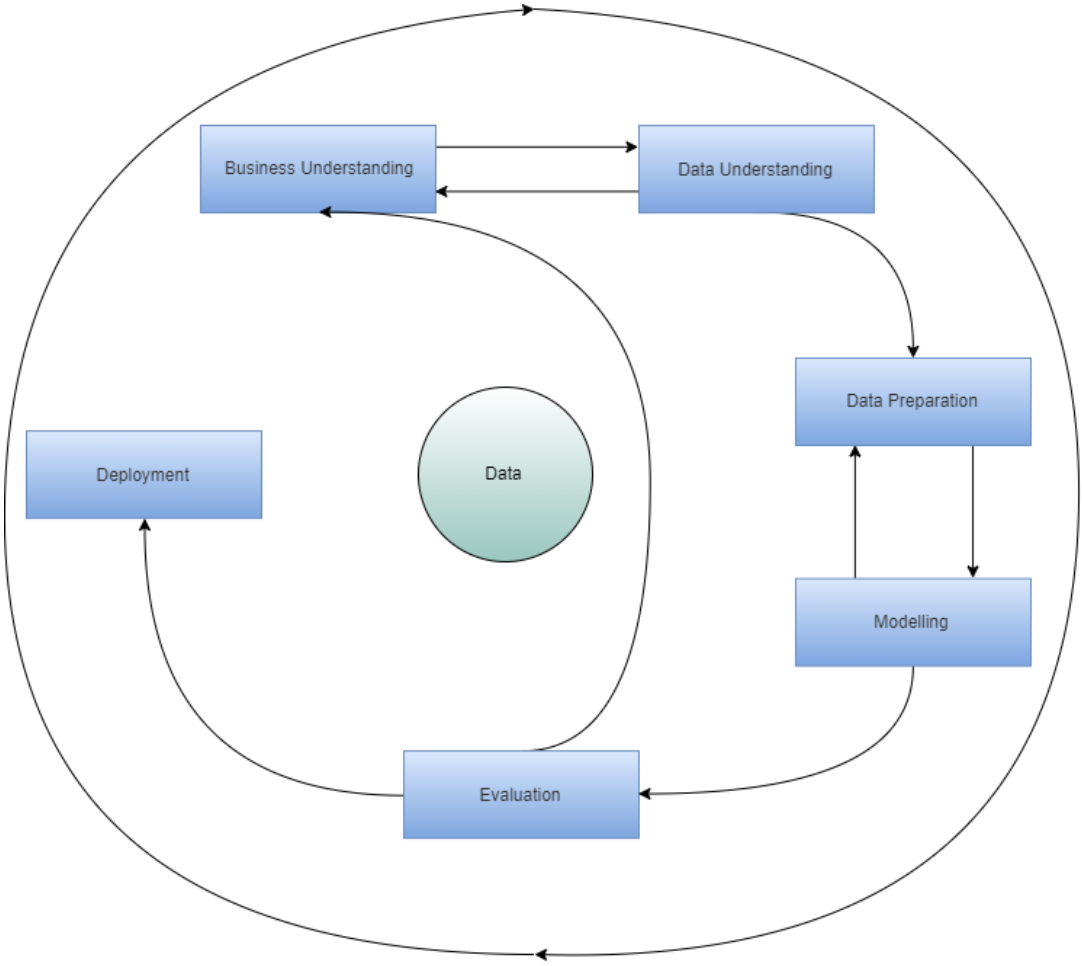


Figure 1.1: CRISP-DM Methodology

(Adopted from (Martinez-Plumed et al., 2019))

CRISP-DM is a data mining methodology that was initially proposed in 1999. It is regarded as the more comprehensive data mining methodology when compared to other data mining methodologies such as KDD and SEMMA. As a result, it has become a standard for data mining projects, and this methodology was chosen for this research. Figure 1.1 depicts the six steps in CRISP-DM. This elaborates the KDD methodology into six steps: business understanding, data understanding, data preparation, modelling, evaluation, and deployment (Daaderman and Rosander, 2018; Martinez-Plumed et al., 2019). The steps involved in CRISP-DM are described and the implementation of each phase in this research is detailed in Table 1.1.

Table 1.1: Description and Implementation of CRISP-DM in Research.

Steps	Description	Implementation
Business understanding	A process of understanding the project initiatives and requirements.	Understand the background of existing network intrusion data sets and transform the observed problem into a data mining problem. Investigate potential solutions and evaluate their effectiveness in mitigating the problem. In this phase, six network intrusion data sets and five oversampling methods have been researched.
Data understanding	A process in which data is collected, explored, and familiarised to comprehend the data better.	Explore, analyse, and visualise the distribution of data and its patterns. The unbalanced class distribution problem was discovered in all studied data sets. The unbalanced distribution problem exists not only between the normal attack traffics, but also among attack traffics.
Data preparation	A sequence of activities that transforms the initial raw data into a final data set. It is to increase the classifier's efficiency by removing noisy, inconsistent data from the data set.	Clean, format and transform the data and implement over sampling techniques before the modelling phase. For example, remove duplicate, null and non-informative columns.
Modelling	Various classifiers are selected, tuned, and implemented into the data set.	The hyperparameters for Logistic Regression, Naïve Bayes, and Decision Tree were tuned, built, and implemented.
Evaluation	A way to determine if the project's objective has been achieved is by	Determine which over sampling technique has the best overall

	evaluating the implemented classifiers.	performance, with the highest increase in detection rates and top occurrences.
Deployment	Organize and present the knowledge gained for further usage.	Documenting the analysis and discussion of the researched results, findings.

1.5 Scope of the Project

In this FYP project, a few unbalanced class network intrusion data sets and over-sampling techniques shall be identified, researched, and explored. Python and other data mining or statistics tools shall be used for data pre-processing, data visualisation, over-sampling technique implementation, predictive model implementation, and technique evaluation. Lastly, the well-performed oversampling techniques for the intrusion network data sets will be identified. The chosen oversampling technique is the method with the highest increase in detection rates among all other techniques used.

CHAPTER 2

LITERATURE REVIEW

Cyber-attacks have increased in tandem with the exponential expansion of digitalisation throughout the world. According to a report in CNBC business news (Scott Steinberg, 2019), Hiscox, an insurance carrier, revealed that the average cost of a cyber-attack in the year 2019 was $ 200,000 for all business sizes. As a result, implementing an IDS is one of the approaches to overcome the security problem in a network. Anomaly detection and misuse detection are the most common intrusion detection approaches. Anomaly detection is the technique of identifying unusual observations, occurrences that are out of the ordinary, comparing behaviour to a profile. Misuse detection, on the other hand, will detect an attack if the sequence of activities in the network meets known attack signatures; it compares behaviour to rules. When compared to anomaly detection, this approach yields fewer false positives, but it can recognise known attacks only for which the signature has been specified (Vigna and Kemmerer, 2002).

Many network intrusion data sets were introduced to train classifiers and evaluate the IDS. The process of effectively recognising network intrusions may be regarded as a classification issue. In reality, network traffic distribution is often skewed because some classes appear more frequently; in a network, normal traffic usually outnumbers attack traffic. The data distribution poses difficulty in classification since the minority class (attack traffic) is frequently underrepresented in the data and overwhelmed by the dominant class (normal traffic). A classifier may produce high overall accuracy if the ratio of class imbalance between majority and minority classes is extreme (Leevy et al., 2018). Till then, the classifier will be useless because the minority class (attack traffic) is essential for intrusion detection. Figure 2.1 shows two possible unbalanced learning scenarios: unbalanced binary and multi-class classification.

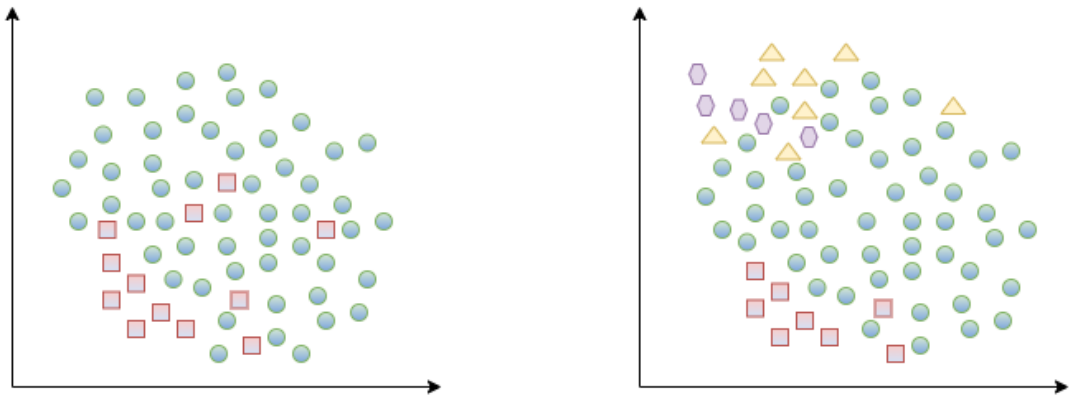


Figure 2.1: Two Possible Scenarios : (a) Unbalanced Binary Classification (b) Unbalanced Multi-class Classification (Adopted from (Sáez, Krawczyk and Woźniak, 2016))

The unbalanced class distribution for the binary class arises in between two classes, one of which is the dominant class, and the lesser sample is the rare class. In contrast to the binary scenario, identifying the data set's possible bias is more challenging in the multi-class scenario. Since the pairwise relation between each class does not reflect the major problem, one class might be the dominant group in one subset while being a rare group in others (Sáez, Krawczyk and Woźniak, 2016).

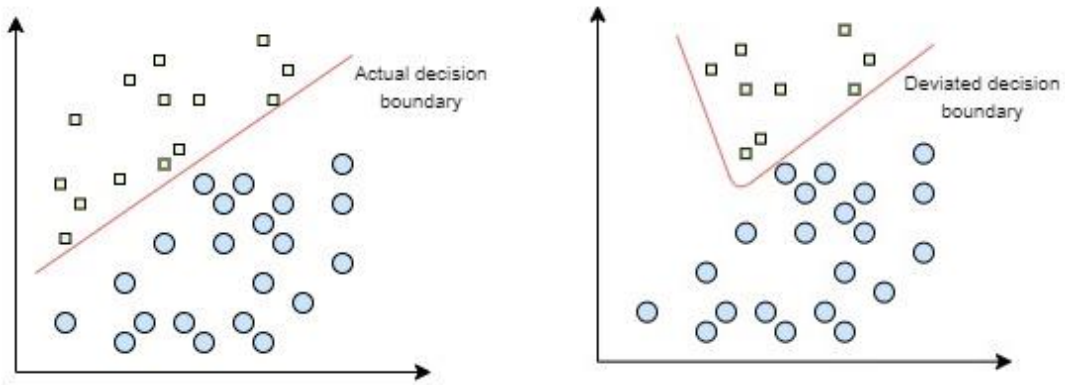


Figure 2.2: Decision Boundary : (a) Actual Decision Boundary (b) Deviated Decision Boundary

Algorithms often categorize an instance based on the boundaries drawn between classes. The number of samples in one class will help to rectify its decision boundary from other classes, hence improving the algorithm's performance. The true/actual decision boundary can be reflected when there have insufficient samples,

as shown in Figure 2.2 (a). Nonetheless, when there are insufficient representative data samples in one class, the decision boundary will deviate, as shown in Figure 2.2 (b). This will result in a data point being misclassified, affecting the algorithm's performance. For example, if one yellow data point is positioned near the bottom of the yellow class in (b), the deviated boundary may cause it to be misclassified as blue.

2.1 Network intrusion data sets

Some authors have concentrated their efforts on network intrusion data sets rather than the IDS techniques. Ring et al. (2019) reviewed 34 intrusion detection data sets, detailing the packets and network statistics as well as their shortcomings. Hindy et al. (2020) examined 30 data sets and discussed the limitations of the existing data sets used for IDS model prediction. In this section, a few network intrusion data sets with unbalanced class distribution problems are identified and described. Lastly, data sets containing a range of classes with low detection rates are selected to evaluate the over sampling techniques. The network intrusion data sets studied in this research were NSL KDD, UNSW-NB15, CICIDS2017, Kyoto2006, ISCX 2012 and Consolidate dataset from CICDDOS2019 with CICIDS2017.

2.1.1 NSL_KDD

Many researchers use the NSL_KDD data set as a benchmark to evaluate their IDS. This data set was created in 2009 as a revised version of the KDD99. As a result, few changes have been made to improve the KDD99 data set, such as eliminating duplicated records, and ensuring adequate data for both training and testing data set while preserving the ratio for each class. However, the underlying network activity in this data set dates from 1998. There are 41 features in this data set that were inherited from the KDD99 data set. The details of the feature in each category are tabulated in Table 2.1.

Table 2.1: 41 Features Descriptions in NSL_KDD

(Adopted from (Siva and Ramani, 2011))

Basic		
No.	**Features**	**Descriptions**
1	Duration	Connection time in seconds
2	Protocol type	Protocol used
3	Service	Network service for destination
4	Src bytes	Bytes send from src to dst
5	Dst bytes	Bytes send from dst to src
6	Flag	Connection status
7	Land	One when connection is receive/send to same host/port; zero vice versa
8	Wrong Fragment	Num of incorrect fragments
9	Urgent	Num of urgent packets
Content		
10	Hot	Num of host
11	Num failed logins	Num of unsuccessful logins
12	Logged in	one for success login; zero for fail login
13	Num compromised	Num of "compromised"
14	Root shell	indicates one when root shell is acquired; else is zero
15	Su attempted	Indicates one when “su root” is attempted; else is zero
16	Num root	Num of “roots’' access
17	Num files creations	The number of times a file has been created.
18	Num shells	Num of shell prompts
19	Num access files	Num of access control files activities
20	Num outbound cmds	Num of outward requests in ftp session
21	Is hot login	indicates one for "host" login
22	Is guest login	indicates one for guest login; else is zero
Time		

23	Count	Num of cons to the identical host as the present con in the last two sec.
24	Serror rate	Percentage of cons with “SYN" errs.
25	Rerror rate	Percentage of cons with “REJ" errs.
26	Same srv rate	Percentage of cons to identical service.
27	Diff srv rate	Percentage of cons to diff service
28	Srv count	Num of cons to the identical host as the present con in the last two sec.
29	Srv serror rate	Percentage of cons with “SYN" errs.
30	Srv rerror rate	Percentage of cons with “REJ" errs.
31	Srv diff host rate	Percentage of cons t
Host		
32	Dst host count	Num of cons having identical dst host
33	Dst host diff srv rate	Percentage of diff services on present host
34	Dst host rerror rate	Percentage of cons to present host that have RST err
35	Dst host same src port rate	Percentage of conns to present host having the identical source port
36	Dst host same srv rate	Percentage of cons having the identical dst host while utilising same service
37	Dst host serror rate	Percentage of cons to present host that obtained S0 err
38	Dst host srv count	Num of cons having the identical dst host and utilising the same service
39	Dst host srv diff host rate	Percentage of cons to the identical service from diff hosts
40	Dst host srv rerror rate	Percentage of cons to the present host and specified service that have RST err
41	Dst host srv serror rate	Percentage of cons to the present host and specified service that have S0 err

The features are categorised into four groups: primary features of a single TCP connection, where some features are extracted from packet headers. Content features hold the domain knowledge of the packets. Time features are intended to capture traits with a two-second window. Host features use historical windows to retrieve the number of connections and access attacks that have more than two seconds time intervals. This data collection contains 37 distinct attack types in total, which may be further classified into four attack categories: Denial of Service, Probes, User to Root and Remote to Local. Table 2.2 categories the typical attacks into four main categories. The class distribution for each attack category is shown in Table 2.3. Even though normal and attack traffic distributions are not significantly different, with 51.89 % and 48.11 %. The results in Table 2.3 show that the class distribution in NSL_KDD is highly unbalanced among the attack classes, with probe, r2l, and u2r accounting for just 9.48 %, 2.52 %, and 0.17 % of the overall data distribution (Rodda and Erothi, 2016).

Table 2.2: Attack Categories in the NSL_KDD

Categories	**Description**	**Class**
DoS	This attack aims to disrupt the service of a computer system or network by overwhelming the system and preventing response to service request	Pod, teardrop, smurf, back, land, Neptune, worm, apache2, mailbomb, process table, UDP storm
R2L	Unauthorised users try to access an account on the victim machine.	warezmaster, warezclient, spy, multihop, imap, phf, password guessing, ftp, snmpguess, snmpgetattack, httptunnel, named, sendmail, xlock, xsnoop
U2R	Unauthorised users have gained local access on the target device and attempting to escalate its privilege.	buffer overflow, load module, perl and root kit, ps, SQL attack, xterm

Probe	Unauthorised users attempts to obtain information of the victim machine by installing some programmes into their system.	nmap, portsweep, ipsweep, satan, mscan, saint

Table 2.3: Class Distribution in NSL_KDD Data Set

Traffic Types	**Class Distributions (%)**
Normal	51.89
DoS	35.94
R2L	2.52
U2R	0.17
Probe	9.48

2.1.2 UNSW-NB15

In 2016, the UNSW-NB15 was introduced by the Australian Center for Cyber Security Lab. It was created with the help of the IXIA Perfect Storm tool to generate hybrid data that compromises both actual and synthetic attack traffic in the network. Tcpdump tools were used to capture the raw traffics in the network, and the data set contains source files in the pcap, Argus, and CVS formats. According to some academics (Chou and Jiang, 2020), it has become another standard data set for the IDS evaluation. However, the reliability of the data generated is a concern for this synthetic data set. This data collection has 49 features and has a total traffic of 2,540,000 records (Choudhary and Kesswani, 2020). The features in UNSW-NB15 can be divided into six categories: basic features, flow features, time-based features, content features, additional features, and labelled features. The description of each feature is tabulated in Table 2.4.

Table 2.4: 47 Categorised Features and Their Descriptions in UNSW-NB15 (Adopted from (Moustafa and Slay, 2015b; Bagui et al., 2019))

Flow features		
No	**Features**	**Description**
1	Srcip	Ip address for src
2	Sport	Port number for src
3	Dstip	Ip address for dest
4	Dsport	Port number for dest
5	Proto	Protocol used
Basic Features		
6	State	Indicates state and the protocol on which it depends on
7	Dur	Total time taken
8	Sbytes	Bytes send from src to dest
9	Dbytes	Bytes send from dest to src
10	Sttl	Time to live from src to dest
11	Dttl	Time to live from dest to src
12	Sloss	Src packets resent or discarded
13	Dloss	Dest packets resent or discarded
14	Service	Like HTTP, DNS, SSH, FTP, SMTP
15	Sload	Src bits in one sec
16	Dload	Dest bits in one sec
17	Spkts	Num of packets from src to dest
18	Dpkts	Num of packets from dest to src
Content Features		
19	Swin	The window advertisement value for src TCP
20	Dwin	The window advertisement value

		for dest TCP
21	Stcpb	TCP base sequence number of src
22	Dtcpb	TCP base sequence number of dest
23	Smeansz	Flow packet size mean sent by source
24	Dmeansz	Flow packet size mean sent by the dest
25	Trans_depth	Pipelined depth into the cons of HTTP request/response transaction
26	Res_bdy_len	Real uncompressed data size transmitted from the server's http service
Time features		
27	Sjit	Src jitter (millisecond)
28	Djit	Dest jitter (millisecond)
29	Stime	Start time
30	Ltime	End time
31	Sintpkt	Arrival time for src interpacket in milliseconds
32	Dintpkt	Arrival time for dest interpacket in milliseconds
33	Tcprtt	The sum of ’synack’ and ’ackdat’ represents the TCP cons in RTT
34	Synack	Time taken for a TCP connection to be established, represent by the transmission time between SYN and SYN_ACK packets
35	Ackdat	Time taken for a TCP connection to be established, represent by the transmission time between the

		SYN_ACK and the ACK packets
Additional features		
36	is_sm_ips_ports	When srcip = dstip and sport = dsport. Assign one, else assign zero.
37	ct_state_ttl	Num for each state based on a particular range of sttl and dttl values
38	ct_flw_http_mthd	Num of flows with methods like DELETE and PUT in HTTP service
39	is_ftp_login	Assign one if a ftp session is established using user and password; else assign zero
40	ct_ftp_cmd	Num of flows in an ftp session that have a command
41	ct_srv_src	Num of records that have identical service and srcip in 100 records depends on ltime
42	ct_srv_dst	Num of records that have identical service and dstip in 100 records depends on ltime
43	ct_dst_ltm	Num of records of the identical dstip in 100 records depends on ltime
44	ct_src_ ltm	Num of records of srcip in 100 records depends on ltime
45	ct_src_dport_ltm	Num of records of identical srcip and dsport in 100 records depends on ltime
46	ct_dst_sport_ltm	Num of records of identical dstip and sport in 100 records depends on ltime
47	ct_dst_src_ltm	Num of records of identical srcip

		and dstip in in 100 records depends on ltime
Labelled Features		
48	Attack_cat	Attack category name
49	label	Zero for normal traffic, one for attack traffic

The flow category stores features that can be used to identify the features between hosts, such as client-to-server and vice versa. While the basic features include attributes used to represent the protocol connections, the content features include TCP/IP attributes that have been encapsulated. For time features, it stores all time-related traffic information, such as packet arrival time and RTT; for additional produced features, it stores general features to secure the protocol service and the connection features store the connection details. Furthermore, the UNSW-NB15 data set contains normal traffic and nine different attack categories known as Analysis, Backdoors, Generic, Shellcode, Exploits, DoS, Worms, Fuzzers and Reconnaissance (Moustafa and Slay, 2016). The attack categories for this data set are described in Table 2.5. Table 2.6 shows that this data set not only has high unbalanced data distribution between normal (87.35 %) and attack classes (12.65 %). It also has a significant difference in the attack classes distribution itself, with generic labels accounting for 8.48 % and the two lowest attacks accounting for less than 0.1 % of the total distribution. Those are Shellcode which have 0.06 % and worms with 0.007 % from the data distributions (Bagui et al., 2019; Chou and Jiang, 2020).

Table 2.5: Attack Types in UNSW-NB15

Attack Type	**Description**
Analysis	Intrusions that attack the web applications via port scan, email scam and web scripts penetration.
Backdoors	A method of bypassing normal authentication in a computer and access the computer's data

	remotely.
DoS	This attack type aims to disrupt the service of a computer system or network by overwhelming the system and preventing response to the service request.
Reconnaissance	It is a foot printing activity in which attackers gather information from a computer network before circumventing security controls.
Generic	A strategy that works against a block-cipher regardless of the block-cipher's structure.
Shellcode	It is a special code injected remotely into the target and attackers use it to exploit the shell control of the compromised machine.
Exploit	It can be a software program, data or sequence of instruction that exploits the vulnerability of a computer software, hardware and cause unanticipated behaviour to occur on the target.
Worms	A malware that will self-replicate in the target machine to distribute the infected code over a network with low protection.
Fuzzers	Attackers try to inject a large volume of randomly generated data into a program, network, or operating system to find the security flaws.

Table 2.6: Data Distribution in the UNSW-NB15 Data Set

Traffic Types	Class Distributions (%)
Normal	87.35
Generic	8.48
Exploit	1.75
DoS	0.64
Fuzzers	0.96
Reconnaissance	0.55
Backdoors	0.09
Shellcode	0.06
Worms	0.007
Analysis	0.113

2.1.3 CICIDS 2017

CICIDS2017 was collected by the Canadian Institute of Cyber Security and was based on the abstract behaviour of 25 users during 5 days in an emulated environment (Kurniabudi et al., 2020). Some protocols, such as HTTP, email, FTP, SSH and HTTPS, were identified as part of those user behaviour It contains 2,830,108 packet and bidirectional flow traffics. This is one of the data sets that include current attacks covering all 11 criteria with common attacks: full network configuration, annotated data set, full capture, presented protocols, attack diversity, complete interaction, complete traffic, available features, contains meta data and heterogeneity. CICFlowMeter software was utilised to excerpt traffic features from the data set and over 80 features for both normal and attack flows were extracted. There are 11 types of attack and 83 features extracted for the data set (Sharafaldin, Lashkari and Ghorbani, 2018). The extracted features for the CICIDS2017 data set are tabulated in Table 2.7.

Table 2.7: 84 Features of Network Traffic in CICIDS 2017

(Adopted from (Abdulhammed et al., 2019))

No	Features	No	Features	No	Features
1	Flow ID	**29**	Fwd IAT Max	**57**	Down/Up Ratio
2	Src IP	**30**	Fwd IAT Min	**58**	Avg Packet Size
3	Src Port	**31**	Bwd IAT Total	**59**	Avg Fwd Segment Size
4	Dest IP	**32**	Bwd IAT Mean	**60**	Avg Bwd Segment Size
5	Dest Port	**33**	Bwd IAT Std	**61**	Fwd Avg Bytes/Bulk
6	Protocol	**34**	Bwd IAT Max	**62**	Fwd Avg Packets/Bulk
7	Time stamp	**35**	Bwd IAT Min	**63**	Fwd Avg Bulk Rate
8	Flow Dur	**36**	Fwd PSH Flags	**64**	Bwd Avg Bytes/Bulk
9	Total Fwd Pcks	**37**	Bwd PSH Flags	**65**	Bwd Avg Packets/Bulk
10	Total Bwd Pcks	**38**	Fwd URG Flags	**66**	Bwd Avg Bulk Rate
11	Total Len of Fwd Pck	**39**	Bwd URG Flags	**67**	Subflflow Fwd Packets
12	Total Len of Bwd Pck	**40**	Fwd Header Len	**68**	Subflflow Fwd Bytes
13	Fwd Pck Length Max	**41**	Bwd Header Len	**69**	Subflflow Bwd Packets
14	Fwd Pck Length Min	**42**	Fwd Packets/s	**70**	Subflflow Bwd Bytes
15	Fwd Pck Length Mean	**43**	Bwd Packets/s	**71**	Init_Win_bytes_fwd
16	Bwd Pck Length Max	**44**	Min Pck Length	**72**	Act_data_pkt_fwd
17	Bwd Pck Length Min	**45**	Max Pck Length	**73**	Min_seg_size_fwd
18	Bwd Pck Length Mean	**46**	Packet Len Mean	**74**	Active Mean
19	Bwd Pck Length Std	**47**	Packet Len Std	**75**	Active Std
20	Flow Bytes/s	**48**	Packet Len. Variance	**76**	Active Max
21	Flow Packets/s	**49**	FIN Flag Count	**77**	Active Min
22	Flow IAT Mean	**50**	SYN Flag Count	**78**	Idle Mean
23	Flow IAT Std	**51**	RST Flag Count	**79**	Idle Std
24	Flow IAT Max	**52**	PSH Flag Count	**80**	Idle Std
25	Flow IAT Min	**53**	ACK Flag Count	**81**	Idle Max
26	Fwd IAT Total	**54**	URG Flag Count	**82**	Idle Min
27	Fwd IAT Mean	**55**	CWE Flag Count	**83**	label
28	Fwd IAT Std	**56**	ECE Flag Count		

The attacks in this data set include Brute Force, Botnet, Heart Bleed, DoS, infiltration, DDoS, SQL injection, port scan, XSS, ftp attack and SSH attack. Table 2.8 describes the attack type (Abdulhammed et al., 2019), whereas table 2.9 provides the class distribution in CICIDS 2017 (Panigrahi and Borah, 2018). In this data set,

the unbalanced class distribution problem not only exists in between normal, and attack traffics, which were 83.3 % and 16.7 %, respectively. However, several attack classes were also significantly underrepresented among the attack classes. For example, heart bleed attacks, infiltration attacks, and SQL injection, which account for less than 0.01 % of the entire distribution. The only attack classes with more than 1 % class distribution among the total distribution were DoS and Port Scan (Sharafaldin, Lashkari and Ghorbani, 2018).

Table 2.8: Attack Types in the CICIDS2017 Data Set

Attack Type	**Description**
Botnet	Malicious programs such as trojans were used to penetrate the target machine's security, steal its data, and get remote access to the target machine.
DoS	This attack seeks to interrupt a computer systems or network's service by overwhelming the system and preventing it from responding to service requests.
FTP	Attackers utilise a brute force strategy for password guessing by using the FTP patator
Heart bleed	This attack occurs in the transport layer where the assailant use the OpenSSL protocol to inject malicious information into its memory to elicit a reaction from the victim. As a result, the attacker will have access to their essential information.
Infiltration	Attackers use tools to infiltrate a network and back door will be installed after exploitation in order to gain full unauthorized access to the network data.

Port Scan	Attackers try to gather information about the target machine by sending packets to various destination ports of the target machine.
SSH	Attackers use the brute force approach for password guessing by using the SSH patator.
SQL Injection	A code injection approach that uses the SQL command to cause the database to respond, allowing the code injection process to extract application data.
XSS	Using a web app that transmits malicious scripts, attackers attempt to inject codes into a legitimate or trustworthy website.
Brute Force	It is a common attack that uses a trial-and-error approach to not only gain privilege information like a password or identification number, but also to locate hidden information in a web app.
DDoS	It is similar to DoS attack, but the attackers use multiple machines to overwhelm the target machine at the same time with a large amount of traffics.

Table 2.9: Data Distribution in CICIDS 2017 Data Set

Traffic Types	Class Distributions (%)
Normal	83.344
Botnet	0.069
DoS	8.91
FTP	0.28
Heart bleed	0.00039
Infiltration	0.00127
Port Scan	5.61
SSH	0.20800
SQL Injection	0.00074
XSS	0.02303
Brute Force	0.05324

2.1.4 KYOTO 2006

Kyoto 2006 data set was created through diverse honeypot in 3 years from August 2006 to November 2009. The IDS detection session was distributed to more than 200+ countries. Approximately 50 % of the cyber-attacks were launched from South Korea, the US, and China. This data set does not specify the attack label, as the traffic was only labelled into three categories: 1 (normal traffic), -1 (known attacks), and -2 (unknown attacks). Web attacks, brute force, DoS, Port Scan, Backdoor, DNS, and other attacks are included in this data set. Kyoto 2006 has a total of 24 statistical features, 14 features were inspired by the KDD 99 data set, and an additional ten features were added to create a new intrusion detection data set for IDS evaluation. Table 2.11 tabulates the percentage and number of records for the class in the Kyoto 2006 (Song et al., 2011) and Table 2.10 tabulates all feature descriptors for the Kyoto 2006.

One of the data set's drawbacks is that the number of protocols available was limited to TCP, UDP, and ICMP, with HTTPS protocol being omitted, causing most stimulated attacks to fail to retain real-world statics (Yahia and Atwell, 2018). Moreover, this data set was not captured from the real-world network traffic, where the normal traffic produces only DNS and mail traffic data. As a result, there were no

false positives in this data set, which significantly reduces the number of alerts. (Sharafaldin, Lashkari and Ghorbani, 2018).

Table 2.10: 24 Features in Kyoto 2006 Data Set
(Adopted from (Song, Takakura and Okabe, 2009; Protić, 2018))

Conventional Features	
Features	**Description**
Duration	Connection length (second)
Service	Connection service's type
Src bytes	Bytes send from src to dst
Dst bytes	Bytes send from dst to src
Count	Num of cons to the identical host as the present con in the last two sec.
Same srv rate	Percentage of cons to identical service.
Serror rate	Percentage of cons with "SYN" errs in count feature
Srv serror rate	Percentage of cons with "SYN" errs.
Dst host count	Num of cons having identical dst host
Dst host srv count	Num of cons having the identical dst host and utilising the same service
Dst host same src port rate	Percentage of conns to present host having the identical source port
Dst host serror rate	Percentage of cons to present host that obtained S0 err
Dst host srv serror rate	Percentage of cons to the present host and specified service that have S0 err

Flag	Connection status
Additional Features	
IDS detection	It reflects if the IDS activated an alert for the cons with three conditions: Zero - no alert Arabic numeral – types of alerts Parenthesis - the number of the identical alert identified in the cons.
Malware detection	It indicates whether a malicious software, was found in the cons with three conditions: Zero - no malware String – represents the malware discovered in the cons. The malwares was detected by Clamav. Parenthesis - number of the identical malware identified in the cons.
Ashula detection	It determines if the shellcodes and exploit codes were used in the cons: Zero - no shellcodes and exploit codes Arabic numeral - various shellcodes or exploit codes was discovered. Parenthesis - the number of the identical shellcode or exploit code found in the cons.
Label	This indicate whether the connection is an attack traffic: One - normal traffic Negative one - known attack Negative two - unknown attack.

Src IP Address	Source IP address
Src Port Number	Source Port Number
Dest IP Address	Destination IP address
Dest Port Number	Destination Port Number
Start Time	Records when the session was begun.
Protocol	Protocol used by connection

Table 2.11: Data Distribution in the Kyoto 2006 Data Set

Traffic Types	**Number of records**	**Class Distribution (%)**
Normal	50,033,015	53.75
Known Attack	42,617,536	45.79
Unknown Attack	425,719	0.46
Total	93,076,270	100

2.1.5 ISCX 2012

The ISCX2012 data set was generated in 2012 at the University of New Brunswick by ISCX. It was intended to solve the problem in other data sets, such as DARPA and DEFCON. It contains normal traffic as well as four attack types acquired over a week: botnet, infiltration, DoS, and SSH; however, they are all labelled as “attack” rather than their specific attack type. Normal traffic has generated in only two days out of seven, and each attack scenario has generated on a single day. Two profiles in this data set are used to create the network traffic: alpha and beta. Attack traffic was created by Alpha profile, whereas normal traffic, such as e-mail and web surfing, were created by a Beta profile and a realistic network environment (Kamarudin et al., 2017). The protocols used in network traffic are HTTP, IMAP, SMTP, SSH, POP3, and FTP. However, the HTTPS protocol, which reflects the most recent network activity, has been disregarded (Sharafaldin, Lashkari and Ghorbani, 2018). As a result, the simulated attacks might not reflect the real network.

Table 2.12 tabulates all 19 characteristics for the ISCX 2012 data set and Table 2.13 tabulates the class distribution in percentage and number of records for each attack scenario in the ISCX 2012 data set from 12 Jun 2021 (Monday) to 17 June 2021 (Sunday). In this data set, the class unbalanced distribution problem is severe occurs between the normal-attack class. Several attack scenarios are significantly underrepresented among attack traffics, with HTTP DoS and SSH accounting for only 0.21 % and 0.42 % of the overall distribution, respectively.

Table 2.12: 19 Features in the ISCX2012 Data Set
(Adopted from (Ghurab et al., 2021))

Additional Features	
No	**Features**
1	Application name
2	Src IP Address
3	Dest IP Address
4	Src Port
5	Dest Port
6	Direction of flow
7	IP Protocol
8	Flow duration
9	Total source bytes
10	Total destination bytes
11	Total bytes
12	Total source packets
13	Total destination packets
14	Total packets
15	Tag
16	Time Start
17	Time End
18	Src TCP flag descrp
19	Dest TCP flag descrp

Table 2.13: Data Distribution in ISCX 2012 Data Set

Traffic Types	Class Distributions (%)	Number of records
Normal	96.30	1,816,609
Botnet	1.99	37,460
Infiltration	1.08	20,358
HTTP denial of service	0.21	3,776
SSH	0.42	7,316

2.1.6 CICDDOS 2019

The CIC-DDoS dataset was created in the year 2019 for the detection of Distributed Denial of Service (DDoS) attacks in two days. The design of a data set has a significant impact on the evaluation of machine learning algorithms. As a result, Sharafaldin et al. (2019) generated this data set to address these flaws by presenting the most significant feature for detecting different types of DDoS. This data set contains two major types of DDoS attacks that use the TCP or UDP protocols: reflection-based and exploitation-based. The abstract behaviour of 25 users was built on the HTTPS, FTP, SSH, and email protocols. Since this data set focused on DDoS attacks, only a limited proportion of normal traffic is created. Therefore, Enoch and Khor (2021) supplemented this data set with Benign tuples from CICIDS 2017 to increase the proportion of normal traffic.

There are 12 DDoS attacks in this consolidated data set. Those are TFTP, SYN, WebDDoS, UDP-Lag, UDP, SSDP, SNMP, NetBIOS, MSSQL, LDAP, DNS and NTP. Table 2.14 tabulates all 77 characteristics of the CIC-DDoS 2019 data set, where most of them are similar to CICICS 2017. Table 2.15 tabulates the class distribution in percentage and number of records for each attack. Normal and TFTP attacks account for the majority of traffic in this data set, accounting for roughly 46 % of the entire distribution. Each of the remaining attack classes has less than 10 % of the entire distribution, with some classes such as UDP Lag, Port Map, and WebDDoS having even less than 1 %. As a result, the class distribution in this data set is unbalanced between the majority (normal & TFTP) and attack classes, and within the attack classes.

Table 2.14: 77 Features of Network Traffic in CIC-DDoS 2019

(Adopted from (Sharafaldin et al., 2019))

No	Features	No	Features	No	Features
1	Flow Duration	**29**	Fwd PSH Flags	**57**	Bwd Avg Bytes/Bulk
2	Total Fwd Pcks	**30**	Bwd PSH Flags	**58**	Bwd Avg Packets/Bulk
3	Total Bwd Pcks	**31**	Fwd URG Flags	**59**	Bwd Avg Bulk Rate
4	Total Len of Fwd Pck	**32**	Bwd URG Flags	**60**	Subflflow Fwd Packets
5	Total Len of Bwd Pck	**33**	Fwd Header Len	**61**	Subflflow Fwd Bytes
6	Fwd Pck Length Max	**34**	Bwd Header Len	**62**	Subflflow Bwd Packets
7	Fwd Pck Length Min	**35**	Fwd Packets/s	**63**	Subflflow Bwd Bytes
8	Fwd Pck Length Mean	**36**	Bwd Packets/s	**64**	Init_Win_bytes_fwd
9	Bwd Pck Length Max	**37**	Min Pck Length	**65**	Init_Win_bytes_bwd
10	Bwd Pck Length Min	**38**	Max Pck Length	**66**	Act_data_pkt_fwd
11	Bwd Pck Length Mean	**39**	Packet Len Mean	**67**	Min_seg_size_fwd
12	Bwd Pck Length Std	**40**	Packet Len Std	**68**	Active Mean
13	Flow Bytes/s	**41**	Packet Len. Variance	**69**	Active Std
14	Flow Packets/s	**42**	FIN Flag Count	**70**	Active Max
15	Flow IAT Mean	**43**	SYN Flag Count	**71**	Active Min
16	Flow IAT Std	**44**	RST Flag Count	**72**	Idle Mean
17	Flow IAT Max	**45**	PSH Flag Count	**73**	Idle Std
18	Flow IAT Min	**46**	ACK Flag Count	**74**	Idle Std
19	Fwd IAT Total	**47**	URG Flag Count	**75**	Idle Max
20	Fwd IAT Mean	**48**	CWE Flag Count	**76**	Idle Min
21	Fwd IAT Std	**49**	ECE Flag Count	**77**	label
22	Fwd IAT Max	**50**	Down/Up Ratio		
23	Fwd IAT Min	**51**	Avg Packet Size		
24	Bwd IAT Total	**52**	Avg Fwd Segment Size		
25	Bwd IAT Mean	**53**	Avg Bwd Segment Size		
26	Bwd IAT Std	**54**	Fwd Avg Bytes/Bulk		
27	Bwd IAT Max	**55**	Fwd Avg Packets/Bulk		
28	Bwd IAT Min	**56**	Fwd Avg Bulk Rate		

Table 2.15: Data Distribution in CIC-DDoS 2019 Data Set

Traffic Types	Class Distributions (%)	Number of records
Normal	25.88	2,384,051
TFTP	21.19	1,951,336
MSSQL	10.84	998,191
NetBIOS	8.12	747,772
UDP	7.47	688,393
SYN	6.45	594,129
SNMP	5.59	514,957
DNS	5.33	490,813
LDAP	4.45	410,301
SSDP	2.79	256,832
NTP	1.30	119,528
UDP Lag	0.38	34,891
Port Map	0.19	17,767
WebDDoS	0.0047	439

2.1.7 Common problem: Low Detection rate

This section investigates the overall and multiclass detection rate obtained from other research for each selected data set. The class distributions from the studied data sets show a highly unbalanced class distribution between normal and attack traffics, or between attack traffics and itself, or both. As a result of the underrepresented minority classes in the data set, the classifiers have a low detection rate for those classes. For example, for the NSL KDD data sets, Ingre and Yadav (2015) had 74.5 % detection accuracy, while Liu et al. (2020) had a 74.34 % detection rate by implementing Random Forest. Moreover, other researchers' results on the detection rate for normal traffic and each attack category on NSL KDD are shown in Table 2.16. Most of the majority class have satisfied detection rates, for instance, normal traffic and DoS traffic. While detection rates for highly underrepresented categories like U2R (0.17 %) and R2L (2.52 %) are often below 80 %, the lowest detection rate for U2R is 4.36 %, and for R2L is 4.48 %.

Table 2.16: Detection rate of the classifiers on the NSL KDD data set

(Notes : the bold numbers indicate the minority class in the data set)

Paper	Classifier	Normal	DoS	Probe	U2R	R2L	Overall DR	Remarks
(Rodda and Erothi, 2016)	Naïve Bayes	96.6	83.81	**78.26**	**19.04**	**57.03**	-	ten-fold cross validation
	Bayes Net	97.3	94.9	**82.31**	**16.84**	**55.97**	-	
	J48	94.9	95.1	**76.57**	**14.28**	**5.61**	-	
	Random Forest	98.9	89.9	**98.13**	**14.28**	**93.35**	-	
(Bedi, Gupta and Jindal, 2020)	DNN	97.35	81.03	**55.64**	**4.36**	**4.48**	-	Apply 1000 epochs ReLU activation function was used in each hidden layer Siam-IDS is own proposed system using Keras API
	CNN	97.48	76.98	**49.57**	**6.44**	**17.91**	-	
	Siam IDS	91.22	85.37	**48.66**	**33.25**	**56.72**	-	
(Mallissery, Kolekar and Ganiga, 2013)	Bayes Net	69.1	68.4	**69.2**	**63.1**	**72.1**	**67.05**	Using all 41 features
	Naïve Bayes	70.1	72.7	**70.4**	**69.5**	**68.5**	**67.53**	
(Ingre and Yadav, 2015)	ANN	-	77.7	**76.6**	**10.5**	**34.6**	**74.5**	Using all 41 features 19 ANN hidden layers & 771 epochs using Levenberg-Marquardt algorithm
(Liu et al., 2020)	Random Forest	-	-	-	-	-	**74.34**	Removed outliers Performed hyper parameter tunning
	SVM	-	-	-	-	-	**73.66**	
	ROS + Random Forest	-	-	-	-	-	**75.15**	Removed outliers Performed hyper parameter tunning Implement random over-sampling technique
	SMOTE + Random Forest	-	-	-	-	-	**74.09**	Removed outliers Performed hyper parameter tunning Implement smote over sampling technique
(Subhy, Ibrahim and Basheer, 2013)	SOM	-	-	-	-	-	**75.49**	SOM is self-proposed IDS system. It describes the neighbourhood relations of data points in a topology map by discover the underlying structure of input data. Using 1000 epochs with learning rate 0.9. While change in learning rate in each epoch is 0.7.

Kumar et al. (2020) applied the UNSW-NB15 data set to validate an IDS system, and the findings demonstrate that their proposed IDS model had a detection rate of 57.01 % in binary classification. Khan et al. (2020) implemented a few classifiers with feature selections for this data set. Those were KNN, Random Forest, Bagging with the highest detection rate of 76 %, with 11 features extracted using feature importance. Gao et al. (2019) proposed an IDS that combined the I-Extreme Learning Machine with the A-PCA algorithm, and they used this data set to evaluate the IDS's performance. Their research revealed that their approach has a detection rate of 77.36 %. Furthermore, another researcher (Kasongo and Sun, 2020) had performed an analysis of IDS using a feature selection method on this data set. They obtained a detection rate of 70.21 % with 42 features and 77.24 % with 19 features in the predictions. Bagui and Li (2021) had implemented an ANN algorithm for this data set on a local machine without resampling and the highest detection rate it had obtained was 41.22 %. Table 2.17 summarises the overall detection rate and Table 2.18 tabulates the multi-classification detection rate on the UNSW-NB15 from other researchers. The detection rate for most underrepresented classes is less than 75 %, particularly for Analysis traffic, which its highest detection rate was 20.45 % among all classifications and obtained 0 % on five occasions.

Table 2.17: Overall Detection Rate on the UNSW-NB15 Data Set

(Notes : the bold numbers indicate the minority class in the data set)

Paper	**Classifier**	**Overall DR**	**Remarks**
(Kumar et al., 2020)	Proposed Integrated Model	57.01	Use information gain to extract features.
	Decision Tree	70.65	extracted 22 out of 47 features.
Khan et al. (2020)	KNN w/o FS	71	-
	Random Forest w/o FS	74	
	Bagging meta estimator w/o FS	75	
	KNN w FS	74	Implemented feature importance for feature selection.
	Random Forest w FS	76	
	Bagging meta estimator w/o FS	75	Extracted 11 out of 47 features.

Gao et al. (2019)	SVM	63.57	Implement A-PCA to perform feature selection
	CNN	67.54	
	ELM	60.35	
	I-ELM with A-PCA	77.36	
(Kasongo and Sun, 2020)	LR w/o FS	65.54	-
	KNN w/o FS	70.21	
	SVM w/o FS	62	
	Decision Tree w/o FS	66.04	
	LR w FS	70.88	Implement filter-based method (XGBoost) for feature selection. Extracted 19 out of 47 features.
	KNN w FS	77.24	
	SVM w FS	61.52	
	Decision Tree w FS	67.56	
Bagui and Li (2021)	ANN	41.22	-
	ANN + RU	50	Implemented random Undersampling technique
	ANN + RO	76	Implemented random over sampling technique
	ANN + RURO	77.58	Implemented random oversampling and Undersampling techniques

Table 2.18: Multi Classification Detection Rate on the UNSW-NB15 Data Set

(Notes : the bold numbers indicate the minority class in the data set)

Paper	(Kumar et al., 2020)	(Bagui et al., 2019)			(Moustafa and Slay, 2015a)	(Meftah, Rachidi and Assem, 2019)		
Classifier	Dendron	Naïve Bayes w/o FS	J48 w/o FS	J48 W FS	Naïve Bayes	Decision Tree	Naïve Bayes	SVM
Normal	97.39	-	-	-	81	74.93	86.29	97.41
Backdoor	**67.32**	**66**	**2.22**	**0**	**20**	**4.97**	**22.47**	**0**
Analysis	**20.45**	**74**	**6.64**	**0**	**0**	**0**	**0**	**0**
Fuzzer	**64.42**	**57**	**75.18**	**2.42**	**33.2**	**55.24**	**36.28**	**76.26**
Shellcode	**16.39**	**72**	**43.12**	**0**	**0**	**60.84**	**1.32**	**0**
Recon.	**46.04**	**65**	**79.5**	**41.47**	**69.9**	**80.77**	**49.57**	**68.44**
Exploit	**76.22**	**56.96**	**64.27**	**52.25**	**54.6**	**90.08**	**24.97**	**75.22**
DoS	**14.29**	**66**	**5.99**	**1.76**	**71.1**	**8.83**	**0**	**1.07**
Worms	**18.37**	**84**	**40.9**	**0**	**0**	**72.72**	**38.64**	**0**
Generic	81.37	83	97.8	96.2	94.3	96.96	96.29	96.24
Overall DR	52.21	-	-	-	37.5	75.53	60.70	70.21

Remarks	Use information gain to extract features. extracted 22 out of 47 features.	-	Implement K-means clustering and CFS for feature selection Extracted 27 out of 47 features.	Implement ARM for feature selection Extracted 31 out of 47 features.	Implement random forest algorithm with ten-fold cross validation for feature selection Use ten-fold cross validation for hyperparameter tuning

Ahmim et al. (2018) proposed a new classifier for an IDS that integrates decision tree and rule-based concepts, and the proposed IDS was evaluated using the CICIDS 2017 data set. A few algorithms were implemented in the research to compare with their proposed classifier, such as WISARD, with an overall detection rate of 48.18 % and LIBSVM, with 56.60 %. In addition, Abdulhammed et al. (2019) were implemented a few machine learning techniques with a feature dimensionality reduction approach on this data set to determine the optimal number of features to maximise detection rate, but the PCA – LDA classifier only achieved 64.3 % of the detection rate when the features were reduced from 81 to 30. Sharafaldin et al. (2018) evaluate the performance of the existing data set, CICIDS 2017 classifiers, to find the best combination of features for detecting the attack. However, the ADA Boost and MLP only achieved a 77 % detection rate. Table 2.19 summarises the detection rate on the CICIDS 2017 data. The unbalanced class distribution among the attack classes resulted in a wide range of detection rates; for example, the detection rate for infiltration attacks ranged from 0 % to 83.33 %. While in the summarised table, the highest detection rate for web attacks (SQL & XSS) was only 50 %.

Table 2.19: Detection Rate of the Classifiers on the CICIDS 2017 Data Set

(Notes : the bold numbers indicate the minority class in the data set)

Paper	(Kurniabudi et al., 2020)			(Ahmim et al., 2018)			(Ravi et al., 2019)		Abdulhammed et al. (2019)	Sharafaldin at al. (2018)
Classifier	Bayes Net	J48	RF	Furia	LIBSVM	WISARD	LR	KNN	LDA	ADA & MLP
Normal	94.3	96.1	96	96.83	94.87	97.13	90.5	88.4	-	-
Brute Force	**99.1**	**79**	**79.2**	**49.8**	**80.81**	**4.7**	**-**	**-**	**-**	**-**
Heart Bleed	**-**	**-**	**-**	**40**	**0**	**80**	**-**	**-**	**-**	**-**
Botnet	**64.2**	**38.1**	**43.8**	**48.07**	**0**	**1.442**	**0**	**50.5**	**-**	**-**

DoS	99.6	99.1	99.2	71.76	71.5	23.35	93.4	99.5	-	-
DDoS				99.76	55.97	54.7			-	-
SQL	**3.1**	**7.2**	**7.2**	**50**	**0**	**0**	**8.6**	**8.3**	-	-
XSS				**38.75**	**0**	**1.25**			-	-
Port	99.2	99.5	99.5	87	48.51	51.4	99.4	100	-	-
Infiltration	**0**	**0**	**0**	**83.33**	**0**	**50**	**-**	**-**	**-**	**-**
FTP	-	-	-	99.6	0	0	48.8	94.5	-	-
SSH	-	-	-	80	0	0	49.9	97.9	-	-
Overall DR	96.2	-	96.5	90.50	54.60	48.18	87	90	64.3	77
Remarks	Implement information gain for feature selection Extracted 77 out of 84 features Utilise 20% of the data Used ten-fold cross validation			Removed null and duplicate data based on Flow Packet's feature. Extracted 20000 normal traffic and total of 20000 attack traffics as both training and testing data set.			Extracted 80000 records for benign traffic and total of 41,981 records for attack traffics in proportion to their distributions		Implemented PCA for feature selection. Reduce the dimensionality of features from 84 to 30 Split the whole data set into 70:30 ratio	Extracted 80 features from a pcap file using Random Forest Regressor

Moreover, Chitrakar and Chuanhe (2012) implemented a hybrid approach to evaluate the detection rate of Kyoto 2006 for IDS performance, integrating the k-Medoids technique with the Naive Bayes classification strategy. According to their findings, the applied techniques increased the detection rate from 66.51 % to 70.69 %. Ambusaidi, He and Nanda (2015) used the Kyoto 2006 to assess the IDS performance and found that the 1NN and SVM classifiers had a detection rate of 62.91 % for 1NN and 63.21 % for the SVM method. Other researchers (Vinayakumar et al., 2019) thoroughly evaluated DNN and other classifiers using the Kyoto 2006. They obtained a 52.6 % detection rate and 67 % f1 score for binary classification using the Naïve Bayes algorithm. Table 2.20 summarises the detection rate on the Kyoto 2006 data set.

Table 2.20: Detection Rate of the Classifiers on Kyoto 2006 Data Set

Paper	Classifier	Overall DR	Remarks
Chitrakar and Chuanhe (2012)	k-means + Naïve Bayes	66.51	Dataset distribution: - 238,179 records from 1,2,3 Nov 2007
	k-medoid + Naïve Bayes	70.69	
Ambusaidi, He and Nanda (2015)	1NN	62.91±	Dataset distribution: 100,000 records with full features from 27,28,29,30,31 August 2009. Implement Laplacian score algorithm to select features Each IDS run ten times and takes the average detection rate.
	SVM	63.21±	
(Vinayakumar et al., 2019)	Naïve Bayes	52.6	Dataset distribution extracted from year 2015: - 2,384,645 normal traffic for training set - 670,037 attack traffic for testing set - 1,405,391 normal traffic for training set - 158,523 attack traffic for testing set

Likewise, Kamarudin et al. (2017) presented a Unified Intrusion Anomaly Detection (UIAD) technique to identify unknown webserver attacks, and ISCX 2012 was one of the data sets they utilised to assess the approach. The SVM anomaly detection technique Nyakundi (2015) yielded a 72.70 % detection rate in the study, but the classic classifiers such as Decision Table only yielded a 47.75 % detection rate. Tan et al. (2015) evaluated their proposed EMD-based detection method using the ISCX 2012 data set. The results showed that their suggested method detected the DoS attack with a detection rate of 90.21 %, whereas traditional classification techniques such as Naive Bayes only achieved a detection rate of 51.54 %. Awad and Alabdallah (2019) employed the WELM and SVM algorithm with polynomial function, and the ISCX 2012 data set was used to test the technique performance on IDS. The detection rate for one of the minority attacks classes in this data set (DoS) only achieved 25 % for SVM and 37.78 % for WELM. Furthermore, botnet attacks

have the lowest detection rate in WELM, which is 0 %. Table 2.21 summarises the detection rate on the ISCX 2012 data set.

Table 2.21: Detection Rate of the Classifiers on ICXS 2012 Data Set

Paper	Classifier	Overall DR	Remarks
Chitrakar and Chuanhe (2012)	Decision Tree	47.75	Dataset distribution: - 8000 records were used in training data while a total of 34,957 records for testing data. - Extracted date from 11 June 2010 to 17 June 2010 Extracted four features (f2,f3,f9,f11).
Nyakundi (2015)	SVM	72.70	Dataset distribution: - Extracted 10% of all labelled data as test set. Which was 13391 records - Extracted 1% of the entire labelled dataset as training set, 1340 records.
Tan et al. (2015)	Naïve Bayes	51.54	Implement PCA for feature selection. Adopted Tuesday network records that contains 8,720 attack flows.

Table 2.22: Detection Rate of the Classifiers on CICDDOS 2019 Data Set

(Notes : the bold numbers indicate the minority class in the data set)

Paper	(Chartuni and Márquez, 2021)	(Ferrag et al., 2021)		(Scaranti et al., 2020)		(Jia et al., 2020)		(Adhao and Pachghare, 2021)		
Classifier	Proposed NN	DNN	CNN	NB	RF	NB	LR	NB	SVM	DT
Normal	95	100	100	-	-	-	-	-	-	-
DNS	**42**	**61**	**58**	-	-	-	-	-	-	-
LDAP	**56**	**30**	**30**	-	-	-	-	-	-	-
MSSQL	92	67	67	-	-	-	-	-	-	-
NTP	95	61	52	-	-	-	-	-	-	-
NetBIOS	**82**	**47**	**46**	-	-	-	-	-	-	-
SNMP	**83**	**93**	**73**	-	-	-	-	-	-	-
SSDP	**28**	**55**	**55**	-	-	-	-	-	-	-
UDP	77	47	46	-	-	-	-	-	-	-
Port Map	**54**	**93**	**73**	-	-	-	-	-	-	-
Syn	**100**	**64**	**65**	-	-	-	-	-	-	-
TFTP	100	100	94	-	-	-	-	-	-	-
UDP Lag	**100**	**99**	**97**	-	-	-	-	-	-	-

WebDDoS	-	**23**	**20**	-	-	-	-	-	-	-
Overall DR	94.38	96.8	95.54	16.91	93.49	11	2	57.91	79.76	74.03
Remarks	- remove non-informative attributes - remove null data, with NaN value, contributes to 3%.	- 75 training and 25 test samples from the full data set. - use full features.		- implement feature selection, same feature found in NSL KDD		- self defined a flow handler to set the filtration rules.		- pre-processing stage involve normalisation, and feature selection using Krill Herd.		

Table 2.22 summarises the detection rate on the CICDDOS 2019 data set. Chartuni and Márquez (2021) proposed a neural network to increase the detection rate for Denial-of-Service (DDoS), achieving a high overall detection rate of 94.38 %, despite some minority classes having a low detection rate, such as SSDP with 28 % and DNS with 42 %. Ferrag et al. (2021) evaluated their proposed deep learning-based IDS for DDoS in Agriculture 4.0 with CICDDOS 2019. The applied techniques increased the overall detection rate to 96.80 %, while some minority classes had a low detection rate, such as LDAP achieved a 30 % detection rate and WebDDoS obtained a 20 % detection rate. Subsequently, other researchers only provide the overall detection rate, which ranges from 2 % in Logistic Regression (Jia et al., 2020) to 93.49 % in Random Forest (Scaranti et al., 2020).

2.1.8 Data set Selection

According to Ring et al. (2019) , an ideal data set should contain real network traffic that is up-to-date, accurately labelled, and provides comprehensive traffic information such as user behaviour and payloads. However, such an ‘ideal’ data set probably never be created due to privacy concerns. As a result, five data sets were investigated to evaluate if the unbalanced distribution problem was presented. Ultimately, four data sets were selected to evaluate the efficacy of over sampling techniques in mitigating the rare class problem in the network intrusion data sets. It aimed to avoid the classifiers from overfitting to a specific data set. The chosen data sets were CICIDS 2017, UNSW-NB15, NSL KDD and Consolidate CICDDOS 2019.

Table 2.23 summarises the information of examined network intrusion data sets. Kyoto 2006 was not chosen because this large data set lacks a standardized set, where the attacks are even unclassified. Hence, it is difficult to be used for comparison. Furthermore, ICXS 2012 has a significant number of data set files, and the majority of researchers did not explicitly specify the distribution utilised in their studies. Additionally, most studies report the overall accuracy for ICXS 2012 rather than the detection rate for individual classes. Therefore it is hard to compare the results as the detection rate for each class is not provided.

Among the investigated data sets, UNSW-NB15 and CICIDS 2017 had a high unbalanced class ratio between normal and attack traffic (8:2). It also contains a wide range of specific attack types. Therefore, these two data sets were chosen for further investigation. The consolidated CICDDOS 2019 and NSL KDD also have an unbalanced class ratio among the normal and attack classes. As a result, these two data sets were also selected to evaluate the methods. There will be further justifications for the data set selection in chapter four.

Table 2.23: Overview of Network Intrusion Data Sets.

	NSL KDD	**UNSW-NB15**	**CICIDS17**	**Kyoto2006**	**ISCX 2012**	**CICDDOS 19**
Year	2009	2016	2017	2006-2009	2012	2019
Number of features	41	47	84	24	19	77
Number of attacks covered	4 categories (21 attack types)	9	11	-	4 categories	12
Attack types	R2L, DoS, U2R, Probe	Analysis, Backdoors, DoS, Reconnaissance, Generic, Shellcode, Exploits, Worms and	Brute Force, Heart Bleed, Botnet, DoS, DDoS, SQL injection, infiltration,	Known attack, unknown attack	Botnet, SSH, infiltration, DoS	NTP, DNS, LDAP, MSSQL, NetBIOS ,SNMP. SSDP, UDP, UDP-Lag, WebDDoS

		Fuzzers	port scan, XSS, ftp and SSH			, SYN, TFTP
Traffic environment	emulated	emulated	emulated	real	emulated	emulated
normal-to-attack ratio	52%-48%	87%-13%	83%-17%	54%-46%	96.3%-3.7%	25%-75%
Unbalance conditions	Among attack classes	Between normal and attack traffic, among attack classes.	Between normal and attack traffic, among attack classes.	Among attack classes	Between normal and attack traffic, among attack classes.	Among Attack classes

2.2 Techniques to mitigate the rare class problem

Most of the standard classifiers are accuracy driven, and they presume that the data sets distribution is always evenly distributed. As a result, unbalanced class distribution becomes one of the issues for classification. In other words, many classification algorithms strive to reduce total error while increasing accuracy. Nonetheless, the accuracy of an unbalanced data set is often driven by the majority class and provides little insight into the minority class (attack classes). These may result in erroneous and misleading information regarding the classifier's performance. In this section, possible solutions to mitigate the rare class problem in the data set were identified, and four oversampling techniques were chosen and implemented to find out which technique generally performed well for the selected network intrusion data sets.

2.2.1 Overview of possible techniques

There are three main approaches to mitigate the rare class problem in an unbalanced data set: Data-level, Algorithm-level, and hybrid techniques.

2.2.1.1 Algorithm level techniques

Algorithm-level techniques focus on modifying existing learning algorithms to reduce bias towards majority classes and skewed the data before mining them. This technique demands a comprehensive understanding of the revised learning algorithm as well as the underlying cause of its failure to handle unbalanced distribution. This technique can be sub-grouped into one class learning method, ensemble learning method and cost-sensitive method. Table 2.24 tabulated the overview for the algorithm level methods.

One-class learning method models the algorithm on the representation of rare class; it is also known as the recognition-based method. It solely focuses on learning from the example of the rare class rather than the difference pattern between dominant and rare classes. However, Ali et al. (2013) pointed out that the crucial part of this method is its efficacy in determining the border threshold since the boundary threshold would divide the distinct classes. Ripper is one example of this approach; it is a rule induction system that uses a divide-and-conquer technique to learn minority class rules repeatedly until all rules have been built (Kanellopoulos, Kotsiantis and Pintelas, 2016).

Cost-sensitive learning method is designed to impose varying penalties or costs on a classifier when misclassification occurs. It takes the form of a cost matrix, with rare classes having a greater cost to emphasise their value during the learning process. It will choose the class with the least conditional risk to minimise the cost of misclassification and no cost will be imposed on the correct classification. The difficulty with this approach is that finding the real cost matrix is challenging due to the huge number of elements to examine and searching for the effective cost will result in overhead in the cost learning task. (Krawczyk, 2016).

Ensemble learning is a hybrid method that involves training two or more classifiers and then making a final classification decision based on their evaluations. Bagging and boosting are the two most popular methods for ensemble learners. Boosting is the process of iteratively adjusting the weights of each classifier

depending on their misclassifications to construct a strong classifier. The goal of bagging is to minimise the variance of the final classifier by combining many algorithms in training rather than using a single algorithm. The ensemble learning method avoids the problem of overfitting. However, compared to traditional classifiers, it generally has a greater computing cost (Leevy et al., 2018).

Table 2.24: Overview of Algorithm Level Techniques.

	One-class learning	**Ensemble learning**	**Cost sensitive**
Description	Model the classifier to learn from minority class pattern.	Combine 2 or more algorithm to produce one final prediction.	Imposed cost/penalty on learning algorithm when misclassification occurs.
Pros	- Useful to deal with a data set that have high dimensional noisy. - costless to implement compared to feature selection method	- avoid overfitting problem - able to decrease variance, bias and improve predictions. - versatile approach	- achieve good performance for cost sensitive algorithm. - important to tackle classifier problem and allow further improvement.
Cons	- Specialized methods are needed for more complex problem - Only focus with examples from 1 class	- higher computational cost and complexity grows with more classifiers involved. - does not assist to improve generalization performance for robust learning algorithm.	- difficult to identify the cost for matrix - additional cost incurred to explore for effective cost matrix
Approach	Ripper	Bagging, Boosting	CSARF, cost sensitive SVM

2.2.1.2 Data level techniques

Data level techniques modify the data set samples to balance the data distribution for the standard classifier. It is used in the pre-processing step to rebalance the distribution of classes. Resampling methods and feature selection methods are two subcategories of this approach. Table 2.25 tabulated the overview for the data level techniques.

The purpose of the data resampling approach is to alleviate the rare class problem by changing the size of the data distribution for pre-processing. It uses techniques such as Undersampling, Oversampling, and Hybrid sampling. Oversampling increases the data distribution of rare classes, and random sampling is the standard approach in this method. However, it often introduces useless new samples due to the duplication record, which can lead to overfitting of the classifier. Therefore, advanced sampling methods were introduced to remove the noisy instances that may hinder the classifier performance. In contrast, under-sampling method is used to reduce the data distribution of majority classes at random to reduce the disparity between two classes. Under-sampling can sometimes result in the loss of valuable information by eliminating large samples from dominant classes. The hybrid method is to apply the combination of both Undersampling and Oversampling techniques to manipulate the data set distributions (Krawczyk, 2016; Leevy et al., 2018).

The most significant or influential attributes that might offer unique information for inter-class discrimination are chosen using the feature selection method. It improves classifier performance by reducing the negative impacts of rare class problems. Two feature selection methods are the filter and wrapper algorithms. The filter approach examines the intrinsic features of the data to determine the usefulness of the feature, whereas the wrapper approach explores the feature with an algorithm. The feature selection approach reduces the dimensionality of the classifier, but it is computationally expensive to implement (Ali et al., 2015; Leevy and Khoshgoftaar, 2020; Sleeman and Krawczyk, 2020).

Table 2.25: Overview of Data Level Techniques.

	Data Resampling	**Feature Selection**
Description	Modify the size of data distribution, either upsizing the minority class or downsizing the majority class.	increase the accuracy by performing feature selection on the attributes.
Pros	- easy to implement compared to algorithm level technique - feasible method to handle large data sets with unbalanced class distribution.	- avoid the classifier suffer from dimensionality problem. - useful to mitigate class overlapping problem.
Cons	- over sampling might cause overfitting problem - under sampling method might loss important information from the data sets.	- computational expensive.
Approach	Over sampling – SMOTE, ADASYN, random sampling Under sampling – RUS, informed under sampling, near miss Hybrid – SMOTE + Tomek links	Wrapper, filter

2.2.1.3 Hybrid techniques

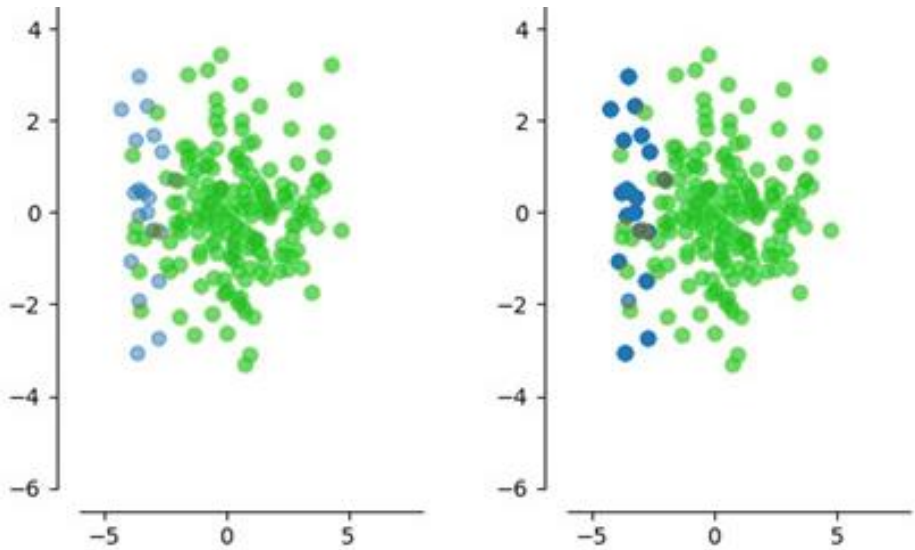
It is a method for building a robust and efficient learner by combining the strengths of both data-level and algorithm-level techniques. It often combines data sampling techniques with ensemble learning and it is called hybridisation that combined multiple learning algorithms to improve the results. It is designed to mitigate the problem in both techniques as well as fine-tuning the standard learning algorithms.

However, a careful selection is required to complement the variations across applied methods. Fuzzy classifier is one of the hybrids, data-driven algorithms that learn from the data distribution patterns. (Krawczyk, 2016; Rout Neelamand Mishra, 2018).

2.2.2 Oversampling techniques

Oversampling strategies are the focus of this research because under-sampling techniques remove informative data from data sets. This approach involves replicating instances from minority classes to get a more balanced class distribution in the data set. There are two ways in over-sampling to increase the minority class instance: replicating the instances from existing data sets or creating synthetic data based on the feature space similarities across minority classes (Dattagupta Jayanta, 2018). This section covered a few widely used sampling methods, including ROS, SMOTE, SMOTE-borderline, SMOTE-kmeans, MWSMOTE, and ADASYN.

2.2.2.1 ROS



(a) **(b)**

Figure 2.3: Illustration of the Replication of New Sample by ROS in Feature Space : (a) Original Feature Space, (b) Oversampled Feature Space (Adopted from (Lemaitre et al., 2016))

The most basic approach for oversampling techniques is ROS. This technique balances the class distribution by reproducing randomly selected instances from the

minority class in the feature space until both the majority and minority ratios are equal (Suh et al., 2017). Figure 2.3 Illustrate the replication of a new sample by Random Over Sampling in feature space. It shows that the minority class (blue plots) had more duplicated data points in the right scatter plot (b) as the blue colour plots were denser than the left scatter plot (a). This technique is commonly used since it is simple to apply and does not result in information loss when compared to the under-sampling method.

However, the major drawback of this technique is that the boundary between classes is difficult to generalise since it creates identical replication of data points from minority classes. It causes the class distribution shifts as it involves many iterations for the replication process and results in overfitting to the training data. Additionally, ROS introduces an additional computational task on the large unbalanced distributed data set. As a result, more advanced techniques such as SMOTE were developed to mitigate the problem with this method (Dattagupta Jayanta, 2018).

2.2.2.2 SMOTE

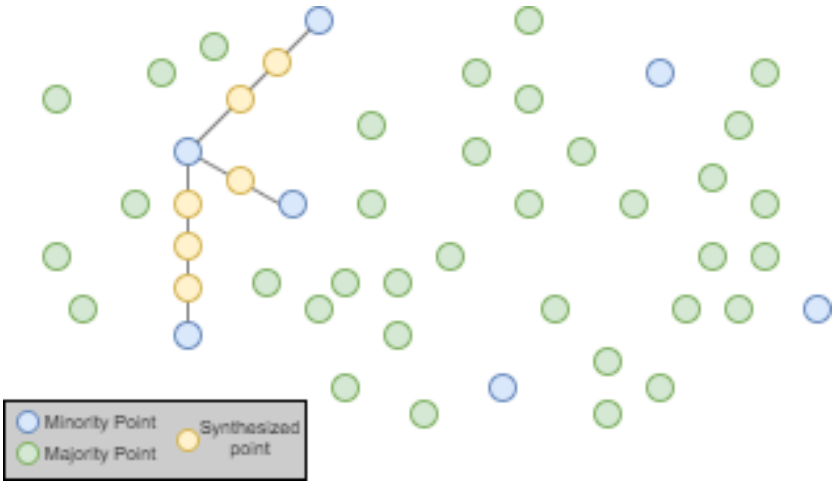
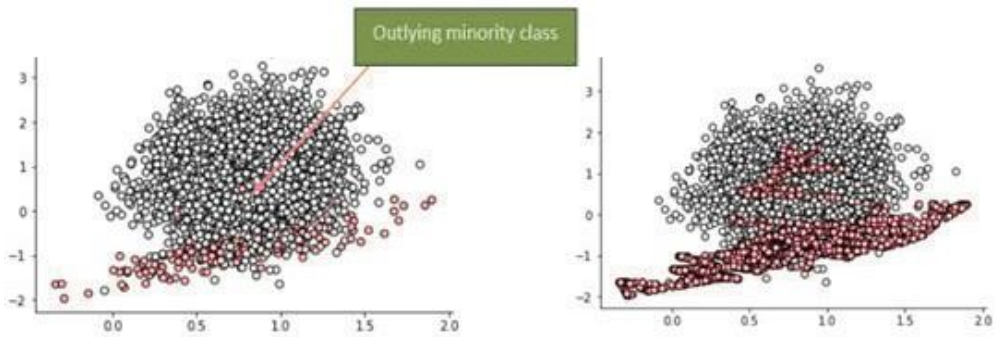


Figure 2.4: Illustration of the Synthesis of a New Sample by SMOTE in Feature Space

(Adopted from (Vivek Vinushanth, 2020))

Chawla et al. (2002) proposed the first SMOTE method. It generates synthetic minority data points in linear interpolation based on the feature space

similarities between rare classes. The linear for interpolation was determined using the KNN technique, in which the neighbours in KNN are randomly selected based on the amount of oversampling that is required (Amin et al., 2016). Figure 2.4 illustrates the synthesis of new data points generated by SMOTE in feature space. For every rare data point, its nearest neighbours (k = 3) from the same class are identified, and a linear is formed between those 2 data points. After that, a new synthetics data point is randomly generated along the linear according to the oversampling rate. Fernández et al. (2017) suggested that this method can avoid the overfitting problem that occurs in ROS since it creates new synthetic instances instead of replicating them. The addition of synthetic data enhances the overall data distributions' balance and expands rare class decision boundaries into other class spaces.



(a) **(b)**

Figure 2.5: Overlapping Issue in SMOTE

: (a) Before Generated Synthetic Data, (b) After Generated Synthetic Data

(Adopted from (Saptarsi, 2020))

Figure 2.5 depicts different examples in the multi-class unbalanced data set. If the data point's nearest neighbour is one of the outliers, it is conceivable that some of the synthetic data points were overlapped with the instance of another class using this technique. As a result, some data points may fall into the wrong region if their nearest neighbours are far apart, as illustrated in Figure 2.5 (b). This might lead to an over generalization of minority class space. Moreover, this technique becomes less effective when dealing with high dimensional data (Dattagupta Jayanta, 2018; Ali, Salleh and Hussain, 2019). Hence, similar to the ROS technique, more variants of SMOTE techniques have been introduced to improve this technique.

2.2.2.3 Borderline-SMOTE

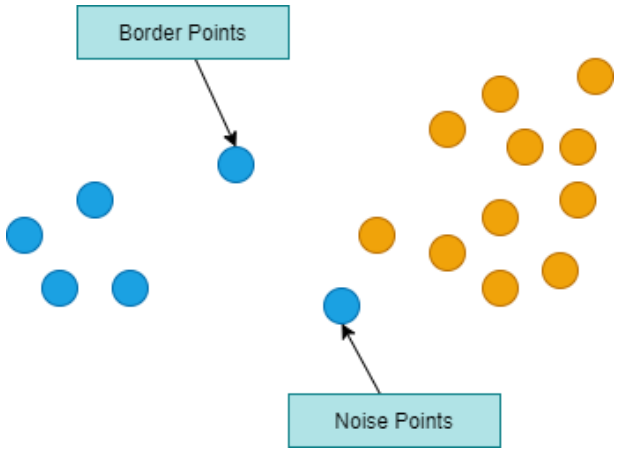


Figure 2.6: Factor to Determine Border Boundary in Borderline-SMOTE (Adopted from (Saptarsi, 2020))

Borderline-SMOTE is an extensions technique from SMOTE introduced by Han, Wang and Mao (2005). It also generates synthetic data points from the minority class, but only when the data point is near the decision border, rather than the whole minority class (Sáez, Krawczyk and Woźniak, 2016). Figure 2.6 depicts the points used by Borderline-SMOTE to establish the minority class decision border. Assuming that the blue point in Figure 2.6 represents a minority class, the top blue point is utilised to construct a decision boundary between its class and other classes, resulting in all synthetic points being generated near the boundary region. Nevertheless, towards the bottom, there is a rare class point that is close to another class space. Hence this noisy point was disregarded during the boundary formation for generating synthetic data.

There is another variation of Borderline-SMOTE, where the difference is that version 2 Borderline-SMOTE uses both positive and negative nearest neighbours. It solves the problem in SMOTE by preventing the algorithm to create synthetic data points from noise instances, which helps to create a better and clearer boundary between classes (Dattagupta Jayanta, 2018). However, this technique also has its drawback. Since this technique only creates the synthetics data points near the border points, it focuses on extreme values and might ignore certain key instances in the minority class.

2.2.2.4 MWMOTE

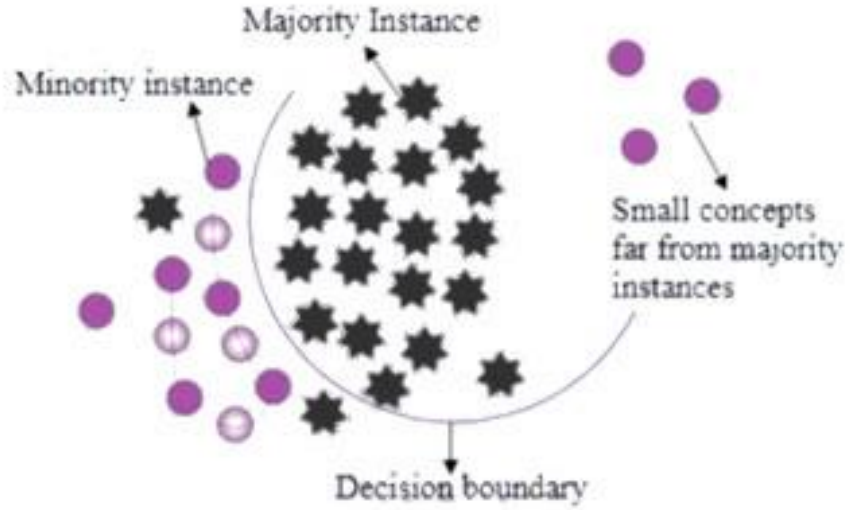


Figure 2.7: Small Concept in MWMOTE that are Not Oversampled

(Adopted from (Ali, Salleh and Hussain, 2019)

Barua et al. (2012) proposed MWMOTE, a strategy that improves on SMOTE to solve the problem of unbalanced class distribution. This method clusters the data points and weights the minority data points depending on their Euclidean distance from the majority data points. It generates synthetic data between the boundaries of two classes performed KNN multiple times to determine the borderlines. Figure 2.7 depicts the small concept in MWMOTE. In this small concept, small disjoints of the minority class far from the majority distance are ignored and not oversampled. This method can mitigate the overlapping problem in SMOTE and the overfitting problem by clustering. However, this small concept also results in a within-class unbalanced problem, because it might cause the classifier to be biased towards the oversampled majority sub-clusters (Ali, Salleh and Hussain, 2019).

2.2.2.5 K-means SMOTE

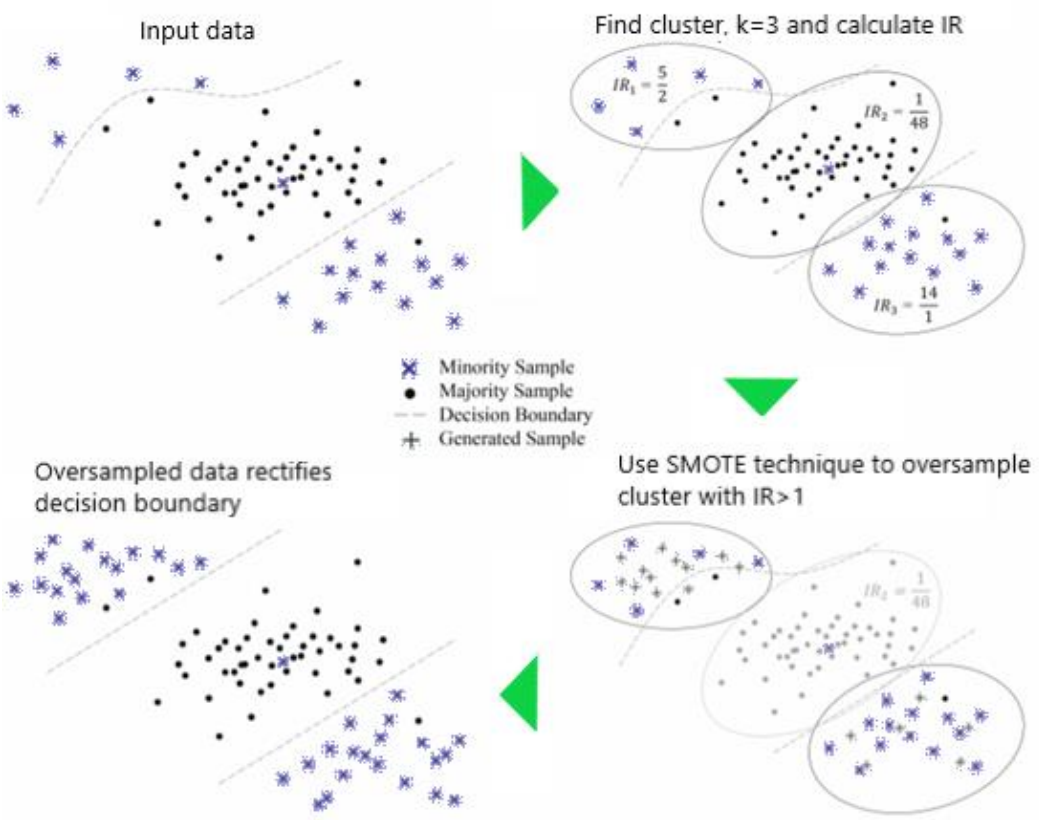


Figure 2.8: K-means SMOTE Sampling Process
(Adopted from (Douzas, Bacao and Last, 2018))

Douzas, Bacao and Last (2018) introduced another technique based on SMOTE, k-mean SMOTE. It uses the density factor that combines K-means clustering with SMOTE to generate synthetic data. It focuses on the unbalanced problem not just between dominant and rare classes, but also within rare classes. Figure 2.8 depicts the process of K-means SMOTE in generating synthetic data for the minority class. Clustering, filtering, and oversampling are the three phases of this technique. Firstly, the K-means method is used to group similar data points into clusters that were represented in Euclidean space repeatedly. The algorithm will then begin to converge if no further observations are reallocated.

The filtering process will find out if the minority cluster is oversampled and determine the amount of synthetic data to be created for each cluster. This phase gives sparse minority clusters more synthetic data than dense ones. The imbalanced

ratio is calculated after filtering the cluster to decide which cluster should implement the SMOTE technique:

$$imbalance\ ratio\ per\ cluster\ (IR) = \frac{majority\ count\ (c) + 1}{minority\ count\ (c) + 1} \tag{2.1}$$

After determining the ratio, the SMOTE technique will be used to oversampled clusters with an IR larger than 1 to produce synthetic data points until the needed amount of synthetic data is obtained. Eventually, the oversampled data rectify the decision boundary. In a nutshell, this technique implemented clustering for input data and generated synthetic samples depending on the cluster's weight. (Ali, Salleh and Hussain, 2019). This method is easy to implement, and the overlapping problem was solved by identifying the clusters with IR ratios. However, it is less efficient when dealing with big data sets, as K-means clustering may be sluggish.

2.2.2.6 ADASYN

Table 2.26: Example of ADASYN Impurity Ratio

(Adopted from (Saptarsi, 2020))

Minority Class	Minority Neighbours	Majority Neighbours	Impurity Ratio
O1	3	2	0.4
O2	4	1	0.2
O3	1	4	0.8
O4	5	0	0

He et al. (2008) proposed the ADASYN approach, which is based on the SMOTE technique. It is a probabilistic oversampling technique that uses the distribution density for different minority instances to determine their learning difficulty and how much synthetic data should be generated. It utilizes the KNN technique to locate the nearest neighbours of the minority data point and determine the impurity ratio for that observation.

This strategy uses a weightage approach to apply higher weights to the minority cases that are difficult to learn, resulting in more synthetic data being

created. In other words, the more majority nearest neighbour, the more synthetic data will be generated. The goal of this strategy is to avoid misclassification of difficult-to-learn minority cases by prioritising them. (Ali, Salleh and Hussain, 2019). Table 2.26 shows an example of the ADASYN impurity ratio.

$$impurity\ ratio\ (IR) = \frac{majority\ neighbour}{total\ neighbours} \tag{2.2}$$

The impurity ratio is transformed into probability distribution that describes the fraction of minority class in its neighbour space using the KNN method. Since O3 has the greatest impurity ratio of all the observations, more synthetic data points will be created for this group.

In contrast to Borderline-SMOTE, it generates synthetics data points based on the density of minority class rather than only depends on the boundary instances. Therefore, the boundary is much softer than Borderline-SMOTE. However, this technique does not identify the noisy instances and hence it is easily influenced by outliers in the data set. Furthermore, this technique may miss some of the rare instances located along the borderline. (Dattagupta Jayanta, 2018; Wei et al., 2020).

2.2.3 Summary: Pros & Cons

Researchers have proposed many oversampling approaches, especially SMOTE technique, having over 100 variants. In this section, the characteristic of the researched oversampling techniques was identified and compared (Kovács, 2019).

C1 Ordinary sampling – to oversampled data that locates along the line segment linking neighbouring minority instances.

C2 Borderline – to identify the minority data points near the decision border and generate new samples near the boundary.

C3 Use Clustering – clustering method was used to cluster the minority class and implement the oversampling technique to the identified cluster independently.

C4 Density-Based – apply weightage technique to minority class, then normalise the weighted score to determine the sampling density.

The summary for the researched oversampling strategies was tabulated in Table 2.26. Random oversampling is the standard oversampling technique that is widely used before more sophisticated techniques were introduced. It is an easy and

rapid approach. However, since the data was replicated from original rare instances, it may cause overfitting to the training data. As a result, SMOTE was developed to address the ROS's overfitting problem by generating synthetic data utilising the connecting concept amongst its neighbours. However, it presents overlapping problems between classes since this technique does not manage noise in the synthesis process and performs slower than ROS. Then, similar to ROS, more techniques have been introduced based on SMOTE to mitigate its shortcomings.

Borderline-SMOTE is designed to address the overlapping issue from SMOTE by creating new samples along the class border. However, it may cause information loss in minority classes because it concentrates on extreme observations. K-means SMOTE that uses the clustering method to identify classes before oversampling, helps to prevent interpolation generalization. Nonetheless, it is time-consuming to implement for large data sets and possible to overlook important data points near the boundary. In addition, MWMOTE uses clustering method with borderline concept able to mitigate the overfitting and class overlapping issues. However, it may lead the classifier tends to oversample the majority sub-cluster in minority class and lose information from the sub clusters that are distant from majority classes.

Lastly, ADASYN combines the clustering method with density-based decision making to create better decision boundaries and gives more attention to difficult-to-learn minority classes, yet it is susceptible to outliers. In a nutshell, more algorithms are being developed to improve upon one another, but fewer algorithms can efficiently deal with noise in feature space. Dealing with noise directly impacts over sampling effects since it affects the distribution of new data points.

Table 2.27: Summary for Oversampling Techniques.

Technique	**Data create based on**	**Data Origin**	**Time Taken (Rank)**	**Characteristics**	**Pros**	**Cons**
ROS	1 minority data instance	existing data	1	- replicate original data	- easy to implement - it does not lead to information loss compared	- may cause overfitting - difficult to generalise the borderline

					to under sampling	between classes
SMOTE	Some minority data around new data	synthetic data	3	C1	- mitigate overfitting problem in ROS. - expand decision boundary for rare classes	- might cause overlapping of classes. - might introduce additional noise. - less effective for high dimensional data.
Borderline-SMOTE	Similar to SMOTE but only concentrate at the near boundary	synthetic data	2	C1, C2	- have better decision boundary between classes. - solve the overlapping problem	- might miss useful examples in a minority class - only focus on extreme observations
K-means SMOTE	Minority data by rectifying and exclude majority data	synthetic data	5	C3	-prevent interpolation generalization. - avoid overfitting problem - easy to implement	- time consuming to implement for a large data set - easy to ignore important boundary data points.
MWMOTE	synthesis minority sub-cluster data after clustering	synthetic data	6	C2, C3	- overcome the overlapping and overfitting problem.	might cause classifier tends toward oversampled sub-cluster - loss important

						information from farther sub-cluster.
ADASYN	Both minority and majority data	synthetic data	4	C1, C2, C4	- softer boundary decision compared to borderline-SMOTE - more attention is given to hard-to-learn minority class	- does not identify noise - easily affected by outliers.

2.2.4 Oversampling Technique selection

All in all, there are two types of oversampling techniques: self-replication and producing synthetic data. In this research, five oversampling techniques were chosen after identifying and comparing several oversampling techniques to assess their effectiveness on unbalanced network intrusion data sets. Those selected techniques were ROS, SMOTE, Borderline-SMOTE, ADASYN and K-mean Smote. Firstly, Random Over Sampling was chosen because of its easy and fast implementation, which is crucial because the time taken is also another performance factor. Furthermore, SMOTE was also chosen because of its ease of implementation and to find out how well this technique can handle high dimensionality data to what extent.

In addition, Borderline-SMOTE can mitigate both SMOTE and ROS issues such as unclear decision boundaries and overfitting problems. As a result, this technique was chosen as one of the techniques to be evaluated. Subsequently, ADASYN was chosen as the last technique to be evaluated since it incorporates the density-based concept to weight the minority class for creating synthetic data. Therefore, it can be used to solve the rare-class problem for the within-class distribution. Lastly, K-mean SMOTE is also included in this research.

2.3 Performance Measures

The main aim to mitigate the unbalanced class distribution problem is to reduce the misclassification of minority classes and increase its detection rate while maintaining

satisfactory performance regarding the majority class. In this section, available performance measures for classifiers were studied to evaluate the effect of the oversampling techniques on mitigating the unbalanced class distribution problem for the network intrusion data set. Performance matrices such as precision, recall, f1 score, accuracy and confusion matrix were discussed, and a set of suitable matrices were identified for the evaluation process.

2.3.1 Confusion Matrix

Table 2.28: Confusion Matrix for Binary Classification Problem.

	Predict Positive	**Predict Negative**
Actual Positive	TP	FN
Actual Negative	FP	TN

The confusion matrix is a 2x2 matrix that allows us to compare actual and predicted classes conveniently. It provides necessary values for developing several performance measurements: TP, TN, FP, FN. (Salo et al., 2018; Leevy and Khoshgoftaar, 2020). In binary classification problems, the more significant label is usually chosen as positive, while the less important label is usually chosen as negative. In the following example, attack traffic is labelled as positive and normal traffic as negative.

True Positive (TP) – represents the number of positives samples accurately classified as positive. For example, an actual attack is predicted correctly as an attack.
True Negative (TN) – represents the number of negative data accurately classified as negative. For example, a normal is predicted as normal.
False Positive (FP) – represents the number of positive data inaccurately classified as negative. For example, an actual attack is predicted as normal. It is also known as a type I error.
False Negative (FN) – represents the number of negative data inaccurately classified as positive. For example, an actual normal is predicted as an attack. It is also known as type II error.

2.3.2 Evaluation Matrix

2.3.2.1 Accuracy

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.3}$$

$$error = 1 - accuracy \tag{2.4}$$

Accuracy is used to find the effectiveness of a classifier to accurately predict an object, either positive or negative from the whole test data set. In other words, it is to find out the correct predictions over total predictions. In contrast, the error is the inverse of accuracy, which predicts a positive instance as negative and vice versa (Vinayakumar et al., 2019).

2.3.2.2 Precision

$$precision - \frac{TP}{TP+FP} \tag{2.5}$$

Precision measures how effectively a classifier can accurately predict the number of positive samples that is actual in a positive class. It represents the true positive divided by the total positive instances. It is used to measurement for classifier precision, a low precision rate indicating a high FPR value (Abdulhammed et al., 2019).

$$Macro\ average\ precision - \frac{P1+Pn\ldots}{n} \tag{2.6}$$

Another performance metric is macro average precision, which calculates precision metrics for each class separately. It sums up the precision value for n class and divides it by n classes to calculate the average (Bagui et al., 2019).

$$Micro\ average\ precision - \frac{TP1+TPn\ldots}{TP1+TPn+FP1+FPn\ldots} \tag{2.7}$$

Micro average precision is to take the sum of true positive from n class and divided by the total positive observations from n classes.

2.3.2.3 G-mean score

$$g - mean = \sqrt{\frac{TP}{P} \times \frac{TN}{N}} = \sqrt{sensitivity \times specificity} \tag{2.8}$$

G-mean score refers to the geometric mean that defined as the geometric mean of sensitivity and specificity. This considers each measure is equally important and refers to them as per-class accuracy (Douzas, Bacao and Last, 2018).

2.3.2.4 FPR / FAR

$$FPR - \frac{FP}{FP+TN} \tag{2.9}$$

FPR is known as fallout score or false alarm rate. It represents the number of negative instances that are mistakenly classified as positive. It is the ratio of normal traffic classified as attack traffic to the overall amount of normal traffic. A lower FPR indicates that the classifier is more effective (Abdulhammed et al., 2019).

2.3.2.5 F measure

$$F\ measure - \ 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.10}$$

F measures combine both precision and recall matrix into a single score to find out the harmonic mean between them. It is a metric for evaluating a classifier's performance; a higher F1 score implies that the classifier is doing better (Salo et al., 2018; Vinayakumar et al., 2019).

2.3.2.6 Recall

$$recall - \frac{TP}{TP+FN} \tag{2.11}$$

Recall, also known as sensitivity or TPR, describes how well the positive instances are predicted correctly out of possible positive instances, also known as the detection

rate, which represents the number of attacks detected as a percentage of the overall number of attacks. The greater the classifier's performance, the higher the TPR rate (Douzas, Bacao and Last, 2018; Panigrahi and Borah, 2018).

2.3.2.7 ROC

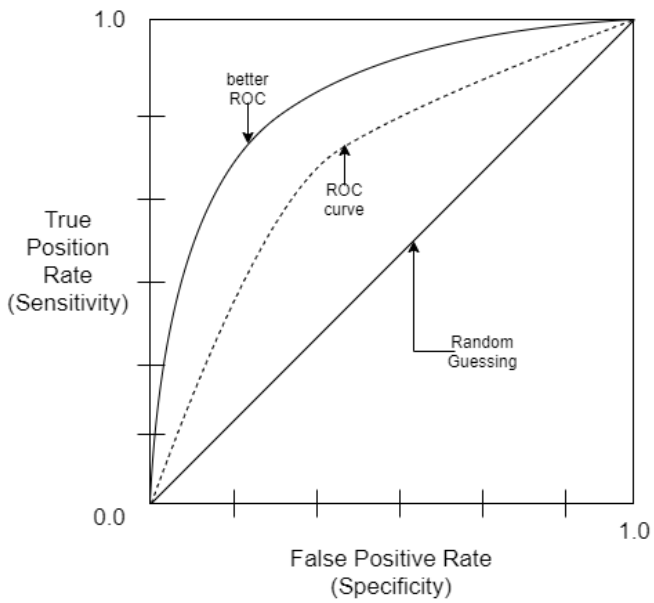


Figure 2.9: Example of ROC Curve

(Adopted from (Lu, Li and Chu, 2017; Ali, Salleh and Hussain, 2019))

ROC is a model performance evaluation to see how capable the modal can differentiate the classes in data sets by plotting the TPR over FPR. It is a graphical presentation that plots the trade-off between intrusion detection rate against the FAR. A strong IDS performance will lead the curve to tend toward the upper left, as shown in Figure 2.9.

2.3.2.8 AUC

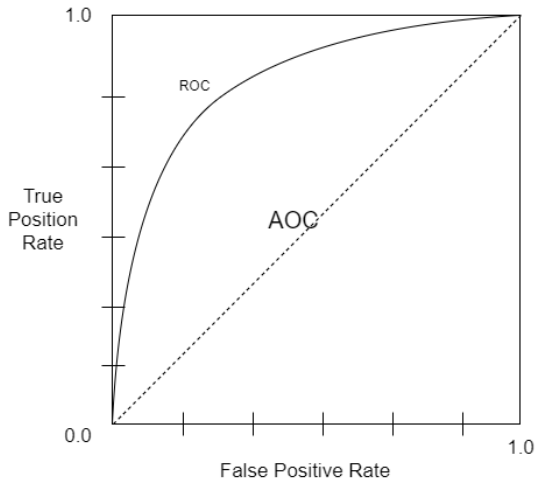


Figure 2.10: Example of AUC

$$AUC = \frac{1+TP-FP}{2} \tag{2.12}$$

The area under the ROC curve is indicated by the AUC curve, which is a matrix that correlates to the ROC curve. The higher the AUC values, the better the classifier's performance, and it is a popular matrix for dealing with the effects of unbalanced class distribution (Lu, Li and Chu, 2017; Leevy et al., 2018).

2.3.2.9 Kappa

$$k = \frac{p_0 - p_e}{1 - p_e} \tag{2.13}$$

Kappa is to measure the agreement between actual and predicted classes where p_0 is the predicted agreement while p_e is expected one (Salo et al., 2018).

2.3.2.10 MCC

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP+FP) \times (TP+FN) \times (TN+FP) \times (TN+FN)}} \tag{2.12}$$

The MCC is used to measure the correlation between predicted outcomes with actual data. It has values between -1 to 1 and a positive 1 indicates 100 % accuracy in prediction and vice versa (Salo et al., 2018).

2.3.2.11 Runtime

It is used to calculate the amount of time it takes for an oversampling technique to complete its operation in seconds.

2.3.3 Performance Matrix Selection

Accuracy is a common performance matrix for evaluating classification performance. However, it is not a suitable evaluation method for unbalanced data set since it will indicate a bias toward the majority classes. In such a situation, high accuracy indicates that the classifier can reliably predict the majority class but not the minority class (Douzas, Bacao and Last, 2018). Same for precision, recall and f1-score, another type of comparable matrix shall be used to handle the unbalanced data set, ensuring that the performance matrices will not bias towards the bigger distribution. Few evaluation matrices are selected to evaluate oversampling performance.

Weighted metrics aggregate the contributions of all classes to calculate the average by assigning a weightage according to the data distributions. It gives a holistic perspective of how well the classifier is handling the unbalanced data set distribution. However, a single metric is insufficient for evaluating the oversampling technique, so weighted f1 measure and weighted recall (detection rate) are employed. The F1 measure was chosen because it better reflects the performance of a classifier since it represents the trade-off between precision and recall. (Lu, Li and Chu, 2017). Finally, the time it takes for the oversampling technique to perform its operation is considered because some techniques may require speedy retraining, resulting in rapid oversampling. In a nutshell, this research utilised weighted f1 measure, recall (detection rate) and runtime to evaluate the performance of the oversampling approach. The optimal oversampling approach shall achieve the highest detection rates with a considerable runtime.

CHAPTER 3

METHODOLOGY AND WORK PLAN

This research employed a modified CRISP-DM since it is a more comprehensive methodology. It is also a cyclic technique that allows researchers to revert to previous phases if necessary. Furthermore, this methodology allows the researcher to conduct the research systematically, reducing the likelihood of the project going astray. In short, this method is implemented because it provides a consistent framework for managing a project's lifecycle with best practises, regardless of its domain. The detailed workflow and algorithm for selected oversampling techniques are explained in this chapter.

3.1 Summary of the workflow

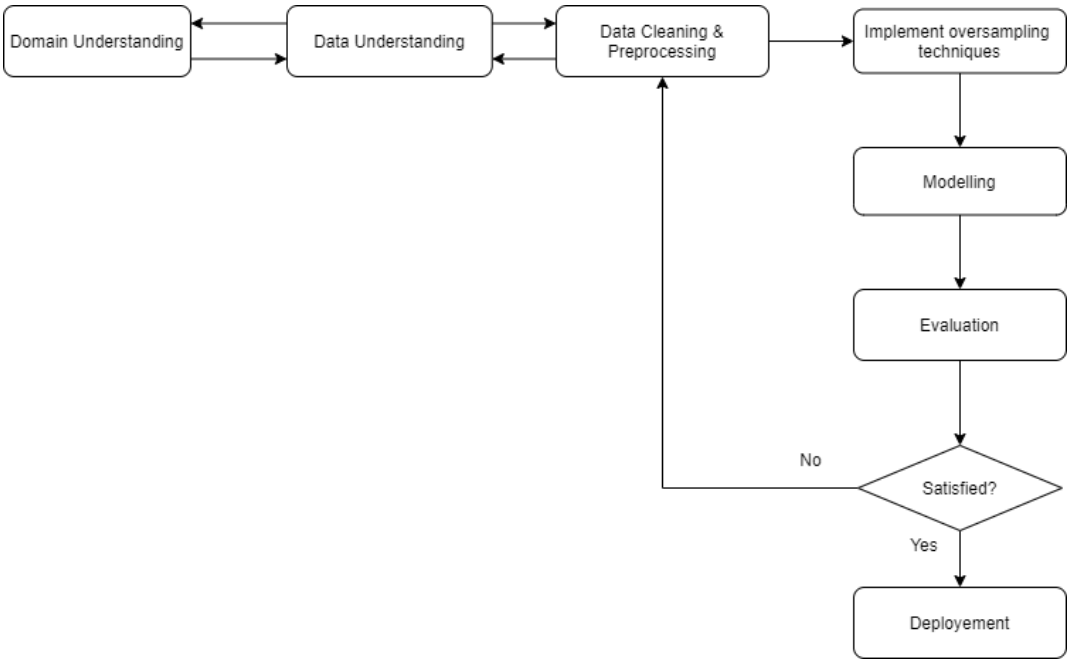


Figure 3.1: Steps of CRISP-DM Implemented in this Research

Figure 3.1 illustrates a modified CRISP-DM methodology to fits the needs of this research. The data pre-processing process has been divided into two steps before proceeding to the modelling phase. The steps for this research are as follows :

3.1.1 Domain Understanding

The domain background, research problem, and objectives were identified and discussed. Throughout the research, we had identified a common phenomenon in all network intrusion data sets, which is an unbalanced class distribution problem. It led to low detection rates for the minority classes due to unbalanced class distribution. Therefore, the focus of this research is to improve the detection rate by using oversampling techniques on those data sets. Finally, recall rate (detection rate) will be utilised to evaluate which technique performed well for those data sets.

3.1.2 Data Understanding

3.1.2.1 Data Collection

In this phase, a few benchmark data sets were identified and evaluated. The background of each data sets, data distributions, features, and detection rates were investigated in the literature review chapter. Those data sets are CICIDS 2017, Kyoto 2006, UNSW-NB15, NSL-KDD, ICXS 2012 and Consolidate CICDDOS 2019. Since the NSL_KDD and UNSW-NB15 contain a standardised training and testing data set. Hence, this research will make use of these data sets rather than retrieving the complete data set.

CICIDS 2017, CICDDOS 2019 and ICXS2012 only consist of the full data set file. Therefore, 10 % of the data will be extracted for this research to reduce the runtime. In the following section, all data sets were explored and visualised before oversampling. Table 3.1 tabulated the data distribution for each data set. Some data sets have a scattered presence of large data files. It makes processing the data very time-consuming and computationally expensive. As a result, for those data sets without a prepared training and testing data set, only 10% of the data were extracted from the entire distribution, while preserving the ratio for each class.

Table 3.1 Data Distributions for each Data Set

Dataset	File Name	Train set	Test set	Total	Remark
UNSW-NB15	UNSW_NB15_testing-set.csv UNSW_NB15_training-set.csv	175,341	82,332	257,673	Utilise available train and test data set
CICIDS 2017	All CSV files in the CICIDS2017 – MachineLearningCSV.zip	198,139	84,918	283,057	Extracted 10% subset of the data:
NSL-KDD	KDDTest+.txt KDDTrain+.txt KDDTrain+_20Percent.txt	151,165	22,544	173,709	Utilise available train and test data set
ICXS 2012	All XML files between 12/6/2012 – 17/6/2012 in ICXS2012.zip	119,642	68,910	188,552	Extracted 10% subset from the full data:
Kyoto 2006	Extracted data from year 2015 data set	210,000	90,000	300,000	300,000 sample data in year 2015.
Cicddos 2019	Consolidate data from CICDDOS 2019 and CICIDS 2017	290,860	124,654	415,514	Extracted 10% subset from the full data set.

3.1.2.2 Data Visualisation

In this phase, the distribution of the data was explored, analysed, and visualise to better comprehend the data. Several graphs have been plotted to get insight into the network intrusion data sets. The findings were discussed as follows:

Attack Type	Normal	Generic	Exploits	Fuzzers	DoS	Reconnaissance	Analysis	Backdoor	Shellcode	Worms
Counts	93000	58871	44525	24246	16353	13987	2677	2329	1511	174

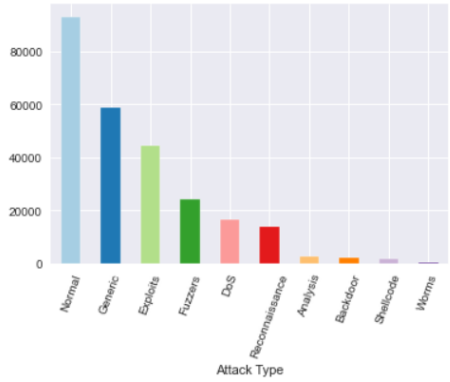


Figure 3.2: Data Distribution in the UNSW-NB15 Data Set

Attack Type	BENIGN	DoS	PortScan	DDoS	FTP	SSH	Bot	Brute Force	XSS	Infiltration	Sql Injection	Heartbleed
Counts	169844	46964	30977	21984	5931	3219	1948	1470	652	36	21	11

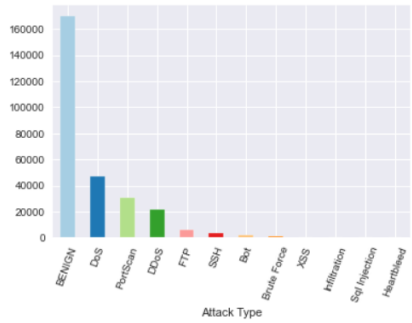


Figure 3.3: Data Distribution in the CICIDS 2017 Data Set

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Attack Type	BENIGN	TFTP	MSSQL	NetBIOS	UDP	Syn	SNMP	DNS	LDAP	SSDP	NTP	UDPLag	Portmap	WebDDoS
Counts	238405	195133	99819	74777	68839	59413	51496	49081	41030	25683	11953	3489	1768	44

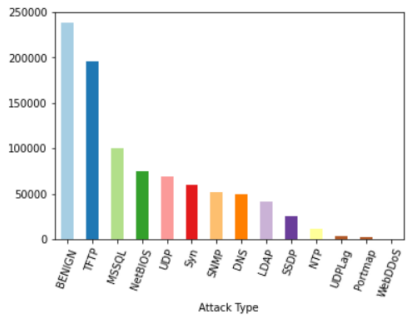


Figure 3.4: Data Distribution in the CICIDOS 2019 Data Set

	0	1	2	3	4
Attack Type	normal	dos	probe	r2l	u2r
Counts	76967	52987	13935	3880	119

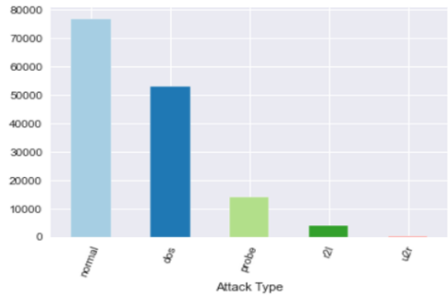


Figure 3.5: Data Distribution in the NSL KDD Data Set

	0	1	2	3	4
Attack Type	Normal	Botnet	infiltration	SSH	DoS
Counts	119642	37460	20358	7316	3776

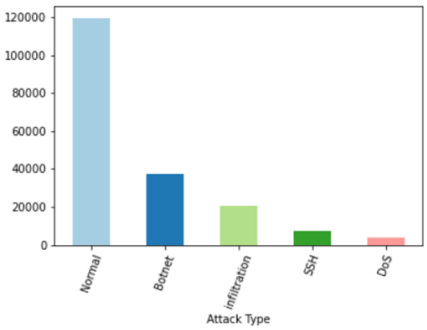


Figure 3.6: Data Distribution in the ICXS 2012 Data Set

	0	1	2
Attack Type	Normal	Known_attack	Unknown_attack
Counts	181600	118264	136

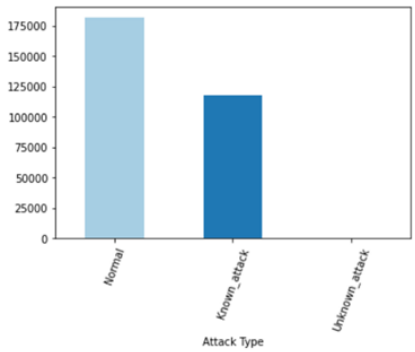
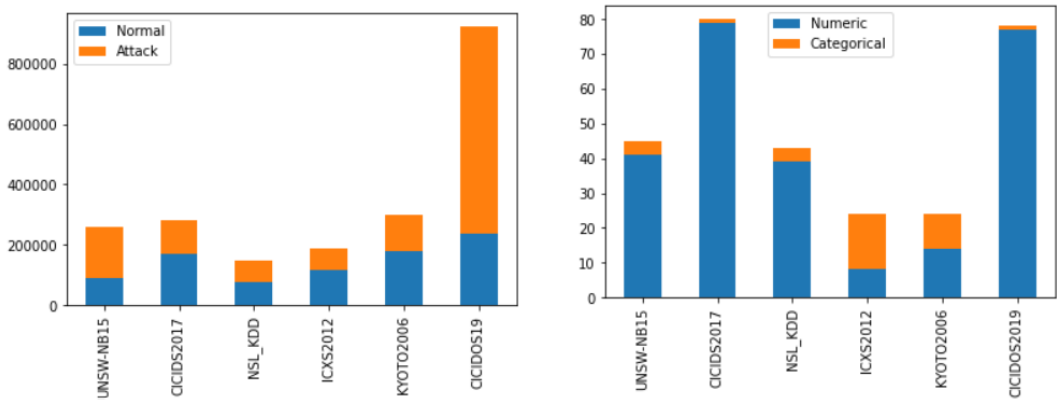


Figure 3.7: Data Distribution in the Kyoto 2006 Data Set

Figures 3.2 to 3.7 describe the data distribution for UNSW-NB15, CICIDS 2017, NSL KDD, ICXS2012, Kyoto2006 and CICDDOS 2019. The normal traffic is the majority class as investigated in the LR section. When comparing the data points

of the majority to each minority class in terms of percentage, the CICIDS 2017 data set suffered the unbalanced class distribution problem the most. This is because the sum of the eight minority classes (<10000 data) only accounted for 8% of the normal traffic. While UNSW-NB15 has four minority classes (<5000 data) accounted for 2.5 % of the total distribution, and three minority classes in NSL KDD accounted for only 10 % of total data distributions. In contrast, the attack classes account for 36 % of the total distribution in ICXS2012 data. While the distributions of unknown attacks in the Kyoto2006 data set are the most severe, they account for only 0.05 percent of the 300,000 data samples.



(a) **(b)**

Figure 3.8: Shape Comparison Among Data Sets

(a) Data Distributions , (b) Features

Figure 3.8 depicts the comparison of rows and columns between data sets. As observed from Figure 3.8 (a), the normal traffic is the majority class in most data sets, except for the prepared data set for UNSW-NB15. While from Figure 3.8 (b) shows that majority of the features in the network intrusion data set are numeric. Moreover, CICIDS 2017 and CICDDOS 2019 have the highest number of features among all data sets. While UNSW-NB15 and NSL-KDD have almost the same number of features. Understanding these two pieces of information can help with the decision-making in the pre-processing phase (dimensional reduction) and evaluation phase (evaluate time performance).

3.1.3 Data Pre-processing

The data pre-processing was divided into two phases: data cleaning and oversampling. The data points and columns were reviewed and renamed in the data cleaning process. For example, the attack target in the NSL KDD data set had categories into four major categories: R2L, Probe, U2R, and DoS, instead of their typical attack type. Since there are large numbers of nan values and columns in ICXS2012, therefore all values remained and were replaced as “none” instead of removing it.

In addition, the attack type for CICIDS 2017 was categorised into 11 types instead of 15. Besides that, features with lower correlation were eliminated to reduce the data dimension, such as ID (record identifier). After that, the data sets were explored to identify and remove records that contained a large number of null values. The duplication of records was dropped. Then, the data sets were divided into train and test sets with a ratio of 70 : 30. Finally, the selected oversampling technique was implemented, then data encoding and scale or without scaling for the modelling phase. Lastly, it is time-consuming to process each data set due to the large data distributions with high dimensionality.

3.1.4 Implementation Oversampling Techniques

This is a part of data pre-processing to implement the identified oversampling techniques to all selected data sets. Those techniques are ROS, SMOTE, Borderline SMOTE, ADASYN, and K-Mean SMOTE. All oversampling techniques were used to increase the sample distribution iteratively from 10 % to 100 %. Lastly, the runtime of each technique was reported in chapter five for evaluation.

3.1.5 Modelling

Few classifiers have been implemented to each data set using Python Scikit-Learn library: Decision Tree, Gaussian Bayes, and Logistic Regression. Each classifier has been tuned with five-cross fold cross-validation to obtain the best hyperparameters. The parameter used to tune each model are listed in Table 3.2.

Table 3.2 Parameters for Each Model

Classifier	Hyperparemeters	Value
Naïve Bayes	var_smoothing	From 1e-2 to 1e-15
Logistic Regression	C	0.001, 0.01, 0.1, 1, 1.5, 2.0, 2.5, 10, 100, 1000
Decision Tree	Max_depth	2,3,5,10,20,100,500,700
	Min_samples_leaf	5,10,20,50,100,500,1000,1500
	Criterion	Gini, entropy

3.1.6 Evaluation

Multiple performance matrices were chosen to provide a holistic evaluation of the classifier and oversampling techniques: Precision, F measure, recall (detection rate), and time taken. The main purpose of this research is to evaluate the oversampling techniques by comparing the result with non-oversampled classifiers. Therefore, the commonly used matrix recall is employed to compare and determine the efficiency of oversampling techniques. In this research, the oversampling technique with the largest increase in recall rate for minority classes will be selected as the best technique. The classification report is employed to measure the classifier performance and the performance of an individual class. Lastly, a confusion matrix is used to further analyse the predictions by generating an $n \times n$ matrix (n is the number of classes). In this phase, the results of all experiments are summarised and carefully documented. More time is required to identify the contributing results, trends and determine whether a new experiment is needed.

3.2 Over Sampling Algorithm

3.2.1 ROS

Table 3.3: Random Over Sampling Algorithm

Algorithm 1 *Random Over Sampling*

Input: data set to implement oversampling technique, number to oversample.
Output: New oversampled data set.
// $\boldsymbol{D}$ – data set
// $\boldsymbol{y}$ – target class
// $\boldsymbol{t_y}$ – threshold for each $\boldsymbol{t_y}$ values
// $\boldsymbol{s}$ – number to oversample

1. Bins $\boldsymbol{D}$ // separate data into multiple bins
2. BinsO ←[Bins: $\boldsymbol{y}$ instances $< \boldsymbol{t_y}$] // select bin to apply oversampling
3. $\boldsymbol{newData}$ ← Bins
4. **For each** $\boldsymbol{B}$ **in** BinsO **do**
5. $\boldsymbol{selData}$ ← SAMPLE ($\boldsymbol{s}$, $\boldsymbol{B}$) // randomly select amount of data from bin $\boldsymbol{B}$
6. $\boldsymbol{newData}$ ← c ($\boldsymbol{newData}, \boldsymbol{selData}$) // add oversampled class to new data set
7. **End for**
8. **return** $\boldsymbol{newData}$

Table 3.3 describes the Random Over Sampling algorithm. This technique aims to balance the distribution of the classes by replicating the existing data points through the binning process. It required two inputs for oversampling: the data set to apply the oversampling technique and the number of data points to duplicate. To begin, the data set was divided into a few bins (Line 1, Algo 1). Only several bins will be selected to apply oversampling technique (Line 2, Algo 1), while the passed-in data set will be stored into a new array named $newData$ (Line 3, Algo 1). Subsequently, a new group of data points will be generated from each selected $BinO$ with the target elements until S samples have been duplicated (Line 4 – 7, Algo 1). Lastly, the new data set was returned, which includes both replicated and original data (Line 8, Algo 1). This is the simplest oversampling technique, but it often introduces an overfitting problem due to the replication of the existing data points.

3.2.2 SMOTE

Table 3.4: SMOTE Algorithm

Algorithm 2 *SMOTE*

Input:

- data set to implement oversampling technique
- number to oversample
- number of nearest neighbours.

Output: New oversampled data set.

// $\boldsymbol{D}$ – data set
// $\boldsymbol{K}$ – number of nearest neighbours
// $\boldsymbol{n}$ – observations
// $\boldsymbol{s}$ – number to oversample
// $\boldsymbol{a}$ – features
// $\boldsymbol{n_{mi}}$ – minority observations

1. initialise $\boldsymbol{syn}$ // empty array for synesthetic samples
2. **For** $\boldsymbol{i} \leftarrow \mathbf{1}$ **to** $\boldsymbol{n_{mi}}$ **do**
3. compute k nearest neighbour and store the index in $\boldsymbol{nn}$ for each $\boldsymbol{i}$
4. $\boldsymbol{inarr} \leftarrow \mathbf{1}$
5. **While** n ! $= \mathbf{0}$ **do** // if more observation to find distance
6. $\boldsymbol{K_c} \leftarrow$ random number between 1 and $\boldsymbol{K}$
7. **For** $\boldsymbol{o} \leftarrow \mathbf{1}$ **to** $\boldsymbol{a}$ **do** //find the distance between target & neighbour
8. $\boldsymbol{dif} \leftarrow \boldsymbol{D}[\boldsymbol{nn}[\boldsymbol{K_c}]][\boldsymbol{o}] - \boldsymbol{D}[\boldsymbol{i}][\boldsymbol{o}]$
9. $\boldsymbol{g} \leftarrow \boldsymbol{uniform}\ (\mathbf{0}, \mathbf{1})$
10. $\boldsymbol{syn}[\boldsymbol{inarr}][\boldsymbol{o}] \leftarrow \boldsymbol{D}[\boldsymbol{i}][\boldsymbol{o}] + \boldsymbol{g} \times \boldsymbol{dif}$
11. **End for**
12. $\boldsymbol{inarr} \mathrel{+}= \mathbf{1}$
13. $\boldsymbol{n} \mathrel{-}= \mathbf{1}$
14. **End while**
15. **End for**
16. **return** $\boldsymbol{D} + \boldsymbol{syn}$

Table 3.4 depicts the algorithm for Synthetic Minority Over-sampling Technique (SMOTE). This technique generated synthetic samples with the K nearest neighbour approach instead of replicating existing data. As a result, it required an additional parameter to indicate the number of K, rather than pass in the data set and the number of samples to be generated. An array will be initialised to store the synthetic data points (Line 1, Algo 2). The indices of the k nearest neighbours for each minority data point are calculated and stored into an array (Line 2- 4, Algo 2). Subsequently, the distance between the specified data point and its nearest neighbour will be computed.

After that, the K number of neighbours with the lowest Euclidean distance between the target and its neighbour will be chosen (Line 5 – 6, Algo 2). Next, the synthetic data point is generated on the gap (line) between the target data and its k neighbours until the sample size reaches S (Line 7 – 15, Algo 2). Finally, the original and synthetic data sets are combined to establish a new data set (Line 16, Algo 2). This technique was introduced to overcome overfitting in random oversampling, yet it causes overlapping issues.

3.2.3 Borderline-SMOTE

Table 3.5: Borderline-SMOTE Algorithm

Algorithm 3 *Borderline-SMOTE*

Input:

- data set to implement oversampling technique
- number to oversample
- number of nearest neighbours
- number of nearest neighbours to create borderline subset.

Output: New oversampled data set.

// $\boldsymbol{D}$ – data set

// $\boldsymbol{M}$ – number of nearest neighbours

// $\boldsymbol{s}$ – number to oversample

// $\boldsymbol{b}$ – number of nearest neighbours to create borderline subset

1. Initialise $\boldsymbol{DANGER}$
2. **For each** $\boldsymbol{P}$ **in** $\boldsymbol{D}$
3. $\boldsymbol{M_p}$ ← compute $\boldsymbol{b}$ nearest neighbours.
4. **If** $\boldsymbol{M'} = \boldsymbol{M}$, Ignore P // ignore noisy point
5. **Else if** $\boldsymbol{0} \leq \boldsymbol{m'} \leq \frac{\boldsymbol{m}}{\boldsymbol{2}}$, ignore P // ignore safe point
6. **Else If** $\frac{\boldsymbol{m}}{\boldsymbol{2}} \leq \boldsymbol{m'} \leq \boldsymbol{m}$, add $\boldsymbol{P}$ to DANGER
7. **End if**
8. **End For**
9. $\boldsymbol{newData}$ ← For each point $\boldsymbol{u}$ in $\boldsymbol{DANGER}$, apply SMOTE algorithm
10. **return** $\boldsymbol{newData}$

Table 3.5 depicts the algorithm for a variation of SMOTE technique named Borderline-SMOTE. This technique will separate the minority class data into three groups: $SAFE$, $NOISY$ and $DANGER$. Only data points that fall into the $DANGER$ category will be utilised to generate synthetic data. Hence, its parameter is similar to SMOTE but with an additional m neighbours to determine the borderline subset.

Firstly, an array will be initialised to store the data points on the boundary (Line 1, Algo 3). The M nearest neighbours of each minority data point will be identified.

The data point is considered as a $NOISY$ point if the data point falls within the majority data points border, where all its neighbours are majority class instances (Line 4, Algo 3). If the data point lies within the large subset of minority data points, with less than a half of the majority instances, it is considered as a $SAFE$ point (Line 5, Algo 3). Otherwise, if the data point is far away from the minority instance and close to the majority data boundary. Where more than half of its neighbours are majority class instances, it is categorised as $DANGER$. The synthetic data is then produced from these data points using the SMOTE algorithm (Line 6 – 9, Algo 3). Lastly, it returns a new data set containing both original and synthetic data points. This technique was designed to overcome the overlapping issue in SMOTE by only producing synthetic data that is near the borderline. Nevertheless, it might lead to information loss of certain key instances in minority classes as it focuses on extreme values near the class boundary.

3.2.4 ADASYN

Table 3.6: ADASYN Algorithm

Algorithm 4 *ADASYN*

Input:

- data set to implement oversampling technique
- desired balance level
- number of nearest neighbours

Output: New oversampled data set.

// $\boldsymbol{D}$ – data set
// $\boldsymbol{K}$ – number of nearest neighbours
// $\boldsymbol{B}$ – desired balance level
// $\boldsymbol{S_{ma}}$ – majority class
// $\boldsymbol{S_{mi}}$ – minority class
// $\boldsymbol{n_{mi}}$ – minority observations
// $\boldsymbol{n_{ma}}$ – majority observations

1. $\boldsymbol{G} \leftarrow (\boldsymbol{n_{ma}} - \boldsymbol{n_{mi}}) \times \boldsymbol{B}$ // determine total number of synthetic data
2. **For** $\boldsymbol{i} \leftarrow \mathbf{1}$ **to** $\boldsymbol{n_{mi}}$ **do**
3. $\boldsymbol{nn} \leftarrow$ indices of the computed k nearest neighbours
4. $\boldsymbol{r}[\boldsymbol{i}] \leftarrow \frac{|\boldsymbol{nn}[\boldsymbol{i}]\boldsymbol{S_{ma}}|}{\boldsymbol{K}}$ // calculate the impurity ratio
5. **End for**
6. **For** $\boldsymbol{i} \leftarrow \mathbf{1}$ to $\boldsymbol{n_{mi}}$ **do**
7. $\boldsymbol{g}[\boldsymbol{i}] = \boldsymbol{proba\ mass\ function} \times \boldsymbol{G}$
   // determine the amount of synthetic data to be generated for each minority class.
8. **End for**
9. $\boldsymbol{syn_G}$ // empty array for new sample
10. $\boldsymbol{z} = \mathbf{1}$
11. **For** $\boldsymbol{i} \leftarrow \mathbf{1}$ **to** $\boldsymbol{n_{mi}}$ **do**
12. $\boldsymbol{inarr} \leftarrow \boldsymbol{g}[\boldsymbol{i}]$
13. **While** $\boldsymbol{inarr} \mathrel{!}= \mathbf{0}$ **do**
14. $\boldsymbol{K_c} \leftarrow$ random number between 1 and $\boldsymbol{K}$
15. $\boldsymbol{dif} \leftarrow \boldsymbol{S_{mi}}\left[\boldsymbol{nn}[\boldsymbol{i}][\boldsymbol{K_c}]\right] - \boldsymbol{S_{mi}}[\boldsymbol{i}]$
16. $\boldsymbol{gap} \leftarrow \boldsymbol{uniform}\ (\mathbf{0}, \mathbf{1})$
17. $\boldsymbol{syn}[\boldsymbol{z}][] \leftarrow \boldsymbol{S_{mi}}[\boldsymbol{i}][] + \boldsymbol{dif} \times \boldsymbol{gap}$
18. $\boldsymbol{z} \mathrel{+}= \mathbf{1}$
19. $\boldsymbol{n} \mathrel{-}= \mathbf{1}$
20. **End while**
21. **End for**
22. **return** $\boldsymbol{D} + \boldsymbol{Syn}$

The algorithm for ADASYN is illustrated in Table 3.6. This is another variation of SMOTE technique that is similar to Borderline-SMOTE. However, it uses the

density-based approach to determine how much synthetic data should be generated according to their distribution rather than using the boundary approach. G indicates how much synthetic data should be generated in total. It was calculated by multiplying the desired balance level (range from 0 to 1) to the difference between minority-majority neighbour samples (Line 1, Algo 4). After that, the K nearest neighbours for each minority data is computed, and their weight is defined in a density distribution, which is called the impurity ratio (Line 2 – 5, Algo 4).

Subsequently, the impurity ratio and G were used to determine the number of synthetic samples required for each minority class (Line 6 – 8, Algo 4). Next, the SMOTE algorithm will be implemented to produce a different number of synthetic data points for each minority class based on their density distribution. The higher the density (g), the more synthetic data to be generated (Line 9 – 21, Algo 4). Lastly, the data set that combines both original data and synthetic data will be returned (Line 22, Algo 4). Nevertheless, since this technique does not identify $NOISE$ data, so it is possible to generate a large amount of synthetic data surrounding such instances. Thereby creating an unrealistic space for the learner.

3.2.5 MWMOTE

Table 3.7: MWMOTE Algorithm

Algorithm 5 *MWMOTE*

Input:

- data sample
- k1 nearest to filter minority noise
- k2 nearest for majority to construct informative minority set
- k3 nearest for minority to construct informative minority set
- number of synthetic samples to be generated

Output: New oversampled data set.

// $\boldsymbol{S_{min}}$ – minority observations

// $\boldsymbol{S_{maj}}$ – majority observations

// $\boldsymbol{S_{minf}}$ **–** majority set from the minority class

// $\boldsymbol{S_{bmaj}}$ **–** borderline majority set

// $\boldsymbol{S_{imin}}$ **–** informative minority set at borderline

1. Determine $\boldsymbol{k1}, \boldsymbol{k2}, \boldsymbol{k3}$ & $\boldsymbol{n\_sample}$
2. **For** $\boldsymbol{S_{min}} \leftarrow \boldsymbol{1}$ **to** $\boldsymbol{S_{minn}}$ **do** // eliminate noise point
3. **While** the $\boldsymbol{k1}$ for the $\boldsymbol{S_{min}}$ consist of $\boldsymbol{S_{min}}$
4. $\boldsymbol{S_{minf}}$ **+=** $\boldsymbol{S_{min}[i]}$
5. **End while**
6. **End for**
7. **For** $\boldsymbol{S_{minf}} \leftarrow \boldsymbol{1}$ **to** $\boldsymbol{S_{minfn}}$ **do** //identify majority borderline
8. **While** all $\boldsymbol{k2}$ for the $\boldsymbol{S_{minf}}$ **=** $\boldsymbol{S_{maj}}$
9. $\boldsymbol{S_{bmaj}}$ **+=** $\boldsymbol{S_{minf}[i]}$
10. **End while**
12. **End for**
13. **For** $\boldsymbol{S_{bmaj}} \leftarrow \boldsymbol{1}$ **to** $\boldsymbol{S_{bmajn}}$ **do** //identify minority borderline
14. **While** all $\boldsymbol{k3}$ for the $\boldsymbol{S_{bmaj}}$ **=** $\boldsymbol{S_{min}}$
15. $\boldsymbol{S_{imin}}$ **+=** $\boldsymbol{S_{bmaj}[i]}$
16. **End while**
17. **End for**
18. **For each** $\boldsymbol{S_{bmaj}} \leftarrow \boldsymbol{1}$ **to** $\boldsymbol{S_{bmajn}}$ **do //** find information weight
19. $\boldsymbol{I_w}\left(\boldsymbol{S_{bmaj}}, \boldsymbol{S_{imin}}\right) = \boldsymbol{C}\left(\boldsymbol{S_{bmaj}}, \boldsymbol{S_{imin}}\right) \times \boldsymbol{D}\left(\boldsymbol{S_{bmaj}}, \boldsymbol{S_{imin}}\right)$
20. **End for**
21. **For each** $\boldsymbol{S_{imin}} \leftarrow \boldsymbol{1}$ **to** $\boldsymbol{S_{iminn}}$ **do //** find selection weight
22. $\boldsymbol{S_w}(\boldsymbol{S_{imin}}) = \sum \boldsymbol{I_w}$ // sum related information weight to find selection weight
23. Convert $\boldsymbol{S_w}$ into probability distribution → $\boldsymbol{S_p}$
23. Find cluster of $\boldsymbol{S_{min}}$ using hierarchical clustering, $\boldsymbol{Z}$ clusters are formed.
24. Initialise set $\boldsymbol{S_{over}}$ **=** $\boldsymbol{S_{min}}$
25. For **j = 1** to **N**

26. Select $\boldsymbol{x}$ from $\boldsymbol{S_{imin}}$ according $\boldsymbol{S_p}$ **,** where $\boldsymbol{x}$ in $\boldsymbol{Clus_k}$, $\mathbf{1} < \boldsymbol{k} < \boldsymbol{Z}$
27. Randomly select $\boldsymbol{y}$ from $\boldsymbol{Clus_k}$
28. $\boldsymbol{s} = \boldsymbol{x} + \boldsymbol{\alpha} \times (\boldsymbol{y} - \boldsymbol{x}), \boldsymbol{where\ \alpha\ in\ range}\ [\mathbf{0}, \mathbf{1}]$
29. Add $\boldsymbol{s}$ into $\boldsymbol{S_{over}}$
30. **End for**
31. Return $\boldsymbol{S_{over}}$

Table 3.7 depicts the algorithm for Majority Weighted Minority Oversampling Technique (MWMOTE). This algorithm combines the borderline and clustering concepts to oversample synthetic data. Where KNN is used to determine the borderline between majority and minority classes, and hierarchical clustering is used to implement the synthetic process. Three k neighbour values must be determined to identify the borderline data and also the majority set of minority instances (Line 1, Algo 5). If all $k1$ neighbour in the minority instance (S_{min}) does not consist of minority instance will be categorised as a noise point. As a result, the majority instance set in the minority class (S_{minf}) will exclude this data point (Line 2 – 6, Algo 5).

Subsequently, $\boldsymbol{k2}$ is utilised to identify the majority neighbours ($\boldsymbol{S_{bmaj}}$) to build informative minority set in $\boldsymbol{S_{imin}}$, where all of the nearest neighbours are majority instances (Line 7-10, Algo 5). Following that, for each majority instance in $\boldsymbol{S_{bmaj}}$ will compute the closest minority set with $\boldsymbol{k3}$ to construct the minority set ($\boldsymbol{S_{imin}}$) **(**Line 13-17, Algo 5). Then an information weight $\boldsymbol{I_w}$ will be calculated by multiplying the density factor $\boldsymbol{D}$ and closeness factor $\boldsymbol{C}$ for each point in $\boldsymbol{S_{bmaj}}$ and $\boldsymbol{S_{imin}}$ (Line 18-20, Algo 5). By summing the information weights of each $\boldsymbol{S_{imin}}$, a selection weight is generated, which is then translated into a probability distribution $\boldsymbol{S_p}$ (Line 21-23, Algo 5)**.** Furthermore, a modified hierarchical clustering is implemented to all minority classes. Where an amount of synthetic data will be generated within the cluster where the data existing in $\boldsymbol{S_{imin}}$ according to $\boldsymbol{S_p}$ (Line 23-30, Algo 5). Finally, the oversampled data are returned (Line 31, Algo 5).

3.2.6 K-Means SMOTE

Table 3.8: K-means SMOTE Algorithm

Algorithm 6 *K-means SMOTE*

Input:

- data sample
- k1 number of clusters
- k2 nearest for neighbour for SMOTE
- imbalance ratio threshold
- number of synthetic samples to be generated

Output: New oversampled data set.

// $\boldsymbol{ir}$ – imbalance ratio

// $\boldsymbol{irt}$ – imbalance ratio threshold

// $\boldsymbol{Ocluster}$ – selected cluster to be oversampled

// $\boldsymbol{AMD}$ – average minority distance

// $\boldsymbol{n}$ – number of samples to be generated

// $\boldsymbol{GSample}$ – new sample array

1. Perform kmeans clustering.
2. **For each** $\boldsymbol{clusters} \rightarrow \boldsymbol{c}$ **do**
3. $\boldsymbol{ir} = \frac{\boldsymbol{majorityCount(c)+1}}{\boldsymbol{minorityCount(c)+1}}$
4. **if** $\boldsymbol{ir} < \boldsymbol{irt}$ **then**
5. $\boldsymbol{Ocluster} \leftarrow$ cluster
6. **End if**
7. **End for each**
8. **For each** $\boldsymbol{Ocluster} \rightarrow \boldsymbol{f}$ **do** //compute sampling weight based on density
9. $\boldsymbol{AMD(f)} \leftarrow \boldsymbol{mean(distance(f))}$
10. $\boldsymbol{density(f)} \leftarrow \frac{\boldsymbol{minority(f)}}{\boldsymbol{amd(F)^{de}}}$
12. $\boldsymbol{sparsity(f)} \leftarrow \frac{\boldsymbol{1}}{\boldsymbol{density}}$
13. **End for each**
14. Sum all $\boldsymbol{sparsity(f)}$
15. $\boldsymbol{Sweight(f)} \leftarrow \frac{\boldsymbol{sparsity}}{\boldsymbol{sum\ of\ sparsity}}$
16. **For each** $\boldsymbol{Ocluster} \rightarrow \boldsymbol{f}$ **do** //SMOTE oversampling with sampling weight.
17. samples $\leftarrow ||\boldsymbol{n} \times \boldsymbol{Sweight(f)}$
18. nSamples $\leftarrow$ $\boldsymbol{GSample} + \boldsymbol{SMOTE(f)}$

19 **End for each**

20. Return $\boldsymbol{S_{over}}$

Table 3.8 illustrates the algorithm for K means SMOTE algorithm that combines clustering with SMOTE technique. To begin this algorithm, the k number of clusters, k neighbour for SMOTE and imbalance ratio threshold have to be determined. Firstly,

the algorithm will perform kmean clustering and compute the imbalance ratio for each cluster, where $\frac{\text{majority count}}{minority\ count}$ (Line 1 - 3, Algo 6). The cluster will be selected for oversampling if its IR exceeds the threshold (Line 4 -7, Algo 7). After filtering the cluster, the sparsity used to determine the weight is calculated using Euclidean distance followed by density factor (Line 8 – 13, Algo 6). Subsequently, the sampling weight for each cluster is computed by dividing its sparsity by the sum of sparsity (Line 14 -15, Algo 6). Eventually, the filtered cluster will be oversampled using SMOTE with the sampling weight (Line 16 – 20, Algo 6).

3.3 Gantt Chart for Research

This section depicts the project schedule for the entire research. All tasks are made sure to abide the schedule to make progress and amendments easier in the future. The overall project schedule is shown in Figure 3.9. The Gantt charts for project one and project two are shown in Figures 3.10 and 3.11. The detailed Gantt Chart are documented in Appendix B and C. Project one was to visualise and model the data sets before oversampling and examine the detection rates. Whereas project two was to implement the oversampling techniques to each data set and compare the detection rate obtained from project one.

Task Name	Duration	Start	Finish
Oversampling on unbalanced dataset	323 days	Mon 7/6/21 8:00 AM	Mon 25/4/22 5:00 PM
Project 1	79 days	Mon 7/6/21 8:00 AM	Tue 24/8/21 5:00 PM
Project 2	91 days	Tue 25/1/22 8:00 AM	Mon 25/4/22 5:00 PM

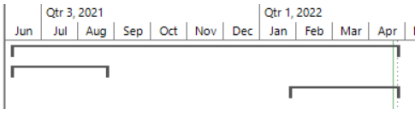


Figure 3.9: Overall Gantt Chart for the Entire Project

Task Name	Duration	Start	Finish
Project 1	79 days	Mon 7/6/21 8:00 AM	Tue 24/8/21 5:00 PM
Project understanding	14 days	Mon 7/6/21 8:00 AM	Sun 20/6/21 5:00 PM
Data collection and understanding	31 days	Mon 21/6/21 8:00 AM	Wed 21/7/21 5:00 PM
Data Cleaning & Preprocessing	2 days	Thu 22/7/21 8:00 AM	Fri 23/7/21 5:00 PM
Modelling	5 days	Sat 24/7/21 8:00 AM	Wed 28/7/21 5:00 PM
Evaluation	27 days	Thu 29/7/21 8:00 AM	Tue 24/8/21 5:00 PM

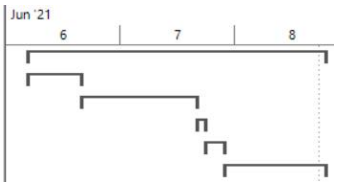


Figure 3.10: Gantt Chart for Project 1

Task Name	Duration	Start	Finish
Project 2	91 days	Tue 25/1/22 8:00 AM	Mon 25/4/22 5:00 PM
Project understanding	2 days	Tue 25/1/22 8:00 AM	Wed 26/1/22 5:00 PM
Data collection and understanding	32 days	Thu 27/1/22 8:00 AM	Sun 27/2/22 5:00 PM
Data Preprocessing	1 day	Mon 28/2/22 8:00 AM	Mon 28/2/22 5:00 PM
Oversampling	29 days	Fri 18/2/22 8:00 AM	Fri 18/3/22 5:00 PM
Modelling	29 days	Fri 18/2/22 8:00 AM	Fri 18/3/22 5:00 PM
Model Evaluation	29 days	Fri 18/2/22 8:00 AM	Fri 18/3/22 5:00 PM
Reporting	38 days	Sat 19/3/22 8:00 AM	Mon 25/4/22 5:00 PM

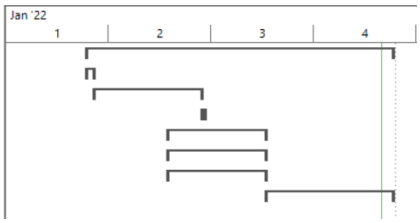


Figure 3.11: Gantt Chart for Project 2

CHAPTER 4

RESULTS AND DISCUSSION

Weighted recall (detection rate) for each class was used for comparison, while the overall weighted recall rate, f1 and precision were also included in the evaluation section. The oversampling technique used in this research is SMOTE, Borderline-SMOTE, Random Oversampling, K-Mean SMOTE, and ADASYN. Six data sets were explored in this section. Those data sets were ICXS 2012, Kyoto 2006, CICIDS 2017, UNSW-NB15, NSL KDD and CICDDOS 2019. The non-oversampled detection rates for these data sets are described in the non-oversampling results section (4.1), while the oversampled results will discuss the oversampled results (4.2).

4.1 Non-Oversampling Results

4.1.1 Multiclass Performance Evaluation

4.1.1.1 UNSW-NB15

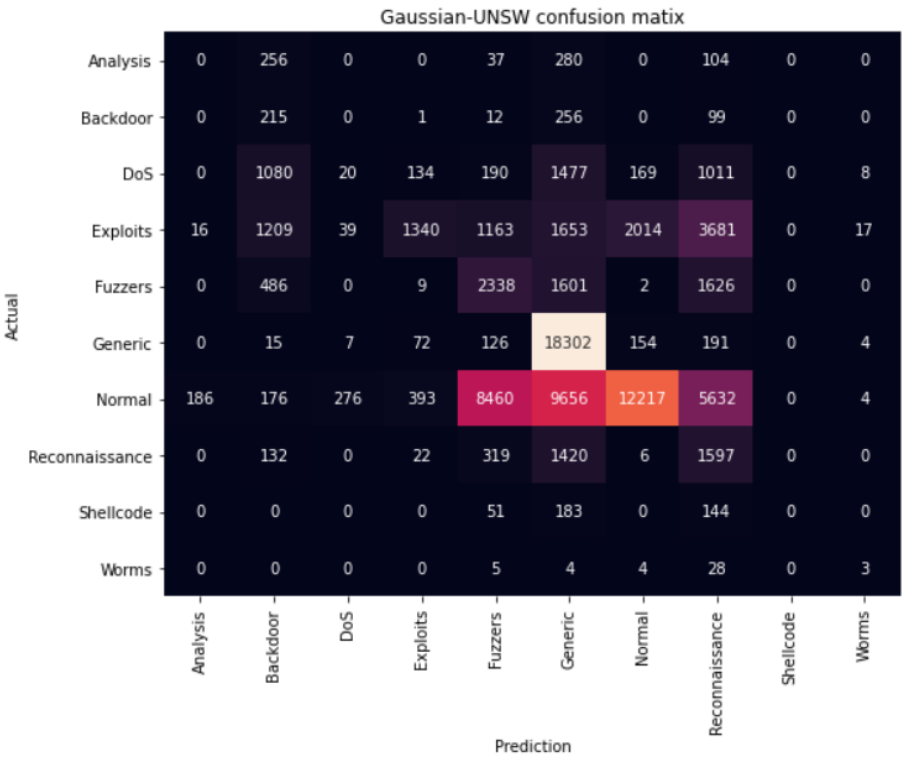
Table 4.1: Comparison of Results Between Classifiers for the UNSW-NB15 Data Set

*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage, **F** indicates **F measures** in percentage, the bold numbers indicate the minority class in the data set)*

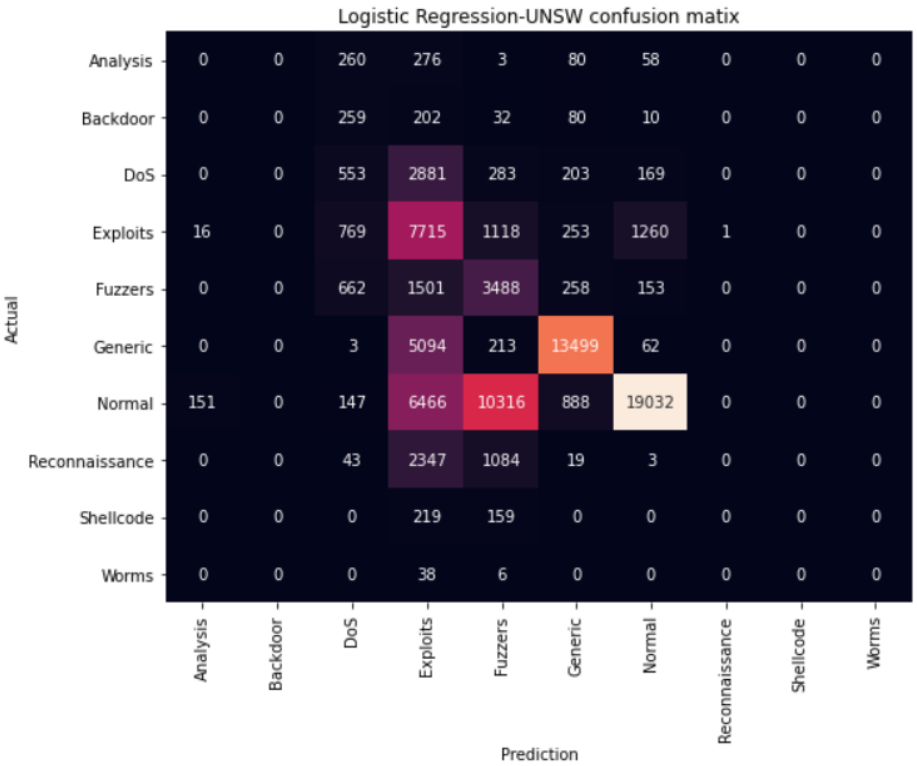
Attack Type	**Gaussian Bayes (GB)**			**Logistic Regression (LR)**			**Decision Tree (DT)**		
	P	**DR**	**F**	**P**	**DR**	**F**	**P**	**DR**	**F**
Analysis	0	**0**	0	0	**0**	0	0	**0**	0
Backdoor	6	**37**	10	0	**0**	0	40	**1**	1
DoS	6	**0**	1	21	**14**	16	10	**4**	6
Exploits	68	**12**	20	29	**69**	41	29	**18**	22
Fuzzers	18	**39**	25	21	**58**	31	11	**67**	19
Generic	53	97	68	88	72	79	0	0	0
Normal	33	84	47	51	92	66	77	73	75
Recon.	11	**46**	18	0	**0**	0	0	**0**	0
Shellcode	0	**0**	0	0	**0**	0	50	**1**	1
Worms	8	**7**	7	0	**0**	0	2	**64**	4

Table 4.1 summarised the comparison results for UNSW-NB15 with several classifiers. The bold numbers represent the minority class in the data set. As observed, most majority classes (generic and normal) have higher detection rates in all three classifiers, except the Generic class in DT, which has 0 % for all matrices. Fuzzer achieved the highest detection rate among the minority classes, ranging from 39 % to 67 %, whereas DoS had detection rates ranging from 0 % to 14 %. Besides, some minority classes perform better in only one classifier. For example, Worm had a recall rate of 64 % in DT but 0 % to 7 % in other classifiers. Whereas Reconnaissance achieved 46 % recall in GB and 0 % in other classifiers.

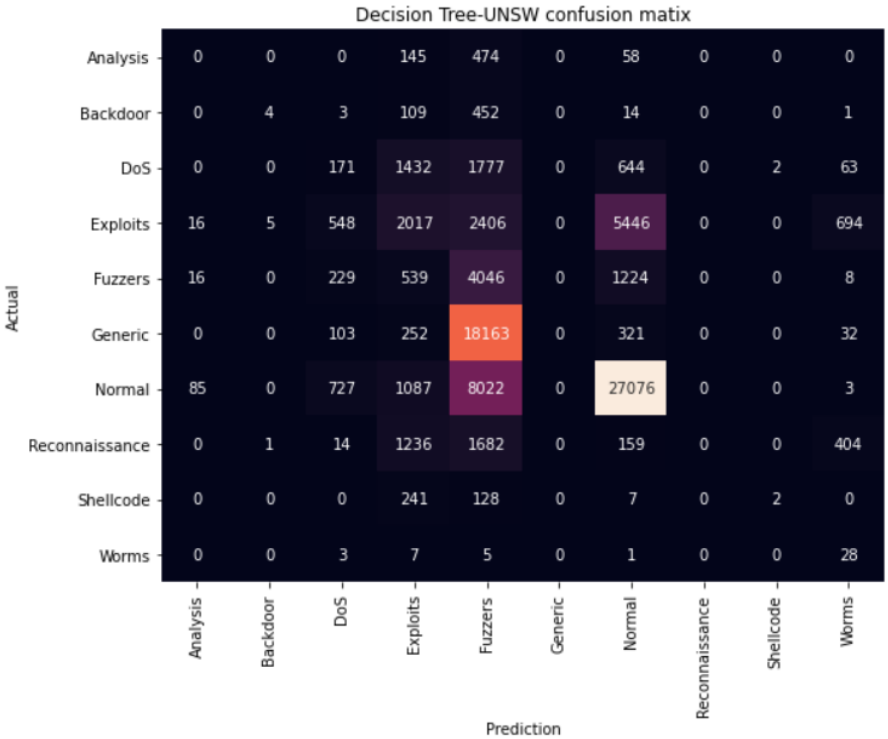
Lastly, Analysis and Shellcodes are the worst-case scenarios, with a recall rate of 0 % to 1 % of all detections. Subsequently, some minority classes, such as Fuzzer and Backdoor are better at recall back the positive case out of all positive cases. Therefore, the recall will be greater than the precision. Conversely, some minority classes like Shellcode and DoS predict better rather than accurately classifying the positive case. For instance, the DoS has higher precision than recall in all classifiers. In conclusion, all minority classes had a low detection rate of less than 70 % in all classifiers as investigated.



(a)



(b)



(c)

Figure 4.1: Confusion Matrix Between Classifiers for UNSW-NB15:

(a) Gaussian Bayes (GB) (b) Logistic Regression (LR) (c) Decision Tree (DT)

The confusion matrix between classifiers for UNSW-NB15 depicts in Figure 4.1. The performance matric in the preceding section is investigated further using the confusion matrix. From the figure above, we can see that the Analysis traffics were detected as Exploits, Fuzzer and Generic, whilst the Shellcode was detected as Exploits and Fuzzers in DT and LR. This may be because these two traffics have similar features to the misclassified classes, resulting in a 0 % recall rate in all classifiers. In addition, the smallest class (Worm) had most of its traffic detected as Reconnaissance in GB and Exploits in LR. Subsequently, most Worm traffic can be predicted accurately by DT. Next, only a small percentage of DoS and Backdoor is correctly labelled in all classifiers, where most traffic is classified into many different classes. These two classes may have no distinguishing characteristics from other classes.

Besides, Fuzzers perform well in LR and DT, yet some Fuzzers are labelled as Generic and Reconnaissance in GB. While for majority of classes, most Normal traffic was detected as Fuzzer, Generic and Reconnaissance in GB, while DT and LR misclassified it as Exploits and Fuzzers. It shows that all classifiers struggle to distinguish between Normal and Fuzzer traffic. Lastly, the 0 % detection rate for Generic traffic in DT was caused by the misclassification to Fuzzers. According to the confusion matrix, Fuzzer is the minority class that might have difficulty reflecting its true decision boundary from Normal and Generic. Hence, Fuzzer might be difficult to differentiate itself from the majority of instances.

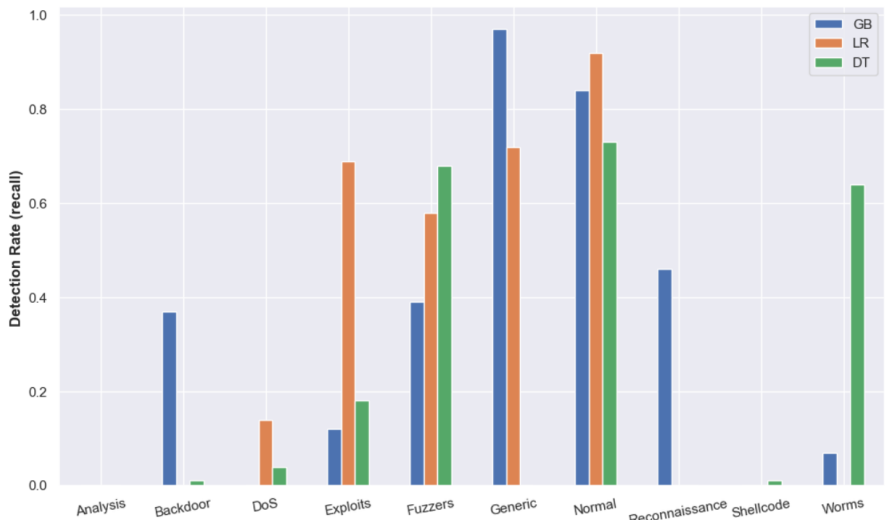


Figure 4.2: Multiclass Recall for UNSW-NB15 with Different Classifiers.

Figure 4.2 shows the multiclass recall rate for UNSW-NB15 with various classifiers. As observed, Analysis and Shellcode have recall rates that are mostly near to zero for all classifiers. As a result, these two minority classes are difficult to detect among all other classes. While some minority classes were identified better by one classifier: GB can identify Backdoor and Reconnaissance better, LR detects exploits better, DT can identify Worms better. It implies that the detection rate for such classes is not solely influenced by the data distribution and features, but also the algorithm used. Although Generic is a majority class, DT still misclassified most Generic traffics to Fuzzer, resulting in 0% recall.

4.1.1.2 CICIDS 2017

Table 4.2: Comparison of Results Between Classifiers for the CICIDS 2017 Data Set

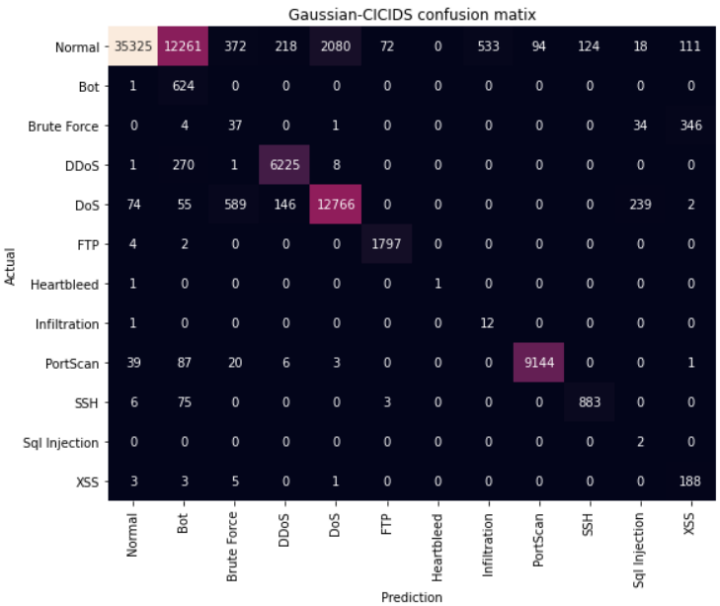
*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage, **F** indicates **F measures** in percentage, the bold numbers indicate the minority class in the data set)*

Attack Type	**Gaussian Bayes (NB)**			**Logistic Regression (LR)**			**Decision Tree (DT)**		
	P	**DR**	**F**	**P**	**DR**	**F**	**P**	**DR**	**F**
Normal	69	100	82	96	97	96	100	100	100
Bot	5	**80**	9	25	**3**	5	97	**85**	96
Brute Force	4	**9**	5	27	**4**	7	71	**75**	73
DDoS	94	**96**	95	100	**99**	99	100	**100**	100
DoS	86	**92**	89	97	**93**	95	100	**100**	100
FTP	96	**100**	98	93	**100**	96	100	**100**	100
Heartbleed	100	**50**	67	100	**100**	100	100	**50**	67
Infiltration	92	**2**	4	60	**23**	33	80	**92**	96
Port Scan	99	**98**	99	89	**100**	94	100	**100**	100
SSH	88	**91**	44	99	**91**	94	100	**99**	100
SQL Injection	100	**1**	1	0	**0**	0	9	**50**	15
XSS	29	**94**	44	43	**1**	3	41	**36**	38

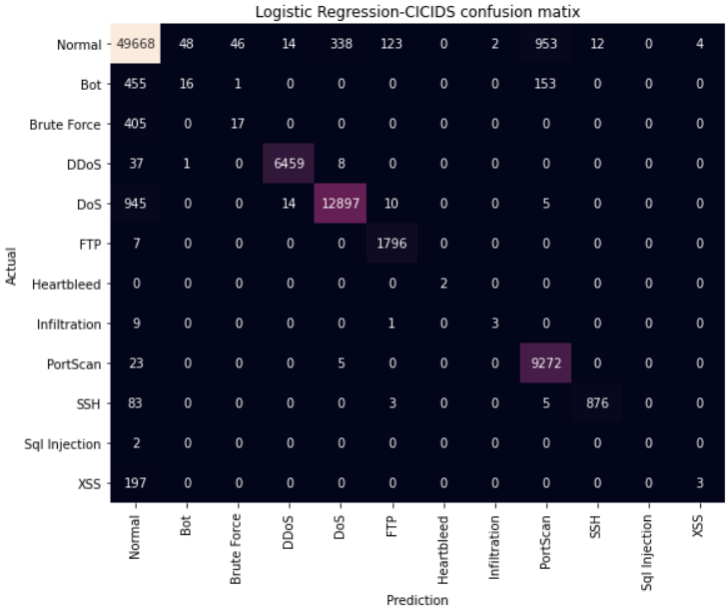
The comparison results for CICIDS 2017 with different classifiers are shown in Table 4.2. As investigated, normal traffic had achieved the highest detection and prediction rate in all classifiers. While all classifiers obtained a satisfactory detection

rate for the minority classes with data samples ranging from 3300 to 47000. Those are DDoS, DoS, FTP, Port Scan, and SSH obtained detection rates of 90 % and above. Especially, FTP has a 100 % detection rate for all classifiers. This might be because all of these classes have distinguishable features that can use to differentiate them. Subsequently, those minority classes with data samples ranging from 650 to 2000 have different detection rates between classifiers. Similar to UNSW-NB15, some minority classes are better detected by certain classifier(s). For instance, GB and DT are better at detecting Bots than LR. While DT can detect Brute Force better and NB is better at detecting XSS.

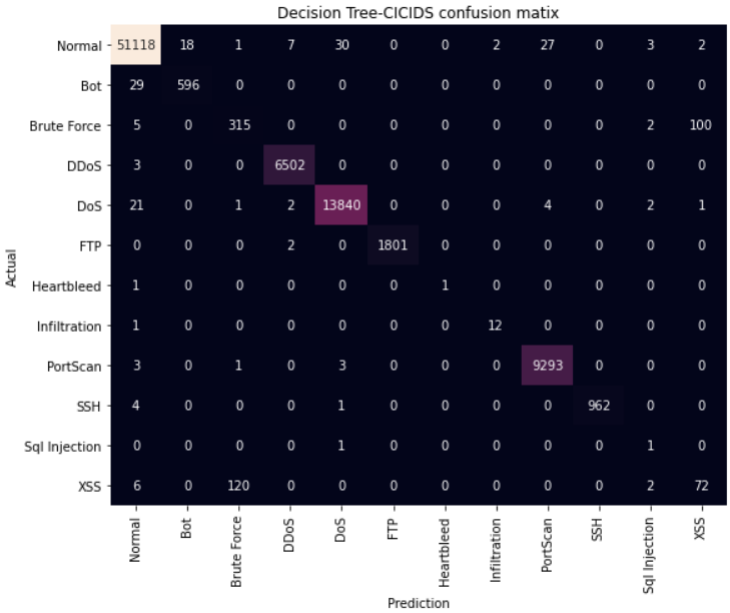
Following that, the minority classes with data samples ranging from 11 to 36 still achieved a high detection rate. For example, Infiltration has a detection rate that ranges from 2 % to 92 %. Despite Heartbleed is the smallest minority class (11 samples) in this data set, it had a 100 % recall in LR, and 50 % for DT and GB. This might be because the test data comprises a very small data distribution, so the classifier only needs to identify only a few of them. Assume that for Heartbleed traffics, train data contains nine samples and test data contains just two samples. If these two samples are successfully detected, the classifier will obtain a 100 % detection rate. However, such a rate cannot accurately reflect the efficiency of the classifiers as too little test data is used for evaluation. Overall, not all minority classes in this data set are hard to detect, but only those minority classes with less than 2000 data points.



(a)



(b)



(c)

Figure 4.3: Confusion Matrix Between Classifiers for CICIDS 2017:
(a) Gaussian Bayes (GB) (b) Logistic Regression (LR) (c) Decision Tree (DT)

Figure 4.3 illustrates the confusion matrix between classifiers for CICIDS 2017. As observed, minority classes with a high detection rate had fewer misclassifications in all classifiers. Only a small percentage of DDoS data is misidentified as Bot in GB. While other minority classes, such as DoS, FTP, Port

Scan, and SSH, have a small miss detection in one of the classifiers. These classes may have unique features that set them apart from others. Additionally, GB detects the XSS better, where LR detect it as normal traffic and DT detects it as Brute Force traffic. From the confusion matrix, there are only two test data for Heartbleed and SQL injection. These results may not accurately reflect classifier efficiency because they only have two samples in test data. In conclusion, the miss detection for minority classes in this data set is lesser than UNSW-NB15.

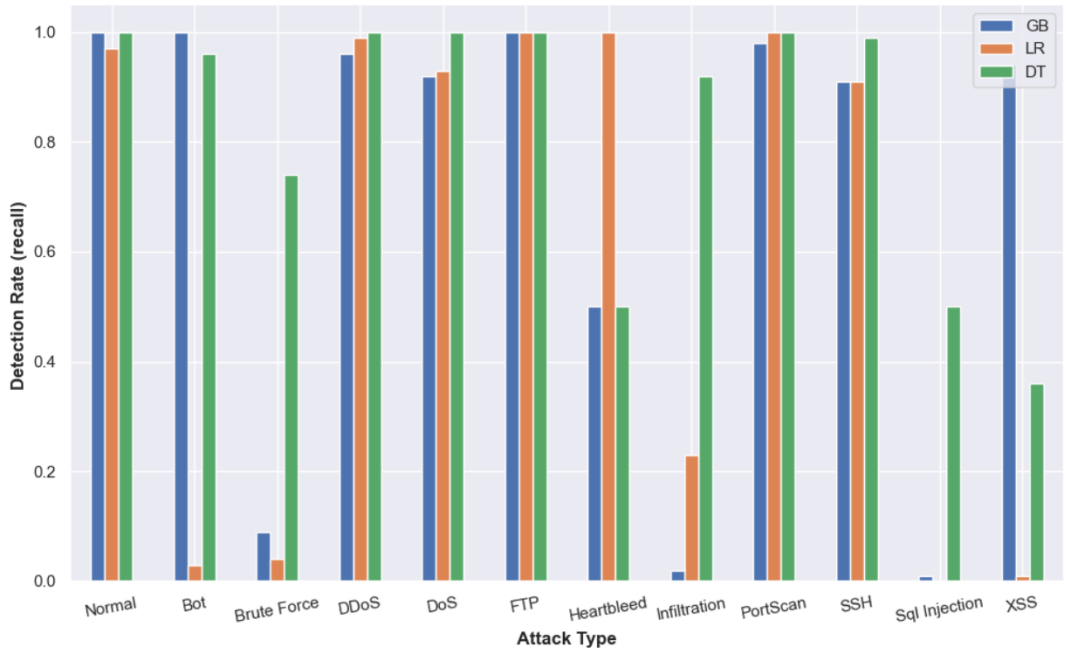


Figure 4.4: Multiclass Recall for CICIDS 2017 Data Set with Different Classifiers.

According to the findings, the majority class (normal) had a high recall in all classifiers. Nonetheless, not all minority classes are difficult to detect in this data set, for instance, DDoS, DoS, FTP, Port Scan, and SSH, which all have a recall rate greater than 80% in all classifiers. This might be because these minority classes have distinct features that differentiate them, despite they consist only of a modest distribution. Subsequently, some minority classes are better recognised by a single classifier. For example, Brute Force, Infiltration, and SQL injection have a greater recall rate in DT. Whereas LR is better at detecting Heartbleed, but it is not suited for BOT and XSS detection (near-zero recall rate). Next, despite being a minority class, DDoS, FTP, and Port Scan had achieved a high recall rate of 95 % and above. Similar to UNSW-NB15, some minority classes in this data set are better detected by

a single classifier. However, almost half of the minority classes can be distinguished from the others. Figure 4.4 shows the multiclass accuracy for CICIDS 2017 with different classifiers.

4.1.1.3 NSL KDD

Table 4.3: Comparison of Results Between Classifiers for the NSL KDD Data Set

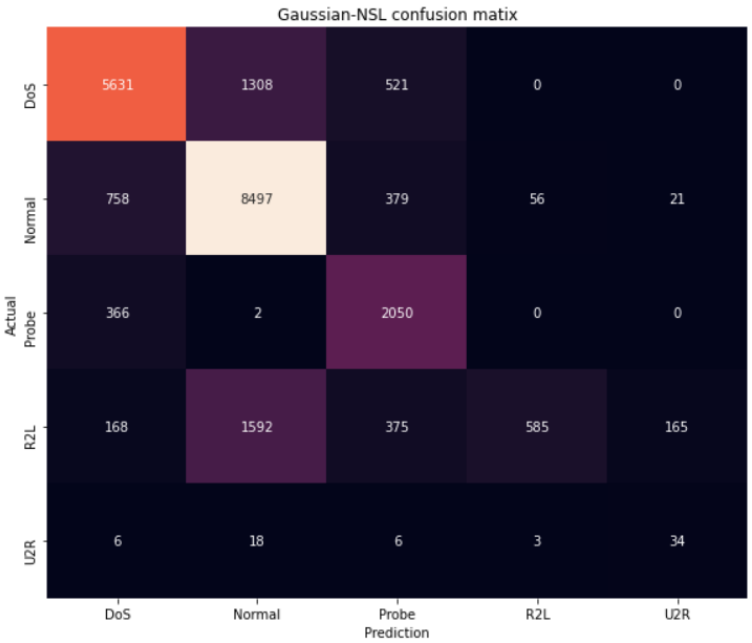
*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage, **F** indicates **F measures** in percentage, the bold numbers indicate the minority class in the data set.)*

Attack Type	**Gaussian Bayes (GB)**			**Logistic Regression (LR)**			**Decision Tree (DT)**		
	P	**DR**	**F**	**P**	**DR**	**F**	**P**	**DR**	**F**
DoS	81	75	78	90	85	87	81	74	71
Normal	74	87	80	69	92	79	71	90	80
Probe	62	**85**	71	76	**76**	76	38	**58**	46
R2L	91	**20**	33	43	**0**	1	36	**7**	12
U2R	15	**51**	24	86	**18**	30	0	**0**	0

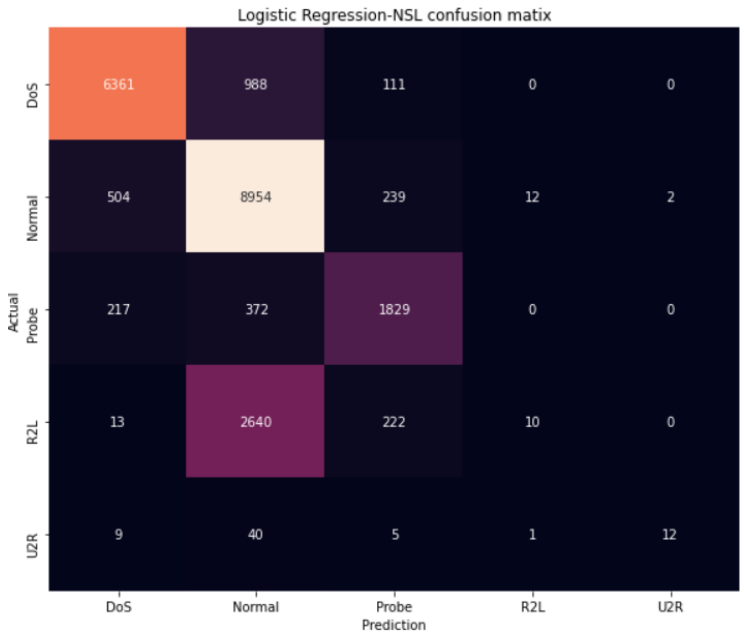
The comparison results of the classifiers for NSL KDD are summarised in Table 4.3. As observed, Normal and DoS have a recall rate of 74 % and above, with the Normal class accounting for at least 87 % of recall. Apart from that, Probes is the minority class that is best detected by all classifiers, with a recall rate ranging from 58 % to 85 %. It had a satisfying recall rate of 76 % and above in GB and LR, but a poor recall rate in DT (58 %). In this relationship, it can be deduced that a probabilistic classifier may be a better choice for detecting Probes rather than the branching technique. Moreover, the smallest minority class (U2R) had a detection rate that varied from 0 % to 51 %, while R2L had the worst performance, with a recall rate ranging from 0 % to 20 %.

U2R has a higher detection rate, despite it has a smaller distribution compared to R2L. This is similar to the Heartbleed traffic in CICIDS 2017, in which one sample contributes a significant percentage in small distributions. As a result, the detection rate has a wider range, but it cannot indicate its performance. Subsequently, all classifiers are effective at detecting Probes, yet the classifiers are better at

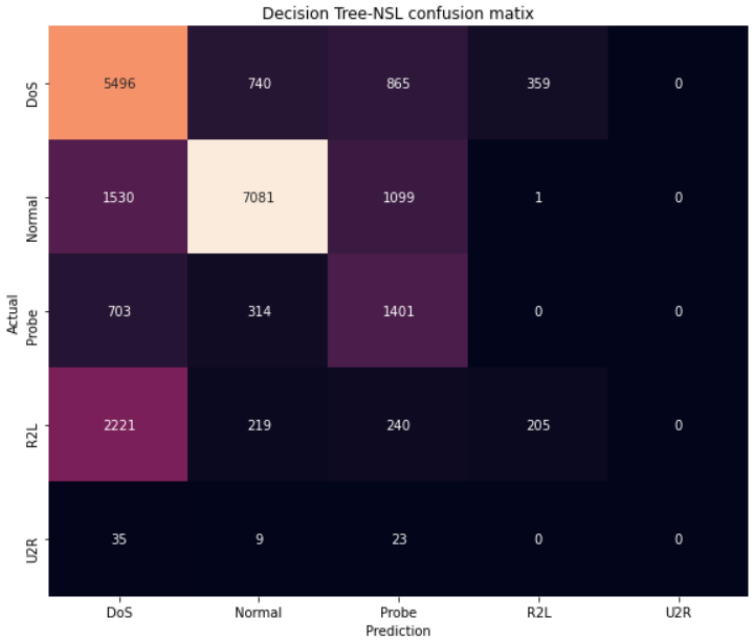
predicting R2L. In summary, the minority classes in this data set have a poor detection rate of less than 80 %.



(a)



(b)



(c)

Figure 4.5: Confusion Matrix Between Classifiers for NSL KDD:

(a) Gaussian Bayes (GB) (b) Logistic Regression (LR) (c) Decision Tree (DT)

As seen in the confusion matrix, the majority of DoS traffic was misclassified as normal traffic by all classifiers. While a substantial amount of Probes was detected as DoS in DT, while most R2L traffics were detected as Normal in LR and GB. R2L is commonly detected as Normal traffic if the classifier uses a probabilistic method, but it may resemble DoS if the branching method is used. Apart from that, all classifiers detected the U2R to others. The U2R may have no distinguishable features and have limited distribution. Figure 4.5 depicts the confusion matrix between classifiers for NSL KDD. Overall, similar to the prior two data sets, more data is needed for minority classes to evaluate their detection rate accurately.

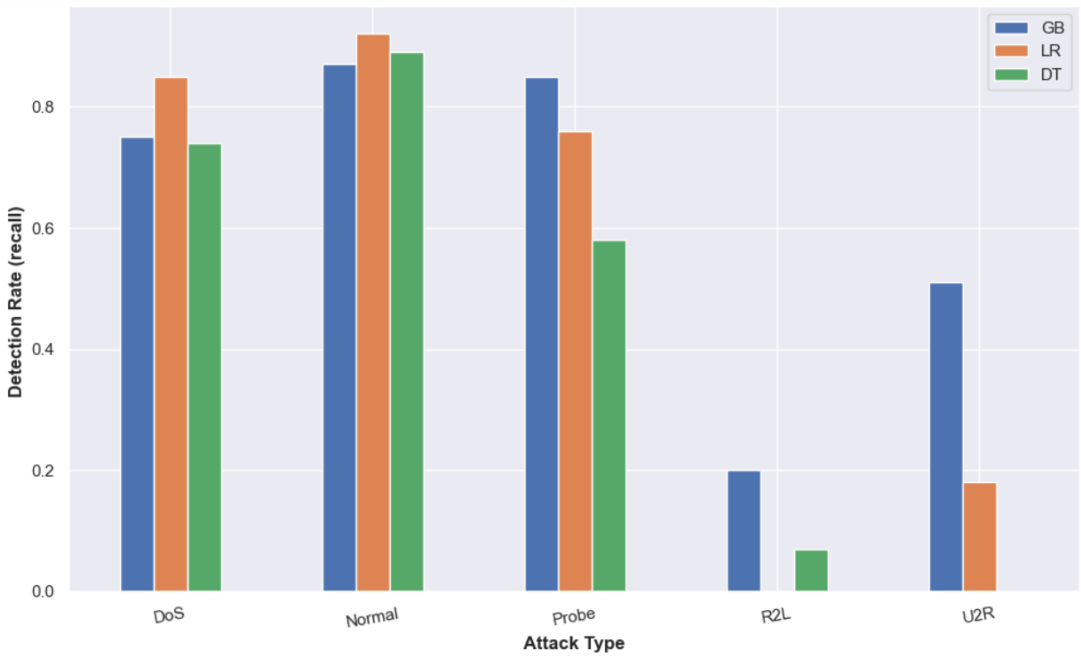


Figure 4.6: Multiclass Recall for NSL KDD Data Set with Different Classifiers.

Figure 4.6 shows the multiclass accuracy for CICIDS 2017 with different classifiers. From the figure above, the Normal traffic achieved the greatest detection rate of all the classifications. While Probe had obtained the highest detection rate among the minority classes in all classifiers. Despite U2R is the modest minority class, but R2L had a lower detection rate than U2R. As mentioned in the previous section, the classifiers might confuse R2L traffic with other classes. So far, GB was performed well in terms of traffic detection. This is because it can retain a satisfactory detection rate for the majority classes (DoS and Normal) while achieving the greatest detection rate for minority classes.

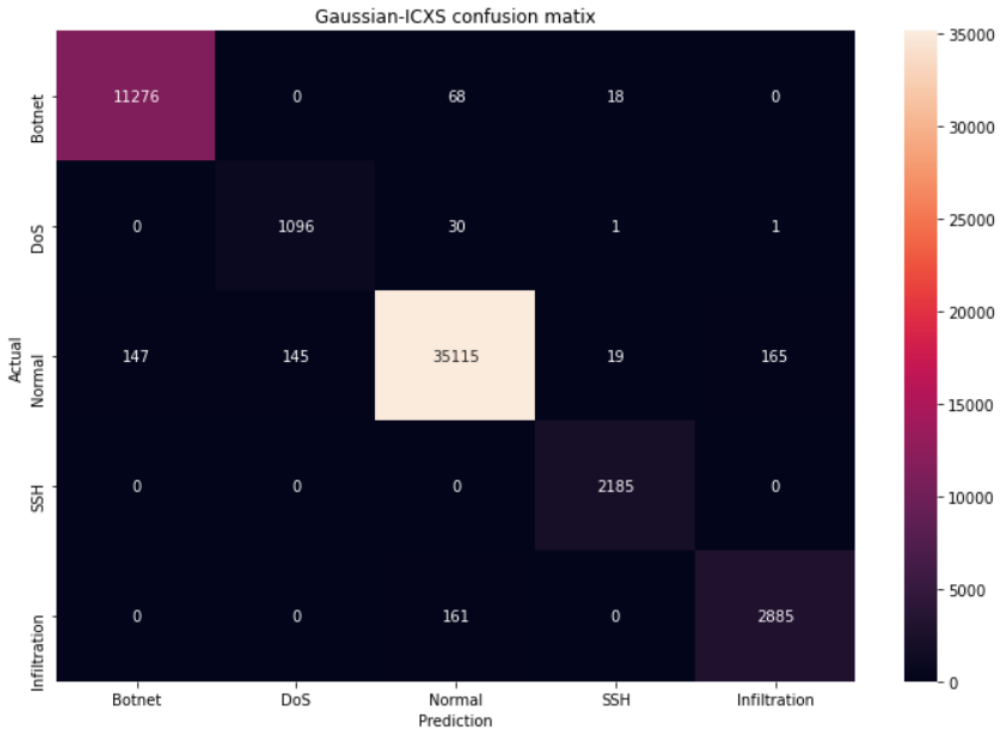
4.1.1.4 ICXS 2012

Table 4.4: Comparison Results Between Classifiers for ICXS 2012 Data Set

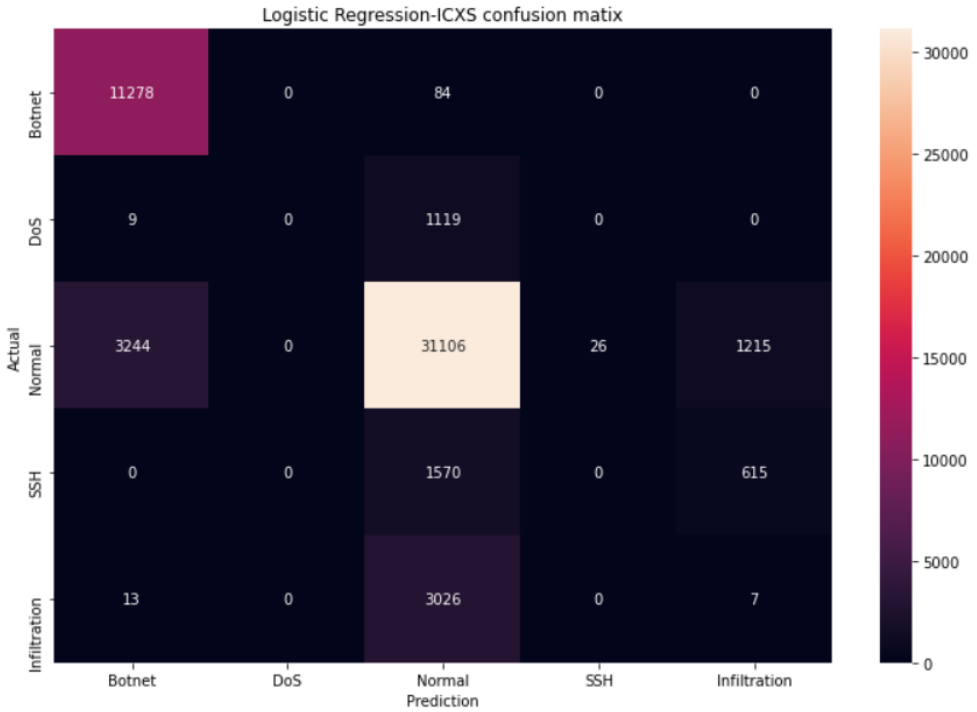
*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage, **F** indicates **F measures** in percentage, the bold numbers indicate the minority class in the data set.)*

Attack Type	**Gaussian Bayes (NB)**			**Logistic Regression (LR)**			**Decision Tree (DT)**		
	P	**DR**	**F**	**P**	**DR**	**F**	**P**	**DR**	**F**
DoS	88	**86**	93	0	**0**	0	90	**100**	100
Normal	99	99	99	84	87	86	100	100	100
Botnet	99	**99**	99	78	**99**	87	100	**80**	97
SSH	98	**81**	99	0	**0**	0	100	**81**	94
Infiltration	95	**95**	95	0	**0**	0	100	**86**	100

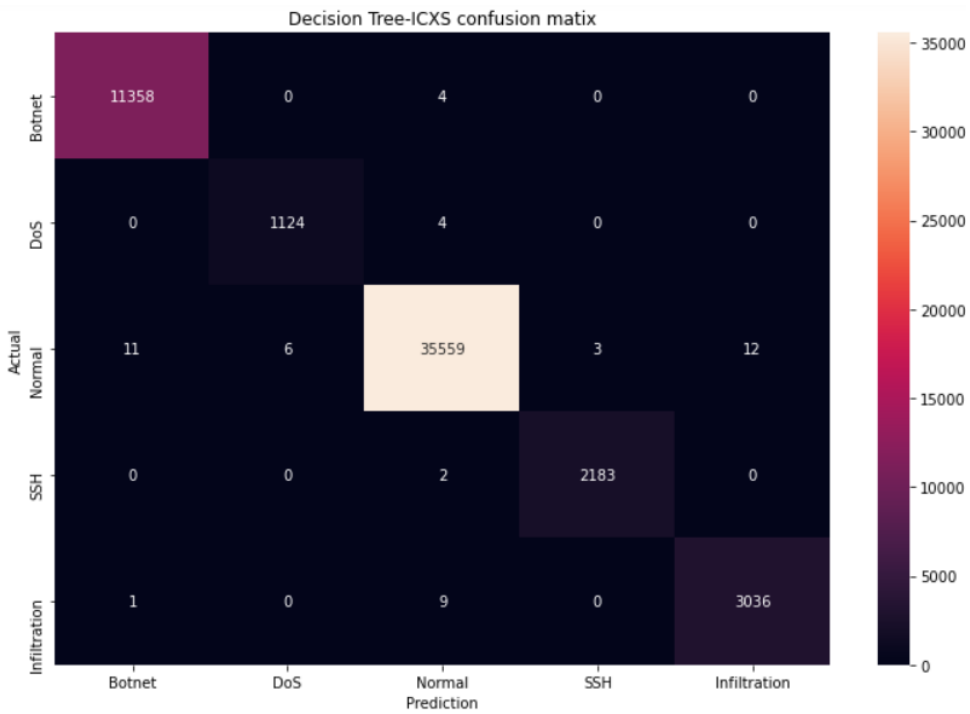
Table 4.4 shows the comparative results for ICXS 2012 using multiple classifiers. Normal traffic had a decent detection and prediction rate in all classifiers, which were all above 80 %. The botnet had a detection rate of 80 % for all classifiers, albeit accounting for only 2 % of the overall distribution. This attack may have distinct traits to differentiate it from others. Subsequently, the other three attack classes: DoS, SSH, and Infiltration, acquired a 0 % detection rate in LR, but obtain an ideal detection from other classifiers. Unlike other data sets, this data set can attain a high detection rate by just selecting the suitable classifier. For example, DT can detect all classes in this data set with a detection rate of 80 % and above, and its F score is 94 % and above. This indicates that DT outperformed all other classifiers, whereas LR performed the worst.


Gaussian-ICXS confusion matix
Botnet
DoS
Normal
SSH
Infiltration
Actual
Prediction
11276
0
68
18
0
0
1096
30
1
1
147
145
35115
19
165
0
0
0
2185
0
0
0
161
0
2885
35000
30000
25000
20000
15000
10000
5000
0

(a)


Logistic Regression-ICXS confusion matix
Botnet
DoS
Normal
SSH
Infiltration
Actual
Prediction
11278
0
84
0
0
9
0
1119
0
0
3244
0
31106
26
1215
0
0
1570
0
615
13
0
3026
0
7
30000
25000
20000
15000
10000
5000
0

(b)



(c)

Figure 4.7: Confusion Matrix Between Classifiers for ICXS 2012:

(a) Gaussian Bayes (GB) (b) Logistic Regression (LR) (c) Decision Tree (DT)

Figure 4.7 illustrates the confusion matrix between classifiers for ICXS 2012. As explored, both GB and DT classifiers performed well in classification. Where there is only a minor misclassification of normal traffic to other classes in GB. Nevertheless, LR had wrongly classified normal as Botnet traffic with a misprediction of 3244 instances. In addition, LR confused the normal traffic with Infiltration traffic, as both classes have the highest rate of misprediction into one another. 99.34 % of infiltrations were misclassified as normal traffic, while 66.39 % of infiltrations' false negatives belong to attack classes.

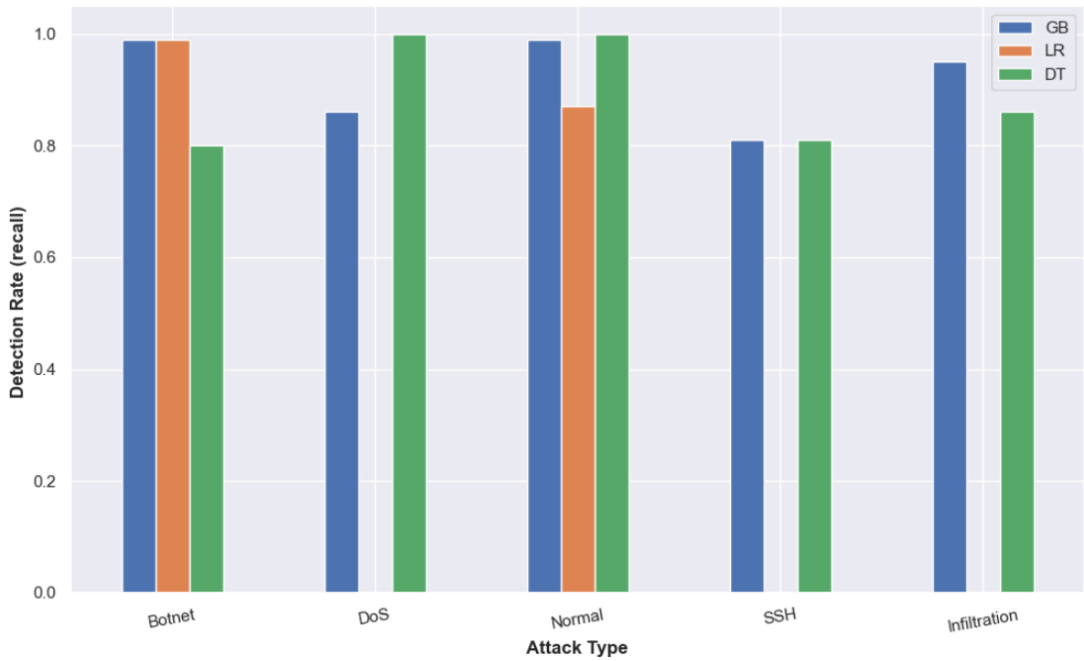


Figure 4.8: Multiclass Recall for ICXS 2012 Data Set with Different Classifiers.

Figure 4.8 shows the multiclass recall rate for ICXS 2012 with multiple classifiers. In this data set, not only did normal traffic attain a satisfactory detection rate, but the Botnet also had a greater detection rate than other attack classes. Furthermore, DoS, SSH, and Infiltration also achieve a decent detection rate in GB and DT, while having the lowest recall in LR. Despite, this data set is having an unbalanced distribution problem, it does not require further pre-processing to increase the detection rates in all classes despite having the unbalanced distribution problem. All classes may contain sufficient instances and distinctive features for a certain classifier to identify their boundary without much hardship.

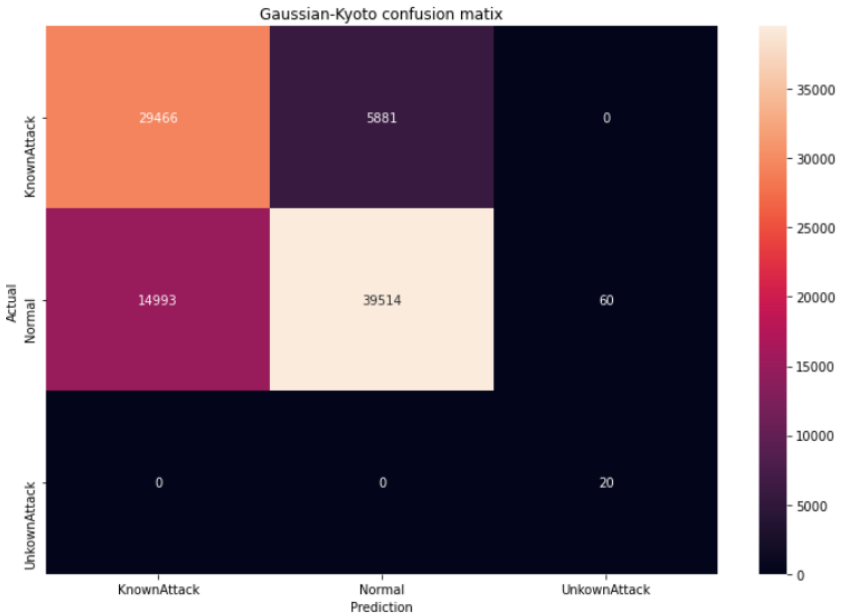
4.1.1.5 Kyoto 2006

Table 4.5: Comparison of Results Between Classifiers for the Kyoto 2006 Data Set

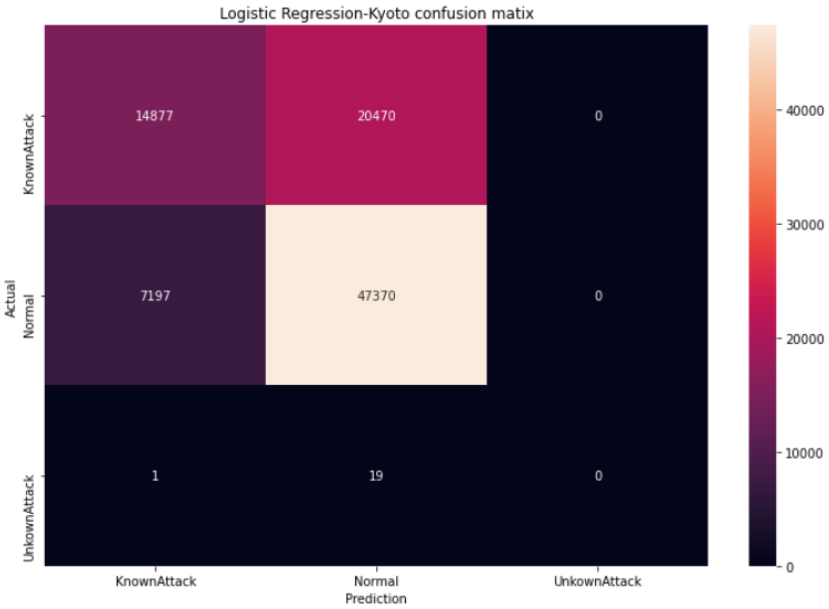
*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage, **F** indicates **F measures** in percentage, the bold numbers indicate the minority class in the data set)*

Attack Type	Gaussian Bayes (NB)			Logistic Regression (LR)			Decision Tree (DT)		
	P	**DR**	**F**	**P**	**DR**	**F**	**P**	**DR**	**F**
Known Attack	66	83	74	67	42	52	100	100	100
Normal	87	100	79	70	87	77	100	100	100
Unknown Attack	25	**72**	40	0	**0**	0	100	**100**	100

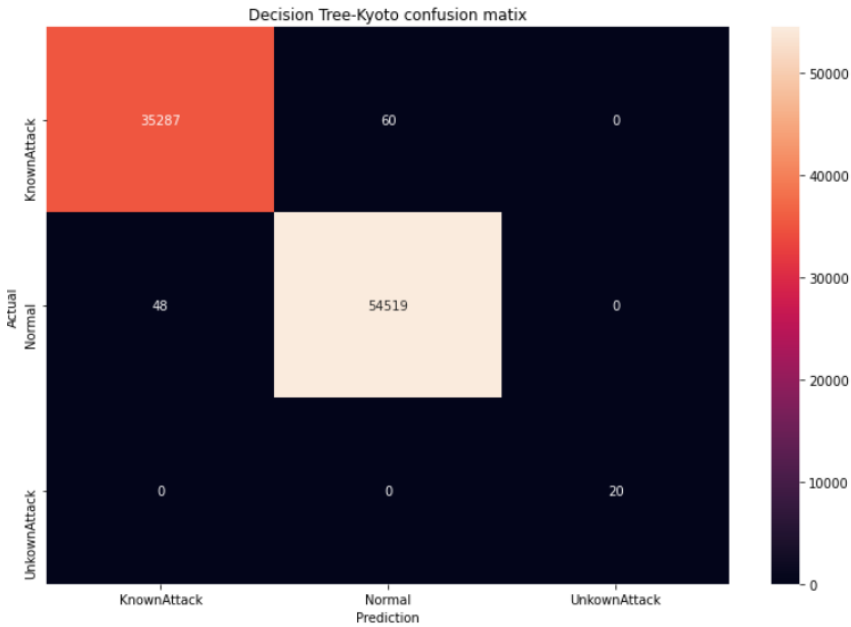
Table 4.5 tabulates the comparative results for Kyoto 2006 with multiple classifiers. This data set did not differentiate all attacks into different labels like other data sets. On the contrary, its categories all the attacks traffic into two categories: known attacks that have been identified, and unknown attacks that have yet to be discovered. Therefore, despite Known attacks account for nearly half of the entire distribution. The known attack's detection rate remains unstable in the classification, with only a 42 % recall in LR. Besides that, the only minority class (unknown attack) has a detection rate ranging from 0 to 72 % in all classifiers, with LR performing the weakest. Similar to ICXS 2012, DT was able to classify all instances with the highest recall, resulting in a 100 % F measure.



(a)



(b)



(c)

Figure 4.9: Confusion Matrix Between Classifiers for Kyoto 2006:
(a) Gaussian Bayes (GB) (b) Logistic Regression (LR) (c) Decision Tree (DT)

Figure 4.9 depicts the confusion matrix between classifiers for Kyoto 2006. As observed, GB and LR were unable to classify between normal and known attack traffics accurately. Whereas GB struggles to classify the normal traffic, LR is poor at

classifying known attacks. Furthermore, DT has the fewest incidences of misclassification, with all unknown attacks accurately predicted. Thus, we may conclude that DT outperformed all other classifiers.

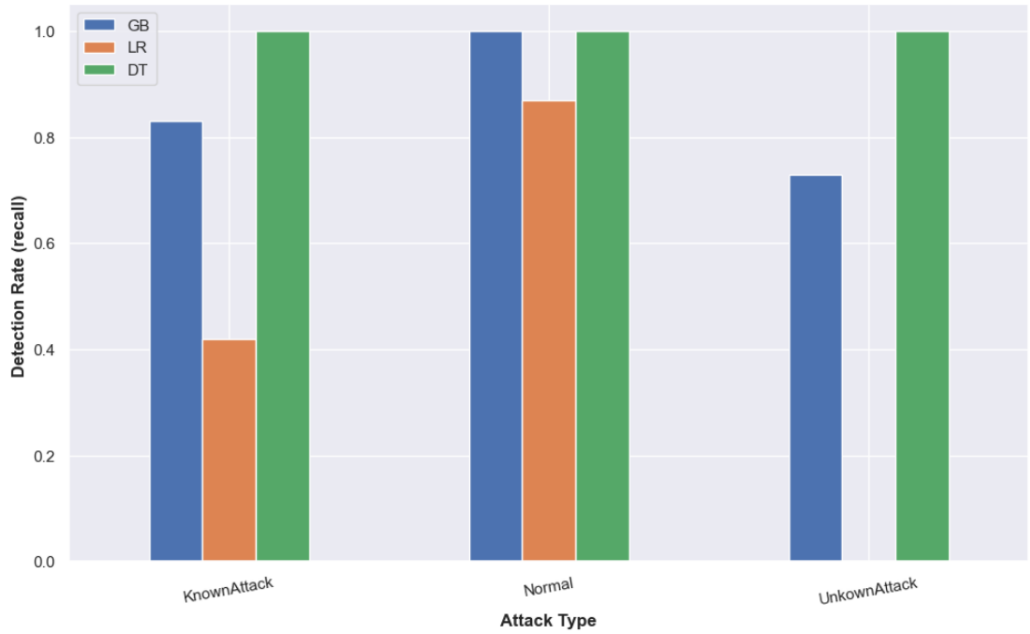


Figure 4.10: Multiclass Accuracy for Kyoto 2006 Data Set with Different Classifiers.

Figure 4.10 describes the multiclass recall rate for Kyoto 2006 with different classifiers. As discussed in the literature review, normal traffic always has a greater detection rate than other classes. Similar to the majority findings, the detection rate for known and unknown traffic is classifier dependent. It means that only a specific classifier can improve the detection of a specific class. For example, DT performed the best in classifying these two traffics, whereas LR performed the worst, with an unknown attack rate of 0 %.

4.1.1.6 CICDDOS 2019

Table 4.6: Comparison of Results Between Classifiers for the CICDDOS 2019 Data Set

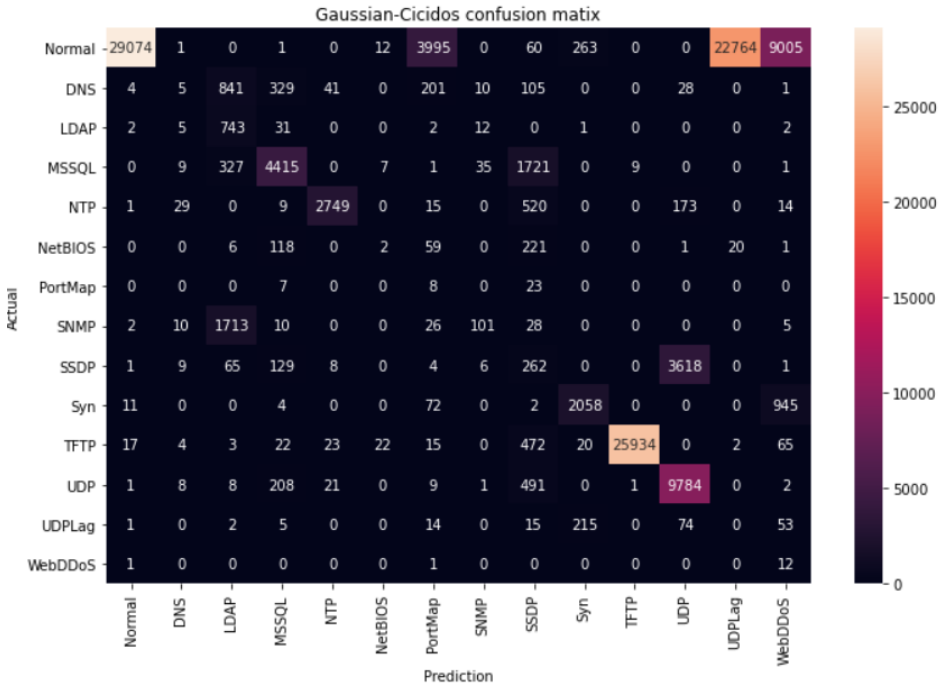
*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage, **F** indicates **F measures** in percentage, the bold numbers indicate the minority class in the data set)*

Attack Type	**Gaussian Bayes (NB)**			**Logistic Regression (LR)**			**Decision Tree (DT)**		
	P	**DR**	**F**	**P**	**DR**	**F**	**P**	**DR**	**F**
Normal	100	95	98	98	100	99	100	100	100
DNS	6	**0**	1	35	**6**	10	66	**38**	48
LDAP	20	**93**	33	64	**1**	2	40	**58**	48
MSSQL	83	68	75	77	94	85	88	95	92
NTP	97	**78**	87	95	**80**	87	100	**99**	99
NetBIOS	5	**0**	1	40	**6**	10	73	**57**	64
PortMap	0	**21**	0	0	**0**	0	0	**0**	0
SNMP	61	**5**	10	47	**87**	62	71	**78**	75
SSDP	7	**2**	7	38	**0**	0	32	**1**	1
Syn	80	**67**	73	85	**62**	72	96	**90**	98
TFTP	100	97	99	100	99	99	100	100	100
UDP	72	**93**	81	68	**94**	79	71	**98**	83
UDPLag	0	**0**	0	14	**1**	1	97	**53**	69
WebDDoS	0	**86**	0	0	**0**	0	40	**14**	21

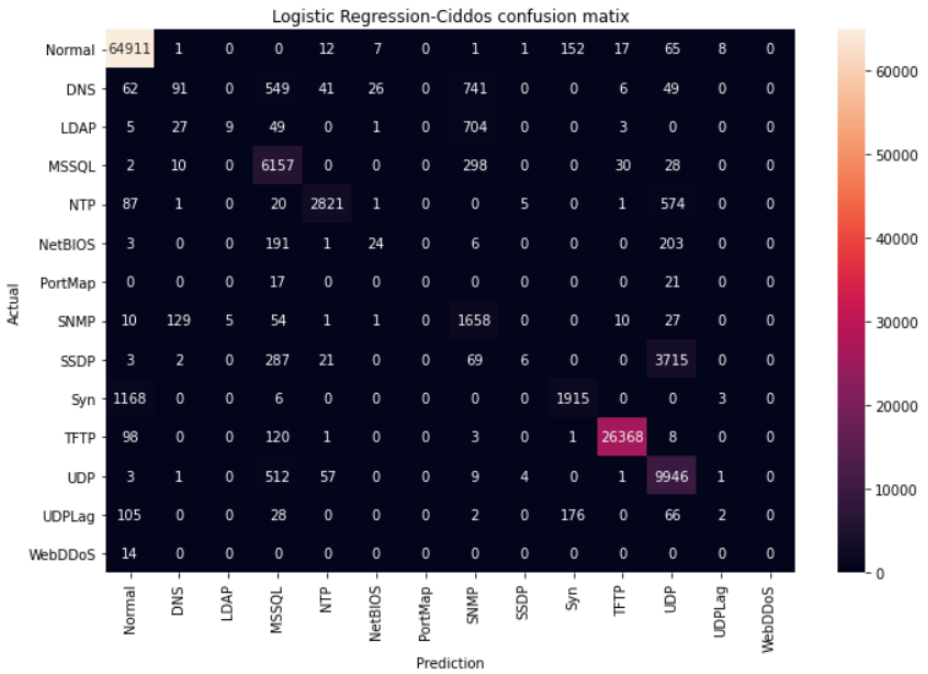
Table 4.6 summarises the comparative results for CICDDOS 2019 between multiple classifiers. Similar to other data sets, normal traffic still obtained the highest detection rate among the classes in all classifiers. TFTP is both attack and majority, resulting in a detection rate similar to normal traffic, with the lowest detection rate maintained at 97 %. On top of that, although being minorities, NTP and UDP can still achieve a fair detection rate of at least 78 % due to the distinguishable boundary.

Subsequently, LDAP, SNMP, Syn, and WebDDoS are all reliant on a particular classifier to obtain a high detection rate. For instance, GB has a 93 % recall for LDAP but just 1 % in LR, while Syn only has a 62 % recall but a 99 % in DT. Finally, the remaining attack classes such as NetBIOS, UDP Lag, DNS, Port Map,

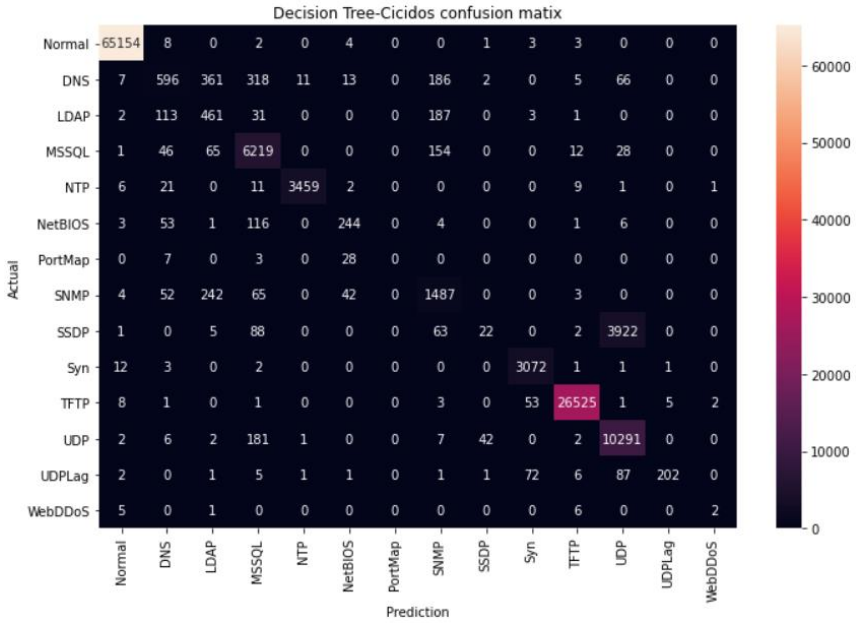
SSDP did not perform well in any classifier. The detection rate for these attacks varies between 0 % and 57 %. In particular, SSDP performed the worst with a recall of only 6 % across all classifiers. This suggests that a minor contribution to distribution (2.6 %) may not be sufficient to capture the true boundary.



(a)



(b)



(c)

Figure 4.11: Confusion Matrix Between Classifiers for CIDDOS 2019:

(a) Gaussian Bayes (GB) (b) Logistic Regression (LR) (c) Decision Tree (DT)

Figure 4.11 shows the confusion matrix between classifiers for CICDDOS 2019. As observed, four attack classes: MSSQL, NTP, TFTP, and UDP can predict most instances accurately in all classifiers. On the other hand, DNS traffic was misclassified as LDAP in GB. However, the LR is unable to distinguish the boundaries between DNS and LDAP with SNMP, causing the traffics to be misclassified as SNMP. Furthermore, SNMP is performing well in both LR and DT classifiers, but it has been misclassified as LDAP in GB. Based on the relationship, there may be a high similarity between LDAP and SNMP, because both were misclassified to each other.

Similar to Port Map, NetBIOS is well predicted in DT, while both traffics are classified as SSDP in GB. This could indicate that NetBIOS, Port Map, and SSDP have some commonalities, resulting in frequent misinterpretation. As shown in the diagram, LR and DT often misclassified WebDDoS as normal traffics. Eventually, UDP Lag and SSDP do not perform well in all classifiers. In addition, UDP Lag is often misclassified as Syn, while SSDP is frequently misclassified as UDP in all classifiers. In general, this data set contains a few groups of attacks that are similar to one another, resulting in misclassification.

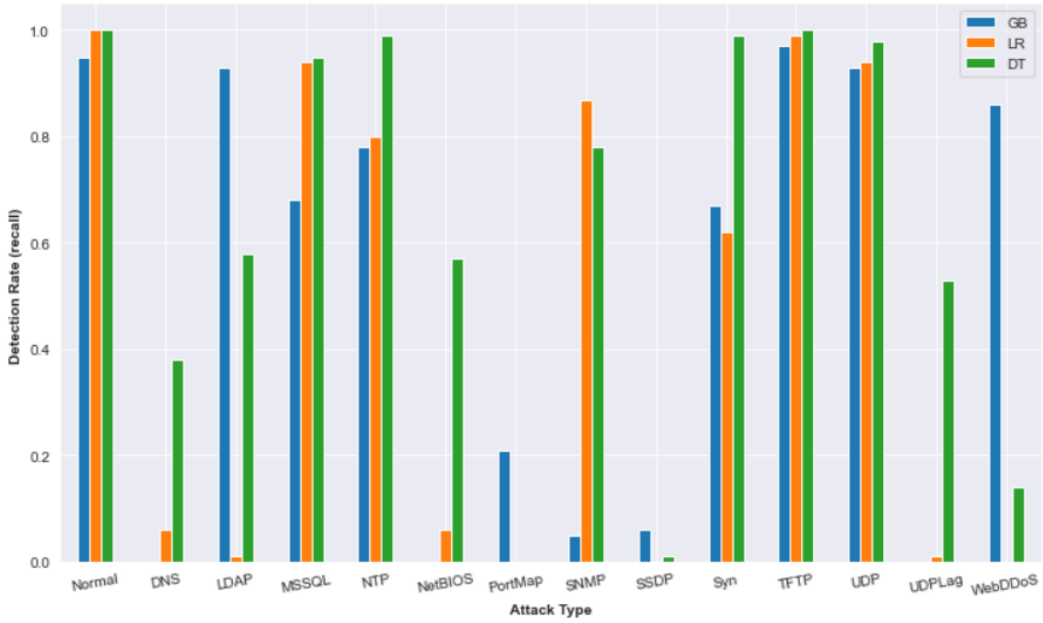


Figure 4.12: Multiclass Accuracy for CICDDOS 2019 Data Set with Different Classifiers.

Figure 4.12 shows that few attacks traffic can be detected well even before oversampling. Those are TFTP, UDP, Normal, NTP, and MSSQL with at least 78 % recall. Additionally, the detection performance of LDAP and WebDDoS is highly dependent on the classifiers chosen. In this example, GB performed the best in detecting these two traffics. Finally, DNS, NetBIOS, Port Map, SSDP, and UDP Lag require efforts in detection because most of their detection rates do not even reach 50 %. Some attack classes, such as SSDP and Port Map, even have a detection rate of no more than 25 % across all classifiers.

4.1.2 Overall Performance Evaluation

Table 4.7: Comparison Results for Different Data Sets

(Notes : the bold numbers indicate the highest F score among all observations as per data set)

Data set	Classifier	Precision(%)	Detection Rate (%)	F score(%)
UNSW-NB15	Gaussian Bayes	61	44	42
	Logistic Regression	68	54	**56**
	Decision Tree	40	41	39
CICIDS 2017	Gaussian Bayes	95	79	85
	Logistic Regression	95	95	**95**
	Decision Tree	83	83	81
NSL_KDD	Gaussian Bayes	77	75	**73**
	Logistic Regression	73	76	71
	Decision Tree	66	67	64
ICXS 2012	Gaussian Bayes	99	99	**99**
	Logistic Regression	73	80	76
	Decision Tree	100	89	93
Kyoto 2006	Gaussian Bayes	79	77	77
	Logistic Regression	69	69	67
	Decision Tree	100	100	**100**
CICDDOS 2019	Gaussian Bayes	90	60	69
	Logistic Regression	90	91	89
	Decision Tree	93	94	**93**

Table 4.7 presents the comparative findings for various data sets. As observed, LR outperformed other classifiers in all data sets, having the highest detection rate. As mentioned in the problem statement, the aggregate performance matrix often

claimed to be biased against the majority classes, it depicts in CICIDS 2017, and CICDDOS 2019. According to findings in sections 4.1.2 (CICIDS 2017) and 4.1.6 (CICDDOS 2019), at least 50 % of the classes have a detection rate below 60 %. Nonetheless, all classifiers for CICIDS 2017 and DT for CICDDOS 2019 still had recall and F measures of 85 % and above, which misled the classifier performance. The cross-validation score as tabulated in Appendix D also shows that the performance for CICIDS 2017, CICDDOS 2019, and NSL KDD classification is inflated by having a detection rate of 90 % in all cv using specific classifiers.

The high detection rate in this situation was influenced not just by majority classes, but also by "majority attack subsets in minority attack classes". In other words, some minority classes with larger data distribution and a high detection rate also increased the aggregates detection rate. While those minority attack classes that contain an underrepresented data distribution still suffered from low detection. As a result, the high detection rate and F score in one classifier in such condition do not reflect the classifier's effectiveness in detecting minorities. Where no classifier can predict the traffics well in UNSW-NB15, because the highest F measure and recall are only 56 % and 54 %, respectively.

Conversely, the aggregate measure for ICXS 2012 and Kyoto 2006 may reflect the classifier performance. This is because all classes in these data sets can obtain a highly satisfying detection rate by selective modelling. Those are Gaussian Bayes for ICXS 2012 and Decision Tree for Kyoto 2006. The cross-validation score for all data sets, as tabulated in Appendix D, indicates that these two data sets can consistently achieve a high recall with a particular classifier. Therefore, these data sets may be of little or no value to our objectives.

4.1.3 Summary of Non-Oversampling Results

In summary, the findings reveal two trends in low-detection minority classes. First, the detection rate may vary in a wider range when the minority classes have an extremely small distribution, often less than 1 % of the overall distribution. This is because a single data points often have a significant impact on the performance matrix. Second, the detection rate is poor with a narrow range. This situation occurs when the data samples are greater than the preceding situation, yet lesser than the majority. These problems occur because the data samples available are insufficient to

determine a clear decision boundary between classes. As investigated in chapters one and two, most majority classes in these all have a high detection rate, whereas the minority classes have a low detection rate.

Wherefore, after analysing the results obtained from all data sets, proving the existence of an unbalanced distribution problem. These four data sets that correspond with our problem statement and objectives will be selected to oversample: UNSW-NB15, CICIDS 2017, NSL KDD, and CICDDOS 2019. Whereas ICXS 2012 and Kyoto 2006 were foregone because they can achieve a high detection rate without requiring additional intervention. Eventually, those attack traffics that fail to detect well by at least one of the classifiers will be oversampled.

4.2 Oversampling Results

This section summarised the findings of oversampling for four data sets: UNSW-NB15, CICIDS 2017, NSL KDD, and CICDDOS 2019. The minority class with a low detection rate in at least two classifiers will be selected to oversample. Each data set will oversample the minority classes with the selected sampling methods: ROS, SMOTE, Borderline SMOTE, ADASYN, and K-Mean SMOTE to generate oversampled data from 10 % to 100 % iteratively. Appendix E tabulated the data distribution of all percentages for all data sets. Throughout the experiments, each data set was conducted twice to compare scaled and unscaled oversample data. All observations will be named in the following format: [Abbrev. Machine learning] – [Oversampling method] – [percentage oversampled].

4.2.1 Multiclass Performance Evaluation

4.2.1.1 UNSW-NB15

Table 4.8: The Highest Detection Rate for all Machine Learning Algorithms of each Minority Class in Both Scaled and Non-scaled UNSW-NB15.

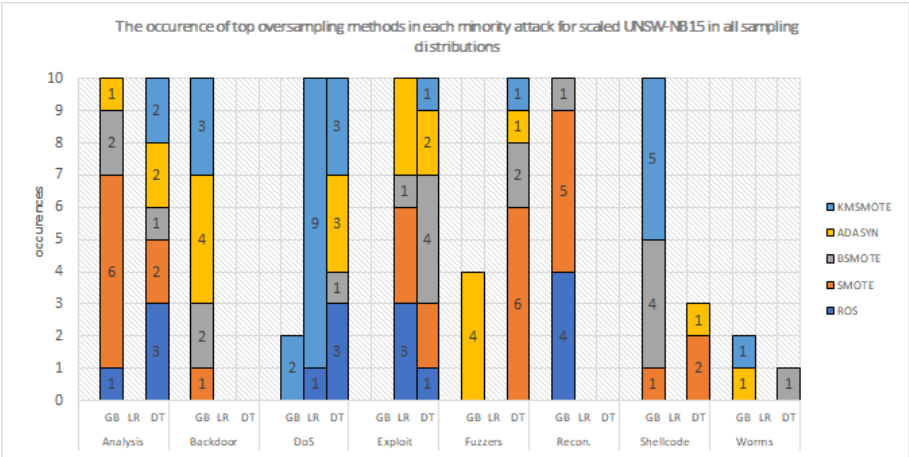
*(Notes : **GB** indicates Gaussian Bayes algorithms, **LR** indicates Logistic Regression Algorithm, **DT** indicates Decision Tree, **DR** indicates detection rates, **S** indicates scaled data, **NS** indicates non scaled data, the bold & italic number indicate the highest detection rate in each minority attack among all observations in the data set, red numbers are the lowest detection rate in each minority class.)*

		Analysis			Backdoor			DoS			Exploit		
		GB	LR	DT	GB	LR	DT	GB	LR	DT	GB	LR	DT
NOS		0	0	0	37	0	1	0	14	4	12	69	18
Top DR in %	S	10	0	***39***	***90***	0	4	34	67	***77***	52	87	95
	NS	0	0	22	2	0	44	62	0	55	66	***99***	92
		Fuzzers			Reconnaissance			Shellcode			Worms		
		GB	LR	DT	GB	LR	DT	GB	LR	DT	GB	LR	DT
NOS		39	58	67	46	0	0	0	0	1	7	0	64
Top DR in %	S	***91***	64	88	70	19	34	***99***	0	3	***66***	0	64
	NS	62	38	72	36	3	***82***	4	0	77	5	0	***66***

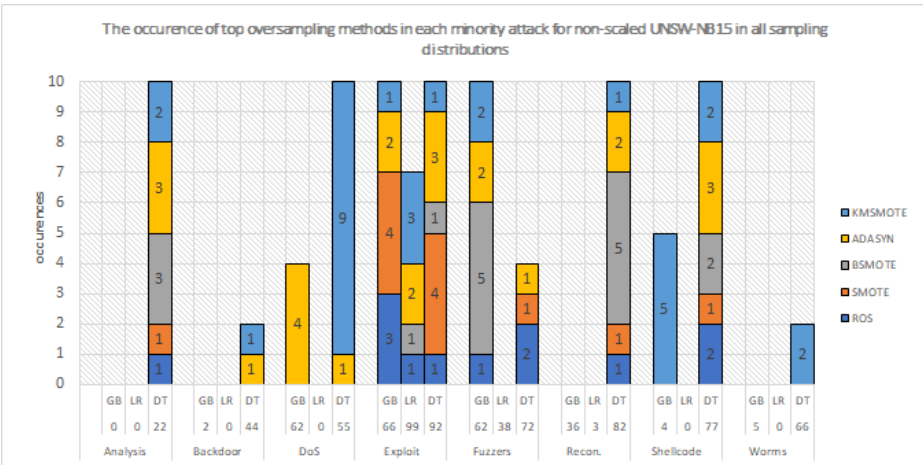
There are eight attacks chosen for oversampling in UNSW-NB15: Analysis, Backdoor, DoS, Exploits, Fuzzers, Reconnaissance, Shellcode, and Worms. Table 4.8 extracted the results of the highest DR for all machine learning of each minority class from Appendix H. The full detection rate for each sampling iteration was

documented in Appendix F & G. Most oversampled attacks that obtained higher individual DR increase in scaled data, except Exploits and Reconnaissance. Some minority classes are able to acquire a satisfied DR regularly in both scaled and non-scaled experiments. For instance, Exploits had a top detection rate ranging from 87 % to 99 % in all LR and DT algorithms.

Furthermore, Backdoor and DoS only have a satisfying performance on scaled data, where most non-scaled DR are below 65 %. Analysis obtained the lowest DR, 39 %, even when the data had oversampled. Most of its traffic was mispredicted as Fuzzers, as shown in Appendix I(a). Lastly, worms have the least changes (2 %) in DR despite implementing the oversampling method, from 64 % to 66 %. To conclude, some attacks were able to obtain a signification DR increase only in a certain model, such as GB for scaled Backdoor. However, no single oversampling method can be outperformed in all machine learning algorithms in one minority class.



(a)



(b)

Figure 4.13: The Occurrence of Top Oversampling Methods for Each Minority Class in UNSW-NB15, Regardless of the Sampling Distributions.

(a) Scaled Experiment (b) Non-scaled Experiment

Figure 4.13 depicts the ten occurrences of top oversampling methods for each minority class in all UNSW-NB sampling distributions. The figure depicts that, some oversampling methods outperform in certain classes. For example, KMSMOTE in DoS for both scaled and non-scaled data accounts for 22 occurrences out of 36. Furthermore, some minority attacks were algorithms dependent, with one to two machine learning algorithms having a DR increase. For instance, only GB in scaled data, and DT in non-scaled data can increase the DR for Backdoor, similar to Reconnaissance. Moreover, worms have the lowest occurrence of top oversampling when compared to other attacks, with a total of 5 in both scaled and non-scaled data. In terms of occurrences and DR increase for worms, the selected oversampling methods are ineffective in increasing the DR for this attack class. Overall, KMSMOTE has the highest number of occurrences of individual top oversampling methods in both scaled and non-scaled UNSW-NB15.

4.2.1.2 NSL KDD

Table 4.9: The highest detection rate for all machine learning algorithms of each minority class in both scaled and non-scaled NSL KDD.

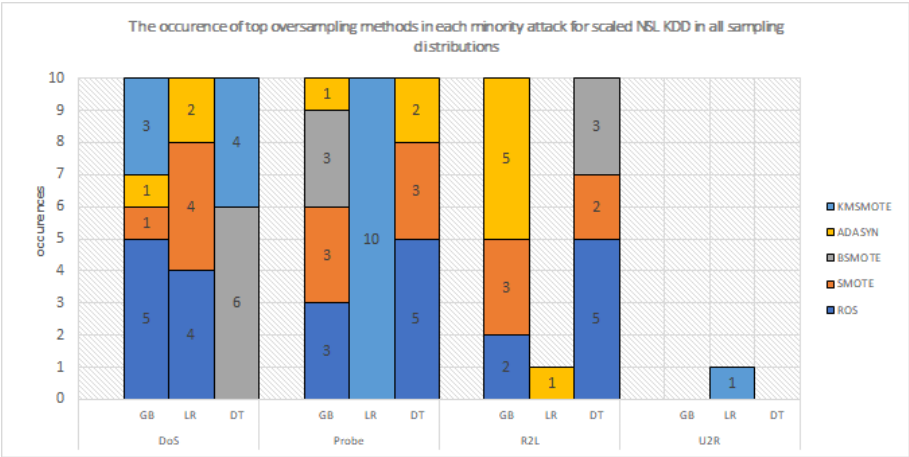
*(Notes : **GB** indicates Gaussian Bayes algorithms, **LR** indicates Logistic Regression Algorithm, **DT** indicates Decision Tree, **DR** indicates detection rates, **S** indicates scaled data, **NS** indicates non scaled data, the bold & italic number indicate the highest detection rate in each minority attack among all observations in the data set, red numbers are the lowest detection rate in each minority class.)*

		DoS			Probe			R2L			U2R		
		GB	LR	DT	GB	LR	DT	GB	LR	DT	GB	LR	DT
NOS		75	85	74	85	76	58	20	0	7	51	18	0
Top DR in %	S	**94**	90	89	86	**100**	89	**90**	1	30	51	57	0
	NS	87	80	83	63	0	82	28	8	28	**73**	0	40

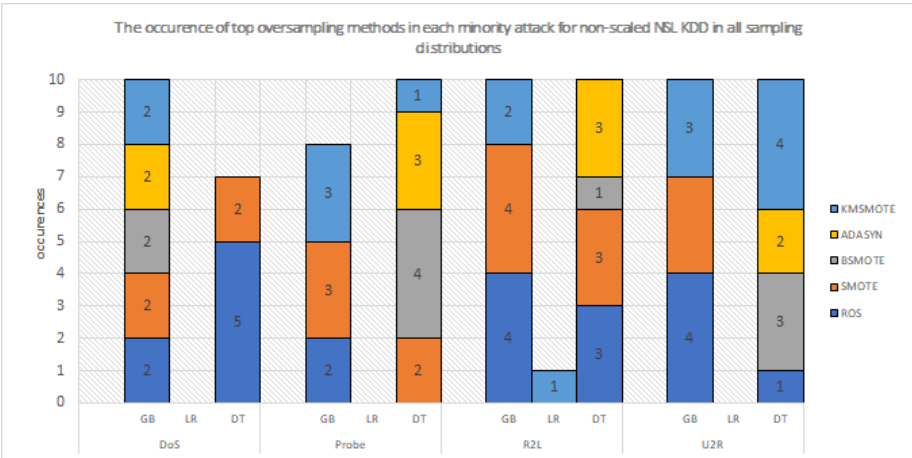
Table 4.9 extracted the results of the highest DR for all machine learning of each minority class in NSL KDD from Appendix L. Four attacks were selected to

oversample: DoS, Probe, R2L, and U2R. The full detection rate for each sampling iteration was documented in Appendix J & K. According to the findings, DoS can maintain satisfactory DR in all top observations. Particularly in scaled data, where DR ranges from 89 % to 94 %, whereas the highest non-oversampled DR is only 85 %. However, referring to the top combinations listed in Appendix L, such DR increases are only effective on specific model combinations. For example, ADASYN or KMSMOTE with GB.

Moreover, Probe has better DR performance in scaled data, similar to DoS where all DR is higher than the non-oversampled DR, ranging from 86 % to 100 % (LR). In addition, R2L and U2R produce the opposite result in both observations. R2L achieved its highest DR, 90 % in scaled data, while U2R obtained the highest in non-scaled data, 73 %. Finally, R2L has the highest increase in DR, with a 70 % increase over its lowest non-oversampled DR (0 % to 90 %).



(a)



(b)

Figure 4.14: The Occurrence of Top Oversampling Methods for each Minority Class in NSL KDD, Regardless of the Sampling Distributions.

(a) Scaled Experiment (b) Non-scaled Experiment

Figure 4.14 depicts the ten occurrences of top oversampling methods for each minority class in all NSL KDD sampling distributions. Some oversampling methods outperform a specific model in certain minority attacks. For example, KMSMOTE in Probe accounts for 10 out of 10 occurrences and achieves a DR of 100 % eight times, according to Appendix L. Unlike UNSW-NB15, the oversampling methods are more effective in this data set, as only scaled U2R has the least occurrences across all machine learning algorithms. Furthermore, the selected oversampling methods perform well in DoS and Probe, in which all the machine learning have top occurrences. Overall, ROS has the highest number of occurrences of individual top oversampling methods in both scaled and non-scaled NSL KDD, with 45 occurrences.

4.2.1.3 CICIDS 2017

Table 4.10: The Highest Detection Rate for all Machine Learning Algorithms of each Minority Class in both Scaled and Non-scaled CICIDS 2017.

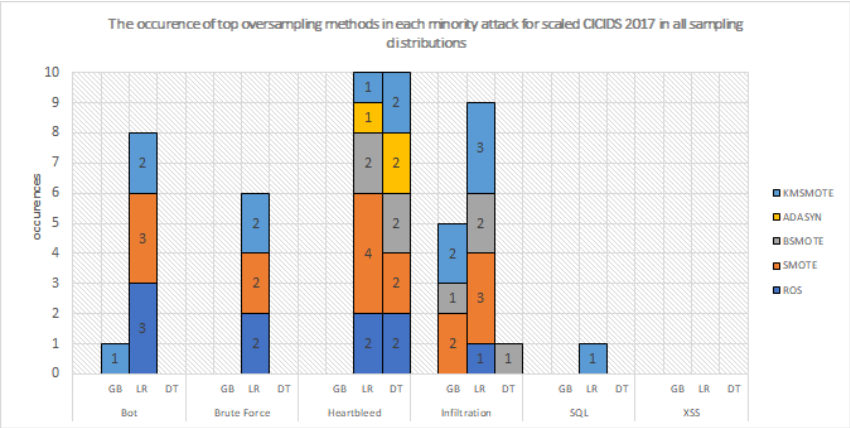
*(Notes : **GB** indicates Gaussian Bayes algorithms, **LR** indicates Logistic Regression Algorithm, **DT** indicates Decision Tree, **DR** indicates detection rates, **S** indicates scaled data, **NS** indicates non scaled data, the bold & italic number indicate the highest detection rate in each minority attack among all observations in the data set, red numbers are the lowest detection rate in each minority class.)*

		Bot			Brute Force			Heart Bleed		
		GB	LR	DT	GB	LR	DT	GB	LR	DT
NOS		80	3	85	9	4	75	50	**100**	50
Top DR in %	S	1	33	26	0	86	9	0	**100**	**100**
	NS	65	0	**99**	90	0	**91**	**100**	0	**100**
		Infiltration			SQL			XSS		
		GB	LR	DT	GB	LR	DT	GB	LR	DT
NOS		2	23	92	1	0	50	94	1	36
Top DR in %	S	33	50	**100**	0	50	40	0	1	1
	NS	67	17	**100**	**100**	0	60	**94**	0	50

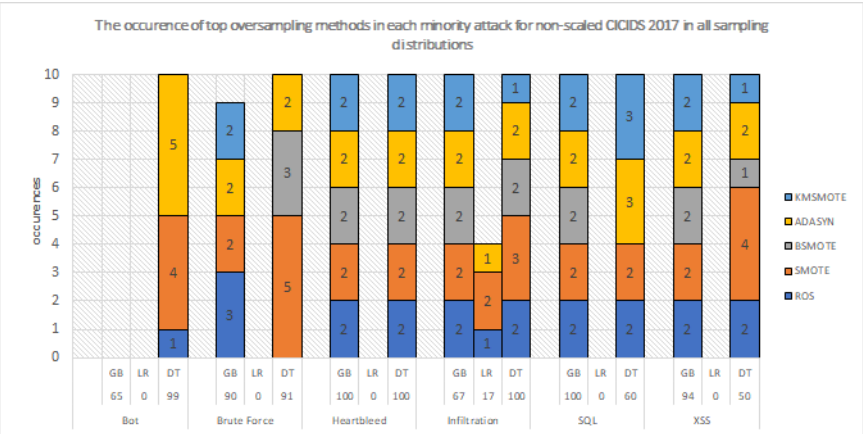
Table 4.10 extracted the results of the highest DR for all machine learning of each minority class in CICIDS 2017 from Appendix O. Six attacks were selected to

oversample: Bot, Brute Force, Heartbleed, Infiltration, SQL injection and XSS. The full detection rate for each sampling iteration were documented in Appendix M & N. Non-scaled experiment observations can obtain a higher increase in DR for most minority attacks, except XSS. For XSS, the oversampling methods can only increase its detection rate in DT from 36 % to 50 %, yet no higher increment than the highest non-oversampled DR (94 %). Referring to Appendix P, confusion metrics for the top DR models, all oversampling methods are less effective in distinguishing between Bot with Normal and Port Scan traffic.

Furthermore, Heartbleed is predicted well in almost all machine learning algorithms and distributions, achieving 100 % in DT. Infiltration also obtained a 100 % DR in DT, with a slight increase in DR in GB and LR. Subsequently, only GB has a good performance for SQL injection in scaled data to obtain a 100 % DR in GB. After all, it shows that the oversampling methods can increase the DR for most minority attack classes, particularly in the non-scaled experiment. Nonetheless, no one oversampling method performs well across all machine learnings.



(a)



(b)

Figure 4.15: The Occurrence of Top Oversampling Methods for each Minority lass in CICIDS 2017, Regardless of the Sampling Distributions.

(a) Scaled Experiment (b) Non-scaled Experiment

Figure 4.15 depicts the ten occurrences of top oversampling methods for each minority class in all CICIDS 2017 sampling distributions. From the figure above, scaled data have a significantly higher occurrence than non-scaled data (133 vs 51). Moreover, most minority classes have at least two machine learning algorithms having an increase in DR, except Bot. These findings indicate that the oversampling methods performed better with scaled data. Subsequently, few minority classes, such as GB & DT in Heartbleed, GB in SQL, and GB in XSS, have an equal occurrence. It illustrates that all oversampling methods have comparable performance, and all contribute to an increase in DR. Furthermore, SMOTE can persist a high presence in all minority classes in both scaled and non-scaled data, 48 counts. All in all, SMOTE has the most occurrences of top oversampling methods in this data set.

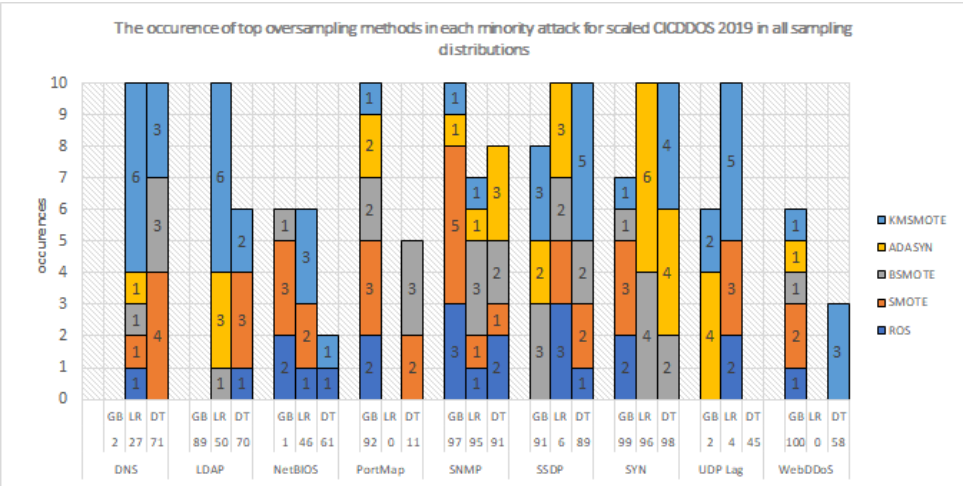
4.2.1.4 CICDDOS 2019

Table 4.11: The Highest Detection ate for all Machine Learning Algorithms of each Minority Class in both Scaled and Non-scaled CICDDOS 2019.

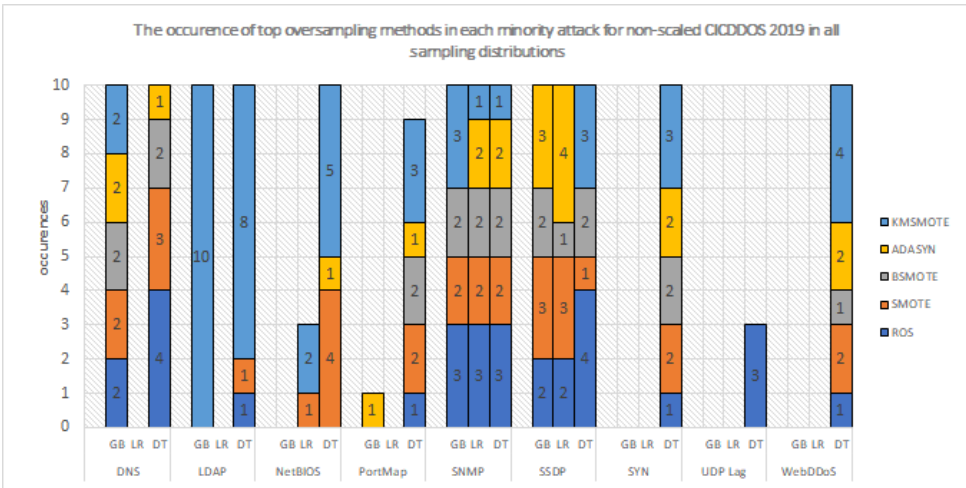
*(Notes : **GB** indicates Gaussian Bayes algorithms, **LR** indicates Logistic Regression Algorithm, **DT** indicates Decision Tree, **DR** indicates detection rates, **S** indicates scaled data, **NS** indicates non scaled data, the bold & italic number indicate the highest detection rate in each minority attack among all observations in the data set, red numbers are the lowest detection rate in each minority class.)*

		DNS			LDAP			NetBIOS		
		GB	LR	DT	GB	LR	DT	GB	LR	DT
NOS		0	6	38	93	1	58	0	6	57
Top DR in %	S	2	27	**71**	**89**	50	70	1	46	**61**
	NS	1	0	52	56	0	54	0	6	58
		Port Map			SNMP			SSDP		
		GB	LR	DT	GB	LR	DT	GB	LR	DT
NOS		21	0	0	5	87	78	2	0	1
Top DR in %	S	**92**	0	11	97	95	91	**91**	6	89
	NS	36	0	11	57	**99**	89	3	3	36
		SYN			UDP Lag			WebDDoS		
		GB	LR	DT	GB	LR	DT	GB	LR	DT
NOS		67	62	90	0	1	53	86	0	14
Top DR in %	S	**99**	96	98	2	4	45	**100**	0	58
	NS	0	59	**99**	0	0	**62**	0	0	58

Table 4.10 extracted the results of the highest DR for all machine learning of each minority class in CICDDOS 2019 from Appendix S. Nine attacks were selected to oversample: DNS, LDAP, NetBIOS, Port Map, SNMP, SSDP, SYN, UDP Lag and WebDDoS. The full detection rate for each sampling iteration was documented in Appendix Q & R. For LDAP, the oversampling methods had increased the DR for LR and DT from 1 % to 50 % and 58 % to 70 %, respectively while maintaining a satisfactory DR in GB (89 %). In scaled data set, SYN and SNMP can maintain a high DR of 90 % and above in all machine learning algorithms, indicating that the oversampling method is effective in increasing the DR. Lastly, UDP Lag has the least DR increment and only DT in scaled data set can achieve 63 %, which is 10 % higher than the non-oversampled DR. Alternative pre-processing steps with oversampling may be required to increase its detection rate.



(a)



(b)

Figure 4.16: The Occurrence of Top Oversampling Methods for each Minority Class in CICDDOS 2019, Regardless of the Sampling Distributions.

(a) Scaled Experiment (b) Non-scaled Experiment

Figure 4.16 depicts the ten occurrences of top oversampling methods for each minority class in all CICDDOS 2019 sampling distributions. In this data set, SNMP and SSDP can achieve high and stable occurrences of top oversampling methods in both data sets, with all machine learning having at least five occurrences. Following that, the KMSMOTE algorithm outperformed for LDAP detection in non-scaled data, contributing 18 out of 20 occurrences. It indicates that KMSMOTE is a better performer in terms of LDAP detection. In the scaled data, DT generally performed well in increasing the DR for all minority classes, with 82 occurrences out of 146. Lastly, KMSMOTE and SMOTE had maintained a high and consistent occurrence in all top observations, with 93 and 67 occurrences, respectively.

4.2.1.5 Multiclass Performance Summary

Table 4.12: Summarise Findings on Multiclass Oversampling Evaluation.

No	Findings
1	Data pre-processing have impact on detection rate.
2	Some oversampling method is very effective in increasing the DR for certain minority classes.
3	No one oversampling method can effectively increase the detection rate of all minority classes in all machine learning algorithms within a data set.

Table 4.12 summarises the findings in this section, multiclass oversampling performance. To summarise, three findings were found in this section. Firstly, different data pre-processing processes contribute differently to the increase in detection rates. For instance, CICIDS 2017 has a significantly higher number of occurrences for the top oversampling methods, with 133 vs 51 for non-scaled data. Such findings indicate that the scaled data set is more consistent in increasing the minority attacks' detection rate.

Additionally, some oversampling methods are more effective than others in increasing the detection rate of specific attack classes. For example, KMSMOTE

dominated all occurrences (10 counts) of top oversampling methods in NSL KDD, Probe with LR algorithm, and CICDDOS 2019, LDAP with GB algorithm. According to Appendix H, the ADASYN contributed the top three highest DR increments for DoS in both scaled and non-scaled UNSW-NB15. Thus, ADASYN is more effective at increasing the DR for DoS. All in all, no single oversampling method outperforms the others, since no one model can improve all minority detection rates in a model. Next, the overall oversampling performance will be evaluated in the following section to further the investigations.

4.2.2 Overall Performance Evaluation

4.2.2.1 UNSW-NB15

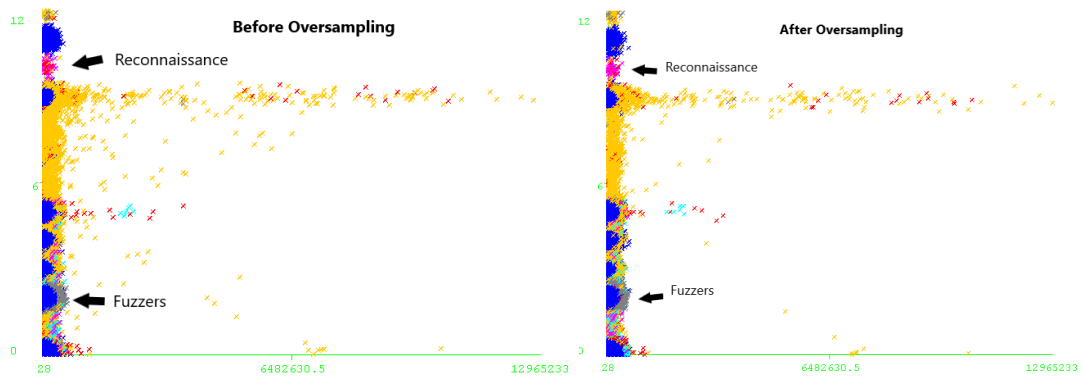
Table 4.13: Top 3 Overall Attack DR for each Oversampling Percentage in UNSW-NB15.

*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage excluding normal traffics, **F** indicates **F measures** in percentage, the bold numbers indicate the highest rate in non-oversampled observation, **green** highlights represent the highest DR in the experiment, **yellow** highlights represents the top 10 DR for each experiment.)*

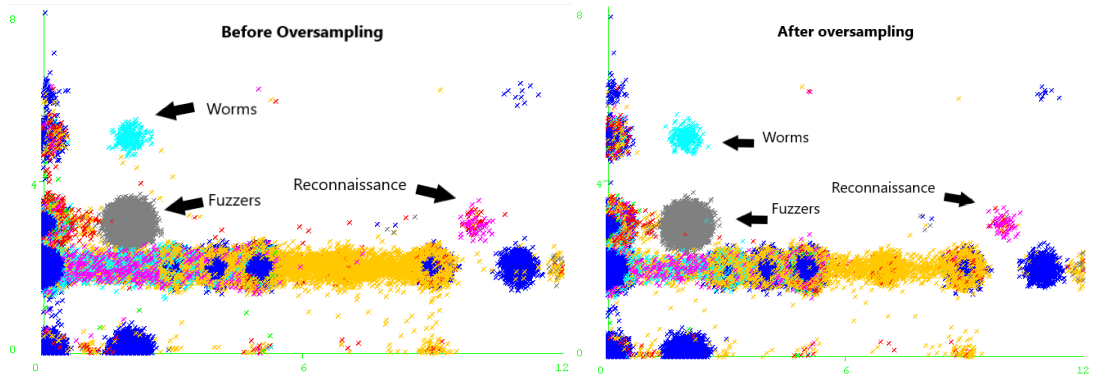
Scaled										*Non-scaled*									
No	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*
Original																			
1	GB	**36.85**	23.25	20.88	-----														
2	LR	31.08	26.66	26.34															
3	DT	8.19	5.40	4.30															
10% oversampled					***60% oversampled***					***10% oversampled***					***60% oversampled***				
1	GB – KMSMOTE	34.17	24.13	27.25	*1*	GB – ROS	30.90	22.67	22.89	*1*	DT – KMSMOTE	43.00	37.59	38.46	*1*	DT – BSMOTE	40.85	35.56	37.17
2	LR - BSMOTE	30.67	26.77	26.39	*2*	LR – ROS	29.85	27.15	25.86	*2*	DT – ADASYN	42.86	38.86	38.31	*2*	DT – ADASYN	40.69	36.02	37.49
3	LR - ROS	30.41	29.66	26.03	*3*	LR – SMOTE	29.46	27.15	24.90	*3*	DT - SMOTE	41.94	37.96	38.73	*3*	DT - KMSMOTE	40.60	37.70	38.91
20% oversampled					***70% oversampled***					***20% oversampled***					***70% oversampled***				
1	LR – SMOTE	30.80	27.04	26.80	*1*	GB - ROS	31.62	23.66	24.00	*1*	DT - ROS	42.46	38.67	38.38	*1*	DT – BSMOTE	40.94	38.56	39.37
2	LR – ROS	30.75	28.40	26.86	*2*	GB - BSMOTE	30.68	22.04	23.76	*2*	DT – ADASYN	41.64	37.76	39.07	*2*	DT – ADASYN	40.78	38.74	39.41
3	LR - BSMOTE	30.42	29.59	26.70	*3*	LR - SMOTE	29.80	25.51	25.51	*3*	DT - KMSMOTE	41.46	37.98	39.39	*3*	DT - SMOTE	40.55	38.13	39.03
30% oversampled					***80% oversampled***					***30% oversampled***					***80% oversampled***				
1	LR - KMSMOTE	30.46	29.47	27.73	*1*	GB - SMOTE	32.54	24.07	24.99	*1*	DT - SMOTE	42.64	38.13	38.37	*1*	DT – BSMOTE	40.91	38.20	38.86
2	LR - ROS	30.37	27.29	26.52	*2*	GB – ROS	31.89	24.52	24.55	*2*	DT – ADASYN	42.61	37.60	37.36	*2*	DT – SMOTE	40.89	35.61	36.58
3	LR - BSMOTE	30.23	27.78	26.63	*3*	LR - ROS	29.93	27.96	25.16	*3*	DT – KMSMOTE	41.20	38.62	39.67	*3*	DT – KMSMOTE	39.93	38.94	39.13
40% oversampled					***90% oversampled***					***40% oversampled***					***90% oversampled***				
1	LR - KMSMOTE	30.15	29.09	27.09	*1*	GB – SMOTE	32.70	25.96	25.50	*1*	DT – SMOTE	42.14	36.76	37.88	*1*	DT – SMOTE	41.19	38.00	39.07
2	GB - ROS	30.07	21.83	22.25	*2*	GB – ROS	32.12	24.23	25.45	*2*	DT – BSMOTE	41.63	38.35	39.24	*2*	DT – ADASYN	40.91	38.11	38.92
3	LR - ROS	29.93	27.99	25.53	*3*	GB – ADASYN	32.06	20.57	23.78	*3*	DT – ROS	40.95	38.50	39.24	*3*	DT – BSMOTE	40.61	37.57	38.80
50% oversampled					***100% oversampled***					***50% oversampled***					***100% oversampled***				
1	GB-KMSMOTE	30.72	22.50	22.93	*1*	GB – SMOTE	32.58	25.06	26.11	*1*	DT – ADASYN	41.96	36.11	37.12	*1*	DT – SMOTE	41.55	39.27	39.51
2	LR - ROS	29.76	30.72	25.82	*2*	GB – ROS	32.46	24.55	25.64	*2*	DT – KMSMOTE	41.07	38.68	39.37	*2*	DT – ADASYN	40.82	37.68	38.63
3	LR – KMSMOTE	29.68	30.73	25.82	*3*	GB - BSMOTE	31.64	24.86	26.35	*3*	DT – ROS	40.46	38.03	38.65	*3*	DT - BSMOTE	40.78	38.61	39.01

Table 4.13 tabulates the top three overall DR metrics for each oversampling percentage in UNSW-NB15, extracted from Appendix U and V. The highest overall detection rate for scaled oversampling had decreased by 7.27 % compared to the highest origin (36.85 % to 34.17 %). The precision rate for its most performing model also decreases by 9.50 %, from 26.66 % to 24.13 %. Non-scaled oversampling performs better than the scaled oversampling, as the DR for all percentages is higher than the highest origin, 36.85 %.

The DT-KMSMOTE-10 % in the non-scaled experiment was able to gain a 16.69 % increase in detection rate from 36.85 % to 43%. It is the highest attack DR in both scaled and non-scaled data sets. The model also performed best in the non-scaled experiment, increasing the F measure to 48.90 %. However, KMSMOTE occurrence is only one-tenth of the top 10 observations, which are highlighted in yellow. While ADASYN contributes to almost half of the top 10 occurrences, indicating that its performance is more consistent for overall attack detection. To conclude, non-scaled data outperforms scaled data because all of the top detection rates are higher than the non-oversample DR.



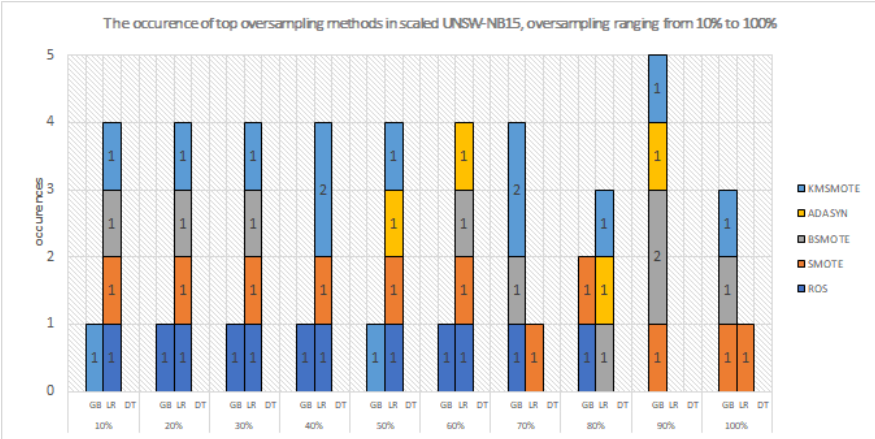
(a)



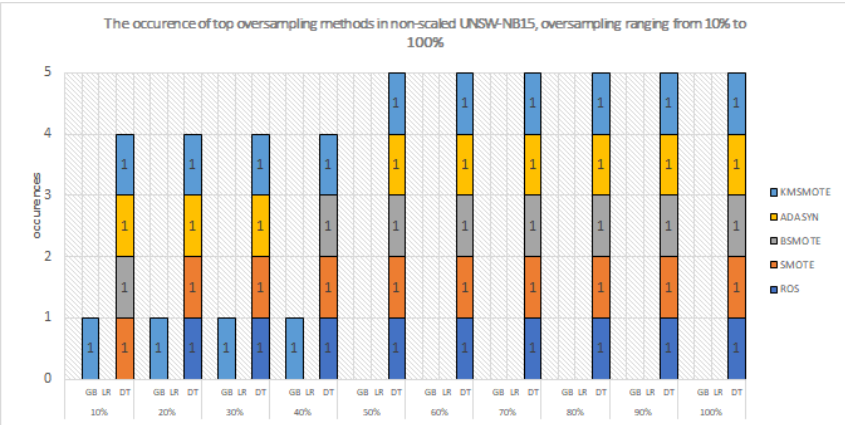
(b)

Figure 4.17: Two Groups of Matrix Plots are shown for the UNSW-NB15: the Sample Before Oversampling on the Left and After Oversampling on the Right. (a) “Sbytes” and “service” Features (b) “service” and “state” Features.

To compare the non-oversampled matrix, the best overall DR performing oversampled method for this data set was chosen, which is KMSMOTE-10 %. From Figure 4.17 (a) and (b), the boundaries for Reconnaissance (purple plot) and Fuzzers (grey plot) are overwhelmed by others. It helps to reduce the overlapping classes in the two minority classes after implementing oversampling, revealing more possible decision boundaries for Reconnaissance and Fuzzers. Moreover, the outliers for Exploits (yellow plot) that were located in between the Worms (light blue plot) and Fuzzers (grey plot) have been removed after oversampling. It helps in defining the decision boundaries for worms, as there are no other classes nearby that could cause confusion. Due to time constraints, only two matrix plots were chosen to visualise, assuming that the some of the matrix plots are similar to these conditions.



(a)



(b)

Figure 4.18: The Occurrence of Top the Five Oversampling Methods in UNSW NB15, Oversampling Ranging from 10% to 100%.

(a) Scaled (b) on-scaled

Figure 4.18 shows the occurrence of the top five oversampling methods in each oversampling percentage for both scaled and non-scaled data sets. As depicted, DT generally performs well in the scaled data set, as it has the highest occurrences among the three machine learning algorithms, especially when the oversample percentage is 50 % and above. In UNSW-NB15, KMSMOTE had the highest total occurrences for scaled and non-scaled data sets, with 14 and 13 counts, respectively. Overall, KMSMOTE is the best and most reliable performer for increasing the overall attack DR in both data sets.

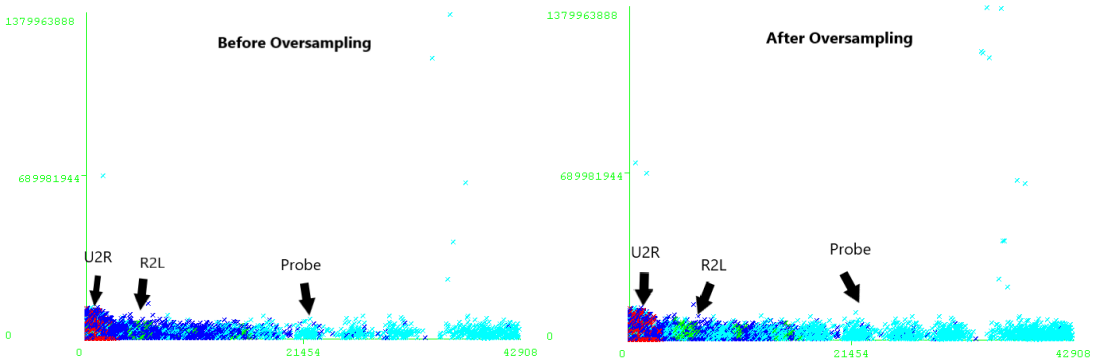
4.2.2.2 NSL KDD

Table 4.14: Top 3 Overall Attack DR for each Oversampling Percentage in NSL KDD.

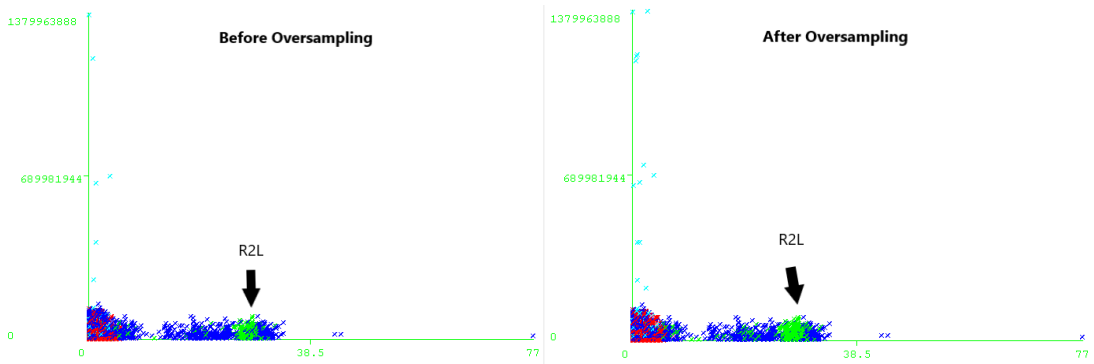
*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage excluding normal traffics, **F** indicates **F measures** in percentage, the bold numbers indicate the highest rate in non-oversampled observation, **green** highlights represent the highest DR in the experiment, **yellow** highlights represents the top 10 DR for each experiment.)*

Scaled										*Non-scaled*									
No	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*
Original																			
1	GB	**32.52**	45.12	34.47	-----														
2	LR	30.37	43.27	36.97															
3	DT	28.67	34.98	29.53															
10% oversampled					***60% oversampled***					***10% oversampled***					***60% oversampled***				
1	GB - BSMOTE	38.28	45.12	38.32	1	GB – ROS	40.06	42.89	38.43	*1*	DT – BSMOTE	34.16	50.77	37.91	*1*	GB - KMSMOTE	38.70	31.26	29.11
2	LR – ADASYN	38.19	41.66	37.83	2	GB – SMOTE	39.60	42.46	38.20	*2*	DT – ADASYN	33.28	50.16	36.92	*2*	DT – ROS	36.13	52.63	40.14
3	GB - ADASYN	38.06	45.67	39.18	3	GB – ADASYN	39.60	43.79	39.31	*3*	DT - SMOTE	33.27	50.15	36.30	*3*	DT - ADASYN	35.31	44.60	38.21
20% oversampled					***70% oversampled***					***20% oversampled***					***70% oversampled***				
1	GB-ADASYN	39.31	45.32	39.84	1	GB – ROS	40.20	42.28	38.66	*1*	GB – SMOTE	33.95	37.18	34.16	*1*	DT – ADASYN	35.40	51.31	40.10
2	GB – BSMOTE	38.76	44.11	38.46	2	GB – SMOTE	40.12	42.05	38.08	*2*	DT – KMSMOTE	33.54	52.47	37.78	*2*	DT – SMOTE	35.07	51.02	38.30
3	GB - SMOTE	38.74	44.71	38.46	3	GB – ADASYN	39.16	42.67	38.11	*3*	DT - SMOTE	33.19	51.87	37.38	*3*	GB – KMSMOTE	35.02	42.39	36.49
30% oversampled					***80% oversampled***					***30% oversampled***					***80% oversampled***				
1	GB – ADASYN	39.76	44.84	39.94	1	GB – SMOTE	40.19	40.45	37.03	*1*	DT - ROS	35.35	48.79	37.76	*1*	DT- ROS	35.37	50.95	38.50
2	GB – SMOTE	39.66	44.50	39.34	2	GB – ADASYN	40.03	43.41	39.50	*2*	GB - KMSMOTE	34.15	40.61	35.20	*2*	DT – KMSMOTE	35.33	52.84	39.94
3	GB – BSMOTE	39.51	44.41	39.72	3	GB – ROS	39.92	33.20	33.60	*3*	DT – ADASYN	33.90	46.89	38.49	*3*	GB – ROS	34.80	42.79	36.74
40% oversampled					***90% oversampled***					***40% oversampled***					***90% oversampled***				
1	GB – SMOTE	39.84	43.90	39.09	1	GB – ROS	40.50	39.83	37.00	*1*	DT – SMOTE	35.91	52.55	39.17	*1*	DT – ROS	35.99	52.47	40.06
2	GB – ADASYN	39.15	44.07	39.27	2	GB – SMOTE	39.71	39.90	36.38	*2*	DT – ADASYN	35.82	50.40	39.95	*2*	DT – KMSMOTE	35.72	53.25	40.36
3	GB - ROS	39.04	43.73	38.41	3	LR – ROS	39.70	30.85	32.92	*3*	GB – KMSMOTE	34.81	39.69	34.62	*3*	GB - KMSMOTE	34.81	41.7	35.65
50% oversampled					***100% oversampled***					***50% oversampled***					***100% oversampled***				
1	GB – BSMOTE	39.36	43.65	38.58	1	GB – ROS	40.66	39.81	37.20	*1*	GB – ROS	34.93	39.80	35.26	*1*	GB – KMSMOTE	34.99	41.56	36.12
2	GB – SMOTE	39.30	43.24	38.43	2	GB – SMOTE	40.42	39.47	36.36	*2*	GB – SMOTE	34.45	39.61	34.76	*2*	DT - ROS	34.84	49.64	39.12
3	GB - ROS	39.16	43.20	37.72	3	GB – BSMOTE	39.73	41.48	37.85	*3*	DT- ADASYN	34.27	49.06	37.90	*3*	DT – SMOTE	32.84	51.66	37.02

Table 4.14 tabulates the top 10 average performances among all observations in NSL KDD, extracted from Appendix W and X. In this data set, both scaled and non-scaled oversampling maximised the detection rate from 32.52 % to 38.70 % and 40.66 %, with a 19 % and 25.03 % increase, respectively. Contrary to UNSW-NB15, all of the top observations in both experiments had gained a greater detection rate compared to non-oversampled. A combination of GB with ADASYN 80 % in scaled oversampling obtained the highest F measure, 39.50 %. While ROS is the highest performing method in this data set because it not only has the highest overall DR (40.66 %) but also consistently contributes to the top 10 highlighted observations (5/10).



(a)

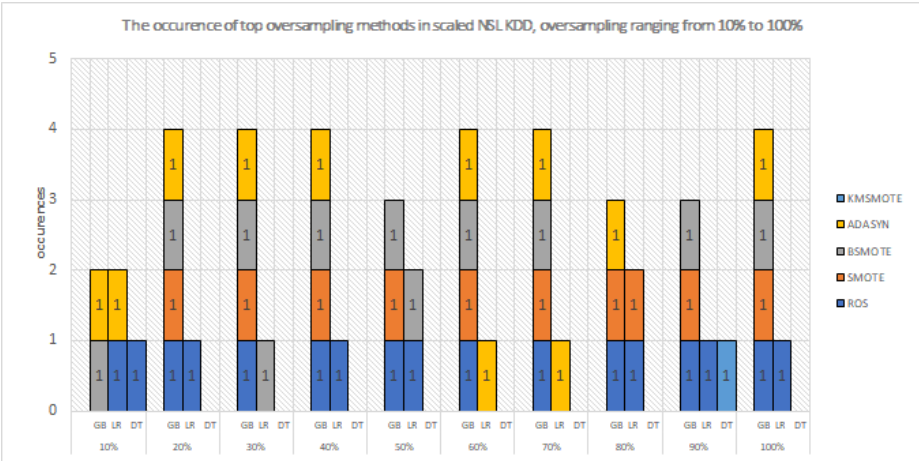


(b)

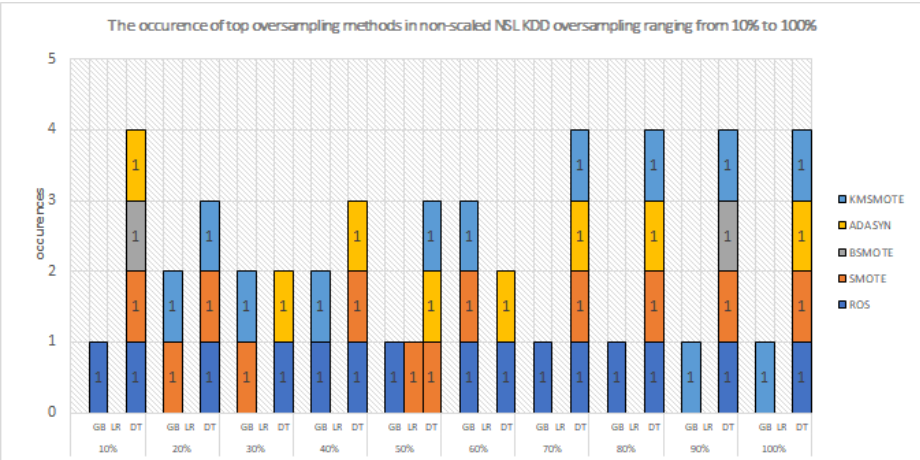
Figure 4.19: Two Groups of Matrix Plots are shown for the NSL KDD: the Sample Before Oversampling on the Left and After Oversampling on the Right.

(a) “Duration” and “srcbytes” Features (b) “hot” and “srcbytes” Features.

In this data set, only two matrix plots were chosen to visualise. The best overall DR performing oversampled method for in data set was chosen for comparison, ROS-100 %. From Figure 4.19 (a) and (b), the oversampling method produced a better possible boundary for U2R (red plot) and R2L (green plot) in both matrix plot groups, especially R2L (green plot) in Figure (b). It explained the detection rate for each minority class. In which the R2L achieved a higher detection rate compared to U2R in ROS-100 % with 71 %, while U2R only obtained 49 %. It is because U2R (red plot) are still being overwhelmed by others, particularly Normal traffic (blue plot). To summarise, this oversampling produced a better possible decision boundary for certain classes, yet the overlapping issue remained.



(a)



(b)

Figure 4.20: The Occurrence of the Top Five Oversampling Methods in NSL KDD, Oversampling Ranging from 10% to 100%.

(a) Scaled (b) Non-scaled

Figure 4.20 shows the occurrence of the top five oversampling methods in each oversampling percentage for both scaled and non-scaled NSL KDD. In these data sets, ROS and ADASYN have a higher occurrence in the scaled data set, whereas ROS and SMOTE have a higher occurrence in the non-scaled data set. Borderline SMOTE has more occurrences in the scaled data compared to the non-scaled data, indicating it can perform better with scaled samples. Overall, ROS is the best and most reliable performer for increasing the overall attack DR in both NSL KDD, with the highest DR 40.66 % and occurrences in 32 counts.

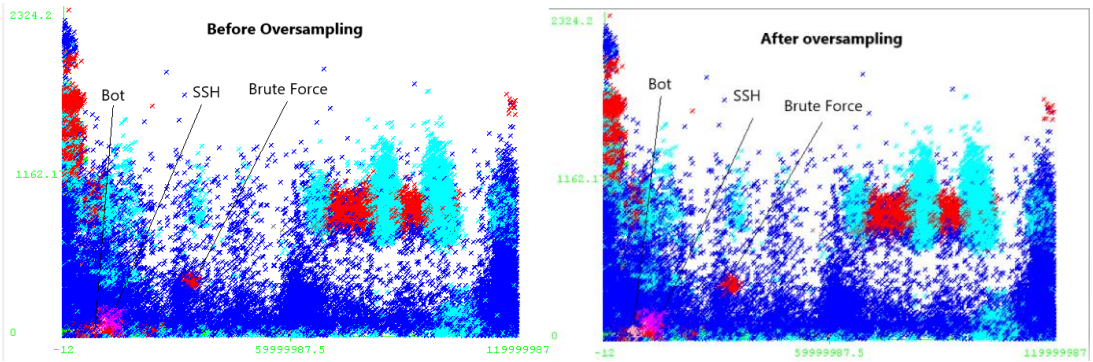
4.2.2.3 CICIDS 2017

Table 4.15: Top 3 Overall Attack DR for each Oversampling Percentage in CICIDS 2017.

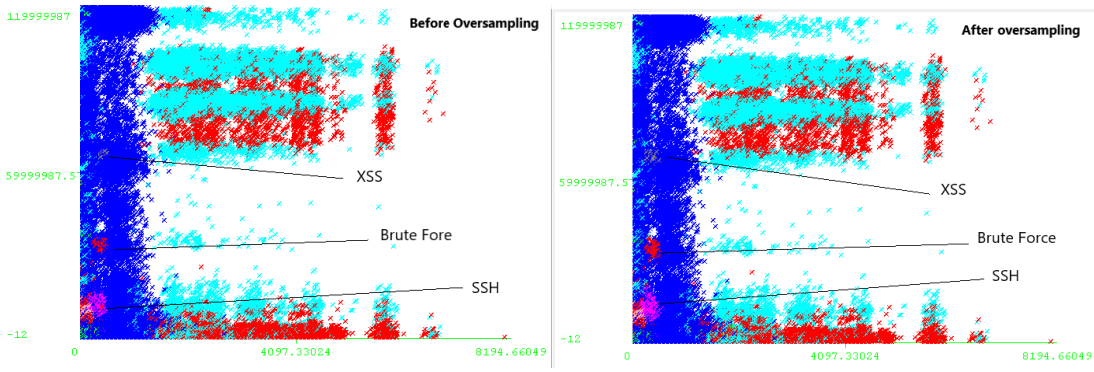
*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage excluding normal traffics, **F** indicates **F measures** in percentage, the bold numbers indicate the highest rate in non-oversampled observation, **green** highlights represent the highest DR in the experiment, **yellow** highlights represents the top 10 DR for each experiment.)*

Scaled										*Non-scaled*									
No	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*
Original																			
1	GB	30.39	34.70	35.55	-----														
2	LR	32.50	37.11	37.10															
3	DT	**36.70**	36.70	36.70															
10% oversampled					**60% oversampled**					**10% oversampled**					**60% oversampled**				
1	LR - ROS	37.90	36.11	37.26	1	LR – KMSMOTE	37.87	36.24	37.29	1	DT - BSMOTE	39.72	39.72	39.72	1	DT – SMOTE	39.71	39.72	39.71
2	LR – KMSMOTE	37.78	36.05	37.17	2	LR – ROS	37.73	36.02	37.13	2	DT- SMOTE	39.71	39.68	39.69	2	DT – ROS	39.70	39.70	39.70
3	LR - SMOTE	37.31	36.15	.36.69	3	LR – SMOTE	37.50	36.34	36.86	3	DT – ROS	39.71	39.71	39.71	3	DT – KMSMOTE	39.69	39.70	39.70
20% oversampled					***70% oversampled***					***20% oversampled***					***70% oversampled***				
1	LR - SMOTE	37.81	36.07	37.20	1	LR – ROS	37.79	36.12	37.21	1	DT – SMOTE	39.73	36.69	39.70	1	DT – BSMOTE	40.52	39.67	39.67
2	LR - ROS	37.29	36.15	36.65	2	LR – SMOTE	37.79	36..49	37.20	2	DT – ROS	39.71	39.70	39.70	2	DT – ROS	39.73	39.74	39.73
3	LR – BSMOTE	37.22	36.09	36.60	3	LR – ADASYN	36.96	35.92	36.34	3	DT – KMSMOTE	39.71	39.70	39.70	3	DT – SMOTE	39.73	39.74	39.74
30% oversampled					***80% oversampled***					***30% oversampled***					***80% oversampled***				
1	LR – BSMOTE	37.80	36.08	37.20	1	LR – ROS	38.04	36.98	37.67	1	DT – SMOTE	40.52	40.12	40.33	1	DT – SMOTE	41.73	39.71	39.71
2	LR – KMSMOTE	37.79	36.06	37.18	2	LR – ADASYN	37.56	35.38	36.38	2	DT – ROS	39.72	39.72	39.72	2	DT – BSMOTE	39.69	39.68	39.69
3	LR – ROS	37.57	36.39	36.94	3	LR - SMOTE	37.53	36.68	37.02	3	DT – BSMOTE	39.70	39.69	39.70	3	DT – ADASYN	39.69	39.70	39.69
40% oversampled					***90% oversampled***					***40% oversampled***					***90% oversampled***				
1	LR – ROS	37.88	36.17	37.28	1	LR – ROS	38.08	37.23	37.27	1	DT – SMOTE	42.74	39.71	40.21	1	DT – KMSMOTE	39.72	39.73	39.72
2	LR – SMOTE	37.86	36.51	37.27	2	LR – KMSMOTE	37.77	37.34	37.64	2	DT – ADASYN	39.71	39.68	39.68	2	DT – ROS	39.70	39.72	39.72
3	LR – BSMOTE	37.48	36.34	36.84	3	LR - SMOTE	37.57	37.47	37.43	3	DT – KMSMOTE	39.69	39.70	39.69	3	DT – ADASYN	39.69	39.69	39.69
50% oversampled					***100% oversampled***					***50% oversampled***					***100% oversampled***				
1	LR – KMSMOTE	37.86	36.16	37.26	1	LR - ROS	38.69	37.09	37.43	1	DT – ROS	40..68	39.74	39.74	1	DT – ADASYN	39.72	39.68	39.68
2	LR – ROS	37.52	36.36	36.89	2	LR – SMOTE	37.96	37.25	37.22	2	DT – KMSMOTE	39.71	39.72	39.72	2	DT – KMSMOTE	39.71	39.71	39.71
3	LR – SMOTE	37.50	36.66	36.91	3	LR – KMSMOTE	37.74	37.22	37.65	3	DT – SMOTE	39.70	39.71	39.70	3	DT – ROS	39.70	39.71	39.70

Table 4.15 tabulates the top three overall attack DR for all percentages in CICIDS 2017, extracted from Appendix Y and Z. Similar to NSL KDD, both scaled and non-scaled oversampling performed well in increasing the detection rates, as both top observations DR is higher than the non-oversampled DR (42.71 %, 38.69 % vs 36.70 %). Non-scaled oversampling outperforms the scaled oversampling with the highest increment rate, 42.71 %, whereas the scaled DR is 38.69%. Although DT-SMOTE achieves the highest overall detection rate in the non-scaled data set, DT-ROS has the highest overall precision and F measures, 41.25 % and 40.67 %. It suggests that ROS had a better model performance. In short, SMOTE is the best-performing in terms of DR in this data set and non-scaled data obtained a better result.



(a)

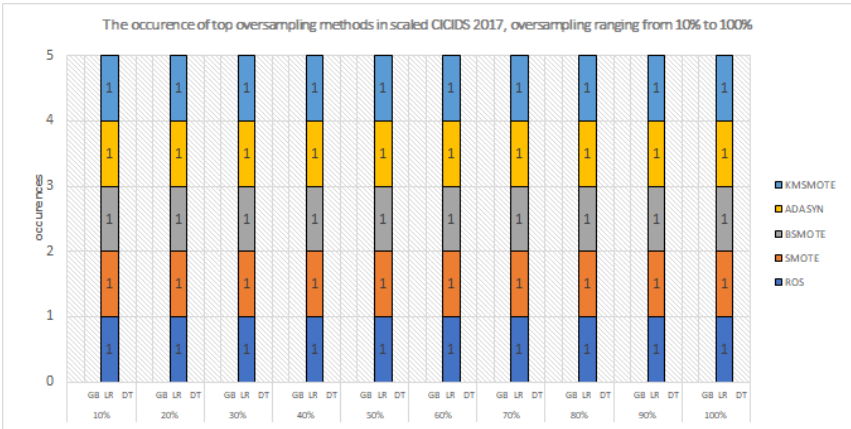


(b)

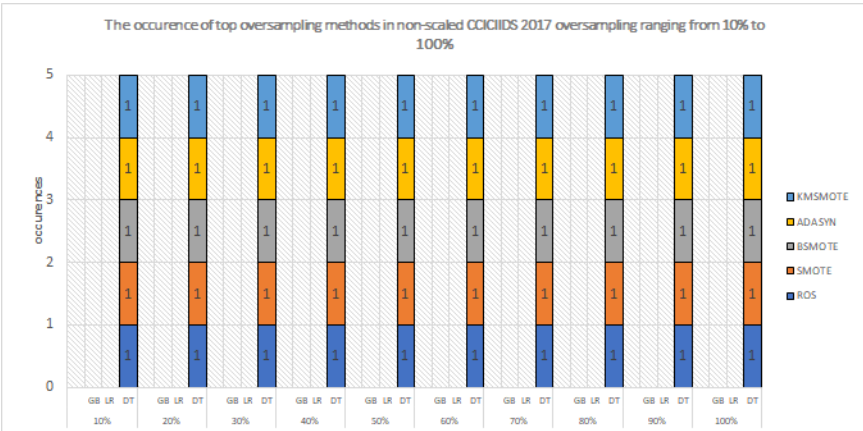
Figure 4.21: Two Groups of Matrix Plots are shown for the CICDS 2017: the Sample Before Oversampling on the Left and After Oversampling on the Right.

(a) “FlowDur” and “AvgPktSize” (b) “BwdPktLen” and “FlowDur”.

In this data set, only two matrix plots were chosen to visualise. DT-SMOTE-40 % were selected for comparison in CICIDS 2017. Figure 4.21 shows two matrix plot groups with a different set of features from the data set. Similarly to other observations, the oversampling method can reflect a better possible boundary for certain attack classes. For example, after oversampling, Brute Force (red plot) and SSH (purple plot) become more visible after oversampling in both (a) and (b). Moreover, the boundaries for Bot (orange plot) become more visible in (a), indicating its possible decision boundary. In contrast to NSL KDD, the overlapping issue in those classes has a lesser impact on their detection rate, as all these three classes can achieve a high detection rate. The detection rate for Brute Force is 91 %, 97 % for Botnet and 99 % for SSH.



(a)



(b)

Figure 4.22: The Occurrence of the Top Five Oversampling Methods in CICIDS 2017, Oversampling Ranging from 10% to 100%.

(a) Scaled (b) Non-scaled

Figure 4.22 shows the occurrence of the top five oversampling methods in each oversampling percentage for both scaled and non-scaled CICIDS 2017. As depicted, most oversampling methods have a comparable performance. The only difference between them is the machine learning algorithm, as each data set has different outperformed machine learning. For example, LR outperformed in scaled data, while DT outperformed in non-scaled data. As aforementioned, a non-scaled data set performs better than scaled data set; thus, DT is the best performing machine learning in this data set.

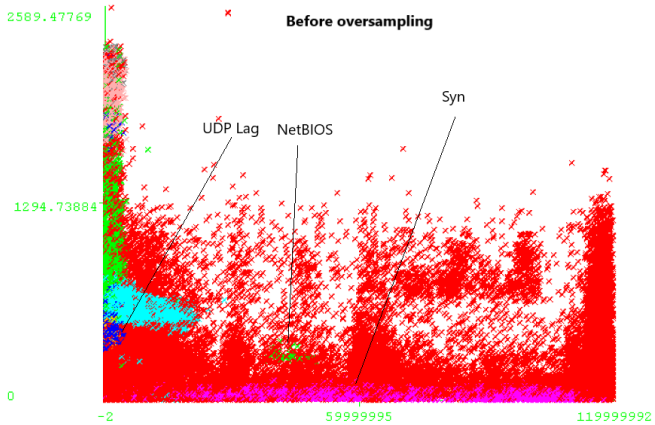
4.2.2.4 CICDDOS 2019

Table 4.16: Top 3 Overall Attack DR for each Oversampling Percentage in CICDDOS 2019.

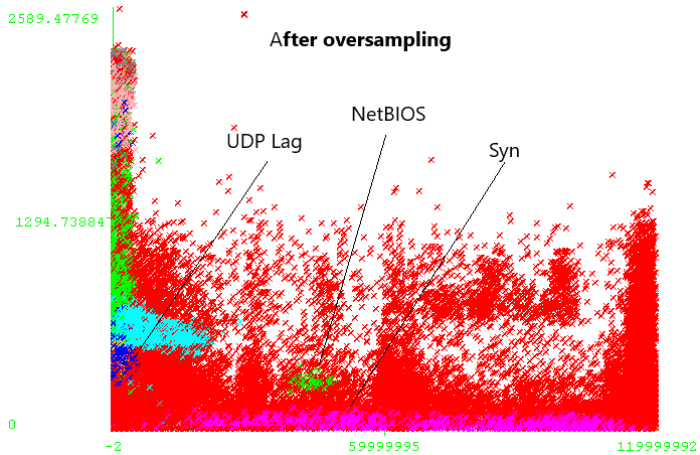
*(Notes : **P** indicates **Precision** in percentage, **DR** indicates **Detection Rate** in percentage excluding normal traffics, **F** indicates **F measures** in percentage, the bold numbers indicate the highest rate in non-oversampled observation, **green** highlights represent the highest DR in the experiment, **yellow** highlights represents the top 10 DR for each experiment.)*

Scaled										*Non-scaled*									
No	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*	*No*	*C*	*DR*	*P*	*F*
Original																			
1	GB	36.47	37.71	36.58	-----														
2	LR	38.72	38.76	37.24															
3	DT	**38.97**	38.72	38.72															
10% oversampled					**60% oversampled**					**10% oversampled**					**60% oversampled**				
1	LR - All methods	40.14	38..14	38..14	1	LR – KMSMOTE	40.16	39.14	39.14	1	DT – SMOTE	43.41	41.06	40.55	1	DT – KMSMOTE	41.61	40.86	41.02
2					2	LR – BSMOTE	40.16	39.14	39.14	2	DT - KMSMOTE	42.04	41.05	40.55	2	DT – SMOTE	41.45	40.79	40.97
3					3	LR – ADASYN	40.16	38.14	38.14	3	DT- ROS	42.01	41.08	40.67	3	DT – BSMOTE	41.44	40.63	40.81
20% oversampled					***70% oversampled***					***70% oversampled***					***70% oversampled***				
1	LR – KMSMOTE	40.20	38..14	42.85	1	LR - All methods	40.14	38..14	38..14	1	DT – KMSMOTE	42.02	40.95	40.62	1	DT – SMOTE	41.46	40.82	40.99
2	LR - ADASYN	40.16	38..14	42.85	2					2	DT – SMOTE	41.99	41.08	40.65	2	DT – ADASYN	41.45	40.83	40.98
3	LR – other methods	40.14	38..14	42.85	3					3	DT – ROS	41.98	40.74	40.65	3	DT - BSMOTE	41.44	40.77	40.93
30% oversampled					***80% oversampled***					***80% oversampled***					***80% oversampled***				
1	LR – SMOTE	40.16	39.14	38.14	1	LR - All methods	40.14	38..14	38..14	1	DT – ADASYN	42.02	41.21	40.57	1	DT – ADASYN	41.43	40.77	40.94
2	LR –BSMOTE	40.15	39.14	38.14	2					2	DT – SMOTE	42.01	40.76	40.67	2	DT – SMOTE	41.40	40.77	40.95
3	LR – ROS	40.15	39.14	38.14	3					3	DT – KMSMOTE	42.00	40.95	40.51	3	DT – BSMOTE	41.32	40.74	40.93
40% oversampled					***90% oversampled***					***90% oversampled***					***90% oversampled***				
1	LR - All methods	40.14	38..14	38..14	1	LR - All methods	40.14	38..14	38..14	1	DT – KMSMOTE	42.01	40.84	40.93	1	DT – SMOTE	41.31	40.83	40.98
2					2					2	DT – SMOTE	41.75	40.94	41.04	2	DT – ADASYN	41.19	40.82	40.94
3					3					3	DT – BSMOTE	41.52	40.64	40.84	3	DT – BSMOTE	40.71	40.63	40.65
50% oversampled					***1000% oversampled***					***100% oversampled***					***100% oversampled***				
1	LR - SMOTE	40.18	39.14	39.14	1	LR - All methods	40.14	38..14	38..14	1	DT – BSMOTE	41.68	40.83	40.94	1	DT – ADASYN	41.26	40.69	40.87
2	LR - KMSMOTE	40.16	38.14	38.14	2					2	DT – SMOTE	41.67	40.86	41.00	2	DT - KMSMOTE	40.78	40.68	40.71
3	LR - other methods	40.14	38..14	38..14	3					3	DT – KMSMOTE	41.66	40.85	40.96	3	DT – SMOTE	40.59	40.60	40.58

Table 4.16 tabulates the top three overall DR in all percentages for CICDDOS 2019, extracted from Appendix AA and BB. In this data set, scaled data oversampling had acquired an identical detection rate of 40.14 % from 70 % to 100 % sampling distribution. It might be because the data become similar after scaling in those percentages, leading all oversampling methods unable to distinguish them further. However, scaled oversampling is still at least a 3% change from 38.97 % to 40.14 %. On the other hand, DT-SMOTE with 10% non-scaled oversampling performed the best, with a 43.41 % of detection rate. Yet, ROS achieves a higher F measure and precision, 41.25 % and 40.67 %, respectively. While KMSMOTE has the highest occurrence (2/5) among the top ten observations. SMOTE only accounted for one-fifth of the top 10 observations, indicating that it may not be the most reliable oversampling method in this data set. To conclude, the non-scaled data set had a better DR increment compared to scaled data, 43.41 % vs 40.20 %.



(a)

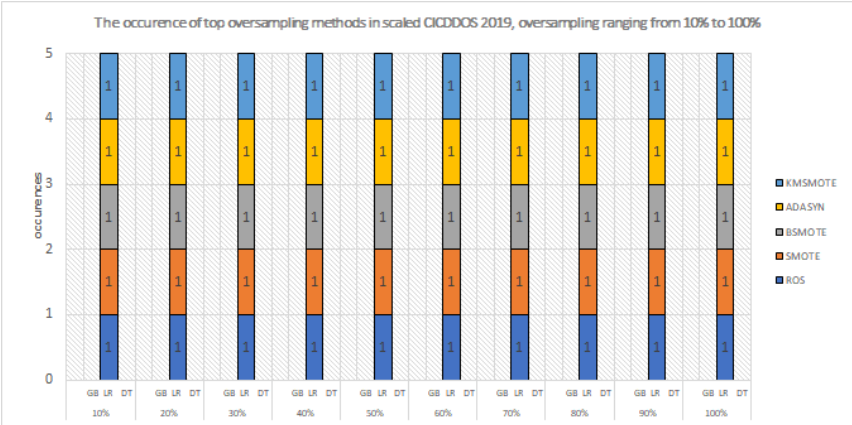


(b)

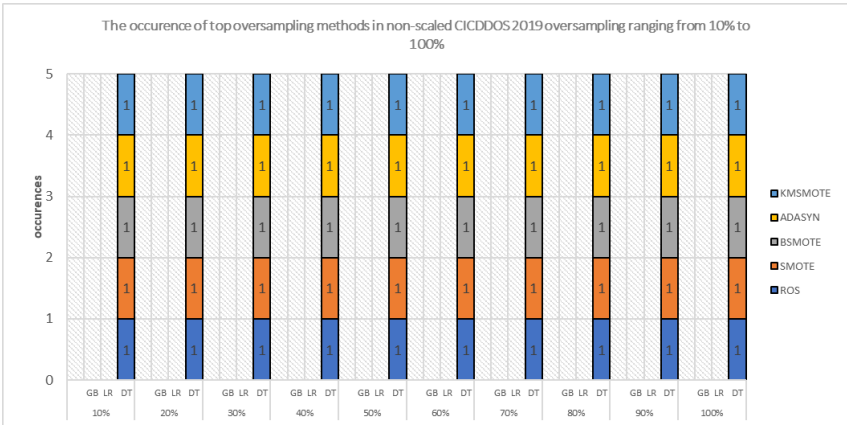
Figure 4.23: The Matrix Plot Shows the Feature Spaces for Flow Duration and Average Packet Size for CICDDOS 2019.

(a) Before Oversampling (b) After Oversampling

In CICDDOS 2019, DT-SMOTE-10 % was chosen for comparison. Figure 4.23 depicts the matrix plot with two features: Flow Duration and Average Packet Size. Similarly to other observations, the oversampling method can reflect a better possible boundary for certain attack classes. For example, the UDP Lag (dark green plot), NetBIOS (sea blue plot) and SYN (purple plot) became more visible after oversampling. Similarly to NSL KDD, the overlapping issue that persists in this data set has an impact on the detection rate even after oversampling. Despite, UDP Lag (dark green plot) and NetBIOS (sea blue plot) had improved their decision boundaries after oversampling, their detection rate remains low compared to other minority classes. UDP Lag achieved 46 % DR, while NetBIOS achieved 54 % DR.



(a)



(b)

Figure 4.24: The Occurrence of Top Five Oversampling Methods in CICDDOS 2019, Oversampling Ranging from 10% to 100%.

(a) Scaled (b) Non-scaled

Figure 4.24 shows the occurrence of the top five oversampling methods in each oversampling percentage for both scaled and non-scaled CICDDOS 2019. Similarly to CICIDS 2017, the only difference between the performance of oversampling methods is the machine learning algorithm, as each data set has different outperformed machine learning. For example, LR outperformed in scaled data, while DT outperformed in the non-scaled data. As aforementioned, a non-scaled data set performs better than scaled data set; thus, DT is the best performing machine learning in this data set.

4.2.2.5 Overall Performance Summary

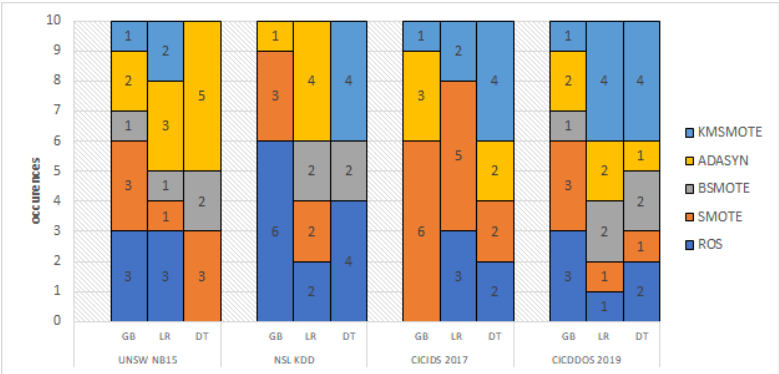


Figure 4.25: The Occurrence of Top Oversampling Methods in each Scaled Data Set, Oversampling Range from 10% to 100%

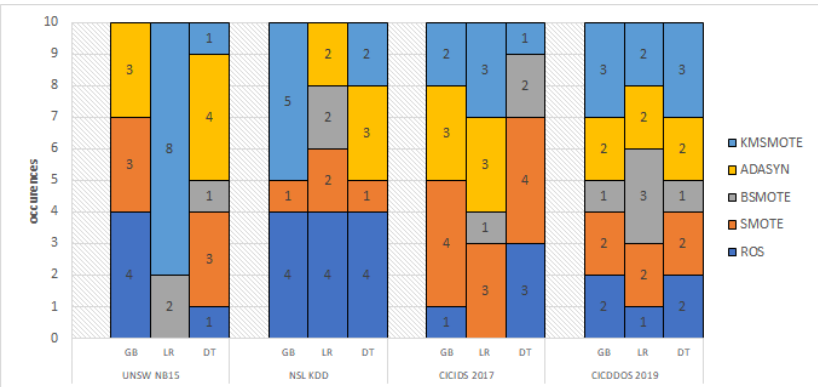


Figure 4.26: The Occurrence of Top Oversampling Methods in each Non-scaled Data Set, Oversampling Range from 10% to 100%

Figure 4.25 and Figure 4.26 charted the occurrences of top oversampling methods in each scaled and non-scaled data set, with an oversampling ranging from 10 % to 100 %. Figure 4.26 depicts that KMSMOTE is more reliable in increasing the UNSW-NB overall attack DR in LR algorithm. Furthermore, some algorithms have a highly consistency occurrences in both scale and non-scaled data sets. For example, ROS in NSL KDD and SMOTE in CICIDS 2017, both with ¼ occurrences (24/60). To conclude, the most consistent oversampling methods in each data set are ADAYSN in UNSW NB15 with 17/60 occurrences, ROS in NSL KDD with 24/60 occurrences, SMOTE in CICIDS 2017 with 24/60 occurrences, and KMSMOTE in CICDDOS 2019 with 17/60 occurrences. Lastly, SMOTE improves the overall attack DR well in the scaled data, with the highest occurrences of 30 times. On the other hand, KMSMOTE improves the overall attack DR well in the non-scaled data, with the highest occurrences 30 times.

Table 4.17: The Highest Overall Attack DR for all Machine Learning Algorithms in both Scaled and Non-scaled Data Sets.

*(Notes : **GB** indicates Gaussian Bayes algorithms, **LR** indicates Logistic Regression Algorithm, **DT** indicates Decision Tree, **DR** indicates detection rates, **S** indicates scaled data, **NS** indicates non scaled data, **NOS** indicates the DR for non-oversampled data, the bold & italic number indicate the highest detection rate in each minority attack among all observations in the data set, red numbers are the lowest detection rate in each minority class.)*

		UNSW NB15			NSL KDD			CICIDS 2017			CICDDOS 2019		
		GB	LR	DT	GB	LR	DT	GB	LR	DT	GB	LR	DT
NOS		36.85	31.08	8.19	32.52	30.37	28.67	30.39	32.50	36.70	36.47	38.72	38.97
Top Overall DR in %	S	34.17	30.80	18.01	**40.66**	39.90	39.09	24.71	38.69	14.95	39.99	40.14	39.14
	NS	31.71	31.53	**43.01**	38.70	26.78	36.13	36.65	27.68	**42.71**	3.88	26.53	**43.41**

In summary, some data sets have underperformed observations in either one of the experiments, resulting in a lower overall detection rate compared to non-oversampled data. For instance, oversampling in scaled UNSW-NB15 had failed to increase the overall detection rate of attack classes, such as GB algorithm had a 2.68 % decrease in the detection rate. However, some data sets have good performance in both scaled and non-scaled data sets, such as the top oversampling DR for NSL KDD is mostly higher than the non-oversample DR.

In the scaled data set, GB performed well in UNSW-NB15 and NSL KDD by achieving the highest overall attack DR, 34.17 % and 40.66 %. While LR performed well in CICIDS 2017 and CICDDOS 2019, with overall attack DR of 38.69 % and 40.14%. Following that, DT generally performed well in increasing the overall attack DR compared to other algorithms. The overall attack DR in UNSW-NB15 is 43.01%, 42.71% for CICIDS 2017 and 43.41 % for CICDDOS 2019. Lastly, the non-scaled data has a greater overall detection rate in three out of four data sets.

4.2.3 Time Performance Evaluation

Table 4.18: Average Runtime Performance for all Oversampling Methods in each Data Set

*(Notes : **S** indicates **scaled** oversampling, **NS** indicates **non scaled** oversampling, the bold numbers indicate the top oversampling method that achieved the highest overall DR for each data set)*

		UNSW-NB15	NSL_KDD	CICIDS 2017	CICDDOS 2019
		Average Time Taken (s)			
ROS	S	6.78	**1.38**	1.10	1.54
	NS	2.29	1.38	1.45	1.54
SMOTE	S	46.28	35.38	1.47	5.71
	NS	35.18	35.38	**1.63**	**5.71**
Borderline SMOTE	S	276.56	132.09	20.63	173.86
	NS	247.19	132.09	17.09	173.86
ADASYN	S	285.61	166.37	13.38	186.90
	NS	264.35	166.37	16.77	186.90
K-Mean SMOTE	S	28.91	41.23	12.04	13.23
	NS	**21.23**	41.23	1.63	9.03

Table 4.18 summarises the runtime performance for all oversampling methods in each data set, and graph visualization is documented in Appendix T. ADSYN has the worst time performance in UNSW-NB15, NSL KDD, and CICDDOS 2019 compared to other oversampling methods, with the least time ranging from 166.37s to most time 285.61s. While ROS consistently outperforms in all methods, taking the least time to oversample, ranging from 1.10s to 6.78s. The top DR performing method in UNSW-NB15 is KMSMOTE, yet ADASYN is the most consistent. KMSMOTE is the second-fastest oversampling method (21.23s), whereas ADASYN

is the slowest oversampling method (264.35s). A trade-off may require between the consistency of DR performance and a reasonable runtime.

Following that, the best performing method in NSL KDD is ROS, which also has the best runtime performance compared to other techniques, taking only 1.38s. In CICIDS 2017, SMOTE is the most reliable and high-performing method, with a runtime of 1.63s, making it an optimal oversampling method for this data set. In addition, SMOTE is also the top DR performance method in CICDDOS 2019 with 5.71s. Where the reliable performer in this data set, KMSMOTE has an average processing time of 9.03s, making it an acceptable choice. The trade-off between consistency of DR performance and runtime performance is lower compared to UNSW-NB15.

According to the runtime results, the runtime relationship for ADASYN and Borderline is always positively related. It is because these two algorithms have a similar operation: identify the difficult-to-learn boundary data for oversampling. Hence, a data set that contains more difficult-to-learn data or overlapping data in feature space may require more time to be oversampled with these two methods. All in all, the time taken for the top DR performing method to oversample in each data set are: 21.23s for K-Mean SMOTE in non-scaled UNSW-NB, 1.38s for ROS in scaled NSL KDD, 1.63s and 5.71s for SMOTE in non-scaled CICIDS 2017 and non-scaled CICDDOS 2017, respectively.

4.2.4 Comparison with other researchers

Table 4.19: Comparison of the Result with Other Research for UNSW-NB15.

*(Notes : bold numbers indicate the highest detection rate, **S** indicate scaled data, **NS** indicates non-scale)*

Paper	**Observation for comparison**		(Bagui et al., 2019)	(Moustafa and Slay, 2015a)	(Meftah, Rachidi and Assem, 2019)	
Classifier	Bayes + KMSMOTE – 10% (S)	Decision Tree + KMSMOTE - 10% (NS)	Bayes	Bayes	Bayes	Decision Tree
Normal	79	62	-	81	***86.29***	74.93
Backdoor	39	40	***66***	20	22.47	4.97
Analysis	0	20	***74***	0	0	0
Fuzzer	57	***63***	57	33.2	36.28	55.24
Shellcode	0	48	***72***	0	1.32	60.84
Recon.	48	79	65	69.9	49.57	***80.77***
Exploit	13	13	56.96	54.6	24.97	***90.08***
DoS	30	55	66	***71.7***	0	8.83
Worms	7	59	84	0	38.64	***72.72***
Generic	***97***	96	83	94.3	96.29	96.96
Overall DR	47.83	74.85	-	37.5	60.70	**75.53**

Table 4.20: Comparison of the Result with Other Research for NSL KDD.

*(Notes : the bold numbers indicate the highest detection rate, **S** indicate scaled data,)*

Paper	**Observation for comparison**	(Rodda and Erothi, 2016)	(Mallissery, Kolekar and Ganiga, 2013)	(Liu et al., 2020)
Classifier	Bayes + ROS – 100% (S)	Bayes		RF + ROS
Normal	74	***96.6***	70.1	-
DoS	***86***	83.81	72.7	-
Probe	***84***	78.26	70.4	-
U2R	49	19.04	***69.5***	-
R2L	***71***	57.03	68.5	-
Overall DR	***79.54***	-	67.53	75.15

Table 4.21: Comparison of the result with other research for CICIDS 2017

*(Notes : the bold numbers indicate the highest detection rate, **S** indicate scaled data, **NS** indicates non-scale)*

Paper	**Observation for comparison**		(Kurniabudi et al., 2020)	(Ravi et al., 2019)
Classifier	LR + ROS – 100% (S)	DT + SMOTE - 40% (NS)	J48	LR
Normal	100	100	96.1	90.5
Brute Force	86	***91***	79	-
Heart Bleed	0	***100***	-	-
Botnet	33	***97***	38.1	0
DoS	***100***	***100***	99.1	93.4
DDoS	96	***100***		
SQL	0	***60***	7.2	8.6
XSS	1	***20***		
Port	***100***	***100***	99.5	99.4
Infiltration	***33***	***33***	0	-
FTP	***100***	***100***	-	48.8
SSH	90	***99***	-	49.9
Overall DR	95.6	***99.6***	-	87

Table 4.22: Comparison of the result with other research for CICDDOS 2019

*(Notes : the bold numbers indicate the highest detection rate, N**S** indicate non-scaled data)*

Paper	**Observation for comparison**		(Chartuni and Márquez, 2021)	(Ferrag et al., 2021)	(Adhao and Pachghare, 2021)	
Classifier	GB – SMOTE – 90% (NS)	DT - SMOTE – 10% (NS)	Proposed NN	DNN	DT	GB
Normal	**100**	**100**	95	**100**	-	-
DNS	49	46	42	**61**	-	-
LDAP	25	50	**56**	30	-	-
MSSQL	58	**96**	92	67	-	-
NTP	80	**98**	95	61	-	-
NetBIOS	40	54	**82**	47	-	-
SNMP	52	**89**	83	93	-	-
SSDP	3	30	28	**55**	-	-
UDP	3	**98**	77	47	-	-
Port Map	0	20	54	**93**	-	-

Syn	70	99	**100**	64	-	-
TFTP	0	**100**	**100**	**100**	-	-
UDP Lag	47	46	**100**	99	-	-
WebDDoS	10	**50**	-	23	-	-
Overall DR	60.18	**97.42**	94.38	96.8	74.03	57.91

Tables 4.19 to 4.22 tabulated the comparison of detection rates in this research with other research for UNSW-NB15, NSL KDD, CICIDS 2017 and CICDDOS 2019. This section only compares the DR for those machine learning algorithms that exist in both research and, section 2.1.7 the LR. In each data set, the top DR performing oversampling method for that particular machine learning was used for comparison. In UNSW-NB15, only Fuzzer traffic can achieve the highest detection rate, 63 %, after oversampling (DT-KMSMOTE). Other researchers, such as Meftah, Rachidi and Assem (2019) who used the random forest with feature selection, had a higher overall performance than any DR in this research, 75.53 % versus 74.85 %.

However, the oversampling model can gain a reasonable increase in all minority attack classes, with most attacks having a detection rate greater than 0 %. Half of the minority attacks in Meftah, Rachidi and Assem (2019) have a greater detection rate, indicating that a combination of feature selection with the oversampling method may enhance the detection rate even further. Subsequently, most minority attack classes in NSL KDD can obtain the highest detection rate for individual comparison, except U2R. in Mallissery, Kolekar and Ganiga (2013) achieves a 69.5 % of detection rate for U2R, while the GB-ROS only obtains 49 %. Despite this, the GB- ROS has the highest overall detection rate among the observations. GB-ROS obtained a detection rate of 79.54 %, while Liu et al. (2020) using random forest and ROS achieved an overall detection rate of 75.15 %.

Additionally, oversampling helps to increase the detection rate for most minority attacks in CICIDS 2017, except Heartbleed and SQL attacks in LR-ROS. The DT-SMOTE outperformed most attack classes since most minority attack classes can achieve a higher detection rate. For example, 91 % for Brute Force, 100 % for FTP, and 99 % for SSH. This could imply that the amount of samples in these classes has a greater impact on the detection rate. Rather than relying solely on oversampling, some attack classes may require more effort to increase the DR, as their detection rate incremental in the model is not significant compared to other

minority classes. For example, infiltration has a maximum DR of 33 %, XSS has a maximum of 20 % DR, and SQL injection with 60 % DR.

For CICDDOS 2019, DT-SMOTE-10 % in the scaled data set can increase the DR for six minority classes to the highest: MSSQL, NTP, SNMP, UDP, TFTP, and WebDDoS. It had achieved the highest overall DR, with 97.42 % among the reviewed research. While Chartuni and Márquez (2021) obtained 89.38 %, and Ferrag et al. (2021) achieved 90.80% DR. Despite, GB-SMOTE-90% has a fair overall DR of 60.18%, yet it is higher than other GB results, 57.91% in Adhao and Pachghare, (2021). It indicates that GB is not a good performer for increasing the DR this data sets. In general, oversampling helps to improve the overall detection rate for attack classes in all data sets.

4.2.5 Summary of Oversampling Results

Table 4.23: Summarised Oversampling Findings for all Data Sets.

*(Notes : **DR** indicates detection rate, **S** indicates **scaled** oversampling data set, **NS** indicates **non-scaled** oversampling data set, **Non-O** indicates **non-oversampled** data set, **L** indicates the lowest value in top 10 overall attack DR range, **H** indicates the highest value in the top 10 overall attack DR range, **yellow** highlights represents the highest value for a particular category)*

	Top 10 overall attack DR, in range						Top DR performer	The occurrence for top oversampling occurrence and time taken															Top occurrence oversampling method
								ROS			SMOTE			BSMOTE			ADASYN			KMSMOTE			
	Non-O	S		NS		% Change between (A) and (B)		S	NS	Avg Time (s)	S	NS	Avg Time (s)	S	NS	Avg Time (s)	S	NS	Avg Time (s)	S	NS	Avg Time (s)	
		L	H	L	H																		
UNSW NB15	36.85 (B)	31.62	34.1	41.64	43.00 (A)	16.69%	DT – KMSMOTE -10%	6	5	4.54	7	6	40.73	4	3	261.8	**9**	7	274.9	3	**9**	25.07	ADASYN - 17
NSL KDD	32.52 (B)	39.70	40.6 (A)	35.35	38.70	20.02%	GB – ROS – 100%	**12**	**12**	1.38	5	4	35.38	4	2	132.0	5	5	166.37	4	7	41.23	ROS - 24
CICIDS 2017	36.70 (B)	37.86	38.69	39.72	42.74 (A)	16.38%	DT – SMOTE – 40%	5	4	1.28	**13**	**11**	1.55	0	3	18.86	5	6	30.15	7	6	6.35	SMOTE - 24
CICDDOS 2019	38.97 (B)	40.14	40.2	42.01	43.41 (A)	11.93%	DT – SMOTE – 10%	6	5	1.54	5	6	5.71	5	5	173.8	5	6	186.9	**9**	**8**	11.13	KMSMOTE - 17
Total occurrence / Average time taken (s)								29	26	2.19	30	27	20.84	13	13	146.6	24	24	518.2	23	30	20.95	-

Table 4.24: The Comparison of the Results with Other Research Work

(Notes: bold and underline indicates the highest detection rate in comparison)

	UNSW NB15	NSL KDD	CICIDS 2017	CICDDOS 2019
Top performing oversampling method in this research	KMSMOTE (DT - 10% - non-scaled)	ROS (GB - 100% - scaled)	SMOTE (DT - 40% - non-scaled)	SMOTE (DT - 10% - non-scaled)
Top Detection Rate in the research	74.85%	**79.54%**	**99.6%**	**97.42%**
DR from other research	**75.53%**	75.49%	96.5%	96.8
Other research	(Meftah, Rachidi and Assem, 2019)	(Subhy, Ibrahim and Basheer, 2013)	(Kurniabudi et al., 2020)	(Ferrag et al., 2021)
Remarks	- use Random Forest with 10-fold cross value for feature selection - use Decision Tree algorithm	- Used self-proposed algorithm, SOM - 1000 epochs with learning rate of 0.9	- Ten-fold cross validation - Extracted 77 out of 84 features - used Random Forest algorithm	-- 75 training and 25 test samples from the full data set. - use full features.

Table 4.22 summarises the results for all data sets. In the multiclass performance evaluation, each data set has a different oversampling method, with higher occurrences of the top oversampling method across all minority attack classes, indicating its persistence. KMSMOTE in UNSW-NB15 has 56 counts, ROS in NSL KDD has 45 counts, SMOTE in CICIDS 2017 has 48 counts and KMSMOTE in CICDDOS 2019 has 93 counts. However, no single oversampling method can provide an increment for all minority classes in a single machine learning. Wherefore, the overall detection rate without accounting for the normal traffics was used to further analyse the oversampling performance.

Two scenarios have been discovered: the method can consistently enhance the detection rate to the highest; the method can achieve the highest detection rate with fewer occurrences. For example, ROS in NSL KDD and SMOTE in CICIDS 2017 are both the best performing and most reliable models, in terms of DR and occurrences from the top ten oversampling methods. The best performing and most prevalent method in UNSW-NB15 and CICDDOS 2019 is different. KMSMOTE is the top performer with 43 % DR and a total of 12 occurrences, but ADASYN is the

top occurrence oversampling method for overall attack DR with 17 occurrences and 42.86 %. Consistency in achieving a high detection rate is another important key factor, as it ensures that the oversampling method performs optimally in most of the iterative.

The top ten overall attack detection rate range shown in the table illustrates that most data sets can increase the overall attack DR. The top ten overall DR had increased from 36.85% to a range of 41.64 % - 43.00 % for non-scaled UNSW-NB, NSL KDD from 32.52 % to 35.35 % - 40.6 %, CICIDS 2017 from 36.70 % to 37.86 % - 42.74 % and CICDDOS 2019 from 38.97 % to 40.14 % - 43.41 %. It implies that most oversampling methods can perform well in all the data sets, except for non-scaled UNSW-NB15.

The runtime performance is also an important key factor for oversampling selection. As tabulated, ROS appears to be the most favourable oversampling method for NSL KDD, because it takes the least time to oversample while providing the highest detection rate and occurrence. SMOTE, on the other hand, is favourable for CICIDS 2017 data set. Additionally, CICDDOS 2019 may have two performing oversampling methods. In this data set, SMOTE can achieve the highest overall attack detection rate (43.41 % with 11 counts) and oversampled in the shortest time (5.71s), compared to a more consistent method KMSMOTE (42.04 % - 11.13s – 17 counts).

For UNSW-NB15 the optimum method may depend on the priority between consistency for DR performance and runtime performance. Unlike CICDDOS 2019, the top DR performing method, KMSMOTE, has a reasonable acceptable average runtime performance (43.00 % - 25.07s) but has fewer occurrences (12 counts). On the other hand, its top occurrence method, ADASYN (17 counts), has the worst runtime performance (274.98s) and a 42.86 % of DR. Therefore, there is a higher cost for a trade-off in terms of time performance and consistency to achieve a high detection rate.

All in all, most oversampling methods have improved the overall attack detection rate for attack classes, as shown in Tables 4.22 and 4.23. The non-scaled data set has a greater overall detection rate in three out of four data sets, which are UNSW-NB15, CICIDS 2017, and CICDDOS 2019. Following that, each data sets have a different outperformed machine learning algorithm and oversampling method.

DT generally performed well in non-scaled and obtained a 16.69 % increase for DR in UNSW-NB15, 16.83 % DR changes in CICIDS 2017, and 11.93% DR changes in CICDDOS 2019. On the other hand, GB performed well in non-scaled data, achieving a 20.02 % change in DR for NSL KDD. To summarise the findings, ROS and SMOTE outperformed the other methods in NSL KDD and CICIDS 2017 by achieving the highest DR rate with the highest consistency and an acceptable runtime. However, there is a trade-off between these two factors for the oversampling method in UNSW-NB15 and CICDDOS 2019. Lastly, SMOTE had improves the overall attack DR well in scaled data, with the highest occurrences of 30, while KMSMOTE outperforms in non-scaled data by achieving the highest occurrences of 30.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

The research objectives were achieved by evaluating five oversampling techniques: ROS, SMOTE, Borderline-SMOTE, ADASYN, and K-mean SMOTE on four unbalanced network intrusion data sets: UNSW-NB15, NSL KDD, CICIDS 2017 and CICDDOS 2019.

In general, oversampling can improve the overall detection rate (DR) for the minority attack classes, with a percentage change ranging from 11.93 % in CICDDOS 2019 to 20.02 % in NSL KDD. SMOTE outperforms when used in the scaled data with the highest top occurrences (30 counts), whereas KMOSTE outperforms when used in the non-scaled data with top occurrences (30 counts).

The pre-processing process, i.e., scaling, also significantly impacts the detection rate, as better performance was achieved in three out of four non-scaled data sets. The distance between data points in the non-scaled data makes the data more differentiable after oversampling. In general, all researched oversampling methods have improved the overall DR for attack classes.

This research can achieve better overall detection rates than the other researchers' work by using oversampling methods on three out of the selected four data sets (refer to Table 4.23). Meftah, Rachidi and Assem (2019) had obtained a higher overall DR using UNSW-NB15 by using a random forest for feature selection and then modelling with DT. For future improvement, it suggests that the oversampling methods can be combined with another pre-processing method, such as feature selection, to extract only valuable information, potentially improving the detection rate further. Finally, the data generated through oversampling methods may have an impact on detection rates. Hence, the parameters required for the oversampling technique, such as the number of K neighbours, can be tuned further to generate a better decision border.

REFERENCES

Abdulhammed, R., Musafer, H., Alessa, A., Faezipour, M. and Abuzneid, A., 2019. Features dimensionality reduction approaches for machine learning based network intrusion detection. *Electronics (Switzerland)*, 8(3). https://doi.org/10.3390/electronics8030322.

Adhao, R. and Pachghare, V., 2021. *Support Based Graph Framework for Effective Intrusion Detection and Classification*. https://doi.org/10.21203/rs.3.rs-1035364/v1.

Ahmim, A., Maglaras, L., Ferrag, M.A., Derdour, M. and Janicke, H., 2018. *A Novel Hierarchical Intrusion Detection System based on Decision Tree and Rules-based Models*. [online] Available at: <http://arxiv.org/abs/1812.09059>.

Ali, A., Mariyam Shamsuddin, S., Ralescu, A. and Ralescu, A.L., 2015. Classification with class imbalance problem: A review. *Classification Int. J. Advance Soft Compu. Appl*, [online] 5(3). Available at: <https://www.researchgate.net/publication/288228469>.

Ali, H., Salleh, M. and Hussain, K., 2019. Improved Adaptive Semi-Unsupervised Weighted Oversampling using Sparsity Factor for Imbalanced Datasets. *International Journal of Advanced Computer Science and Applications*, 10.

Ambusaidi, M.A., He, X. and Nanda, P., 2015. *Unsupervised Feature Selection Method for Intrusion Detection System*.

Amin, A., Anwar, S., Adnan, A., Nawaz, M., Howard, N., Qadir, J., Hawalah, A. and Hussain, A., 2016. Comparing Oversampling Techniques to Handle the Class Imbalance Problem: A Customer Churn Prediction Case Study. *IEEE Access*, 4, pp.7940–7957. https://doi.org/10.1109/ACCESS.2016.2619719.

Awad, M. and Alabdallah, A., 2019. Addressing imbalanced classes problem of intrusion detection system using weighted Extreme Learning Machine. *International Journal of Computer Networks and Communications*, 11(5), pp.39–58. https://doi.org/10.5121/ijcnc.2019.11503.

Aziz, M.N. and Ahmad, T., 2021. CLUSTERING UNDER-SAMPLING DATA FOR IMPROVING THE PERFORMANCE OF INTRUSION DETECTION SYSTEM. *Journal of Engineering Science and Technology*, [online] 16(2), pp.1342–1355. Available at: <https://jestec.taylors.edu.my/Vol%2016%20issue%202%20April%202021/16_2_32.pdf> [Accessed 6 June 2021].

Bagui, S., Kalaimannan, E., Bagui, S., Nandi, D. and Pinto, A., 2019. Using machine learning techniques to identify rare cyber-attacks on the UNSW-NB15 dataset. *Security and Privacy*, 2(6). https://doi.org/10.1002/spy2.91.

Bagui, S. and Li, K., 2021. Resampling imbalanced data for network intrusion detection datasets. *Journal of Big Data*, 8(1). https://doi.org/10.1186/s40537-020-00390-x.

Barua, S., Islam, M.M., Yao, X. and Murase, K., 2012. MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on knowledge and data engineering*, 26(2), pp.405–425.

Bedi, P., Gupta, N. and Jindal, V., 2020. Siam-IDS: Handling class imbalance problem in Intrusion Detection Systems using Siamese Neural Network. *Procedia Computer Science*, [online] 171, pp.780–789. https://doi.org/https://doi.org/10.1016/j.procs.2020.04.085.

Chartuni, A. and Márquez, J., 2021. Multi-classifier of ddos attacks in computer networks built on neural networks. *Applied Sciences (Switzerland)*, 11(22). https://doi.org/10.3390/app112210609.

Chawla, N. v, Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P., 2002. SMOTE synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16, pp.321–357.

Chitrakar, R. and Chuanhe, H., 2012. *Anomaly based intrusion detection using hybrid learning approach of combining k-medoids clustering and naïve bayes classification*. *2012 International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM 2012*. https://doi.org/10.1109/WiCOM.2012.6478433.

Chou, D. and Jiang, M., 2020. Data-Driven Network Intrusion Detection: A Taxonomy of Challenges and Methods. [online] Available at: <http://arxiv.org/abs/2009.07352>.

Choudhary, S. and Kesswani, N., 2020. Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 Datasets using Deep Learning in IoT. In: *Procedia Computer Science*. Elsevier B.V.pp.1561–1573. https://doi.org/10.1016/j.procs.2020.03.367.

Daaderman, A. and Rosander, S., 2018. *Evaluating frameworks for implementing machine learning in signal processing A comparative study of CRISP-DM, semma and kdd*.

Dattagupta Jayanta, S., 2018. *A PERFORMANCE COMPARISON OF OVERSAMPLING METHODS FOR DATA GENERATION IN IMBALANCED LEARNING TASKS*.

Douzas, G., Bacao, F. and Last, F., 2018. Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *Information Sciences*, [online] 465, pp.1–20. https://doi.org/https://doi.org/10.1016/j.ins.2018.06.056.

Dr. Saptarsi, G., 2020. *Class Imbalance, SMOTE, borderline SMOTE, ADASYN*. [online] towards data science. Available at: <https://towardsdatascience.com/class-imbalance-smote-borderline-smote-adasyn-6e36c78d804> [Accessed 4 July 2021].

Dr.S.Siva, S. and Ramani, R., 2011. Discriminant Analysis based Feature Selection in KDD Intrusion Dataset. *International Journal of Computer Applications*, 31, pp.1–7.

Enoch, M. and Khor, K.-C., 2021. DDoS Attacks Data Set - Consolidated from CICDDOS2019 and CICIDS2017. [online] https://doi.org/10.5281/ZENODO.5770290.

Fernández, A., del Río, S., Chawla, N. v. and Herrera, F., 2017. An insight into imbalanced Big Data classification: outcomes and challenges. *Complex & Intelligent Systems*, 3(2), pp.105–120. https://doi.org/10.1007/s40747-017-0037-9.

Ferrag, M.A., Shu, L., Hamouda, D. and Choo, R., 2021. Deep Learning-Based Intrusion Detection for Distributed Denial of Service Attack in Agriculture 4.0. *Electronics*, 10, p.1257. https://doi.org/10.3390/electronics10111257.

Gao, J., Chai, S., Zhang, B. and Xia, Y., 2019. Research on Network Intrusion Detection Based on Incremental Extreme Learning Machine and Adaptive Principal Component Analysis. *Energies*, [online] 12(7). https://doi.org/10.3390/en12071223.

Ghurab, M., Gaphari, G., Alshami, F., Alshamy, R. and Othman, S., 2021. A Detailed Analysis of Benchmark Datasets for Network Intrusion Detection System. *Asian Journal of Research in Computer Science*, pp.14–33. https://doi.org/10.9734/ajrcos/2021/v7i430185.

Han, H., Wang, W.-Y. and Mao, B.-H., 2005. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In: *International conference on intelligent computing*. pp.878–887.

He, H., Bai, Y., Garcia, E.A. and Li, S., 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. pp.1322–1328.

Hindy, H., Brosset, D., Bayne, E., Seeam, A.K., Tachtatzis, C., Atkinson, R. and Bellekens, X., 2020. A Taxonomy of Network Threats and the Effect of Current Datasets on Intrusion Detection Systems. *IEEE Access*, 8, pp.104650–104675. https://doi.org/10.1109/ACCESS.2020.3000179.

Ingre, B. and Yadav, A., 2015. Performance analysis of NSL-KDD dataset using ANN. pp.92–96. https://doi.org/10.1109/SPACES.2015.7058223.

Jia, Y., Zhong, F., Alrawais, A., Gong, B. and Cheng, X., 2020. FlowGuard: An Intelligent Edge Defense Mechanism against IoT DDoS Attacks. *IEEE Internet of Things Journal*, PP, p.1. https://doi.org/10.1109/JIOT.2020.2993782.

Kamarudin, M.H., Maple, C., Watson, T. and Safa, N.S., 2017. A New Unified Intrusion Anomaly Detection in Identifying Unseen Web Attacks. *Security and Communication Networks*, [online] 2017, p.2539034. https://doi.org/10.1155/2017/2539034.

Kanellopoulos, D., Kotsiantis, S. and Pintelas, P., 2016. *Handling imbalanced datasets: A review*. Greece.

Kasongo, S.M. and Sun, Y., 2020. Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset. *Journal of Big Data*, [online] 7(1), p.105. https://doi.org/10.1186/s40537-020-00379-6.

Khan, N., C, N., Negi, A. and Thaseen, S., 2020. Analysis on Improving the Performance of Machine Learning Models Using Feature Selection Technique. pp.69–77. https://doi.org/10.1007/978-3-030-16660-1_7.

Kovács, G., 2019. An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. *Applied Soft Computing*, [online] 83, p.105662. https://doi.org/https://doi.org/10.1016/j.asoc.2019.105662.

Krawczyk, B., 2016. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4), pp.221–232. https://doi.org/10.1007/s13748-016-0094-0.

Kumar, V., Sinha, D., Das, A., Pandey, Dr.S. and Goswami, R., 2020. An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset. *Cluster Computing*, 23. https://doi.org/10.1007/s10586-019-03008-x.

Kurniabudi, Stiawan, D., Darmawijoyo, bin Idris, M.Y. bin, Bamhdi, A.M. and Budiarto, R., 2020. CICIDS-2017 Dataset Feature Analysis with Information Gain for Anomaly Detection. *IEEE Access*, 8, pp.132911–132921. https://doi.org/10.1109/ACCESS.2020.3009843.

Leevy, J.L. and Khoshgoftaar, T.M., 2020. A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data. *Journal of Big Data*, 7(1). https://doi.org/10.1186/s40537-020-00382-x.

Leevy, J.L., Khoshgoftaar, T.M., Bauder, R.A. and Seliya, N., 2018. A survey on addressing high-class imbalance in big data. *Journal of Big Data*, 5(1). https://doi.org/10.1186/s40537-018-0151-6.

Lemaitre, G., Nogueira, F., Oliveira, D. and Aridas, D., 2016. *Random Over Sampling*. [online] Available at: <http://glemaitre.github.io/imbalanced-learn/auto_examples/over-sampling/plot_random_over_sampling.html#sphx-glr-auto-examples-over-sampling-plot-random-over-sampling-py> [Accessed 4 July 2021].

Liu, L., Wang, P., Lin, J. and Liu, L., 2020. Intrusion Detection of Imbalanced Network Traffic Based on Machine Learning and Deep Learning. *IEEE Access*, PP, p.1. https://doi.org/10.1109/ACCESS.2020.3048198.

Lu, W., Li, Z. and Chu, J., 2017. Adaptive Ensemble Undersampling-Boost: A novel learning framework for imbalanced data. *Journal of Systems and Software*, 132, pp.272–282. https://doi.org/10.1016/j.jss.2017.07.006.

Mallissery, S., Kolekar, S. and Ganiga, R., 2013. Accuracy Analysis of Machine Learning Algorithms for Intrusion Detection System using NSL-KDD Dataset. https://doi.org/10.13140/RG.2.1.5018.0247.

Martinez-Plumed, F., Contreras-Ochando, L., Ferri, C., Hernandez Orallo, J., Kull, M., Lachiche, N., Ramirez Quintana, M.J. and Flach, P.A., 2019. CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories. *IEEE Transactions on Knowledge and Data Engineering*, pp.1–1. https://doi.org/10.1109/tkde.2019.2962680.

Meftah, S., Rachidi, T. and Assem, N., 2019. Network based intrusion detection using the UNSW-NB15 dataset. *International Journal of Computing and Digital Systems*, 8(5), pp.477–487. https://doi.org/10.12785/ijcds/080505.

Moustafa, N. and Slay, J., 2015a. The Significant Features of the UNSW-NB15 and the KDD99 Data Sets for Network Intrusion Detection Systems. pp.25–31. https://doi.org/10.1109/BADGERS.2015.014.

Moustafa, N. and Slay, J., 2015b. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). https://doi.org/10.1109/MilCIS.2015.7348942.

Moustafa, N. and Slay, J., 2016. 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*, pp.1–14. https://doi.org/10.1080/19393555.2015.1125974.

Nyakundi, E., 2015. USING SUPPORT VECTOR MACHINES IN ANOMALY INTRUSION DETECTION.

Panigrahi, R. and Borah, S., 2018. *A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems*. [online] *International Journal of Engineering & Technology*, Available at: <https://www.researchgate.net/publication/329045441>.

Protić, D., 2018. Review of KDD Cup ’99, NSL-KDD and Kyoto 2006+ datasets. *Vojnotehnicki glasnik*, 66(3), pp.580–596. https://doi.org/10.5937/vojtehg66-16670.

Ravi, V., Alazab, M., Kp, S., Poornachandran, P., Al-Nemrat, A. and Venkatraman, S., 2019. Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access*, PP, p.1. https://doi.org/10.1109/ACCESS.2019.2895334.

Ring, M., Wunderlich, S., Scheuring, D., Landes, D. and Hotho, A., 2019. A Survey of Network-based Intrusion Detection Data Sets. [online] https://doi.org/10.1016/j.cose.2019.06.005.

Rodda, S. and Erothi, U., 2016. Class imbalance problem in the Network Intrusion Detection Systems. pp.2685–2688. https://doi.org/10.1109/ICEEOT.2016.7755181.

Rout Neelamand Mishra, D.M.M.K., 2018. *Handling Imbalanced Data: A Survey*. *International Proceedings on Advances in Soft Computing, Intelligent Systems and Applications* .

Sáez, J.A., Krawczyk, B. and Woźniak, M., 2016. Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. *Pattern Recognition*, [online] 57, pp.164–178. https://doi.org/https://doi.org/10.1016/j.patcog.2016.03.012.

Salo, F., Injadat, M., Nassif, A.B., Shami, A. and Essex, A., 2018. *Data mining techniques in intrusion detection systems: A systematic literature review*. *IEEE Access*, https://doi.org/10.1109/ACCESS.2018.2872784.

Scaranti, G., Carvalho, L., Barbon Junior, S. and Proenca, M., 2020. Artificial Immune Systems and Fuzzy Logic to Detect Flooding Attacks in Software-Defined Networks. *IEEE Access*, PP, p.1. https://doi.org/10.1109/ACCESS.2020.2997939.

Scott Steinberg, 2019. Cyberattacks now cost companies $200,000 on average, putting many out of business. *CNCB*. [online] 13 Oct. Available at: <https://www.cnbc.com/2019/10/13/cyberattacks-cost-small-companies-200k-putting-many-out-of-business.html> [Accessed 28 June 2021].

Sharafaldin, I., Habibi Lashkari, A., Hakak, S. and Ghorbani, A., 2019. Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy. pp.1–8. https://doi.org/10.1109/CCST.2019.8888419.

Sharafaldin, I., Lashkari, A.H. and Ghorbani, A.A., 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: *ICISSP 2018 - Proceedings of the 4th International Conference on Information Systems Security and Privacy*. SciTePress.pp.108–116. https://doi.org/10.5220/0006639801080116.

Sleeman, W. and Krawczyk, B., 2020. Multi-class imbalanced big data classification on Spark. *Knowledge-Based Systems*, 212. https://doi.org/10.1016/j.knosys.2020.106598.

Song, J., Takakura, H. and Okabe, Y., 2009. *Benchmark Data*. [online] Available at: <http://www.secure-ware.com/contents/product/ashula.html>.

Song, J., Takakura, H., Okabe, Y., Eto, M., Inoue, D. and Nakao, K., 2011. *Statistical Analysis of Honeypot Data and Building of Kyoto 2006+ Dataset for NIDS Evaluation*. BADGERS '11. [online] *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/1978672.1978676.

Subhy, M., Ibrahim, L. and Basheer, D., 2013. A comparison study for intrusion database (KDD99, NSL-KDD) based on self organization map (SOM) artificial neural network. *Journal of Engineering Science and Technology*, 8, pp.107–119.

Suh, Y., Yu, J., Mo, J., Song, L. and Kim, C., 2017. *A Comparison of Oversampling Methods on Imbalanced Topic Classification of Korean News Articles*. Seoul.

Tan, Z., Jamdagni, A., He, X., Nanda, P., Liu, R. and Hu, J., 2015. Detection of Denial-of-Service Attacks Based on Computer Vision Techniques. *IEEE Transactions on Computers*, 64. https://doi.org/10.1109/TC.2014.2375218.

Vigna, G. and Kemmerer, R.A., 2002. Intrusion Detection: A Brief History and Overview (Supplement to Computer Magazine). *Computer*, 35(04), pp.27–30. https://doi.org/10.1109/MC.2002.10036.

Vinayakumar, R., Alazab, M., Soman, K.P., Poornachandran, P., Al-Nemrat, A. and Venkatraman, S., 2019. Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access*, 7, pp.41525–41550. https://doi.org/10.1109/ACCESS.2019.2895334.

Vivek Vinushanth, C., 2020. Handling imbalanced data using Geometric SMOTE. *towards data sciene*. [online] 29 Jul. Available at: <https://towardsdatascience.com/handling-imbalanced-data-using-geometric-smote-770b49d5c7b5> [Accessed 4 July 2021].

Wei, J., Huang, H., Yao, L., Hu, Y., Fan, Q. and Huang, D., 2020. NI-MWMOTE: An improving noise-immunity majority weighted minority oversampling technique for imbalanced classification problems. *Expert Systems with Applications*, [online] 158, p.113504. https://doi.org/https://doi.org/10.1016/j.eswa.2020.113504.

Yahia, A. and Atwell, E., 2018. Network Intrusion Datasets Used in Network Security Education. *International Journal on Integrating Technology in Education*, 7(3), pp.43–50. https://doi.org/10.5121/ijite.2018.7304.

APPENDICES

APPENDIX A: Citation Score of the Selected Papers.

Authors	Citation Score		
	Journal Citation Report	**Scopus**	**Scimago**
(Lu, Li and Chu, 2017)	2.829	8.4	Q1
(Salo et al., 2018)	3.367	4.8	Q1
(Kovács, 2019)	6.725	11.2	Q1
(Wei et al., 2020)	6.954	12.7	Q1
(Barua et al., 2012)	6.977	13.3	Q1
(Douzas, Bacao and Last, 2018)	6.795	12.1	Q1
(Amin et al., 2016)	3.367	4.8	Q1
(Sáez, Krawczyk and Woźniak, 2016)	7.740	15.7	Q1
(Sleeman and Krawczyk, 2020)	8.038	11.3	Q1
(Vinayakumar et al., 2019)	3.367	4.8	Q1
(Protić, 2018)	4.801	8.4	Q1
(Ravi et al., 2019)	3.367	4.8	Q1
(Kurniabudi et al., 2020)	3.367	4.8	Q1
(Bagui and Li, 2021)	1.37	8.6	Q1
(Kasongo and Sun, 2020)	1.37	8.6	Q1
(Liu et al., 2020)	3.367	4.8	Q1
(Hindy et al., 2020)	3.367	4.8	Q1
(Leevy et al., 2018)	1.37	8.6	Q1

(Leevy and Khoshgoftaar, 2020)	1.37	8.6	Q1
(Martinez-Plumed et al., 2019)	6.977	13.3	Q1
(Chawla et al., 2002)	2.776	6.8	Q2
(Tan et al., 2015)	2.663	6.9	Q2
(Ghurab et al., 2021)	3.014	7.1	Q2
(Kamarudin et al., 2017)	1.791	4.2	Q2
(Yahia and Atwell, 2018)	1.824	4.5	Q2
(Abdulhammed et al., 2019)	2.397	2.7	Q2
(Gao et al., 2019)	3.004	4.7	Q2
(Choudhary and Kesswani, 2020)	-	3.0	Q2
(Bagui et al., 2019)	2.58	6.5	Q2
(Moustafa and Slay, 2016)	2.540	2.8	Q2
(Bedi, Gupta and Jindal, 2020)	-	3.0	Q2
(Ali, Salleh and Hussain, 2019)	0.17	1.1	Q3
(Awad and Alabdallah, 2019)	0.19	1.3	Q3
(Kumar et al., 2020)	1.809	3.1	Q3
(Meftah, Rachidi and Assem, 2019)	1.5	1.8	Q3
(Dr.S.Siva and Ramani, 2011)	-	2.4	Q3
(Krawczyk, 2016)	0.28	3.4	Q3
(Aziz and Ahmad,	0.24	1.5	Q3

2021)			
(Subhy, Ibrahim and Basheer, 2013)	0.24	1.2	Q3
(Fernández et al., 2017)	4.92	-	-
(Vigna and Kemmerer, 2002)	2.683	-	-

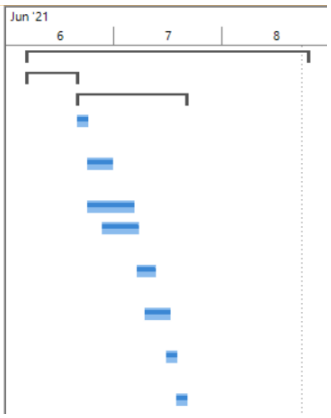
APPENDIX B: Detailed Gantt Chart for FYP 1

I. Project Understanding – FYP1

Task Name	Duration	Start	Finish
Project 1	**79 days**	**Mon 7/6/21 8:00 AM**	**Tue 24/8/21 5:00 PM**
Project understanding	**14 days**	**Mon 7/6/21 8:00 AM**	**Sun 20/6/21 5:00 PM**
Understand project domain	3 days	Mon 7/6/21 8:00 AM	Wed 9/6/21 5:00 PM
Identify problem statement	3 days	Thu 10/6/21 8:00 AM	Sat 12/6/21 5:00 PM
Determine project objective	2 days	Tue 15/6/21 8:00 AM	Wed 16/6/21 5:00 PM
Determine research approach	2 days	Thu 17/6/21 8:00 AM	Fri 18/6/21 5:00 PM
Determine project scope	2 days	Sat 19/6/21 8:00 AM	Sun 20/6/21 5:00 PM

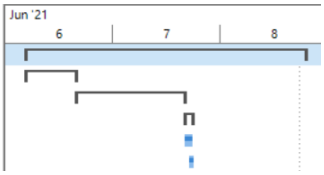
II. Data collection and understanding – FYP 1

Task Name	Duration	Start	Finish
Project 1	**79 days**	**Mon 7/6/21 8:00 AM**	**Tue 24/8/21 5:00 PM**
Project understanding	**14 days**	**Mon 7/6/21 8:00 AM**	**Sun 20/6/21 5:00 PM**
Data collection and understanding	**31 days**	**Mon 21/6/21 8:00 AM**	**Wed 21/7/21 5:00 PM**
Research on unbalanced class problem	3 days	Mon 21/6/21 8:00 AM	Wed 23/6/21 5:00 PM
Identify few unbalanced network intrusion dataset	7 days	Thu 24/6/21 8:00 AM	Wed 30/6/21 5:00 PM
Conduct literature review on data set	13 days	Thu 24/6/21 8:00 AM	Tue 6/7/21 5:00 PM
Review detection rate for the dataset in others literature review	10 days	Mon 28/6/21 8:00 AM	Wed 7/7/21 5:00 PM
Conduct literature review on possible technique	5 days	Thu 8/7/21 8:00 AM	Mon 12/7/21 5:00 PM
Identify and review few oversampling method	7 days	Sat 10/7/21 8:00 AM	Fri 16/7/21 5:00 PM
Identify and determine performance measure	3 days	Fri 16/7/21 8:00 AM	Sun 18/7/21 5:00 PM
Data visualisation	3 days	Mon 19/7/21 8:00 AM	Wed 21/7/21 5:00 PM



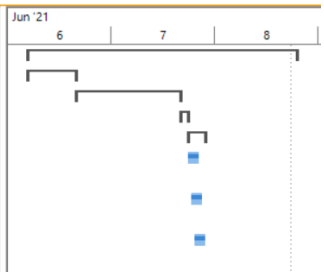
III. Data cleaning & pre-processing – FYP 1

Task Name	Duration	Start	Finish
Project 1	**79 days**	**Mon 7/6/21 8:00 AM**	**Tue 24/8/21 5:00 PM**
Project understanding	**14 days**	**Mon 7/6/21 8:00 AM**	**Sun 20/6/21 5:00 PM**
Data collection and understanding	**31 days**	**Mon 21/6/21 8:00 AM**	**Wed 21/7/21 5:00 PM**
Data Cleaning & Preprocessing	**2 days**	**Thu 22/7/21 8:00 AM**	**Fri 23/7/21 5:00 PM**
Data cleaning	2 days	Thu 22/7/21 8:00 AM	Fri 23/7/21 5:00 PM
Data transformation	1 day	Fri 23/7/21 8:00 AM	Fri 23/7/21 5:00 PM



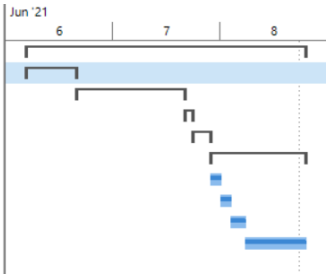
IV. Modelling – FYP 1

Task Name	Duration	Start	Finish
Project 1	**79 days**	**Mon 7/6/21 8:00 AM**	**Tue 24/8/21 5:00 PM**
Project understanding	**14 days**	**Mon 7/6/21 8:00 AM**	**Sun 20/6/21 5:00 PM**
Data collection and understanding	**31 days**	**Mon 21/6/21 8:00 AM**	**Wed 21/7/21 5:00 PM**
Data Cleaning & Preprocessing	**2 days**	**Thu 22/7/21 8:00 AM**	**Fri 23/7/21 5:00 PM**
Modelling	**5 days**	**Sat 24/7/21 8:00 AM**	**Wed 28/7/21 5:00 PM**
Build and Tune Gaussian Bayes model with GridSearchCV	3 days	Sat 24/7/21 8:00 AM	Mon 26/7/21 5:00 PM
Build and Tune Logistic Regression model with GridSearchCV	3 days	Sun 25/7/21 8:00 AM	Tue 27/7/21 5:00 PM
Build and Tune Decision Tree model with GridSearchCV	3 days	Mon 26/7/21 8:00 AM	Wed 28/7/21 5:00 PM



V. Evaluation – FYP 1

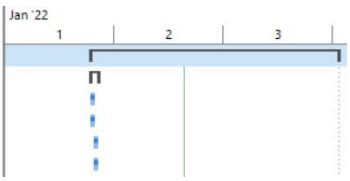
Task Name	Duration	Start	Finish
Project 1	**79 days**	**Mon 7/6/21 8:00 AM**	**Tue 24/8/21 5:00 PM**
Project understanding	**14 days**	**Mon 7/6/21 8:00 AM**	**Sun 20/6/21 5:00 PM**
Data collection and understanding	**31 days**	**Mon 21/6/21 8:00 AM**	**Wed 21/7/21 5:00 PM**
Data Cleaning & Preprocessing	**2 days**	**Thu 22/7/21 8:00 AM**	**Fri 23/7/21 5:00 PM**
Modelling	**5 days**	**Sat 24/7/21 8:00 AM**	**Wed 28/7/21 5:00 PM**
Evaluation	**27 days**	**Thu 29/7/21 8:00 AM**	**Tue 24/8/21 5:00 PM**
evaluation using classification report	3 days	Thu 29/7/21 8:00 AM	Sat 31/7/21 5:00 PM
evaluation using confusion matrix	3 days	Sun 1/8/21 8:00 AM	Tue 3/8/21 5:00 PM
prepare visualisation for metrics	4 days	Wed 4/8/21 8:00 AM	Sat 7/8/21 5:00 PM
document and analyse results in report	17 days	Sun 8/8/21 8:00 AM	Tue 24/8/21 5:00 PM



APPENDIX C: Detailed Gantt Chart for FYP 2

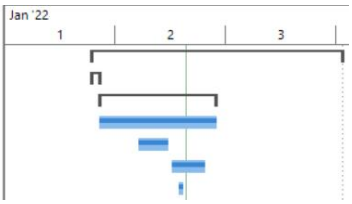
I. Project Understanding – FYP2

Task Name	Duration	Start	Finish
Project 2	**69 days**	**Tue 25/1/22 8:00 AM**	**Sun 3/4/22 5:00 PM**
Project understanding	**2 days**	**Tue 25/1/22 8:00 AM**	**Wed 26/1/22 5:00 PM**
Revise problem statement	1 day	Tue 25/1/22 8:00 AM	Tue 25/1/22 5:00 PM
Revise project objective	1 day	Tue 25/1/22 8:00 AM	Tue 25/1/22 5:00 PM
Revise research approach	1 day	Wed 26/1/22 8:00 AM	Wed 26/1/22 5:00 PM
Revise project scope	1 day	Wed 26/1/22 8:00 AM	Wed 26/1/22 5:00 PM



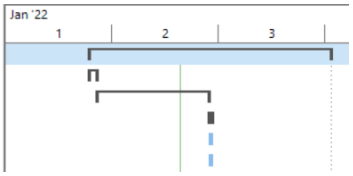
II. Data collection and understanding – FYP2

Task Name	Duration	Start	Finish
Project 2	**69 days**	**Tue 25/1/22 8:00 AM**	**Sun 3/4/22 5:00 PM**
Project understanding	**2 days**	**Tue 25/1/22 8:00 AM**	**Wed 26/1/22 5:00 PM**
Data collection and understanding	**32 days**	**Thu 27/1/22 8:00 AM**	**Sun 27/2/22 5:00 PM**
Revise existing dataset LR	32 days	Thu 27/1/22 8:00 AM	Sun 27/2/22 5:00 PM
Revise Oversampling algorithm	8 days	Mon 7/2/22 8:00 AM	Mon 14/2/22 5:00 PM
Conduct LR on an additional dataset	9 days	Wed 16/2/22 8:00 AM	Thu 24/2/22 5:00 PM
Determine dataset for oversampling	1 day	Fri 18/2/22 8:00 AM	Fri 18/2/22 5:00 PM



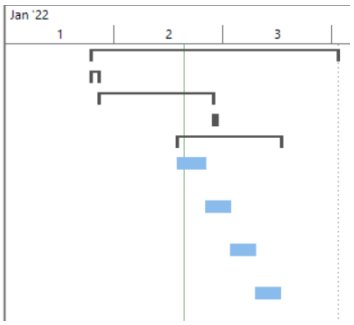
III. Data Pre-processing – FYP2

Task Name	Duration	Start	Finish
Project 2	**69 days**	**Tue 25/1/22 8:00 AM**	**Sun 3/4/22 5:00 PM**
Project understanding	**2 days**	**Tue 25/1/22 8:00 AM**	**Wed 26/1/22 5:00 PM**
Data collection and understanding	**32 days**	**Thu 27/1/22 8:00 AM**	**Sun 27/2/22 5:00 PM**
Data Preprocessing	**1 day**	**Mon 28/2/22 8:00 AM**	**Mon 28/2/22 5:00 PM**
Data cleaning	1 day	Mon 28/2/22 8:00 AM	Mon 28/2/22 5:00 PM
Data conversion	1 day	Mon 28/2/22 8:00 AM	Mon 28/2/22 5:00 PM



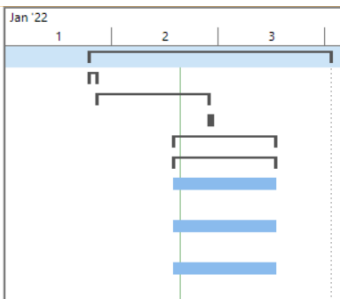
IV. Oversampling – FYP2

Task Name	Duration	Start	Finish
Project 2	**69 days**	**Tue 25/1/22 8:00 AM**	**Sun 3/4/22 5:00 PM**
Project understanding	**2 days**	**Tue 25/1/22 8:00 AM**	**Wed 26/1/22 5:00 PM**
Data collection and understanding	**32 days**	**Thu 27/1/22 8:00 AM**	**Sun 27/2/22 5:00 PM**
Data Preprocessing	**1 day**	**Mon 28/2/22 8:00 AM**	**Mon 28/2/22 5:00 PM**
Oversampling	**29 days**	**Fri 18/2/22 8:00 AM**	**Fri 18/3/22 5:00 PM**
implement all oversampling technique for UNSW-NB15	8 days	Fri 18/2/22 8:00 AM	Fri 25/2/22 5:00 PM
implement all oversampling technique for CICIDS 2017	7 days	Sat 26/2/22 8:00 AM	Fri 4/3/22 5:00 PM
implement all oversampling technique for NSL KDD	7 days	Sat 5/3/22 8:00 AM	Fri 11/3/22 5:00 PM
implement all oversampling technique for CICDDOS 2019	7 days	Sat 12/3/22 8:00 AM	Fri 18/3/22 5:00 PM



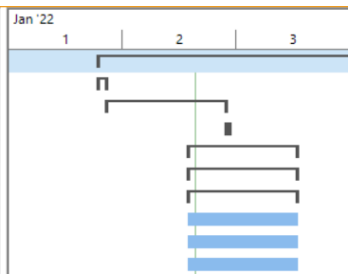
V. Modelling – FYP2

Task Name	Duration	Start	Finish
Project 2	**69 days**	**Tue 25/1/22 8:00 AM**	**Sun 3/4/22 5:00 PM**
Project understanding	**2 days**	**Tue 25/1/22 8:00 AM**	**Wed 26/1/22 5:00 PM**
Data collection and understanding	**32 days**	**Thu 27/1/22 8:00 AM**	**Sun 27/2/22 5:00 PM**
Data Preprocessing	**1 day**	**Mon 28/2/22 8:00 AM**	**Mon 28/2/22 5:00 PM**
Oversampling	**29 days**	**Fri 18/2/22 8:00 AM**	**Fri 18/3/22 5:00 PM**
Modelling	**29 days**	**Fri 18/2/22 8:00 AM**	**Fri 18/3/22 5:00 PM**
Build and Tune Gaussian Bayes model with GridSearchCV	29 days	Fri 18/2/22 8:00 AM	Fri 18/3/22 5:00 PM
Build and Tune Logistic Regression model with GridSearchCV	29 days	Fri 18/2/22 8:00 AM	Fri 18/3/22 5:00 PM
Build and Tune Decision Tree model with GridSearchCV	29 days	Fri 18/2/22 8:00 AM	Fri 18/3/22 5:00 PM



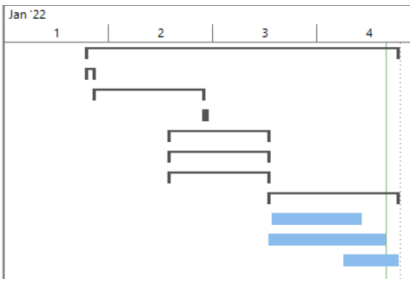
VI. Model Evaluation – FYP2

Task Name	Duration	Start	Finish
Project 2	**69 days**	**Tue 25/1/22 8:00 AM**	**Sun 3/4/22 5:00 PM**
Project understanding	**2 days**	**Tue 25/1/22 8:00 AM**	**Wed 26/1/22 5:00 PM**
Data collection and understanding	**32 days**	**Thu 27/1/22 8:00 AM**	**Sun 27/2/22 5:00 PM**
Data Preprocessing	**1 day**	**Mon 28/2/22 8:00 AM**	**Mon 28/2/22 5:00 PM**
Oversampling	**29 days**	**Fri 18/2/22 8:00 AM**	**Fri 18/3/22 5:00 PM**
Modelling	**29 days**	**Fri 18/2/22 8:00 AM**	**Fri 18/3/22 5:00 PM**
Model Evaluation	**29 days**	**Fri 18/2/22 8:00 AM**	**Fri 18/3/22 5:00 PM**
evaluation using classification report	29 days	Fri 18/2/22 8:00 AM	Fri 18/3/22 5:00 PM
evaluation using model runtime	29 days	Fri 18/2/22 8:00 AM	Fri 18/3/22 5:00 PM
evaluation using ROC curve	29 days	Fri 18/2/22 8:00 AM	Fri 18/3/22 5:00 PM



VII. Reporting – FYP2

Task Name	Duration	Start	Finish
Project 2	**91 days**	**Tue 25/1/22 8:00 AM**	**Mon 25/4/22 5:00 PM**
Project understanding	**2 days**	**Tue 25/1/22 8:00 AM**	**Wed 26/1/22 5:00 PM**
Data collection and understanding	**32 days**	**Thu 27/1/22 8:00 AM**	**Sun 27/2/22 5:00 PM**
Data Preprocessing	**1 day**	**Mon 28/2/22 8:00 AM**	**Mon 28/2/22 5:00 PM**
Oversampling	**29 days**	**Fri 18/2/22 8:00 AM**	**Fri 18/3/22 5:00 PM**
Modelling	**29 days**	**Fri 18/2/22 8:00 AM**	**Fri 18/3/22 5:00 PM**
Model Evaluation	**29 days**	**Fri 18/2/22 8:00 AM**	**Fri 18/3/22 5:00 PM**
Reporting	**38 days**	**Sat 19/3/22 8:00 AM**	**Mon 25/4/22 5:00 PM**
FYP poster	26 days	Sun 20/3/22 8:00 AM	Thu 14/4/22 5:00 PM
Report writing	34 days	Sat 19/3/22 8:00 AM	Thu 21/4/22 5:00 PM
Presentation Preparation	16 days	Sun 10/4/22 8:00 AM	Mon 25/4/22 5:00 PM



APPENDIX D: Cross Validation Score between Classifiers for Researched Data Sets.

(Notes : the bold numbers indicate the best score for the cv scores)

CV	Gaussian Bayes (GB)	Logistic Regression (LR)	Decision Tree (DT)
UNSW-NB15			
1	**47.03**	**67.9**	**77.66**
2	46.54	67.81	77.64
3	46.29	67.73	77.61
4	46.25	67.60	77.61
5	45.83	67.57	77.60
CICIDS2017			
1	**79.36**	**95.42**	**97.53**
2	79.32	95.41	97.52
3	79.26	95.41	97.52
4	79.23	95.38	97.51
5	79.19	95.37	97.51
NSL KDD			
1	**98.53**	**79.90**	**85.92**
2	98.47	79.89	85.91
3	98.46	79.88	85.91
4	98.41	79.88	85.90
5	98.33	79.88	85.90
ICXS 2012			
1	**98.54**	**79.90**	**99.92**
2	98.47	79.89	99.92
3	98.46	79.88	99.91
4	98.41	79.88	99.91
5	98.34	79.87	99.90
Kyoto 2006			
1	**76.04**	**68.00**	**99.87**
2	75.31	67.63	99.86
3	74.76	67.62	99.85

4	73.29	67.51	99.85
5	71.94	67.43	99.84
CICDDOS 2019			
1	**63.61**	**91.43**	**94.43**
2	62.73	91.42	94.41
3	61.00	91.42	94.38
4	60.40	91.41	94.33
5	58.23	91.40	94.32

APPENDIX E: Data Distribution of All Oversampling Percentage for all Data Sets.

i. UNSW-NB15

Sample /Classes	Normal	Generic	Exploits	Fuzzers	DoS	Recon	Analysis	Back door	Shell code	Worm
Initial	56,000	40,000	33,393	18,184	12,264	10,491	2,000	1,746	1,133	130
10%	56,000	40,000	36,732	20,002	13,460	11,540	2,200	1,920	1,246	143
20%	56,000	40,000	40,071	21,820	14,716	12,589	2,400	2,098	1,359	156
30%	56,000	40,000	43,410	23,639	15,943	13,638	2,600	2,269	1,472	169
40%	56,000	40,000	46,750	25,457	17,169	14,687	2,800	2,444	1,586	182
50%	56,000	40,000	50,089	27,276	18,396	15,736	3,000	2,619	1,699	195
60%	56,000	40,000	53,428	29,094	19,622	16,785	3,200	2,793	1,812	208
70%	56,000	40,000	56,768	30,912	20,848	17,834	3,400	2,968	1,926	221
80%	56,000	40,000	60,107	32,731	22,075	18,883	3,600	3,142	2,039	234
90%	56,000	40,000	63,446	34,549	23,301	19,932	3,800	3,317	2,152	247
100%	56,000	40,000	66,786	36,368	24,528	20,982	4,000	3,492	2,266	260

ii. CICIDS 2017

Sample /Classes	Norm	DoS	Port Scan	DDoS	FTP	SSH	Bot	Brute Force	XSS	Infil.	SQL	Heart Bleed
Initial	119,006	32,884	21,592	15,364	4,122	2,255	1,343	1,037	480	30	16	10
10%	119,006	32,884	21,592	15,364	4,122	2,255	1,477	1,140	528	33	17	11
20%	119,006	32,884	21,592	15,364	4,122	2,255	1,611	1,244	576	36	19	12
30%	119,006	32,884	21,592	15,364	4,122	2,255	1,745	1,348	624	39	20	13
40%	119,006	32,884	21,592	15,364	4,122	2,255	1,880	1,451	672	42	22	14
50%	119,006	32,884	21,592	15,364	4,122	2,255	2,014	1,555	720	45	24	15
60%	119,006	32,884	21,592	15,364	4,122	2,255	2,148	1,659	768	48	25	16
70%	119,006	32,884	21,592	15,364	4,122	2,255	2,283	1,762	816	51	27	17
80%	119,006	32,884	21,592	15,364	4,122	2,255	2,417	1,866	864	54	28	18
90%	119,006	32,884	21,592	15,364	4,122	2,255	2,551	1,970	912	57	30	19
100%	119,006	32,884	21,592	15,364	4,122	2,255	2,686	2,074	960	60	32	20

iii. NSL KDD

Sample /Classes	Normal	DoS	Probe	R2L	U2R
Initial	67,343	45,927	11,647	995	52
10%	67,343	50,519	12,811	1,094	57
20%	67,343	55,112	13,976	1,194	62
30%	67,343	59,705	15,141	1,293	67
40%	67,343	64,297	16,305	1,393	72
50%	67,343	68,890	17,470	1,492	78
60%	67,343	73,483	18,635	1,592	83
70%	67,343	78,075	19,799	1,691	88
80%	67,343	82,668	20,964	1,791	93
90%	67,343	87,261	22,129	1,890	98
100%	67,343	101,038	25,622	2,188	114

iv. CICDDOS 2019

Sample /Classes	Norm	TFTP	UDP	MSSQL	SSDP	NTP	Syn	SNMP	DNS	LDAP	Net BIOS	UDP Lag	Port Map	Web DDOS
Initial	151,993	62,162	24,895	15,231	9,597	7,956	7,186	4,323	3,658	1,858	999	889	81	31
10%	151,993	62,162	24,895	15,231	10,556	7,956	7,904	4,755	4,023	2,043	1,098	977	89	34
20%	151,993	62,162	24,895	15,231	11,516	7,956	8,623	5,187	4,389	2,229	1,198	1,066	97	37
30%	151,993	62,162	24,895	15,231	12,476	7,956	9,341	5,619	4,755	2,415	1,298	1,155	105	40
40%	151,993	62,162	24,895	15,231	13,435	7,956	10,060	6,052	5,121	2,601	1,398	1,244	113	43
50%	151,993	62,162	24,895	15,231	14,395	7,956	10,779	6,484	5,487	2,787	1,498	1,333	121	46
60%	151,993	62,162	24,895	15,231	15,355	7,956	11,497	6,916	5,852	2,972	1,598	1,422	129	49
70%	151,993	62,162	24,895	15,231	16,314	7,956	12,216	7,349	6,218	3,158	1,698	1,511	137	52
80%	151,993	62,162	24,895	15,231	17,274	7,956	12,934	7,781	6,584	3,344	1,798	1,600	145	55
90%	151,993	62,162	24,895	15,231	18,234	7,956	13,653	8,213	6,950	3,530	1,898	1,689	153	58
100%	151,993	62,162	24,895	15,231	19,194	7,956	14,372	8,646	7,316	3,716	1,998	1,778	162	62

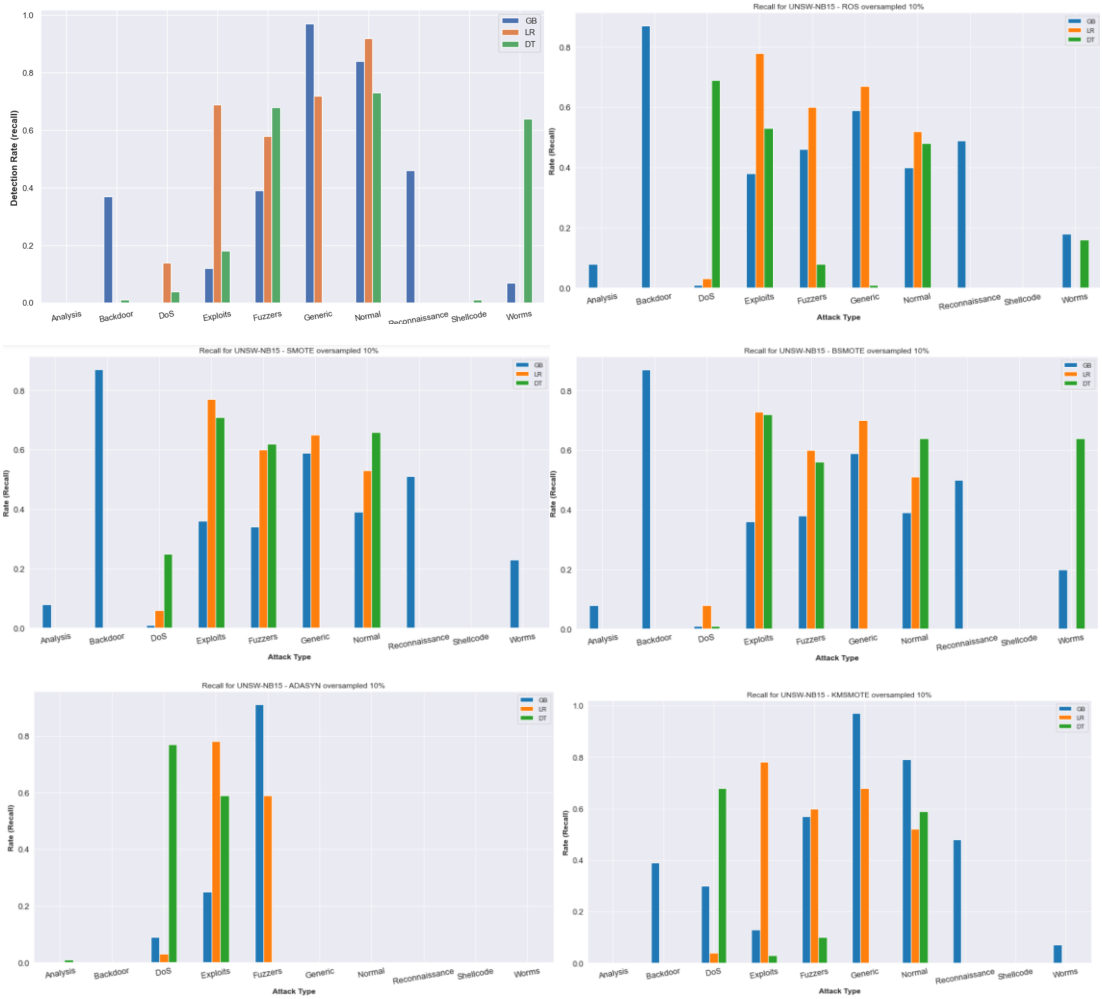
APPENDIX F: UNSW-NB15 oversampling individual DR on scaled data

i. Oversampled 10% detection rate

Result	Analysis	Back door	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	***8***	***87***	1	38	46	59	40	49	0	18
GB SMOTE	***8***	***87***	1	36	34	59	39	***51***	0	23
GB BSMOTE	***8***	***87***	1	36	38	59	39	50	0	20
GB ADASYN	0	0	9	25	***91***	0	0	0	0	0
GB KSMOTE	0	39	30	13	57	97	79	48	0	7
LR ROS	0	0	3	***78***	60	67	52	0	0	0

LR SMOTE	0	0	6	77	60	65	53	0	0	0
LR BSMOTE	0	0	8	73	60	70	51	0	0	0
LR ADASYN	0	0	0	3	78	59	0	0	0	0
LR KSMOTE	0	0	4	***78***	60	68	52	0	0	0
DT ROS	0	0	69	53	8	1	48	0	0	16
DT SMOTE	0	0	25	71	62	0	66	0	0	0
DT BSMOTE	0	0	1	72	56	0	64	0	0	64
DT ADASYN	1	0	***77***	59	0	0	0	0	0	0
DT KSMOTE	0	0	68	3	10	0	59	0	0	0

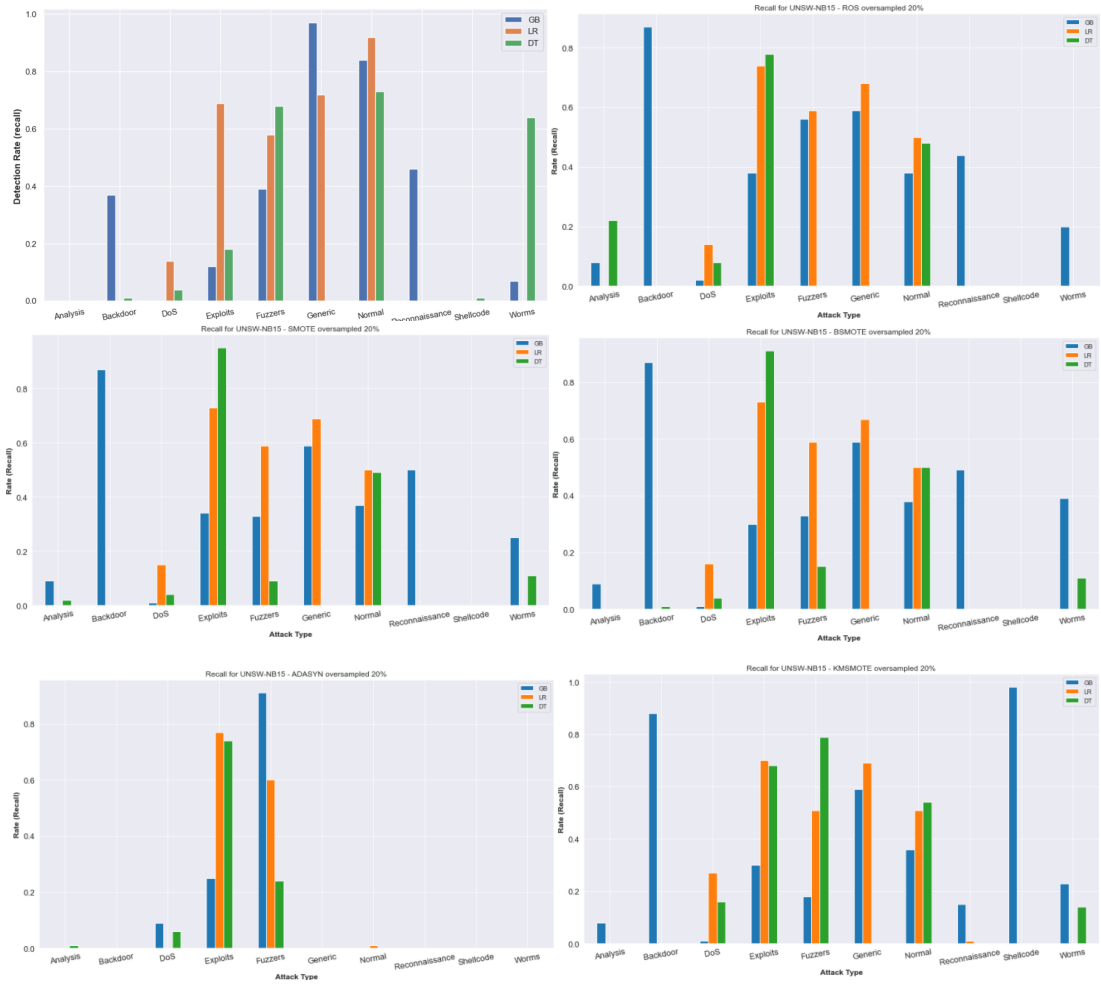
ii. Oversampled 10% comparison results



iii. Oversampled 20% detection rate

Result	Analysis	Back door	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	8	87	2	38	56	59	38	44	0	2
GB SMOTE	9	87	1	34	33	59	37	5	0	25
GB BSMOTE	9	87	1	30	33	59	38	***49***	0	39
GB ADASYN	0	0	9	25	***91***	0	0	0	0	0
GB KSMOTE	8	***88***	1	30	18	59	36	15	***98***	23
LR ROS	0	0	14	74	59	68	5	0	0	0
LR SMOTE	0	0	15	73	59	69	5	0	0	0
LR BSMOTE	0	0	16	73	59	67	5	0	0	0
LR ADASYN	0	0	0	77	6	0	1	0	0	0
LR KSMOTE	0	0	***27***	7	51	69	51	1	0	0
DT ROS	***22***	0	8	78	0	0	48	0	0	0
DT SMOTE	2	0	4	***95***	9	0	49	0	0	0
DT BSMOTE	0	1	4	91	15	0	5	0	0	11
DT ADASYN	1	0	6	74	24	0	0	0	0	0
DT KSMOTE	0	0	16	68	79	0	54	0	0	14

iv. Oversampled 20% comparison results

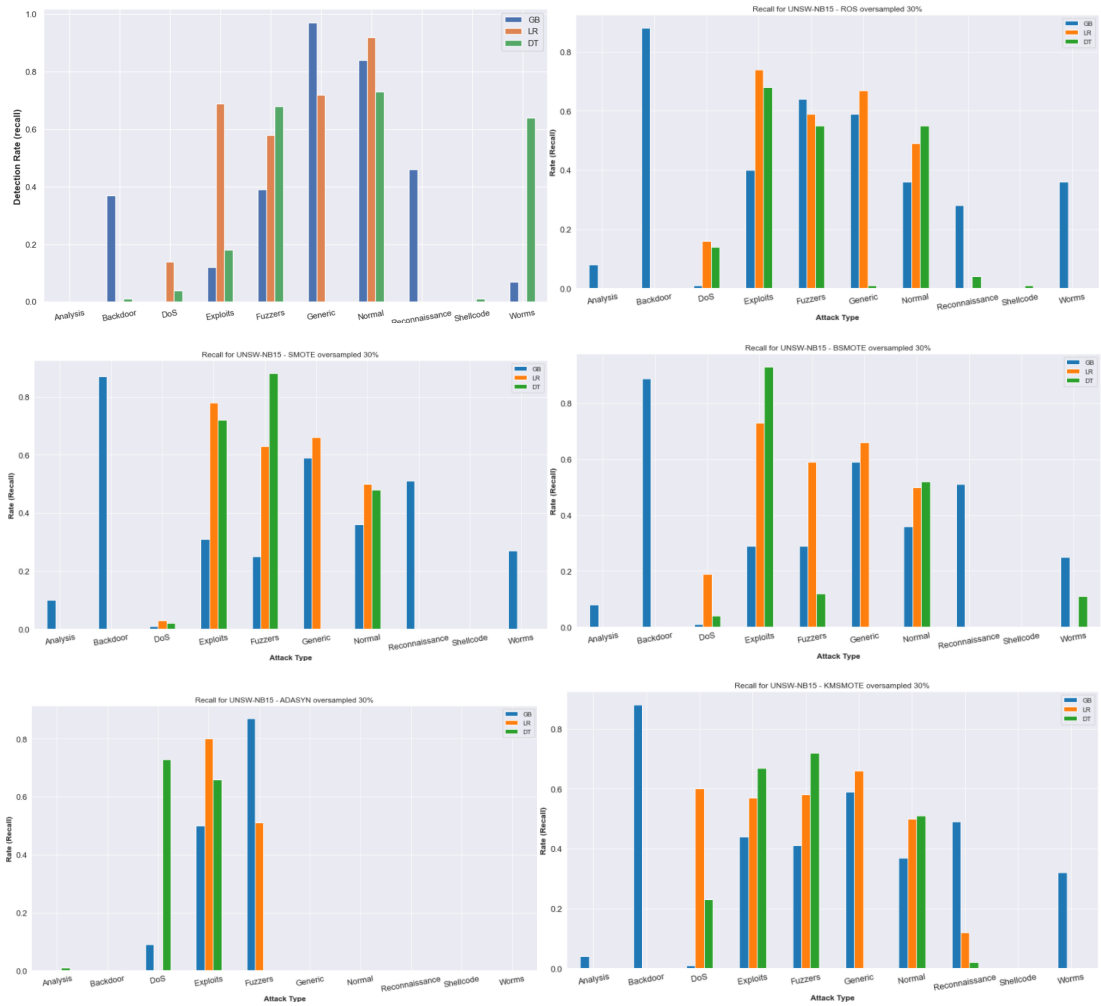


v. Oversampled 30% detection rate

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shellcode	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	8	88	1	4	64	59	36	28	0	36
GB SMOTE	***10***	87	1	31	25	59	36	***51***	0	27
GB BSMOTE	8	***89***	1	29	29	59	36	***51***	0	25
GB ADASYN	0	0	9	50	87	0	0	0	0	0
GB KSMOTE	4	88	1	44	41	59	37	49	0	32
LR ROS	0	0	16	74	59	67	49	0	0	0
LR SMOTE	0	0	3	78	63	66	50	0	0	0
LR BSMOTE	0	0	19	73	59	66	5	0	0	0

LR ADASYN	0	0	0	80	51	0	0	0	0	0
LR KSMOTE	0	0	60	57	58	66	50	12	0	0
DT ROS	0	0	14	68	55	1	55	4	1	0
DT SMOTE	0	0	2	72	***88***	0	48	0	0	0
DT BSMOTE	0	0	4	***93***	12	0	52	0	0	11
DT ADASYN	1	0	***73***	66	0	0	0	0	0	0
DT KSMOTE	0	0	23	67	72	0	51	2	0	0

vi. Oversampled 30% comparison results

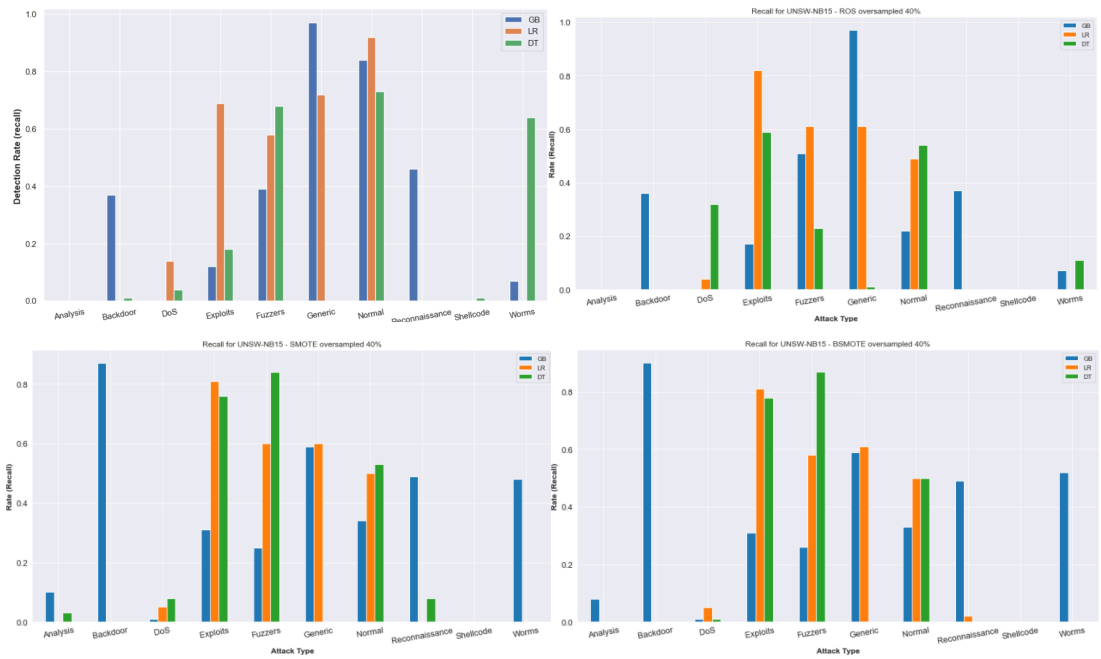


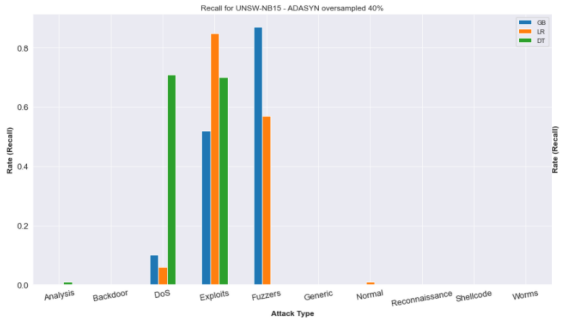
vii. Oversampled 40% detection rate

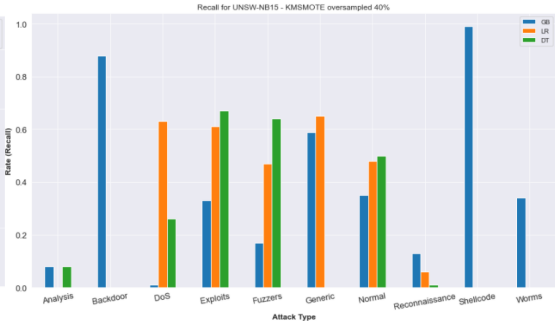
Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0

DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	36	0	17	51	97	22	37	0	7
GB SMOTE	***10***	87	1	31	25	59	34	***49***	0	48
GB BSMOTE	8	***90***	1	31	26	59	33	***49***	0	52
GB ADASYN	0	0	1	52	87	0	0	0	0	0
GB KSMOTE	8	88	1	33	17	59	35	13	***99***	34
LR ROS	0	0	4	82	61	61	49	0	0	0
LR SMOTE	0	0	5	81	60	60	50	0	0	0
LR BSMOTE	0	0	5	81	58	61	50	2	0	0
LR ADASYN	0	0	6	***85***	57	0	1	0	0	0
LR KSMOTE	0	0	63	61	47	65	48	6	0	0
DT ROS	0	0	32	59	23	1	54	0	0	11
DT SMOTE	3	0	8	76	84	0	53	8	0	0
DT BSMOTE	0	0	1	78	***87***	0	50	0	0	0
DT ADASYN	1	0	***71***	70	0	0	0	0	0	0
DT KSMOTE	8	0	26	67	64	0	5	1	0	0

viii. Oversampled 40% comparison results



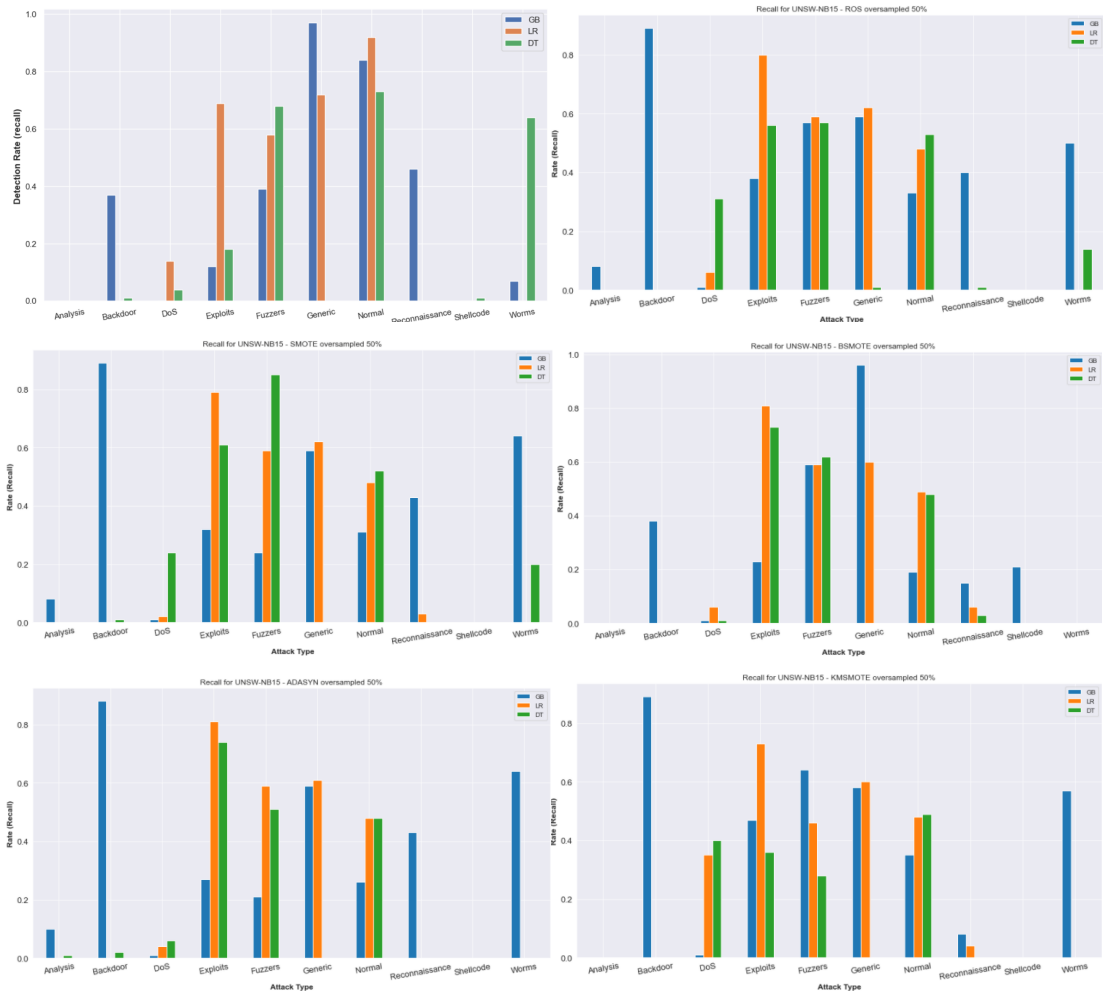




ix. Oversampled 50% detection rate

Result	Analysis	Back door	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	8	***89***	1	38	57	59	33	40	0	50
GB SMOTE	8	***89***	1	32	24	59	31	43	0	64
GB BSMOTE	0	38	1	23	59	96	19	15	***21***	0
GB ADASYN	***10***	88	1	27	21	59	26	43	0	64
GB KSMOTE	0	***89***	1	47	64	58	35	8	0	57
LR ROS	0	0	6	80	59	62	48	0	0	0
LR SMOTE	0	0	2	79	59	62	48	3	0	0
LR BSMOTE	0	0	6	***81***	59	60	49	6	0	0
LR ADASYN	0	0	4	***81***	59	61	48	0	0	0
LR KSMOTE	0	0	35	73	46	60	48	4	0	0
DT ROS	0	0	31	56	57	1	53	1	0	14
DT SMOTE	0	1	24	61	***85***	0	52	0	0	2
DT BSMOTE	0	38	1	23	59	96	19	15	***21***	0
DT ADASYN	1	2	6	74	51	0	48	0	0	0
DT KSMOTE	0	0	***40***	36	28	0	49	0	0	0

x. Oversampled 50% comparison results

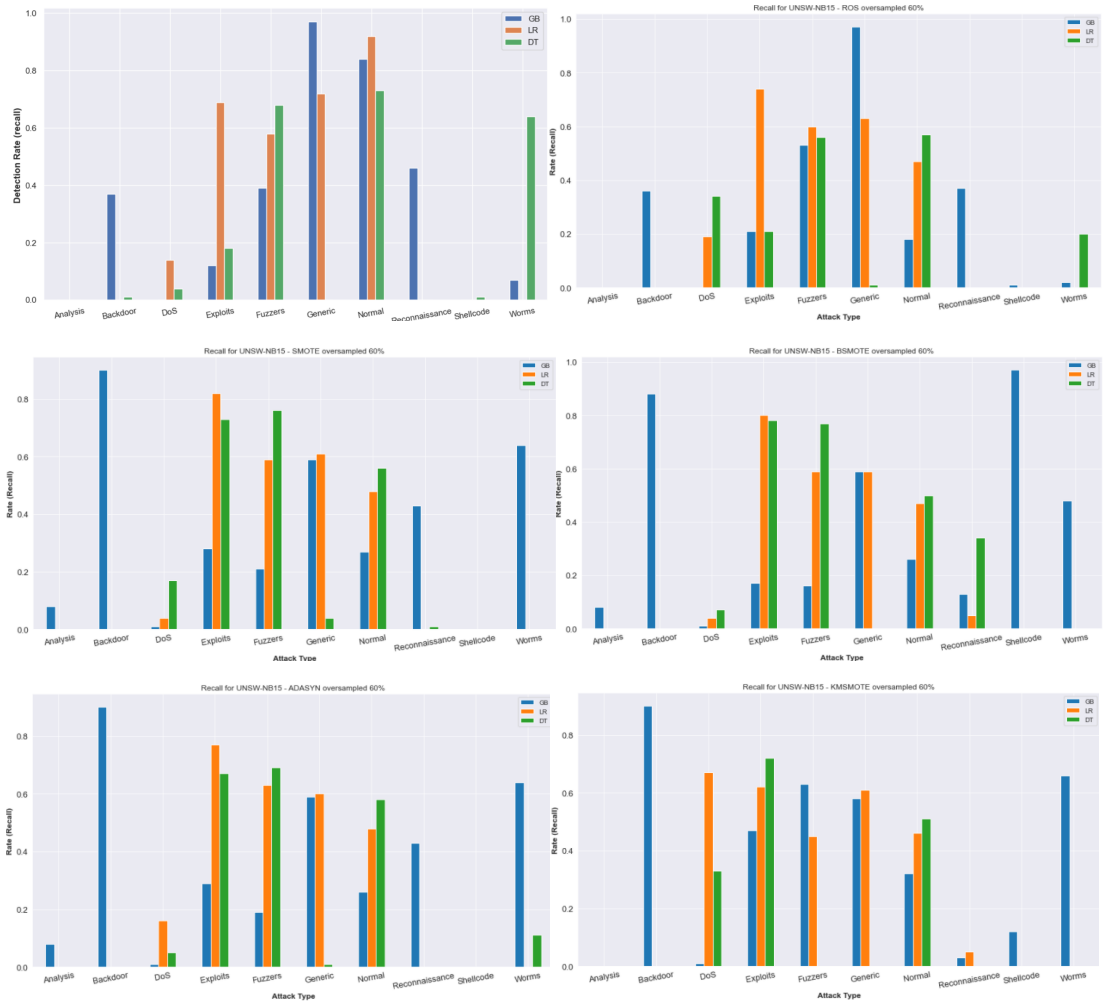


xi. Oversampled 60% detection rate

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shellcode	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	36	0	21	53	97	18	37	1	2
GB SMOTE	***8***	***90***	1	28	21	59	27	43	0	64
GB BSMOTE	***8***	88	1	17	16	59	26	13	***97***	48
GB ADASYN	***8***	***90***	1	29	19	59	26	43	0	64
GB KSMOTE	0	***90***	1	47	63	58	32	3	12	***66***
LR ROS	0	0	19	74	60	63	47	0	0	0
LR SMOTE	0	0	4	***82***	59	61	48	0	0	0
LR BSMOTE	0	0	4	***80***	59	59	47	5	0	0

LR ADASYN	0	0	16	77	63	60	48	0	0	0
LR KSMOTE	0	0	***67***	62	45	61	46	5	0	0
DT ROS	0	0	34	21	56	1	57	0	0	2
DT SMOTE	0	0	17	73	76	4	56	1	0	0
DT BSMOTE	0	0	7	78	***77***	0	50	34	0	0
DT ADASYN	0	0	5	67	69	1	58	0	0	11
DT KSMOTE	0	0	33	72	0	0	51	0	0	0

xii. Oversampled 60% comparison results

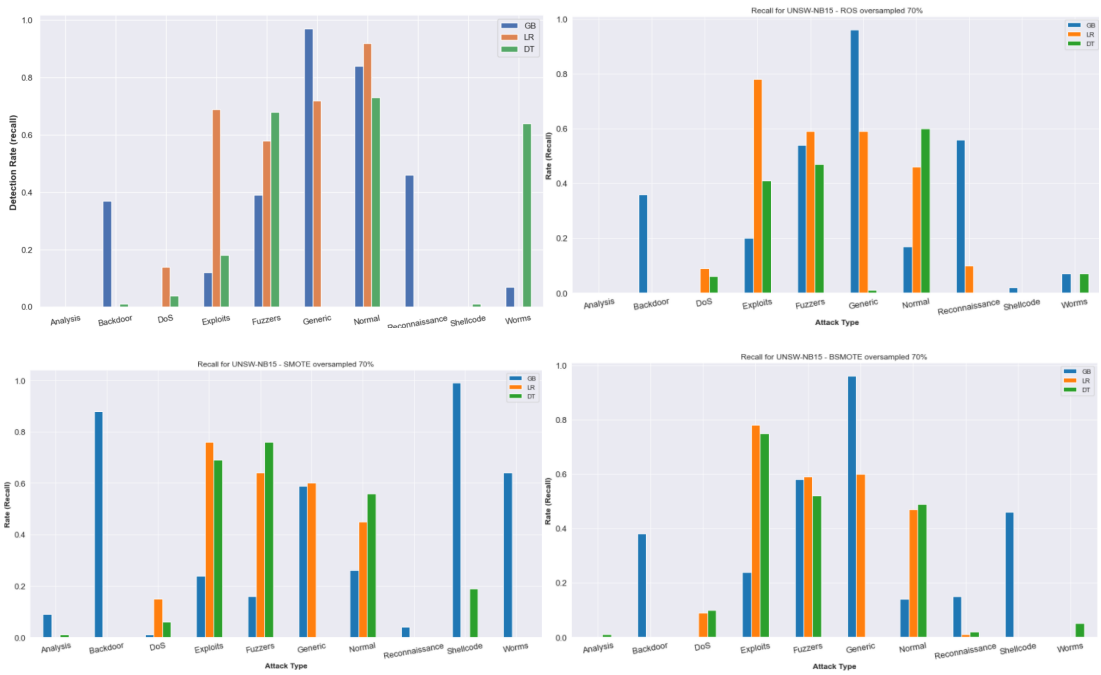


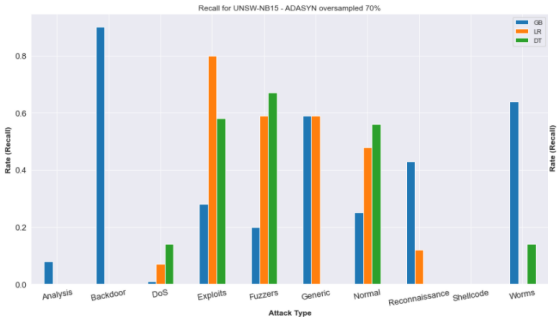
xiii. Oversampled 70% detection rate

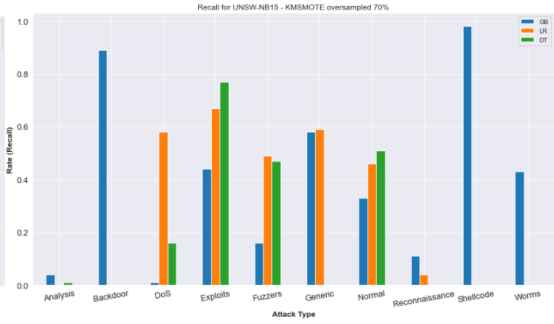
Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0

DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	36	0	20	54	96	17	***56***	2	7
GB SMOTE	***9***	88	1	24	16	59	26	4	***99***	***64***
GB BSMOTE	0	38	0	24	58	96	14	15	46	0
GB ADASYN	8	***90***	1	28	20	59	25	43	0	***64***
GB KSMOTE	4	89	1	44	16	58	33	11	98	43
LR ROS	0	0	9	78	59	59	46	1	0	0
LR SMOTE	0	0	15	76	64	6	45	0	0	0
LR BSMOTE	0	0	9	78	59	60	47	1	0	0
LR ADASYN	0	0	7	***80***	59	59	48	12	0	0
LR KSMOTE	0	0	***58***	67	49	59	46	4	0	0
DT ROS	0	0	6	41	47	1	60	0	0	7
DT SMOTE	1	0	6	69	***76***	0	56	0	19	0
DT BSMOTE	1	0	10	75	52	0	49	2	0	5
DT ADASYN	0	0	14	58	67	0	56	0	0	14
DT KSMOTE	1	0	16	77	47	0	51	0	0	0

xiv. Oversampled 70% comparison results



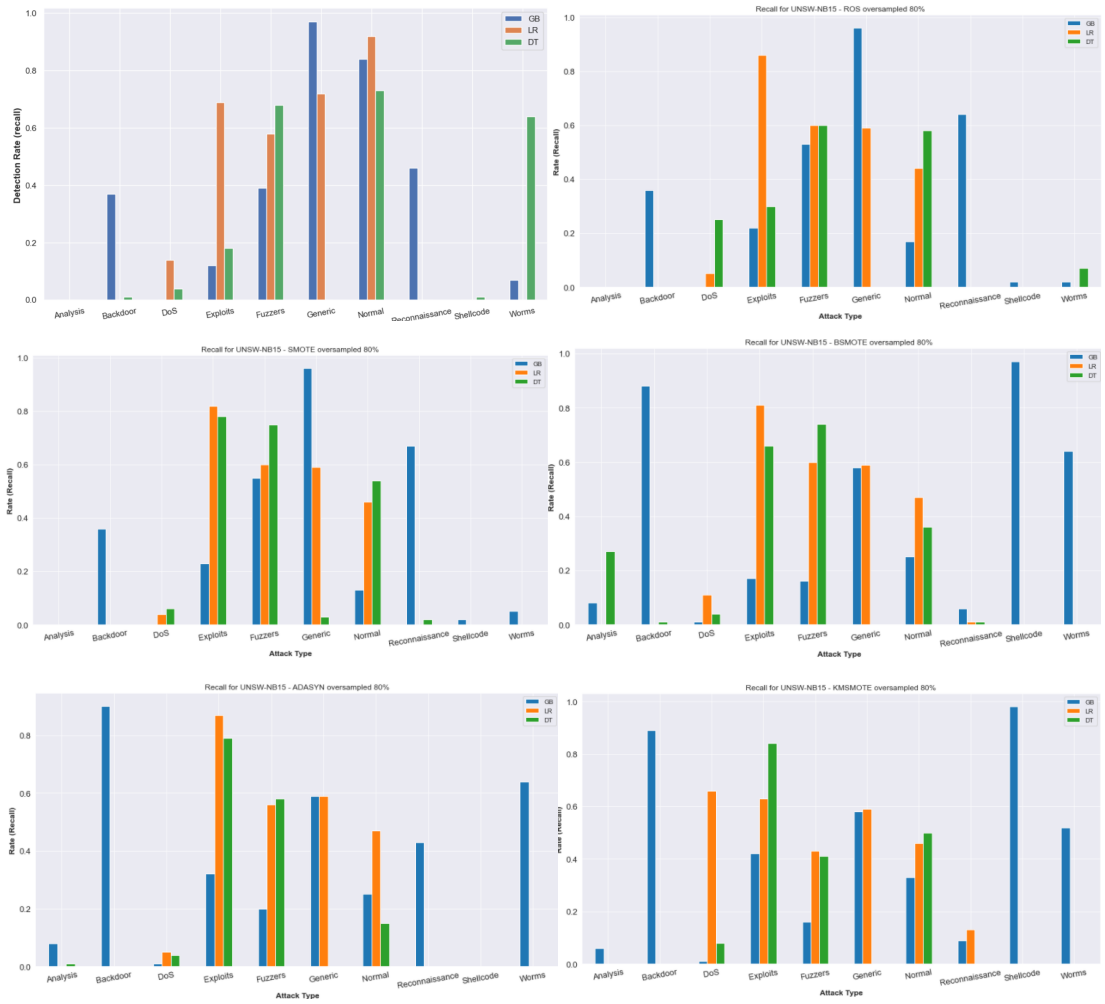




xv. Oversampled 80% detection rate

Result	Analysis	Back door	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	36	0	22	53	96	17	64	2	2
GB SMOTE	0	36	0	23	55	96	13	***67***	2	5
GB BSMOTE	8	88	1	17	16	58	25	6	97	64
GB ADASYN	8	***90***	1	32	20	59	25	43	0	64
GB KSMOTE	6	89	1	42	16	58	33	9	***98***	52
LR ROS	0	0	5	86	60	59	44	0	0	0
LR SMOTE	0	0	4	82	60	59	46	0	0	0
LR BSMOTE	0	0	11	81	60	59	47	1	0	0
LR ADASYN	0	0	5	***87***	56	59	47	0	0	0
LR KSMOTE	0	0	***66***	63	43	59	46	13	0	0
DT ROS	0	0	25	3	60	0	58	0	0	0
DT SMOTE	0	0	6	78	***75***	3	54	2	0	0
DT BSMOTE	***27***	1	4	66	74	0	36	1	0	0
DT ADASYN	1	0	4	79	58	0	15	0	0	0
DT KSMOTE	0	0	8	84	41	0	50	0	0	0

xvi. Oversampled 80% comparison results

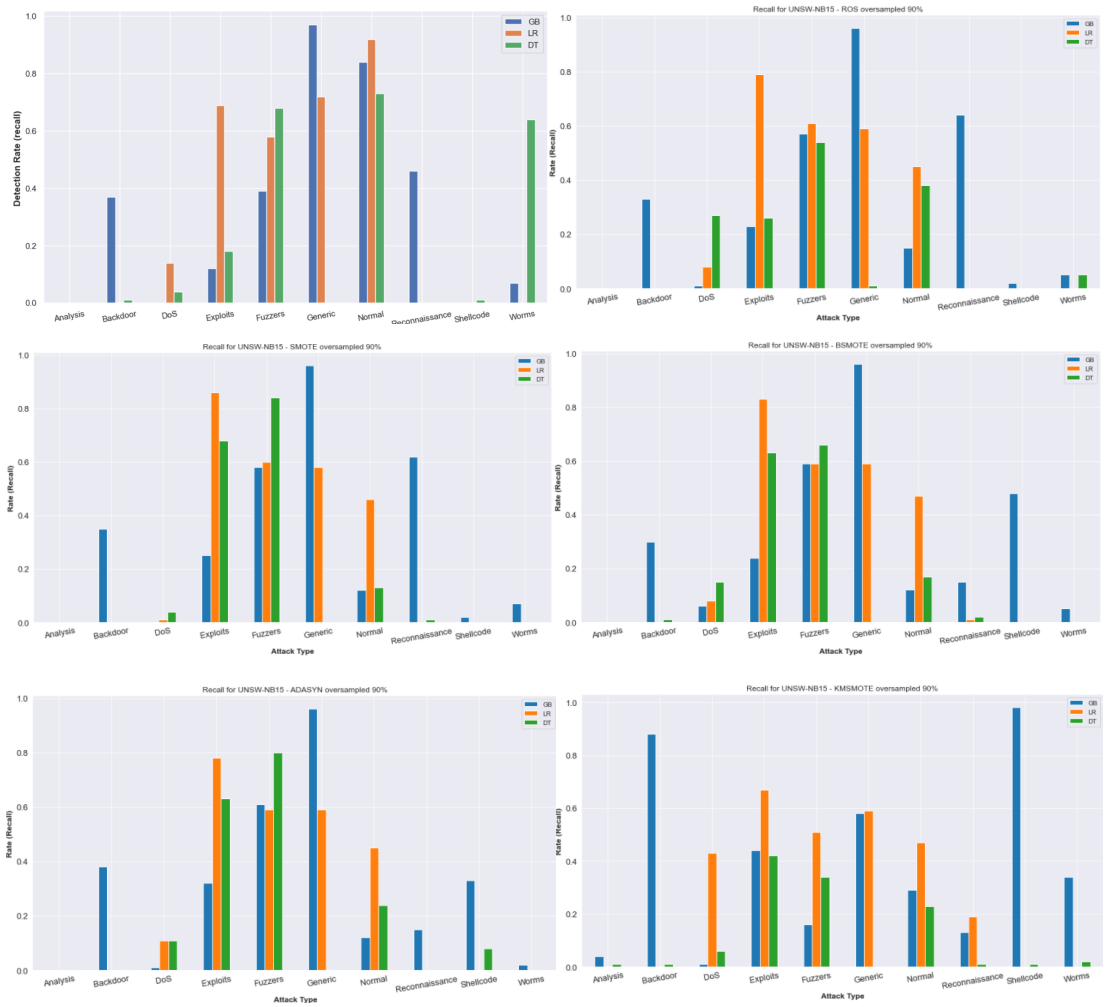


xvii. Oversampled 90% detection rate

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shellcode	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	33	1	23	57	96	15	***64***	2	5
GB SMOTE	0	35	0	25	58	96	12	62	2	7
GB BSMOTE	0	30	6	24	59	96	12	15	48	5
GB ADASYN	0	38	1	32	61	96	12	15	33	2
GB KSMOTE	4	***88***	1	44	16	58	29	13	***98***	34
LR ROS	0	0	8	79	61	59	45	0	0	0
LR SMOTE	0	0	1	***86***	60	58	46	0	0	0
LR BSMOTE	0	0	8	83	59	59	47	1	0	0

LR ADASYN	0	0	11	78	59	59	45	0	0	0
LR KSMOTE	0	0	***43***	67	51	59	47	19	0	0
DT ROS	0	0	27	26	54	1	38	0	0	5
DT SMOTE	0	0	4	68	84	0	13	1	0	0
DT BSMOTE	0	1	15	63	66	0	17	2	0	0
DT ADASYN	0	0	11	63	***80***	0	24	0	8	0
DT KSMOTE	1	1	6	42	34	0	23	1	1	2

xviii. Oversampled 90% comparison results

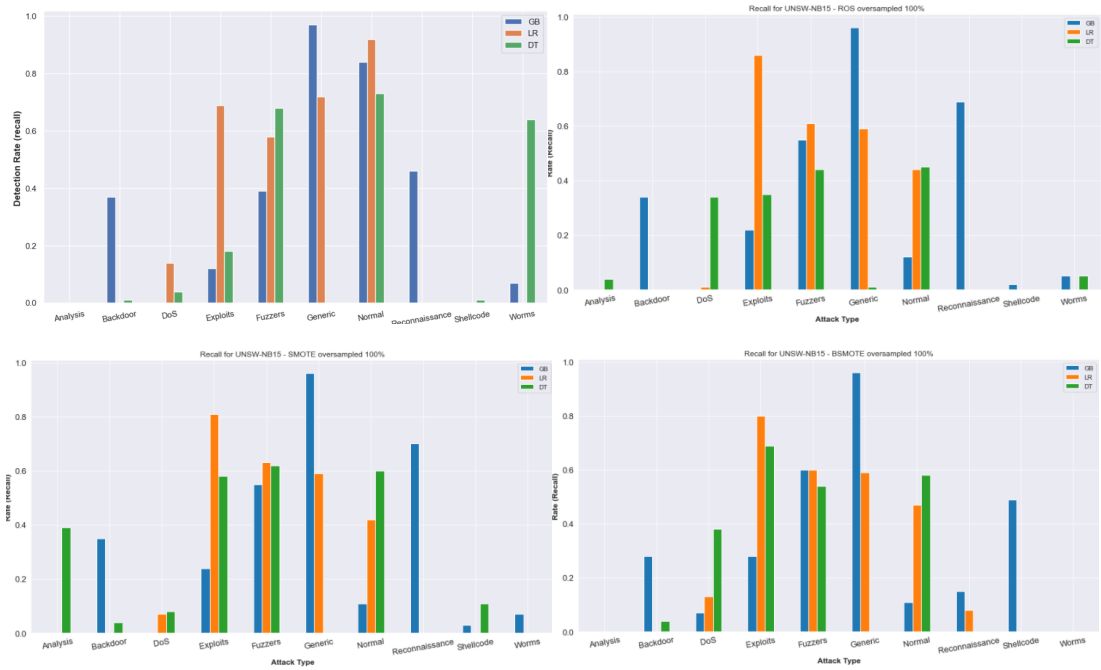


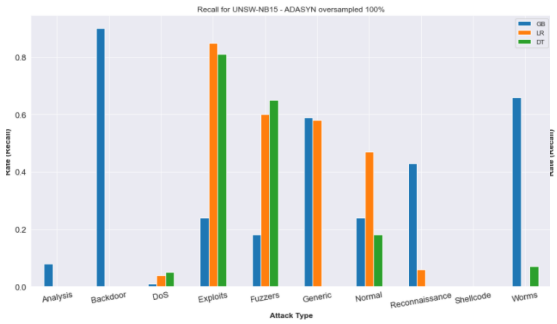
xix. Oversampled 100% detection rate

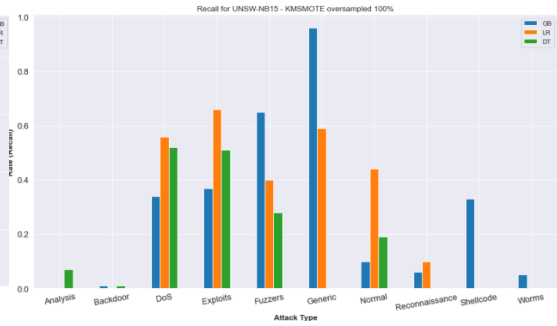
Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shellcode	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0

DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	34	0	22	55	96	12	***69***	2	5
GB SMOTE	0	35	0	24	55	96	11	7	3	7
GB BSMOTE	0	28	7	28	6	96	11	15	***49***	0
GB ADASYN	8	***90***	1	24	18	59	24	43	0	***66***
GB KSMOTE	0	1	34	37	65	96	10	6	33	5
LR ROS	0	0	1	***86***	61	59	44	0	0	0
LR SMOTE	0	0	7	81	63	59	42	0	0	0
LR BSMOTE	0	0	13	8	6	59	47	8	0	0
LR ADASYN	0	0	4	85	60	58	47	6	0	0
LR KSMOTE	0	0	***56***	66	40	59	44	10	0	0
DT ROS	4	0	34	35	44	1	45	0	0	5
DT SMOTE	***39***	4	8	58	62	0	6	0	11	0
DT BSMOTE	0	4	38	69	54	0	58	0	0	0
DT ADASYN	0	0	5	81	65	0	18	0	0	7
DT KSMOTE	7	1	52	51	28	0	19	0	0	0

xx. Oversampled 100% comparison results





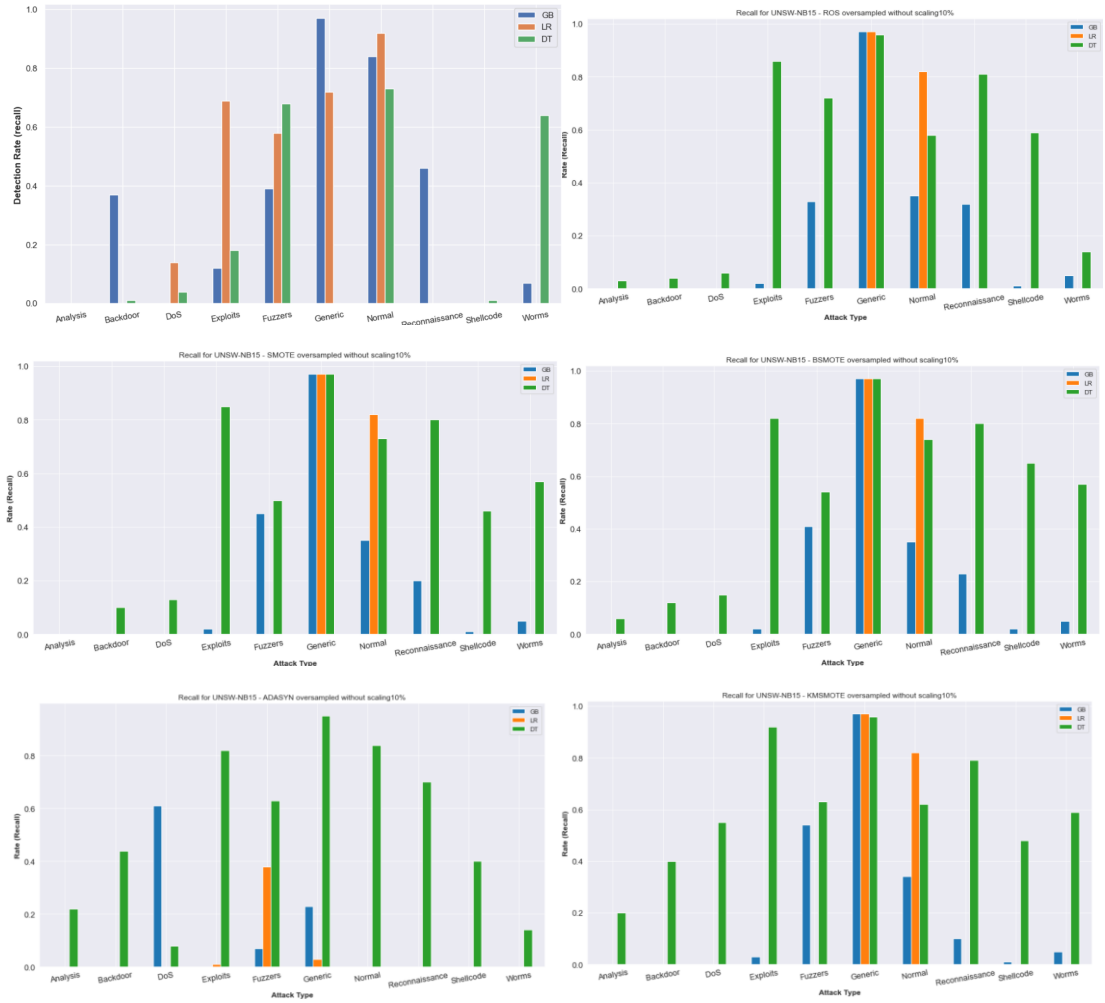


APPENDIX G: UNSW-NB15 oversampling individual DR on non-scaled data

i. Oversampled 10% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	2	33	97	35	32	1	5
GB SMOTE	0	0	0	2	45	97	35	20	1	5
GB BSMOTE	0	0	0	2	41	97	35	23	2	5
GB ADASYN	0	0	***61***	0	7	23	0	0	0	0
GB KSMOTE	0	0	0	3	54	97	34	10	1	5
LR ROS	0	0	0	0	0	97	82	0	0	0
LR SMOTE	0	0	0	0	0	97	82	0	0	0
LR BSMOTE	0	0	0	0	0	97	82	0	0	0
LR ADASYN	0	0	0	1	38	3	0	0	0	0
LR KSMOTE	0	0	0	0	0	97	82	0	0	0
DT ROS	3	4	6	86	***72***	96	58	***81***	59	14
DT SMOTE	0	10	13	85	50	97	73	80	46	57
DT BSMOTE	6	12	15	82	54	97	74	80	***65***	57
DT ADASYN	***22***	***44***	8	82	63	95	81	70	40	14
DT KSMOTE	20	40	55	***92***	63	96	62	79	48	66

ii. Oversampled 10% comparison results – non-scaled

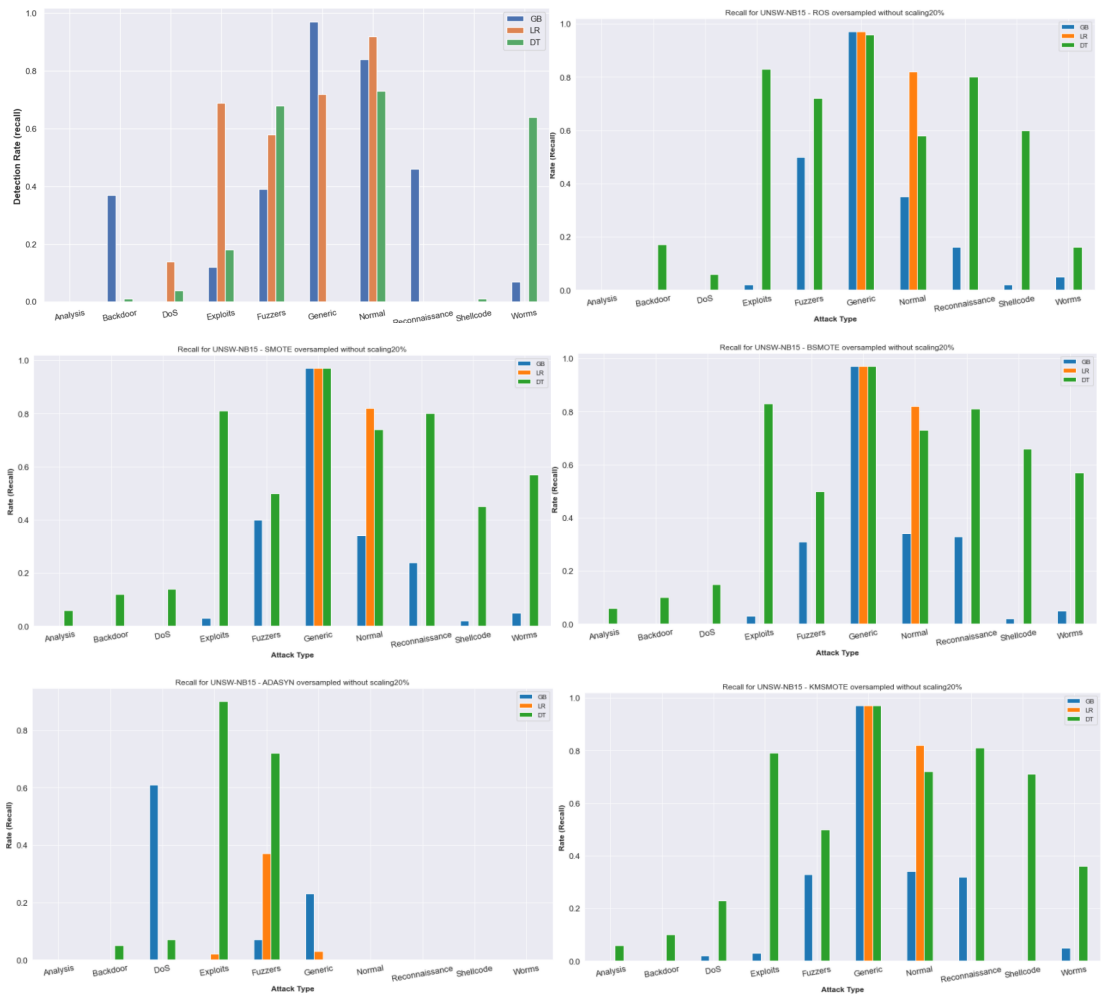


iii. Oversampled 20% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	2	50	97	35	16	2	5
GB SMOTE	0	0	0	3	40	97	34	24	2	5
GB BSMOTE	0	0	0	3	31	97	34	33	2	5
GB ADASYN	0	0	***61***	0	7	23	0	0	0	0
GB KSMOTE	0	0	2	3	33	97	94	32	0	5
LR ROS	0	0	0	0	0	97	82	0	0	0
LR SMOTE	0	0	0	0	0	97	82	0	0	0

LR BSMOTE	0	0	0	0	0	97	82	0	0	0
LR ADASYN	0	0	0	2	37	3	0	0	0	0
LR KSMOTE	0	0	0	**0**	0	97	82	0	0	0
DT ROS	0	17	6	83	***72***	96	58	80	60	16
DT SMOTE	***6***	12	14	81	50	97	74	80	45	57
DT BSMOTE	***6***	10	15	83	50	97	73	***81***	66	57
DT ADASYN	0	5	7	***90***	***72***	0	0	0	0	0
DT KSMOTE	***6***	10	23	79	50	97	72	***81***	***71***	36

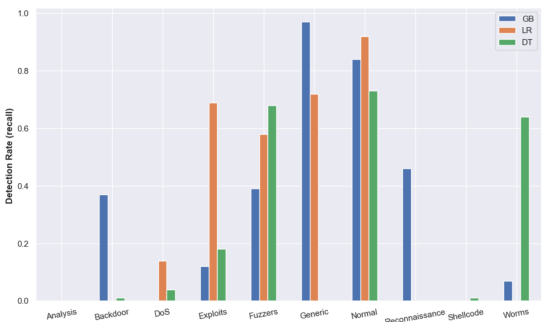
iv. Oversampled 20% comparison results – non-scaled

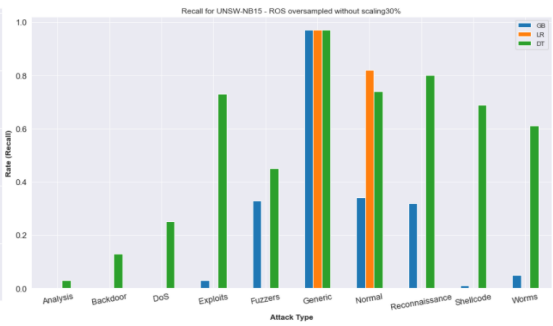


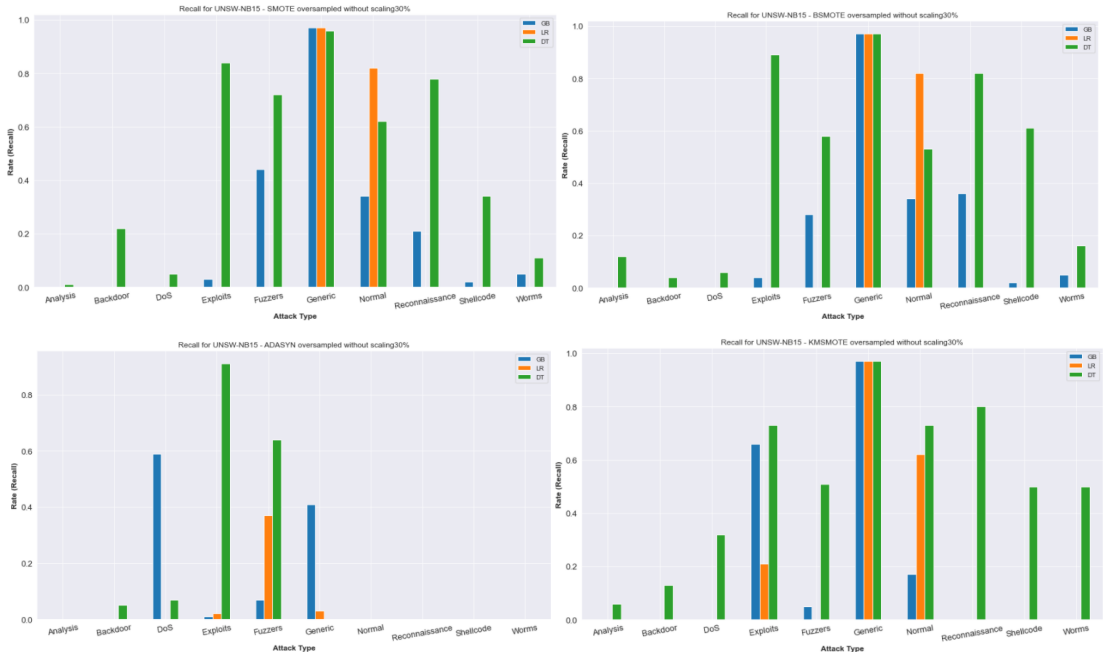
v. Oversampled 30% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	3	33	97	34	32	1	5
GB SMOTE	0	0	0	3	44	97	34	21	2	5
GB BSMOTE	0	0	0	4	28	97	34	36	2	5
GB ADASYN	0	0	***59***	1	7	41	0	0	0	0
GB KSMOTE	0	0	0	66	5	97	17	0	0	0
LR ROS	0	0	0	0	0	97	82	0	0	0
LR SMOTE	0	0	0	0	0	97	82	0	0	0
LR BSMOTE	0	0	0	0	0	97	82	0	0	0
LR ADASYN	0	0	0	2	37	3	0	0	0	0
LR KSMOTE	0	0	0	21	0	97	62	0	0	0
DT ROS	3	13	25	73	45	97	74	80	69	61
DT SMOTE	1	22	5	84	***72***	96	62	78	34	11
DT BSMOTE	***12***	4	6	89	58	97	53	***82***	***61***	16
DT ADASYN	0	5	7	***91***	64	0	0	0	0	0
DT KSMOTE	6	13	32	73	51	97	73	80	50	50

vi. Oversampled 30% comparison results – non-scaled



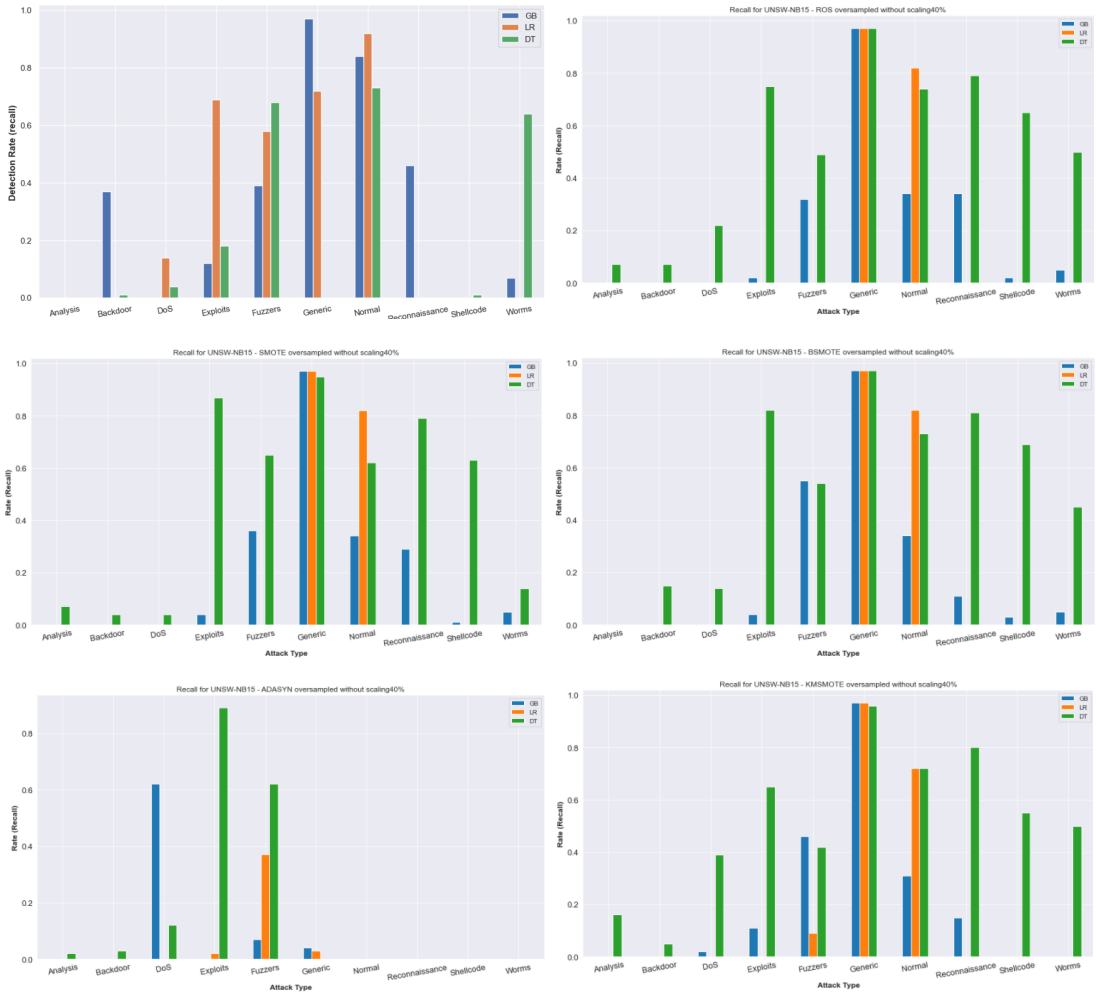




vii. Oversampled 40% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	2	32	97	34	34	2	5
GB SMOTE	0	0	0	4	36	97	34	29	1	5
GB BSMOTE	0	0	0	4	55	97	34	11	3	5
GB ADASYN	0	0	***62***	0	7	4	0	0	0	0
GB KSMOTE	0	0	2	11	46	97	31	15	0	0
LR ROS	0	0	0	0	0	97	82	0	0	0
LR SMOTE	0	0	0	0	0	97	82	0	0	0
LR BSMOTE	0	0	0	0	0	97	82	0	0	0
LR ADASYN	0	0	0	2	37	3	0	0	0	0
LR KSMOTE	0	0	0	0	9	97	72	0	0	0
DT ROS	7	7	22	75	49	97	74	79	65	50
DT SMOTE	7	4	4	87	65	95	62	79	63	14
DT BSMOTE	0	15	14	82	54	97	73	***81***	***69***	45
DT ADASYN	2	3	12	***89***	62	0	0	0	0	0
DT KSMOTE	***16***	5	39	65	42	96	72	80	55	50

viii. Oversampled 40% comparison results – non-scaled

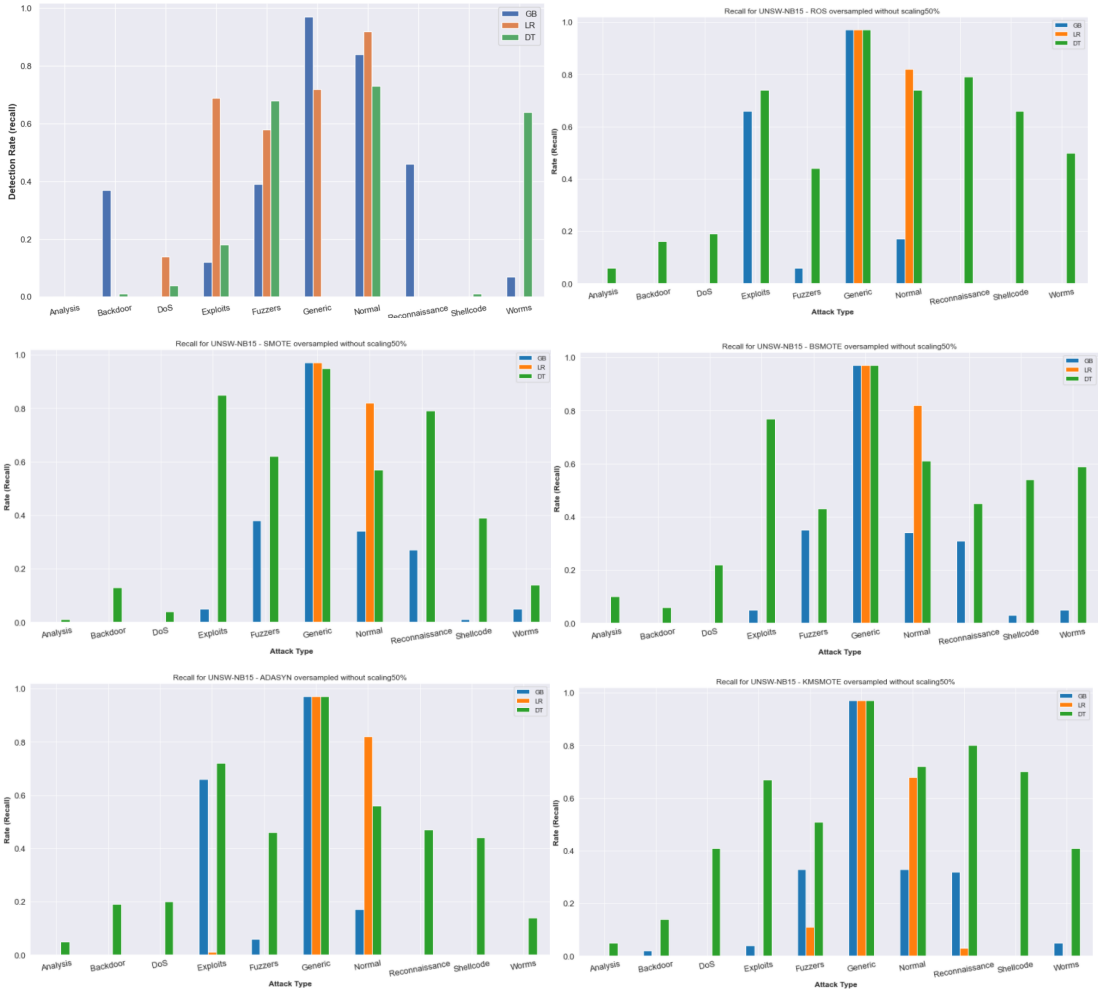


ix. Oversampled 50% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	66	6	97	17	0	0	0
GB SMOTE	0	0	0	5	38	97	34	27	1	5
GB BSMOTE	0	0	0	5	35	97	34	31	3	5
GB ADASYN	0	0	0	66	6	97	17	0	0	0
GB KSMOTE	0	2	0	4	33	97	33	32	0	5
LR ROS	0	0	0	0	0	97	82	0	0	0
LR SMOTE	0	0	0	0	0	97	82	0	0	0

LR BSMOTE	0	0	0	0	0	97	82	0	0	0
LR ADASYN	0	0	0	1	0	97	82	0	0	0
LR KSMOTE	0	0	0	0	11	97	68	3	0	0
DT ROS	6	16	19	74	44	97	74	79	66	50
DT SMOTE	1	13	4	***85***	62	95	57	79	39	14
DT BSMOTE	***10***	6	22	77	43	97	61	45	54	66
DT ADASYN	5	19	20	72	46	97	56	47	44	14
DT KSMOTE	5	14	***41***	67	51	97	72	***80***	***70***	41

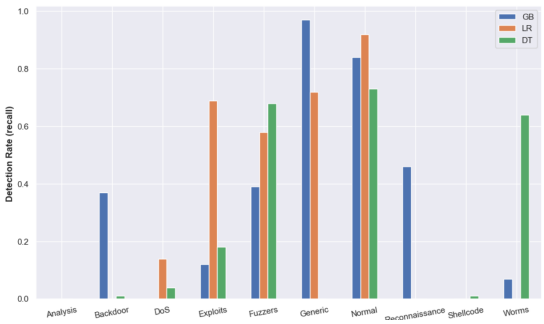
x. Oversampled 50% comparison results – non-scaled

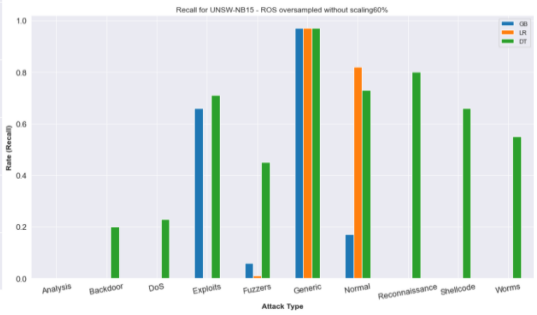


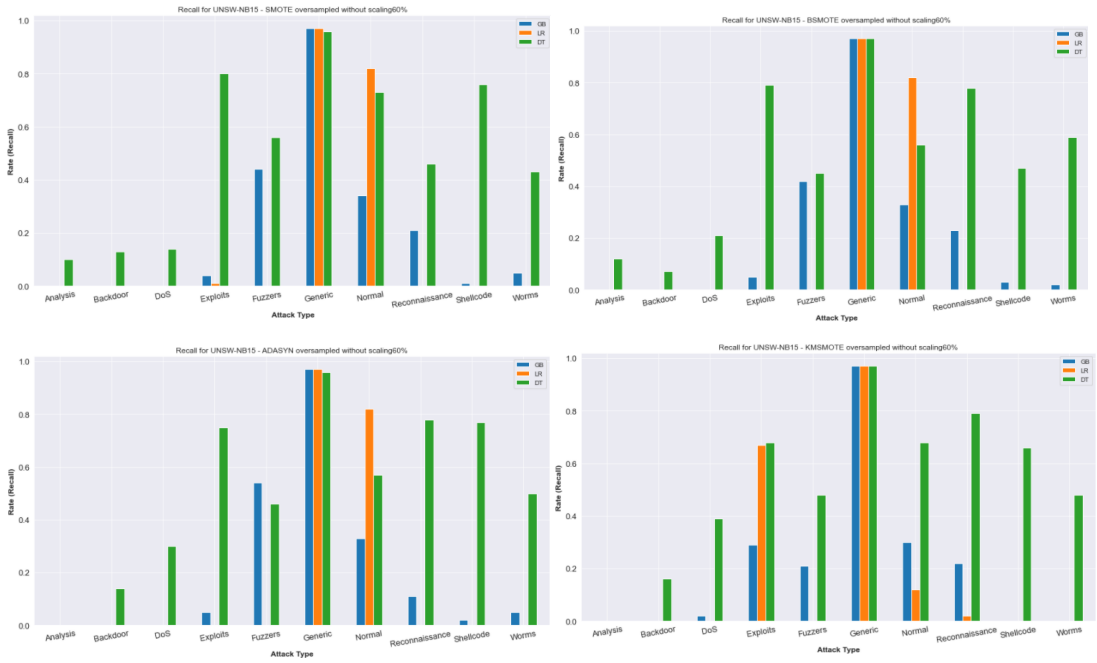
xi. Oversampled 60% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	66	6	97	17	0	0	0
GB SMOTE	0	0	0	4	44	97	34	21	1	5
GB BSMOTE	0	0	0	5	42	97	33	23	3	2
GB ADASYN	0	0	0	5	54	97	33	11	2	5
GB KSMOTE	0	0	2	29	21	97	30	22	0	0
LR ROS	0	0	0	0	1	97	82	0	0	0
LR SMOTE	0	0	0	1	0	97	82	0	0	0
LR BSMOTE	0	0	0	0	0	97	82	0	0	0
LR ADASYN	0	0	0	0	0	97	82	0	0	0
LR KSMOTE	0	0	0	67	0	97	12	2	0	0
DT ROS	0	20	23	71	45	97	73	***80***	66	55
DT SMOTE	10	13	14	***80***	56	96	73	46	76	43
DT BSMOTE	***12***	7	21	79	45	97	56	78	47	66
DT ADASYN	0	14	30	75	46	96	57	78	***77***	50
DT KSMOTE	0	16	***39***	68	48	97	68	79	66	48

xii. Oversampled 60% comparison results – non-scaled





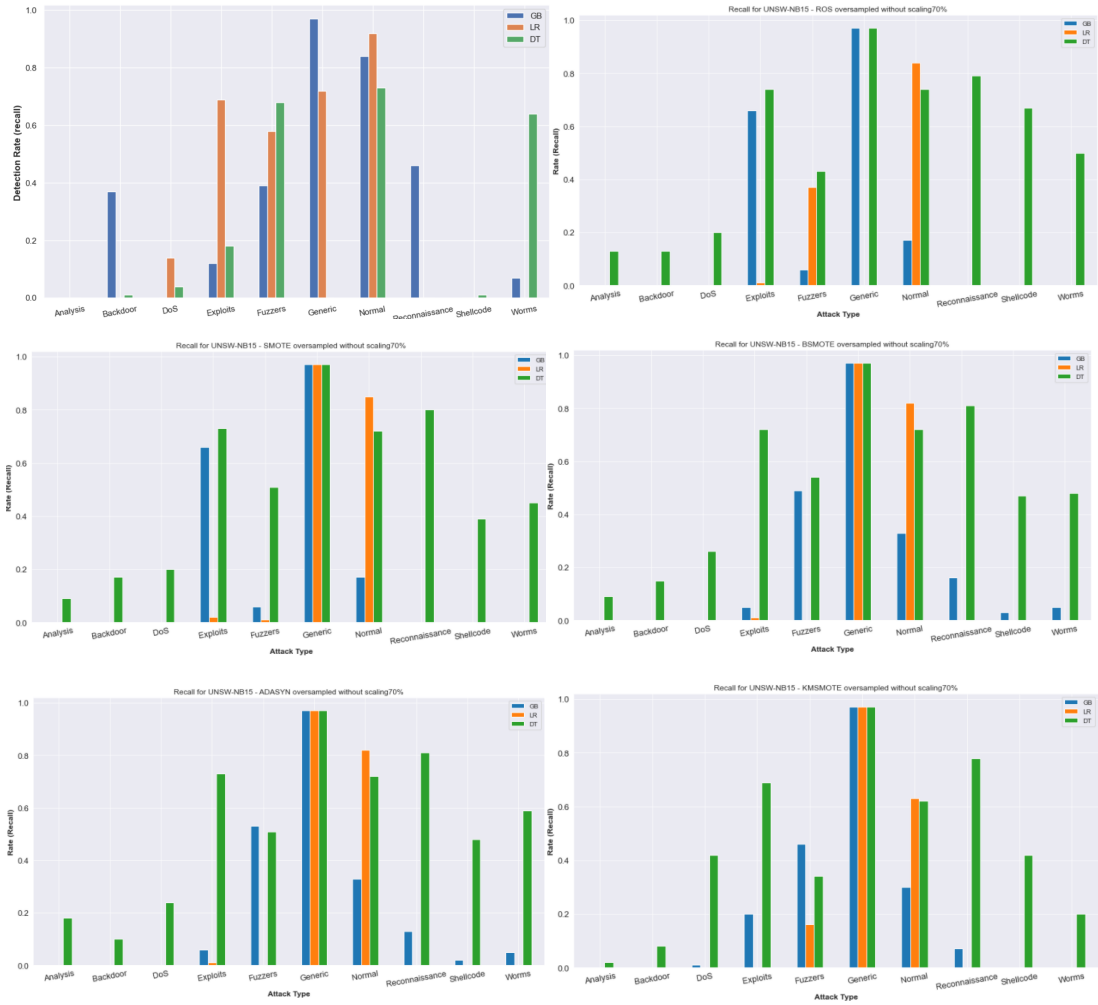


xiii. Oversampled 70% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	66	6	97	17	0	0	0
GB SMOTE	0	0	0	66	6	97	17	0	0	0
GB BSMOTE	0	0	0	5	49	97	33	16	3	5
GB ADASYN	0	0	0	6	53	97	33	13	2	5
GB KSMOTE	0	0	1	20	46	97	30	7	0	0
LR ROS	0	0	0	1	37	0	84	0	0	0
LR SMOTE	0	0	0	2	1	97	85	0	0	0
LR BSMOTE	0	0	0	1	0	97	82	0	0	0
LR ADASYN	0	0	0	1	0	97	82	0	0	0
LR KSMOTE	0	0	0	0	16	97	63	0	0	0
DT ROS	13	13	20	***74***	43	97	74	79	***67***	50
DT SMOTE	9	17	20	73	51	97	72	80	39	45
DT BSMOTE	9	15	26	72	54	97	72	***81***	47	48
DT ADASYN	***18***	10	24	73	51	97	72	***81***	48	66

DT KSMOTE	2	8	***42***	69	34	97	62	78	42	20

xiv. Oversampled 70% comparison results – non-scaled

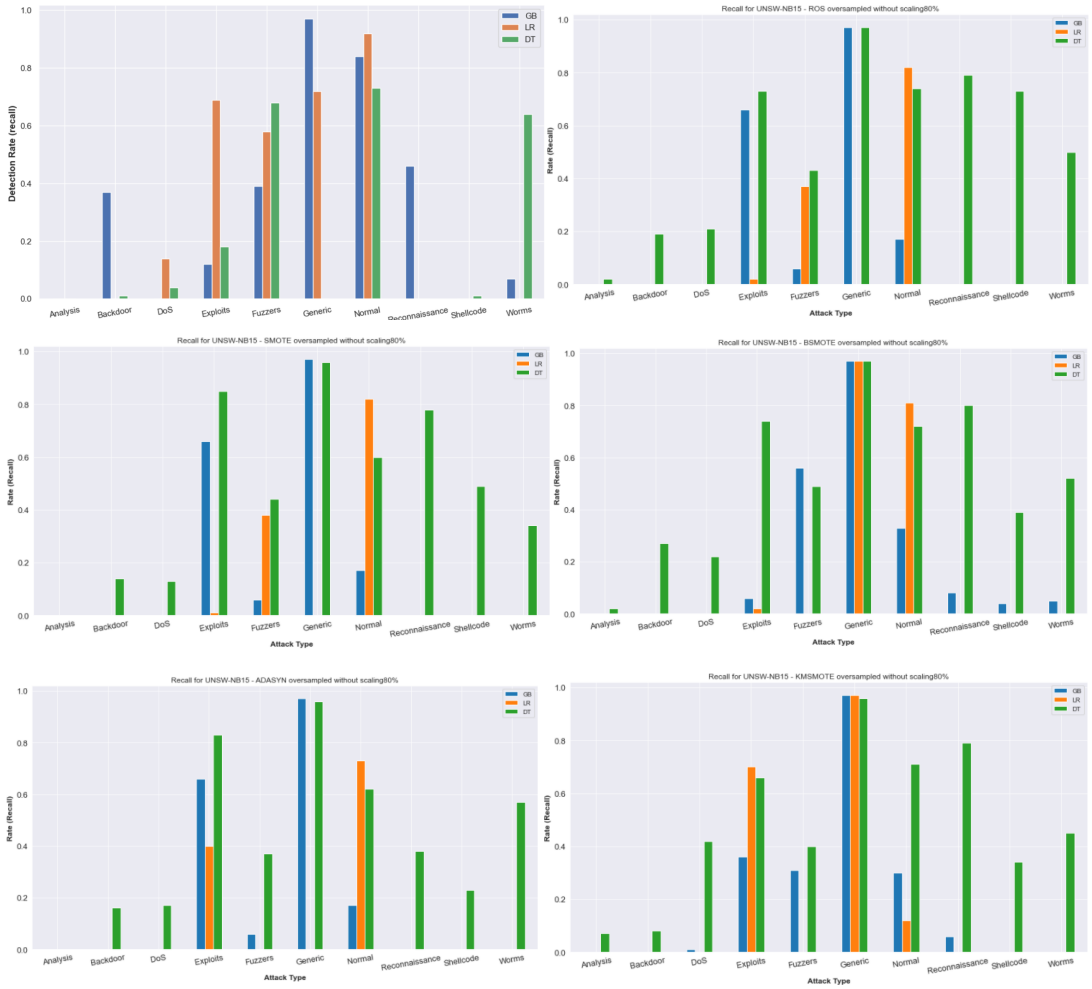


xv. Oversampled 80% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	66	6	97	17	0	0	0
GB SMOTE	0	0	0	66	6	97	17	0	0	0
GB BSMOTE	0	0	0	6	56	97	33	8	4	5
GB ADASYN	0	0	0	66	6	97	17	0	0	0
GB KSMOTE	0	0	1	36	31	97	30	6	0	0

LR ROS	0	0	0	2	37	0	82	0	0	0
LR SMOTE	0	0	0	1	38	0	82	0	0	0
LR BSMOTE	0	0	0	2	0	97	81	0	0	0
LR ADASYN	0	0	0	40	0	0	73	0	0	0
LR KSMOTE	0	0	0	70	0	97	12	0	0	0
DT ROS	2	19	21	73	43	97	74	79	***73***	50
DT SMOTE	0	14	13	***85***	44	96	60	78	49	34
DT BSMOTE	2	27	22	74	49	97	72	***80***	39	52
DT ADASYN	0	16	17	83	37	96	62	38	23	57
DT KSMOTE	7	8	***42***	66	40	96	71	79	34	45

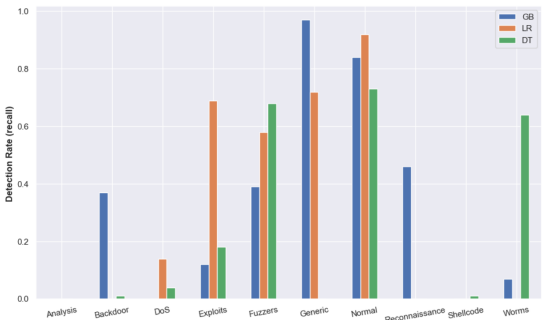
xvi. Oversampled 80% comparison results – non-scaled

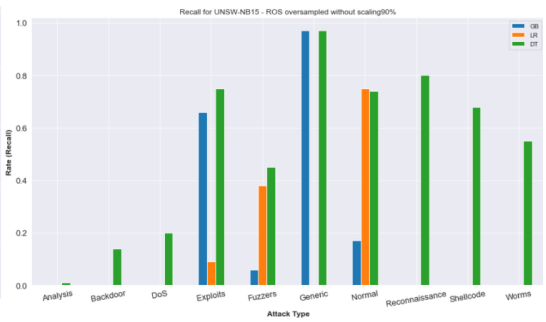


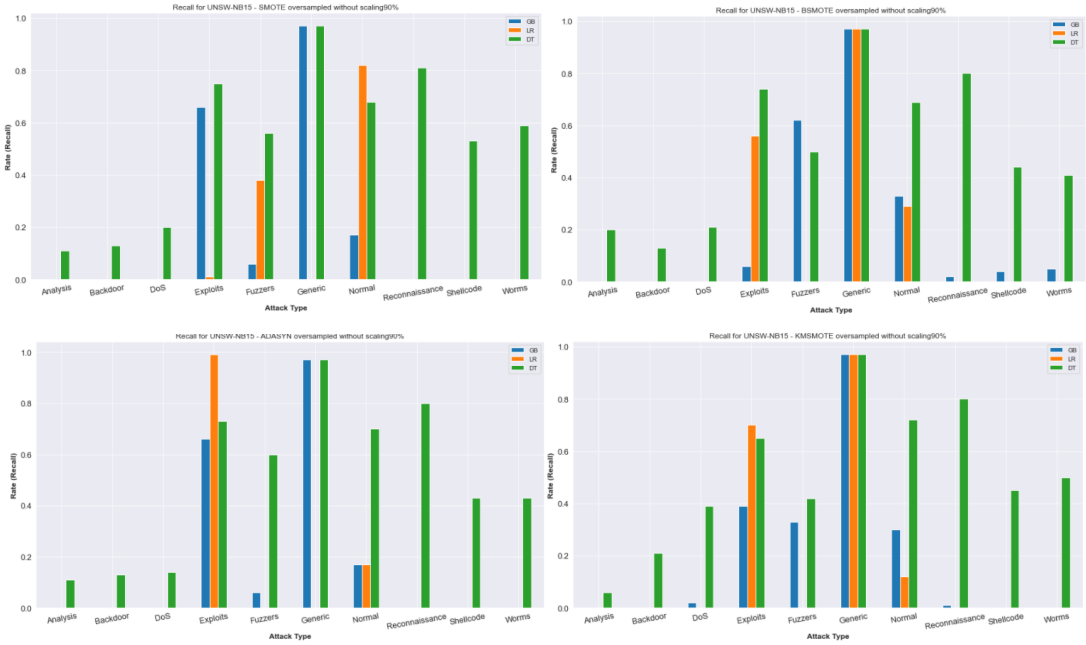
xvii. Oversampled 90% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	66	6	97	17	0	0	0
GB SMOTE	0	0	0	66	6	97	17	0	0	0
GB BSMOTE	0	0	0	6	62	97	33	2	4	5
GB ADASYN	0	0	0	66	6	97	17	0	0	0
GB KSMOTE	0	0	2	39	33	97	30	1	0	0
LR ROS	0	0	0	9	38	0	75	0	0	0
LR SMOTE	0	0	0	1	38	0	82	0	0	0
LR BSMOTE	0	0	0	56	0	97	29	0	0	0
LR ADASYN	0	0	0	***99***	0	0	17	0	0	0
LR KSMOTE	0	0	0	70	0	97	12	0	0	0
DT ROS	1	14	20	75	45	97	74	80	***68***	55
DT SMOTE	11	13	20	75	56	97	68	***81***	53	66
DT BSMOTE	***20***	13	21	74	50	97	69	80	44	41
DT ADASYN	11	13	14	73	60	97	70	80	43	43
DT KSMOTE	6	21	***39***	65	42	97	72	80	45	50

xviii. Oversampled 90% comparison results – non-scaled



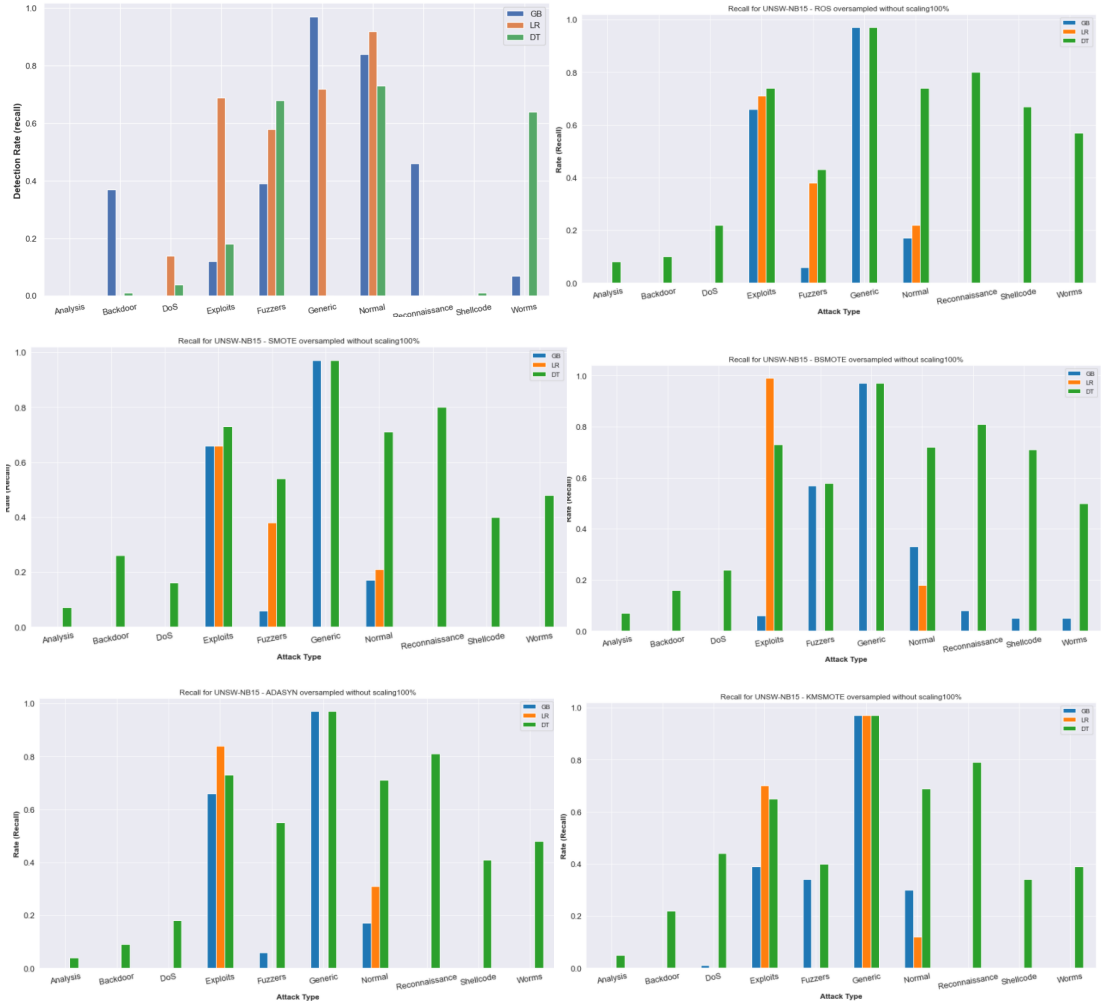




xix. Oversampled 100% detection rate – non-scaled

Result	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Norm	Recon	Shell code	Worms
GB	0	37	0	12	39	97	84	46	0	7
LR	0	0	14	69	58	72	92	0	0	0
DR	0	1	4	18	67	0	73	0	1	64
GB ROS	0	0	0	66	6	97	17	0	0	0
GB SMOTE	0	0	0	66	6	97	17	0	0	0
GB BSMOTE	0	0	0	6	57	97	33	8	5	5
GB ADASYN	0	0	0	66	6	97	17	0	0	0
GB KSMOTE	0	0	1	39	34	97	30	0	0	0
LR ROS	0	0	0	71	38	0	22	0	0	0
LR SMOTE	0	0	0	66	38	0	21	0	0	0
LR BSMOTE	0	0	0	***99***	0	0	18	0	0	0
LR ADASYN	0	0	0	84	0	0	31	0	0	0
LR KSMOTE	0	0	0	70	0	97	12	0	0	0
DT ROS	***8***	10	22	74	43	97	74	80	67	57
DT SMOTE	7	26	16	73	54	97	71	80	40	48
DT BSMOTE	7	16	24	73	58	97	72	***81***	***71***	50
DT ADASYN	4	9	18	73	55	97	71	***81***	41	48
DT KSMOTE	5	22	***44***	65	40	97	69	79	34	39

xx. Oversampled 100% comparison results – non-scaled

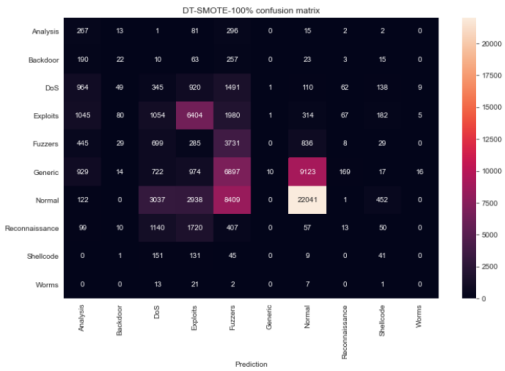


APPENDIX H: Top three detection rate of each minority classes for UNSW-NB15.

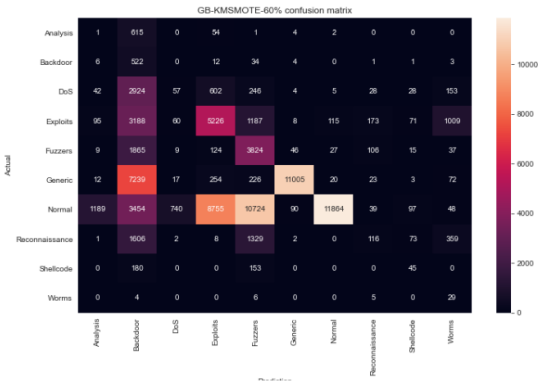
	ANALYSIS		BACKDOOR		DOS		EXPLOITS	
	Combination	**DR**	**Combination**	**DR**	**Combination**	**DR**	**Combination**	**DR**
	Original	0	Original	37	Original	14	Original	69
S1	DT-SMOTE-100%	***39***	GB-BSMOTE-40% GB-SMOTE-60% GB-KMSMOTE-60% GB-ADASYN-60, 70%, 80%, 100%	***90***	DT-ADASYN-10%	***77***	DT-BSMOTE-10%	93
S2	DT-BSMOTE-80%	27	GB-SMOTE-30%, 50%,70% GB-ROS-50% GB-KMSMOTE-50%, 70%, 80%	89	DT-ADASYN-30%	73	LR-ADASYN-80%	87
S3	DT-ROS-20%	22	GB-BSMOTE-60%, 80% GB-KMSMOTE-90%	88	DT-ADASYN-40%	71	LR-SMOTE-90% LR-ROS-100%	86
NS1	DT-ADASYN-10%	22	DT-ADASYN-10%	44	GB-ADASYN-40%	62	LR-ADASYN-90% LR-BSMOTE-100%	***99***
NS2	DT-KMSMOTE-10% DT-BSMOTE-90%	20	DT-KMSMOTE-10%	40	GB-ADASYN-10%, 20%	61	DT-KMSMOTE-10%	92
NS3	DT-ADASYN-70%	18			GB-ADASYN-30%	59	DT-ADASYN-30%	91
	FUZZERS		**RECONNAISSANCE**		**SHELLCODE**		**WORMS**	
	Combination	**DR**	**Combination**	**DR**	**Combination**	**DR**	**Combination**	**DR**
	Original	67	Original	46	Original	1	Original	7

S1	GB-ADASYN-10%	***91***	GB-ROS-100%	69	GB-KMSMOTE-40% GB-SMOTE-70%	***99***	GB-KMSMOTE-60% GB-ADASYN-100%	***66***
S2	DT-SMOTE-30%	88	GB-SMOTE-80%	67	GB-KMSMOTE-70%, 80%, 90%	98		
S3	GB-ADASYN-30%, 40% DT-BSMOTE-40%	87	GB-ROS-80%, 90%	64	GB-BSMOTE-60%, 80%	97		
NS1	DT-ROS-10%, 20% DT-ADASYN-20% DT-SMOTE-30%	72	DT-BSMOTE-30%	***82***	DT-ADASYN-60%	77	DT-KMSMOTE-10% DT-BSMOTE-50%, 60% DT-ADASYN-70% DT-SMOTE-90%	66
NS2			DT-ROS-10% DT-BSMOTE-20%, 40%, 100% DT-KMSMOTE-20% DT-SMOTE-70%, 90% DT-ADASYN-70%, 100%	81	DT-SMOTE-60%	76	DT-SMOTE-10%, 20% DT-BSMOTE-10%, 20% DT-ADASYN-80% DT-ROS-100%	57
NS3			DT-SMOTE-10%, 20% , 70% DT-BSMOTE-10%, 80% DT-ROS-20%, 30% , 60% DT-KMSMOTE-30%, 50% DT-All-90%	80	DT-ROS-80%	73	DT-ROS-60%, 90%	55

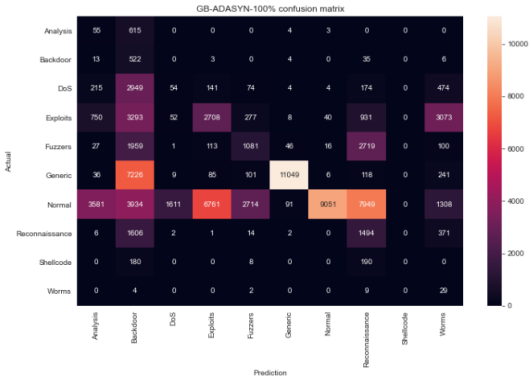
APPENDIX I: UNSW-NB15 reference confusion matrix



(a)



(b)



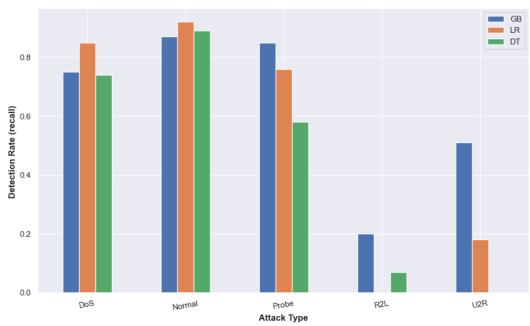
(c)

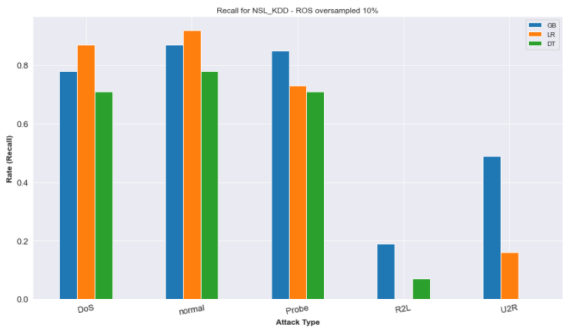
APPENDIX J: NSL KDD oversampling individual DR on scaled data

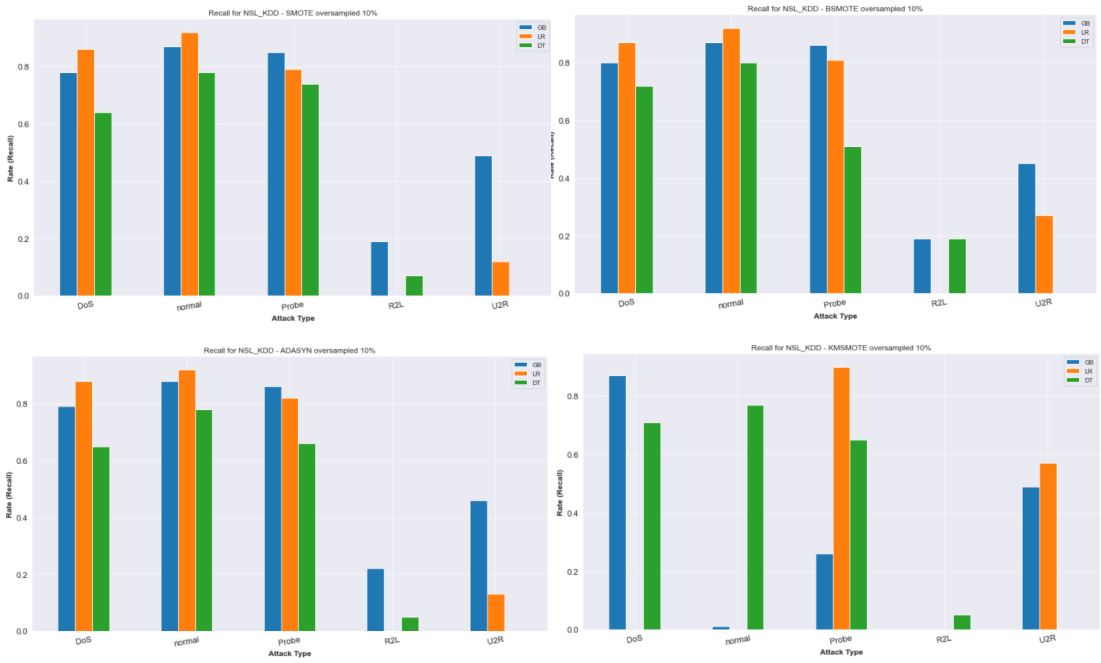
i. Oversampled 10% detection rate

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	78	85	19	49	87
GB - SMOTE	78	85	19	49	87
GB - BSMOTE	80	86	19	45	87
GB - ADASYN	79	86	***22***	46	88
GB - KSMOTE	87	26	0	49	1
LR - ROS	87	73	0	16	92
LR - SMOTE	86	79	0	12	92
LR - BSMOTE	87	81	0	27	92
LR - ADASYN	***88***	82	0	13	92
LR - KSMOTE	0	***90***	0	***57***	0
DT - ROS	71	71	7	0	78
DT - SMOTE	64	74	7	0	78
DT - BSMOTE	72	51	19	0	80
DT - ADASYN	65	66	5	0	78
DT - KSMOTE	71	65	5	0	77

ii. Oversampled 10% comparison results



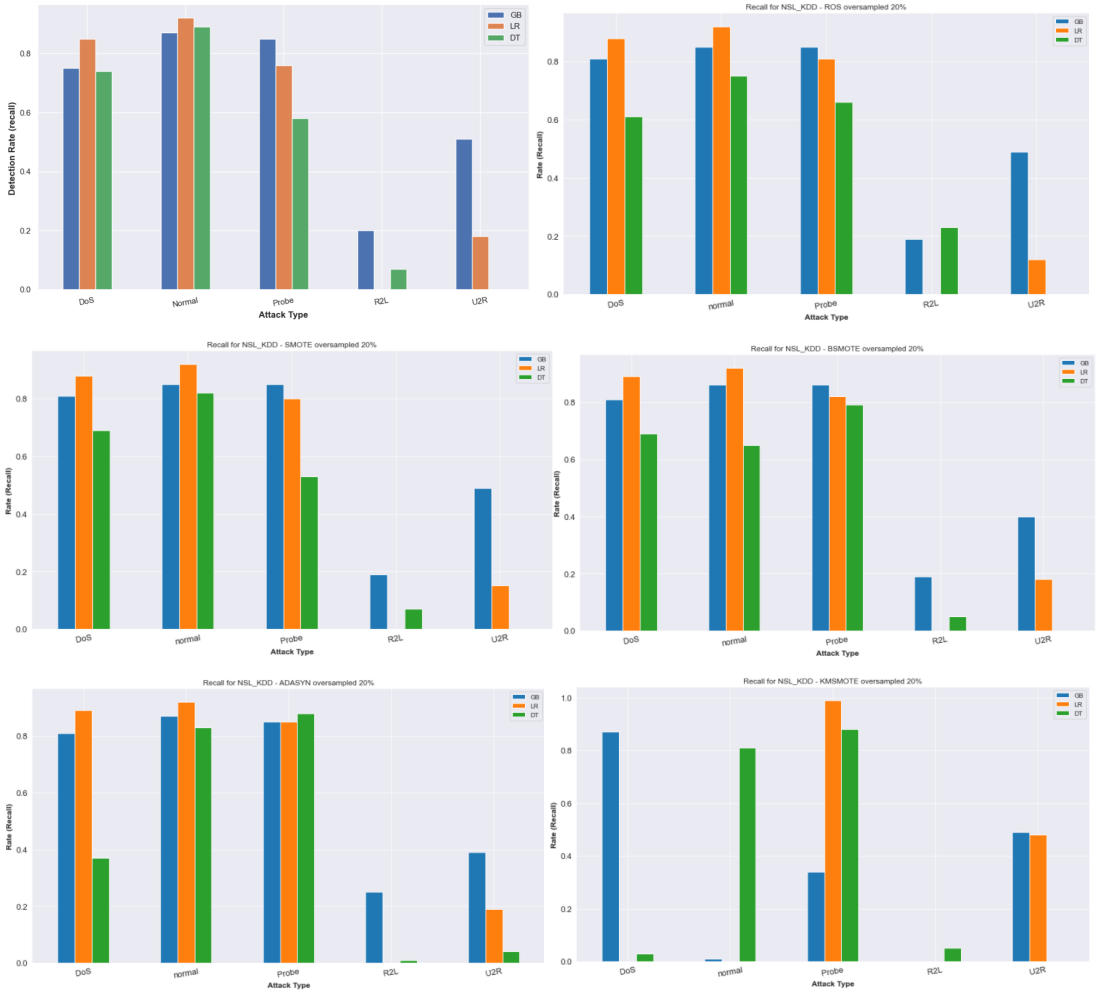




iii. Oversampled 20% detection rate

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	81	85	19	49	85
GB - SMOTE	81	85	19	49	85
GB - BSMOTE	81	86	19	40	86
GB - ADASYN	81	85	***25***	39	87
GB - KSMOTE	87	34	0	49	1
LR - ROS	88	81	0	12	92
LR - SMOTE	88	80	0	15	92
LR - BSMOTE	***89***	82	0	18	92
LR - ADASYN	***89***	85	0	19	92
LR - KSMOTE	0	***99***	0	48	0
DT - ROS	61	66	23	0	75
DT - SMOTE	69	53	7	0	82
DT - BSMOTE	69	79	5	0	65
DT - ADASYN	37	88	1	4	83
DT - KSMOTE	3	88	5	0	81

iv. Oversampled 20% comparison results

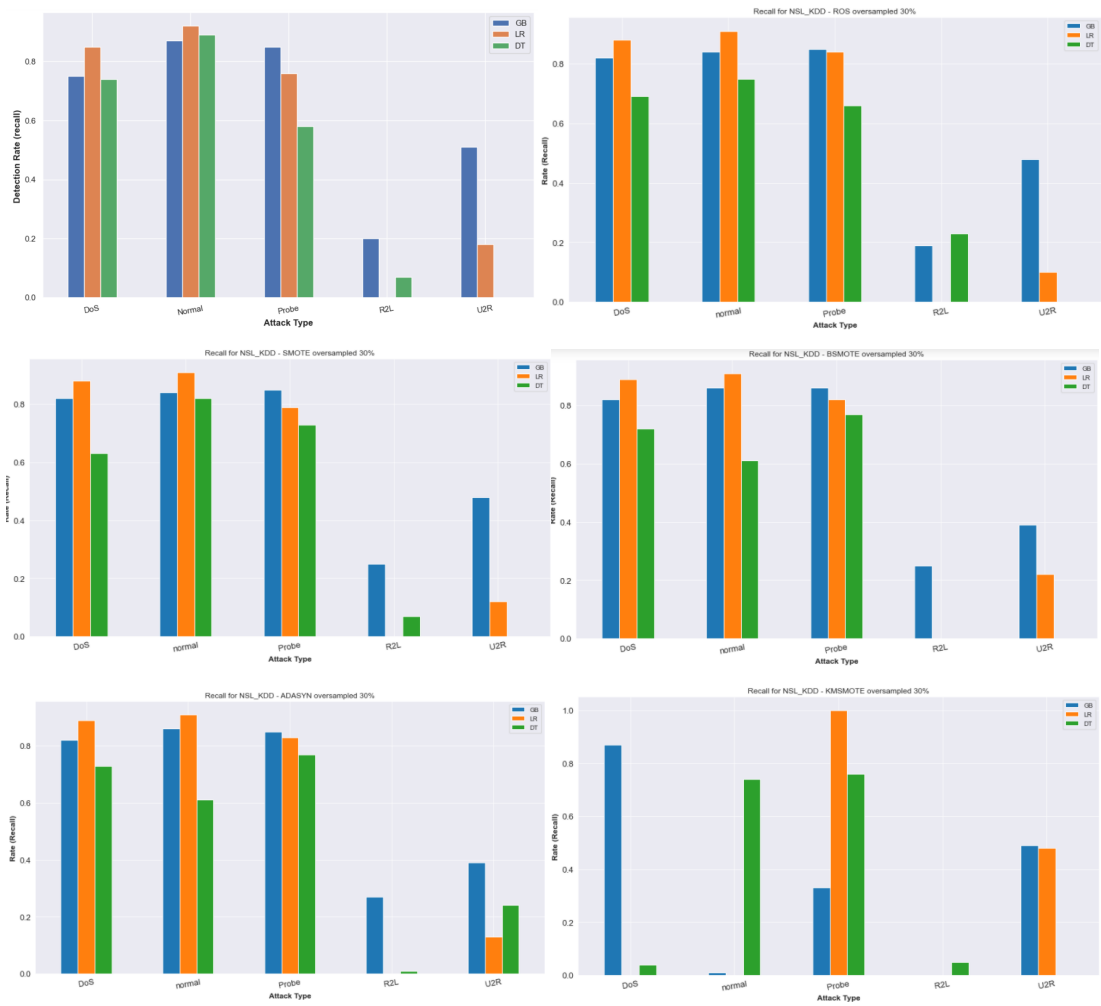


v. Oversampled 30% detection rate

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	82	85	***90***	48	78
GB - SMOTE	82	85	25	48	84
GB - BSMOTE	82	86	25	39	86
GB - ADASYN	82	85	27	39	86
GB - KSMOTE	87	33	0	***49***	1
LR - ROS	88	84	0	10	91
LR - SMOTE	88	79	0	12	91

LR - BSMOTE	**89**	82	0	22	91
LR - ADASYN	**89**	83	0	13	91
LR - KSMOTE	0	***100***	0	48	0
DT - ROS	69	66	23	0	75
DT - SMOTE	69	73	7	0	82
DT - BSMOTE	72	77	0	0	61
DT - ADASYN	73	77	1	24	61
DT - KSMOTE	4	76	5	0	74

vi. Oversampled 30% comparison results

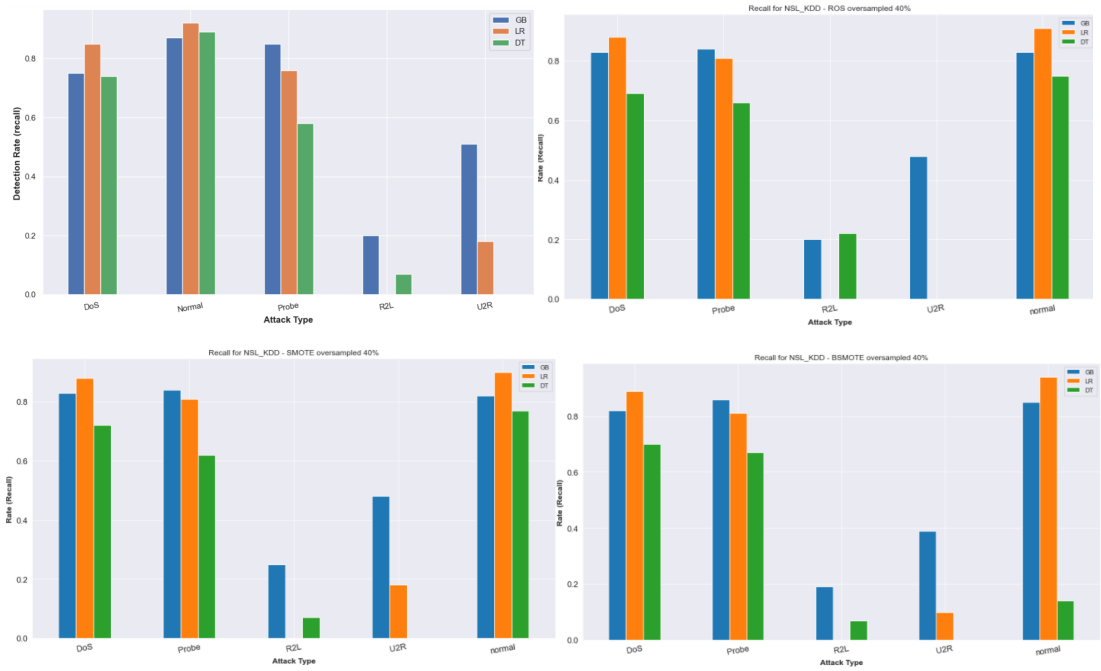


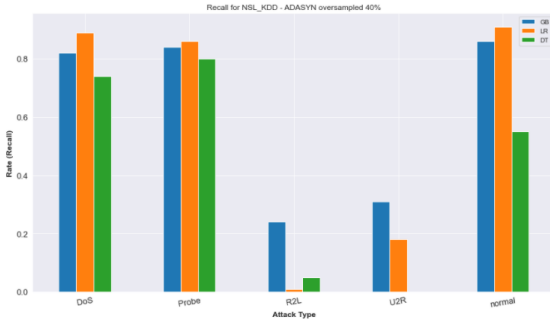
vii. Oversampled 40% detection rate

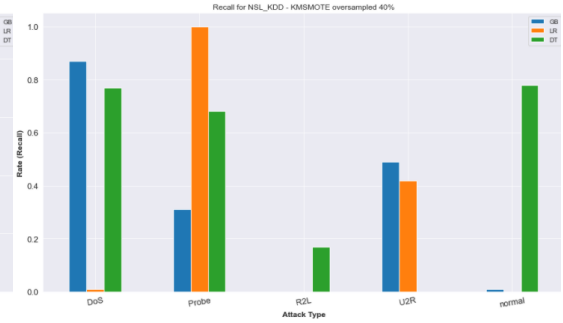
Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87

LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	83	84	20	48	83
GB - SMOTE	83	84	***25***	48	82
GB - BSMOTE	82	86	19	39	85
GB - ADASYN	82	84	24	31	86
GB - KSMOTE	87	31	0	49	1
LR - ROS	88	81	0	0	91
LR - SMOTE	88	81	0	18	90
LR - BSMOTE	***89***	81	0	10	94
LR - ADASYN	***89***	86	1	18	91
LR - KSMOTE	1	***100***	0	42	0
DT - ROS	69	66	22	0	75
DT - SMOTE	72	62	7	0	77
DT - BSMOTE	70	67	7	0	14
DT - ADASYN	74	80	5	0	55
DT - KSMOTE	77	68	17	0	78

viii. Oversampled 40% comparison results



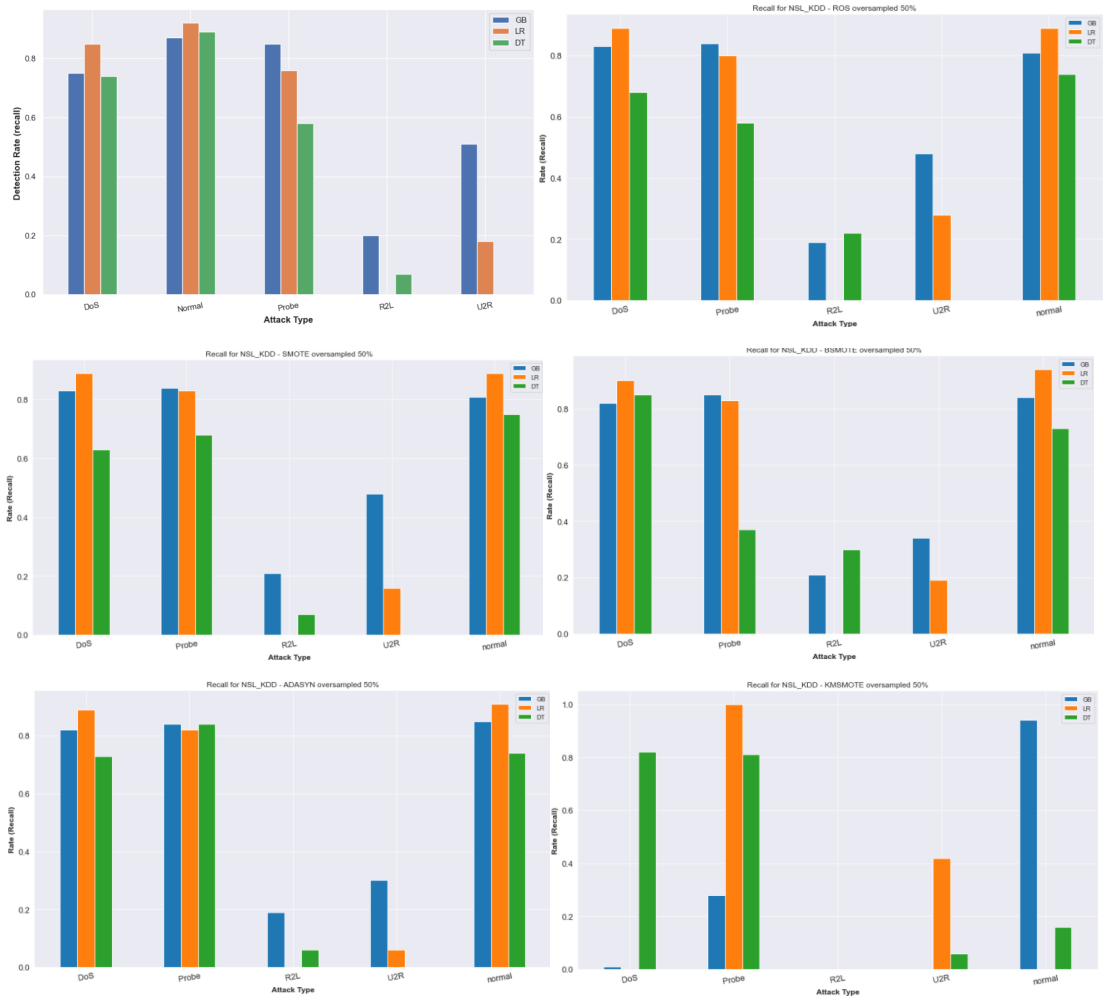




ix. Oversampled 50% detection rate

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	83	84	19	48	81
GB - SMOTE	83	84	21	48	81
GB - BSMOTE	82	85	21	34	84
GB - ADASYN	82	84	19	30	85
GB - KSMOTE	1	28	0	0	94
LR - ROS	89	80	0	28	89
LR - SMOTE	89	83	0	16	89
LR - BSMOTE	***90***	83	0	19	94
LR - ADASYN	89	82	0	6	91
LR - KSMOTE	0	***100***	0	42	0
DT - ROS	68	58	22	0	74
DT - SMOTE	63	68	7	0	75
DT - BSMOTE	85	37	***30***	0	73
DT - ADASYN	73	84	6	0	74
DT - KSMOTE	82	81	0	6	16

x. Oversampled 50% comparison results

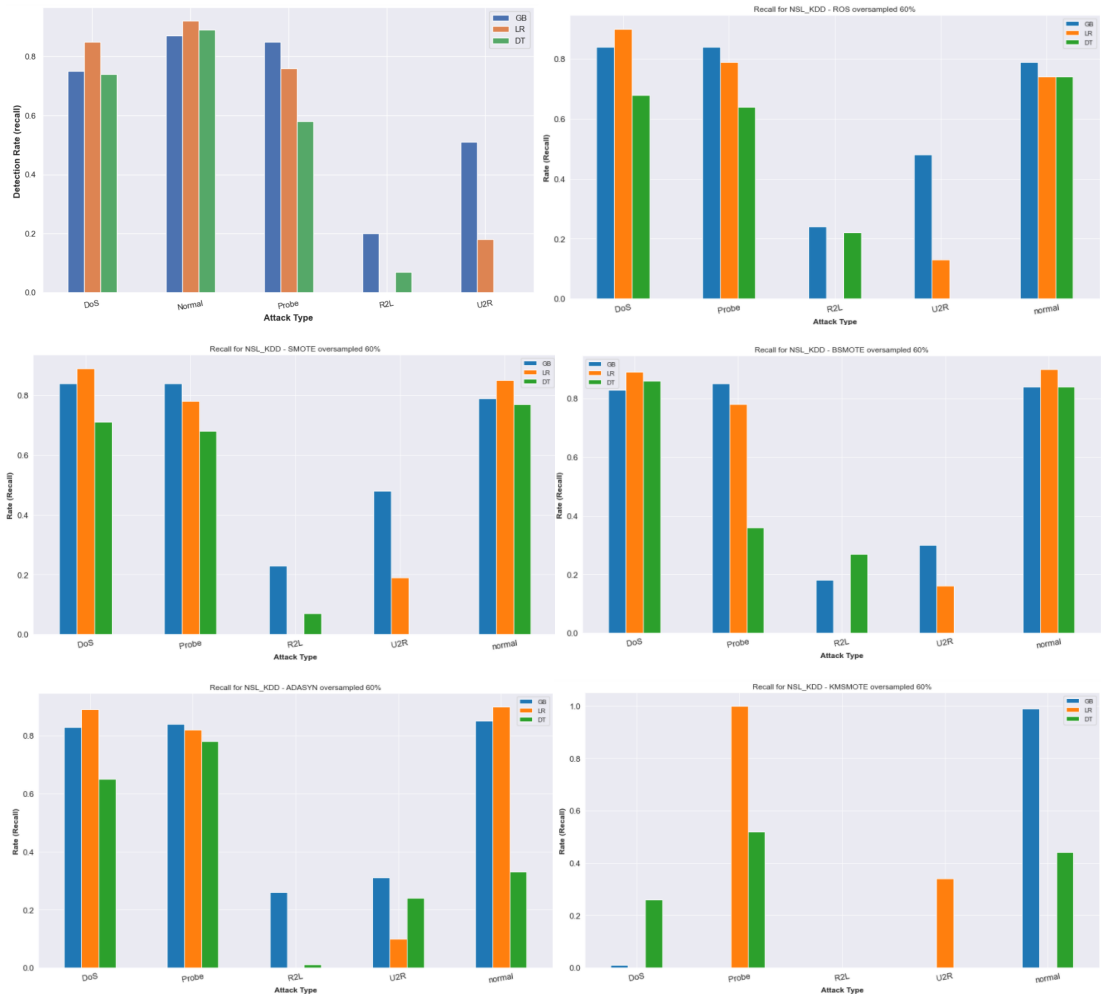


xi. Oversampled 60% detection rate

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	84	84	24	48	79
GB - SMOTE	84	84	23	48	79
GB - BSMOTE	83	85	18	30	84
GB - ADASYN	83	84	26	31	85
GB - KSMOTE	1	0	0	0	99
LR - ROS	***90***	79	0	13	74
LR - SMOTE	89	78	0	19	85
LR - BSMOTE	89	78	0	16	90

LR - ADASYN	89	82	0	10	90
LR - KSMOTE	0	***100***	0	34	0
DT - ROS	68	64	22	0	74
DT - SMOTE	71	68	7	0	77
DT - BSMOTE	86	36	***27***	0	84
DT - ADASYN	65	78	1	24	33
DT - KSMOTE	26	52	0	0	44

xii. Oversampled 60% comparison results

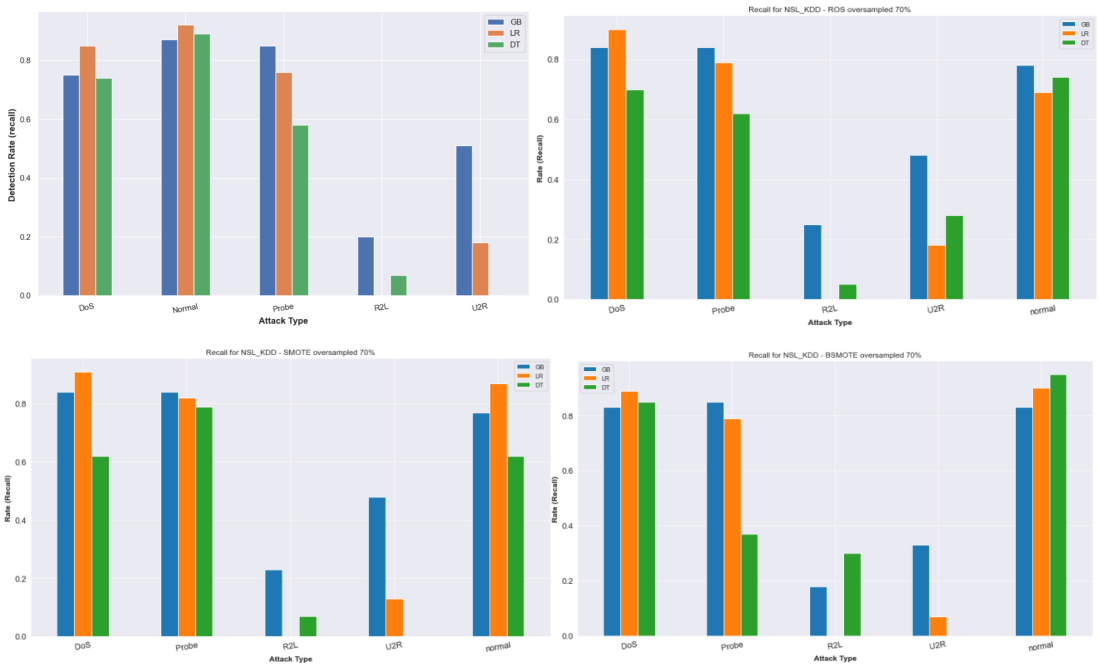


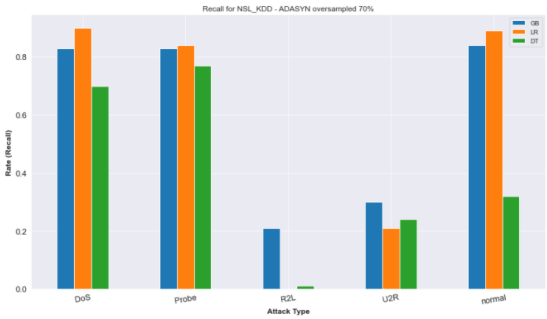
xiii. Oversampled 70% detection rate

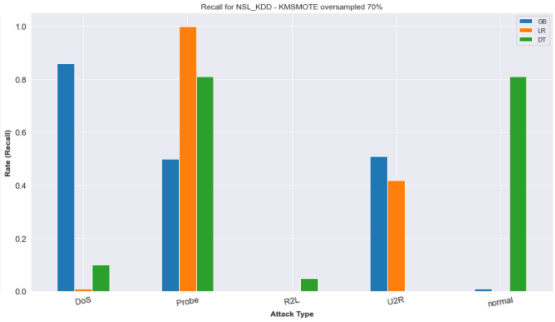
Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90

GB - ROS	84	84	25	48	78
GB - SMOTE	84	84	23	48	77
GB - BSMOTE	83	85	18	33	83
GB - ADASYN	83	83	21	30	84
GB - KSMOTE	86	50	0	51	1
LR - ROS	90	79	0	18	69
LR - SMOTE	***91***	82	0	13	87
LR - BSMOTE	89	79	0	7	90
LR - ADASYN	90	84	0	21	89
LR - KSMOTE	1	***100***	0	42	0
DT - ROS	70	62	5	28	74
DT - SMOTE	62	79	7	0	62
DT - BSMOTE	85	37	***30***	0	95
DT - ADASYN	70	77	1	24	32
DT - KSMOTE	10	81	5	0	81

xiv. Oversampled 70% comparison results



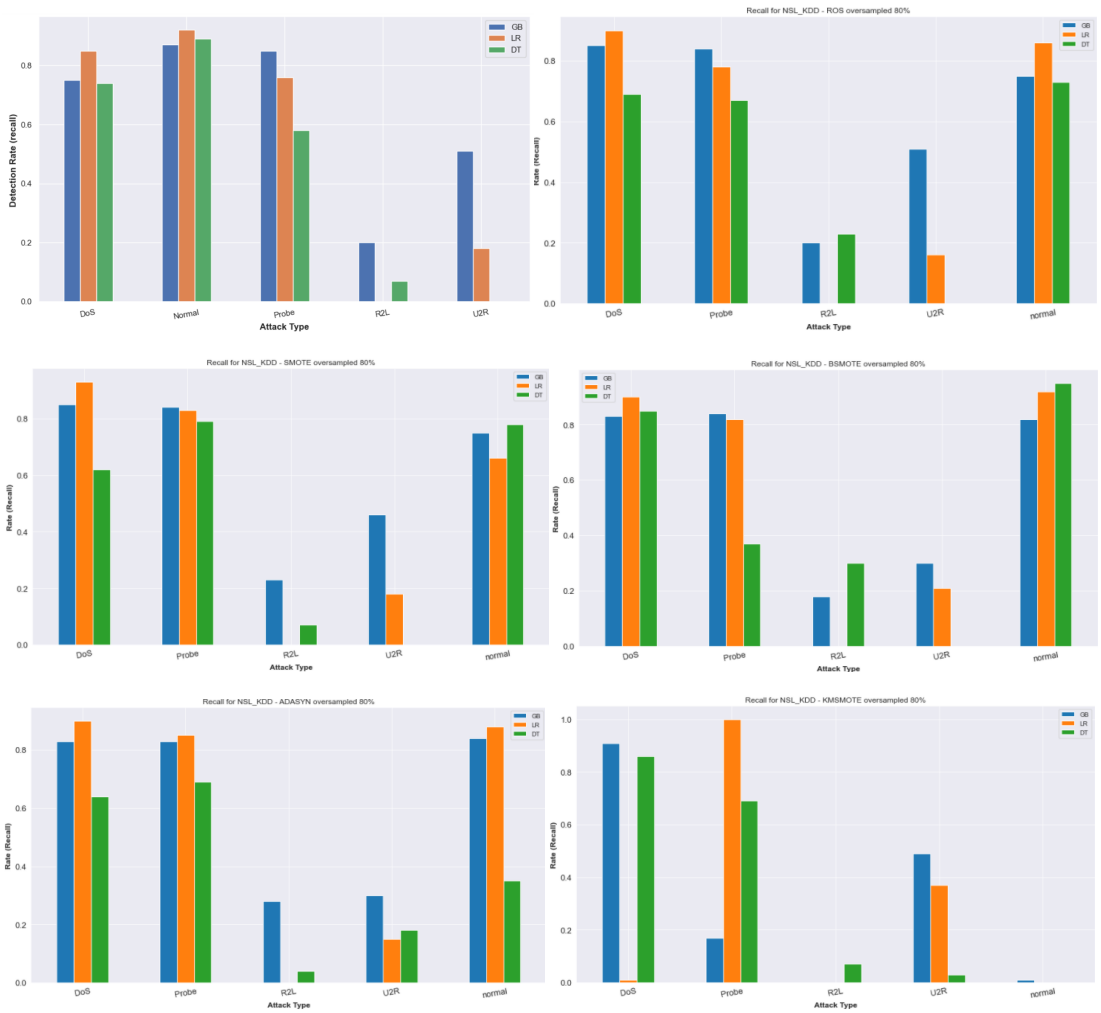




xv. Oversampled 80% detection rate

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	85	84	20	51	75
GB - SMOTE	85	84	23	46	75
GB - BSMOTE	83	84	18	30	82
GB - ADASYN	83	83	28	30	84
GB - KSMOTE	***91***	17	0	49	1
LR - ROS	90	78	0	16	86
LR - SMOTE	93	83	0	18	66
LR - BSMOTE	90	82	0	21	92
LR - ADASYN	90	85	0	15	88
LR - KSMOTE	1	***100***	0	37	0
DT - ROS	69	67	23	0	73
DT - SMOTE	62	79	7	0	78
DT - BSMOTE	85	37	***30***	0	95
DT - ADASYN	64	69	4	18	35
DT - KSMOTE	86	69	7	3	0

xvi. Oversampled 80% comparison results

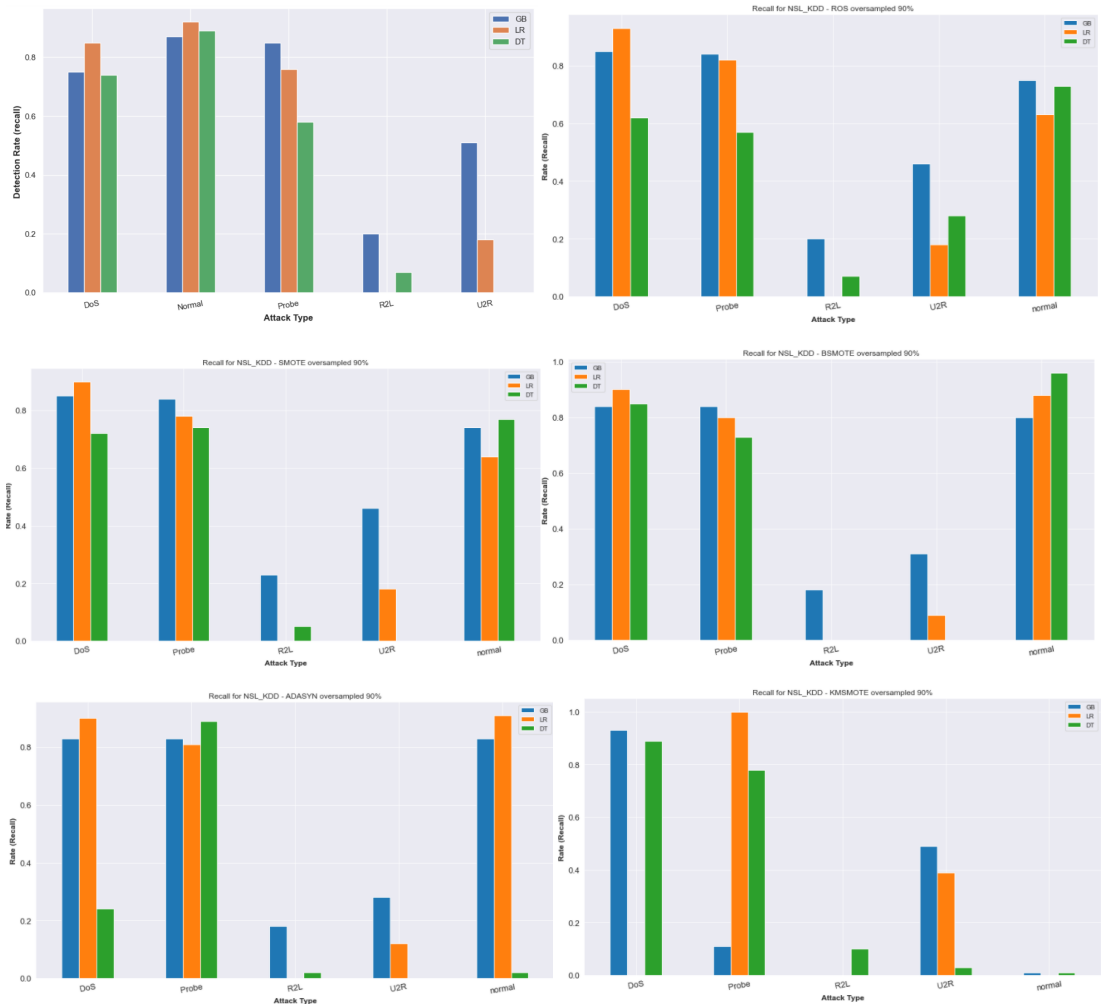


xvii. Oversampled 90% detection rate

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	85	84	20	46	75
GB - SMOTE	85	84	***23***	46	74
GB - BSMOTE	84	84	18	31	80
GB - ADASYN	83	83	18	28	83
GB - KSMOTE	***93***	11	0	49	1
LR - ROS	***93***	82	0	18	63
LR - SMOTE	90	78	0	18	64
LR - BSMOTE	90	80	0	9	88

LR - ADASYN	90	81	0	12	91
LR - KSMOTE	0	***100***	0	39	0
DT - ROS	62	57	7	28	73
DT - SMOTE	72	74	5	0	77
DT - BSMOTE	85	73	0	0	96
DT - ADASYN	24	89	2	0	2
DT - KSMOTE	89	78	10	3	1

xviii. Oversampled 90% comparison results

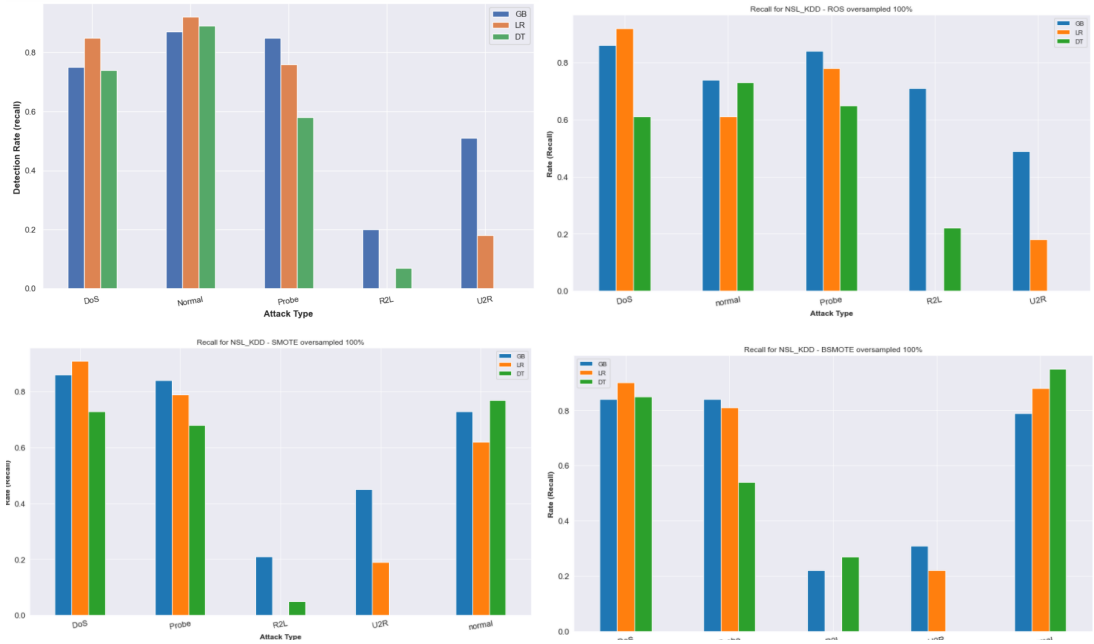


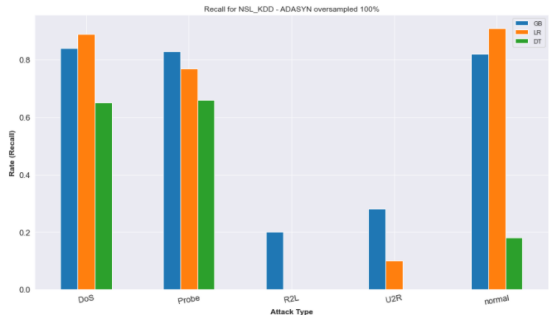
xix. Oversampled 100% detection rate

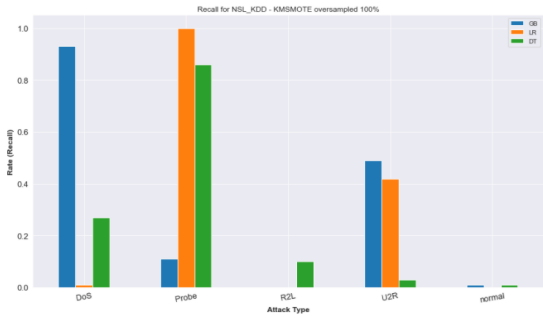
Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92

DR	74	58	7	0	90
GB - ROS	86	84	71	49	74
GB - SMOTE	86	84	21	45	73
GB - BSMOTE	84	84	22	31	79
GB - ADASYN	***94***	93	20	28	82
GB - KSMOTE	93	11	0	49	1
LR - ROS	92	78	0	18	61
LR - SMOTE	91	79	0	19	62
LR - BSMOTE	90	81	0	22	88
LR - ADASYN	89	77	0	10	91
LR - KSMOTE	1	***100***	0	42	0
DT - ROS	61	65	22	0	73
DT - SMOTE	73	68	5	0	77
DT - BSMOTE	85	54	***27***	0	95
DT - ADASYN	65	66	0	0	18
DT - KSMOTE	27	86	10	3	1

xx. Oversampled 100% comparison results





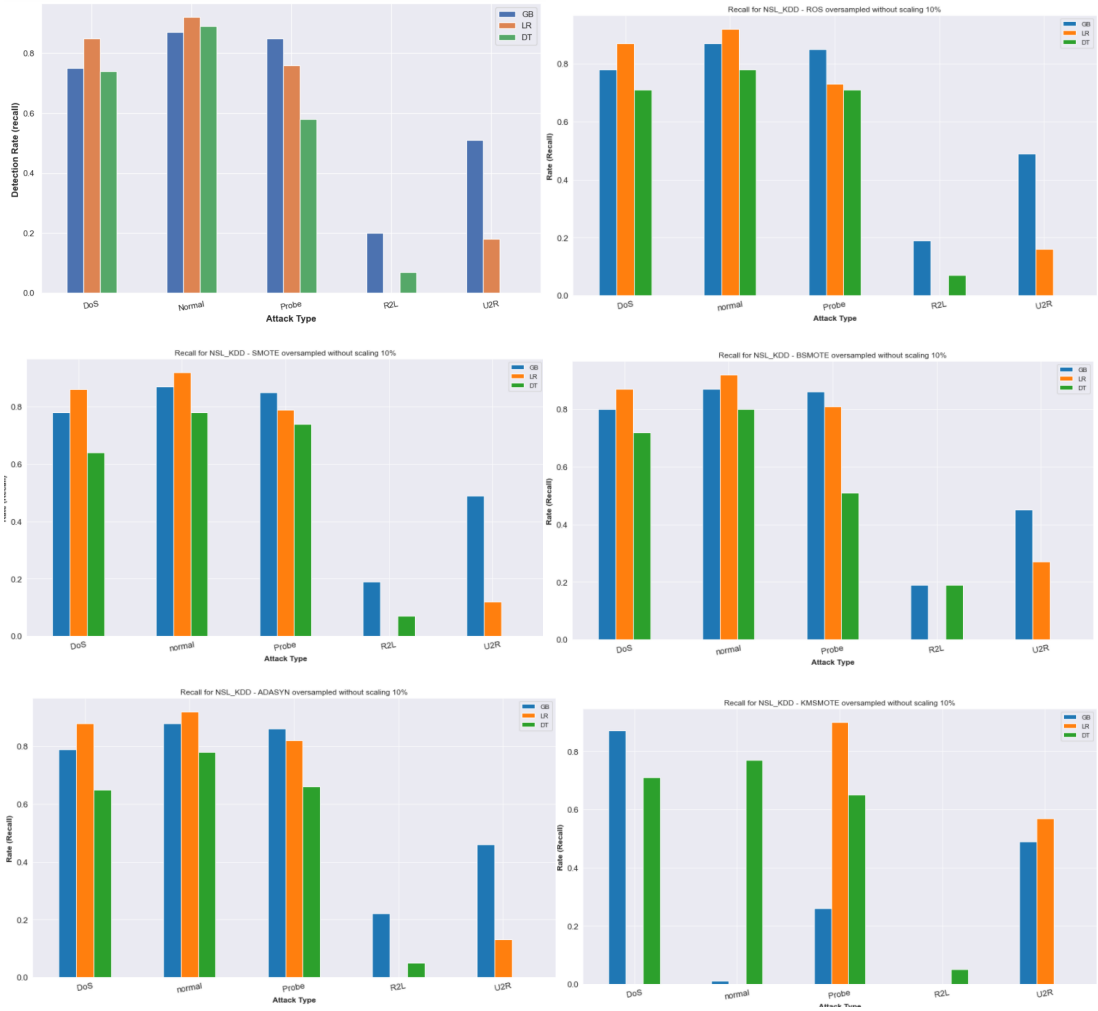


APPENDIX K: NSL KDD oversampling individual DR on non-scaled data

i. Oversampled 10% detection rate – non-scaled

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	77	33	***28***	69	78
GB - SMOTE	77	31	***28***	67	78
GB - BSMOTE	82	2	14	66	66
GB - ADASYN	79	10	***28***	67	75
GB - KSMOTE	77	30	***28***	***70***	79
LR - ROS	66	0	0	0	96
LR - SMOTE	66	0	0	0	96
LR - BSMOTE	64	0	0	0	97
LR - ADASYN	66	0	0	0	96
LR - KSMOTE	66	0	0	0	96
DT - ROS	76	61	9	13	97
DT - SMOTE	76	63	9	13	96
DT - BSMOTE	78	67	9	18	97
DT - ADASYN	77	63	9	12	97
DT - KSMOTE	76	61	9	13	97

ii. Oversampled 10% comparison results – non-scaled

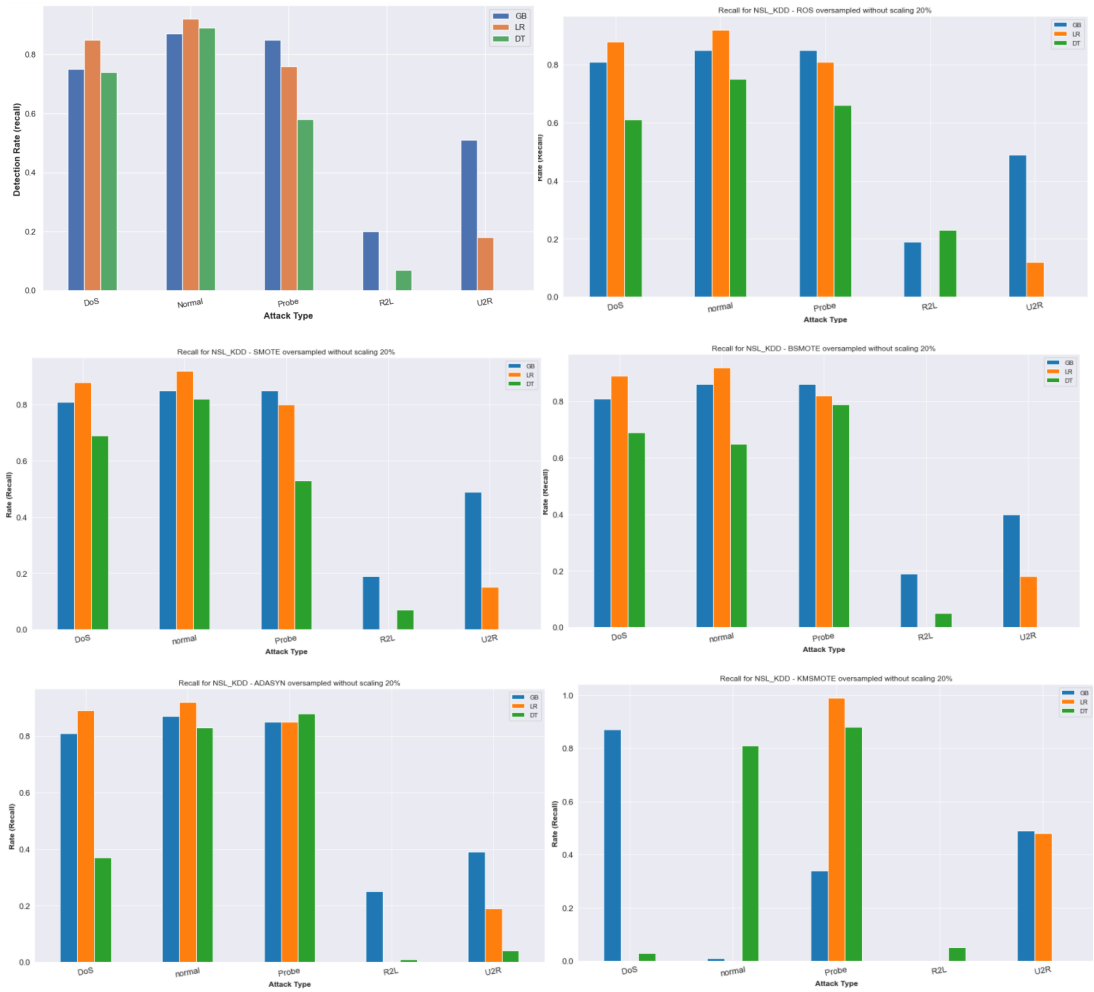


iii. Oversampled 20% detection rate – non-scaled

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	78	18	***28***	67	76
GB - SMOTE	77	44	***28***	70	78
GB - BSMOTE	82	2	12	66	48
GB - ADASYN	80	6	26	66	73
GB - KSMOTE	76	33	***28***	***73***	79
LR - ROS	68	0	0	0	95
LR - SMOTE	68	0	0	0	95
LR - BSMOTE	67	0	0	0	95

LR - ADASYN	67	0	0	0	95
LR - KSMOTE	69	0	0	0	89
DT - ROS	76	62	9	18	97
DT - SMOTE	77	61	9	16	97
DT - BSMOTE	71	66	3	13	96
DT - ADASYN	74	62	2	9	96
DT - KSMOTE	76	67	9	15	97

iv. Oversampled 20% comparison results – non-scaled

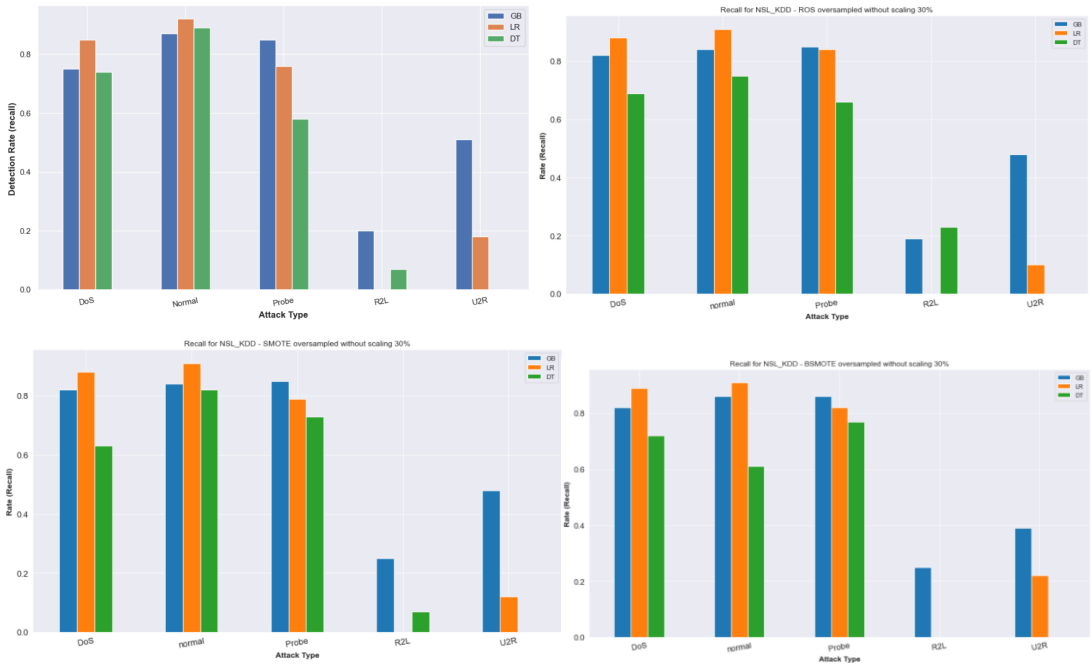


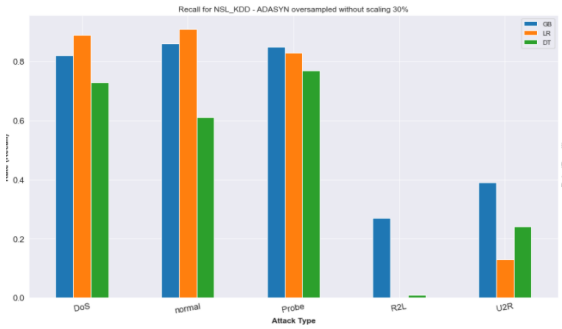
v. Oversampled 30% detection rate – non-scaled

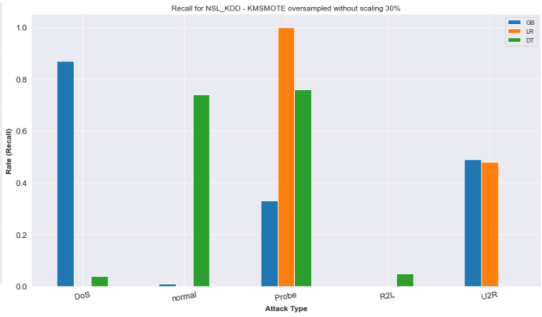
Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90

GB - ROS	78	21	***28***	***72***	76
GB - SMOTE	77	41	***28***	67	77
GB - BSMOTE	82	1	12	66	42
GB - ADASYN	81	3	23	66	72
GB - KSMOTE	75	50	***28***	70	82
LR - ROS	69	0	0	0	91
LR - SMOTE	69	0	0	0	91
LR - BSMOTE	68	0	0	0	95
LR - ADASYN	69	0	0	0	90
LR - KSMOTE	69	0	0	0	91
DT - ROS	82	71	5	18	96
DT - SMOTE	76	63	9	9	97
DT - BSMOTE	71	62	3	12	96
DT - ADASYN	76	71	9	1	97
DT - KSMOTE	77	63	9	18	97

vi. Oversampled 30% comparison results – non-scaled



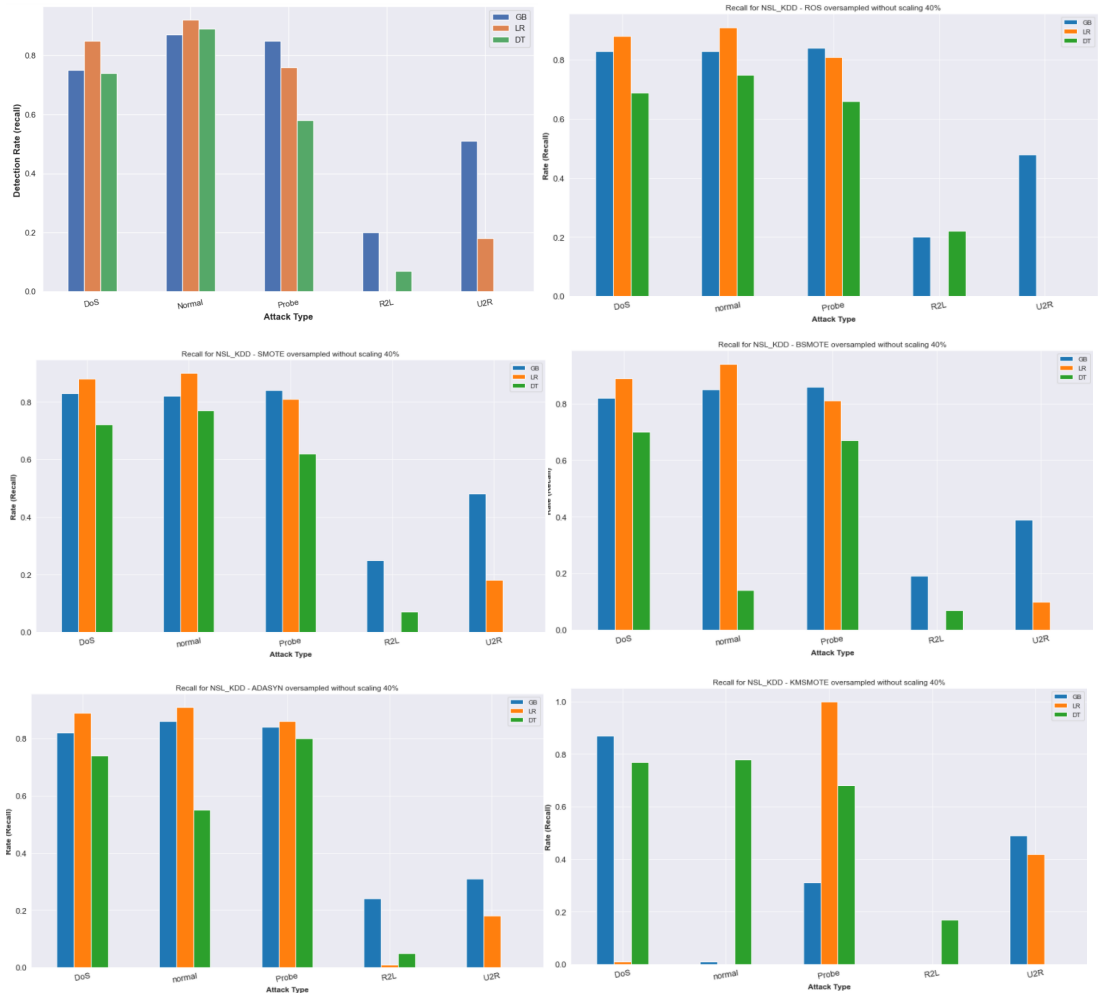




vii. Oversampled 40% detection rate – non-scaled

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	77	49	***28***	***73***	77
GB - SMOTE	78	28	***28***	67	77
GB - BSMOTE	82	1	9	67	43
GB - ADASYN	81	2	17	66	71
GB - KSMOTE	76	52	***28***	69	78
LR - ROS	71	0	0	0	89
LR - SMOTE	73	0	0	0	87
LR - BSMOTE	69	0	0	0	93
LR - ADASYN	69	0	0	0	91
LR - KSMOTE	70	0	0	0	91
DT - ROS	76	73	9	9	97
DT - SMOTE	82	70	9	15	97
DT - BSMOTE	70	82	14	19	97
DT - ADASYN	81	61	21	6	97
DT - KSMOTE	76	61	8	15	97

viii. Oversampled 40% comparison results – non-scaled

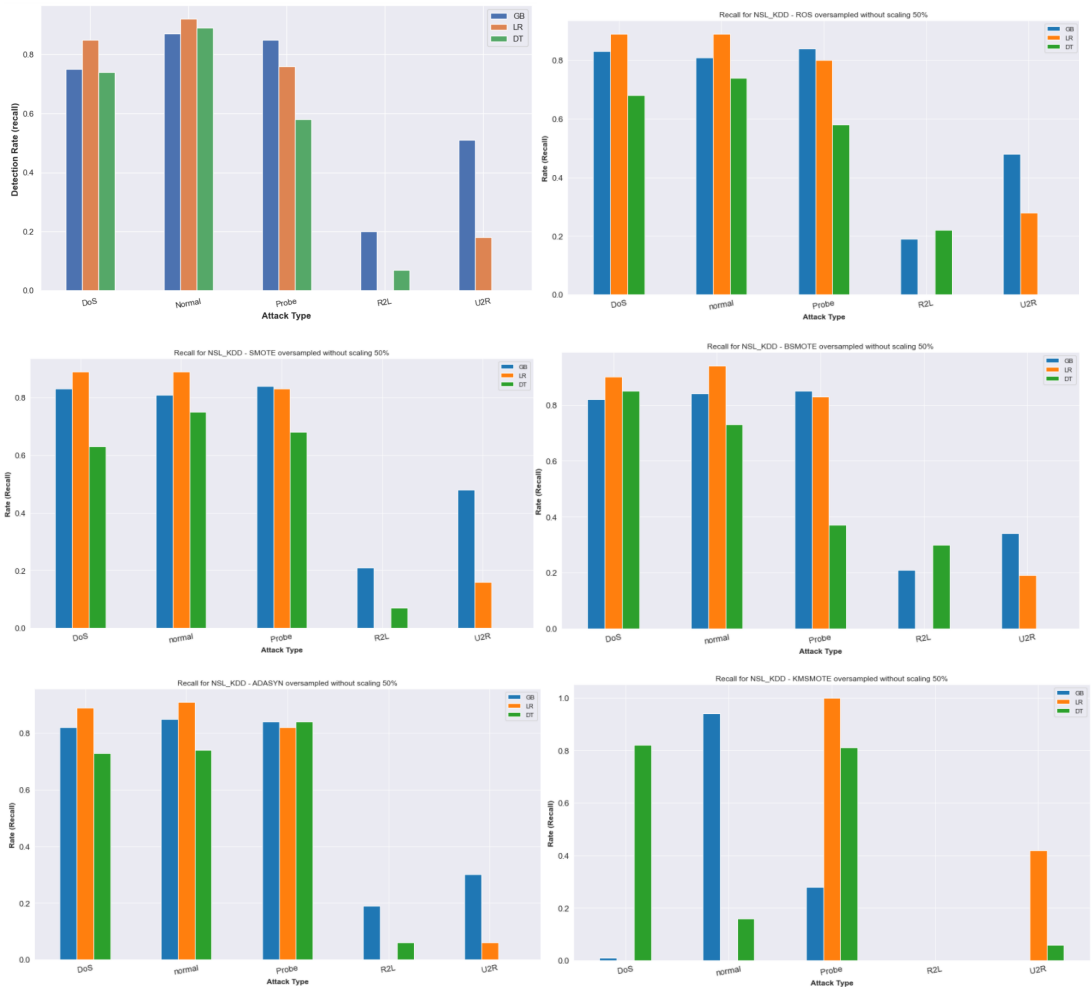


ix. Oversampled 50% detection rate – non-scaled

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	77	53	***28***	67	78
GB - SMOTE	77	50	***28***	***70***	78
GB - BSMOTE	82	0	0	55	42
GB - ADASYN	82	1	12	66	68
GB - KSMOTE	***87***	0	0	12	4
LR - ROS	74	0	0	0	85
LR - SMOTE	72	0	0	0	89
LR - BSMOTE	70	0	0	0	90

LR - ADASYN	71	0	0	0	89
LR - KSMOTE	73	0	0	0	83
DT - ROS	77	62	6	16	97
DT - SMOTE	80	64	9	6	97
DT - BSMOTE	69	74	8	9	96
DT - ADASYN	70	78	20	22	96
DT - KSMOTE	76	68	3	10	97

x. Oversampled 50% comparison results – non-scaled

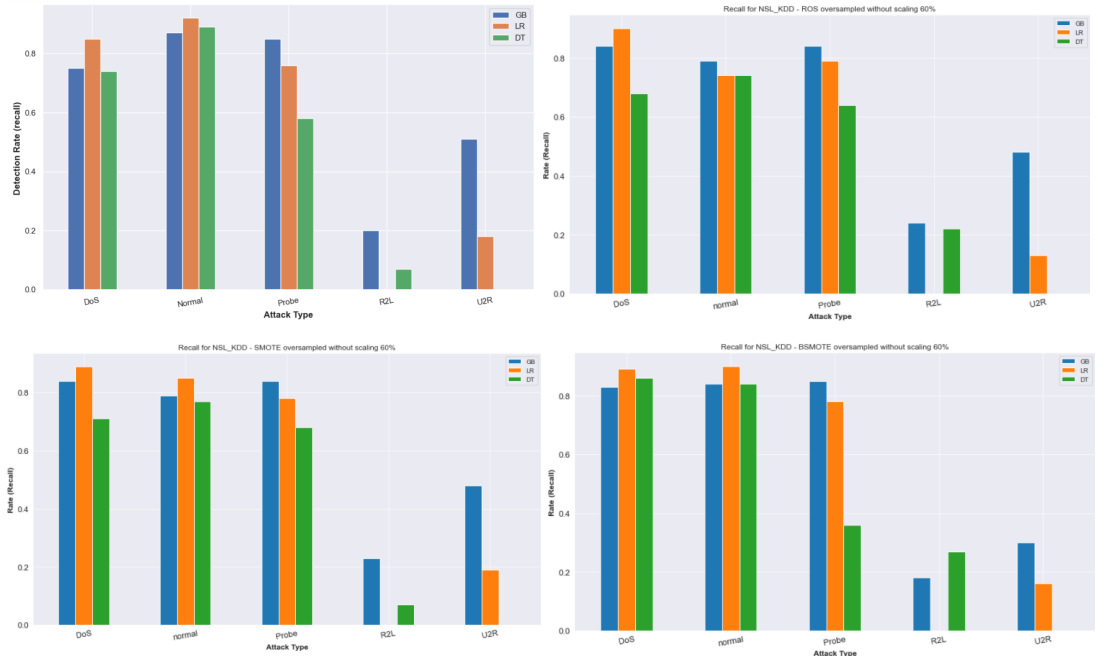


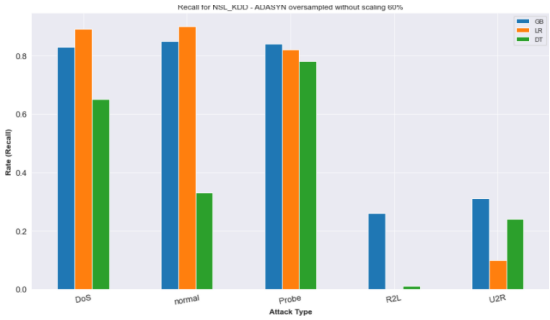
xi. Oversampled 60% detection rate – non-scaled

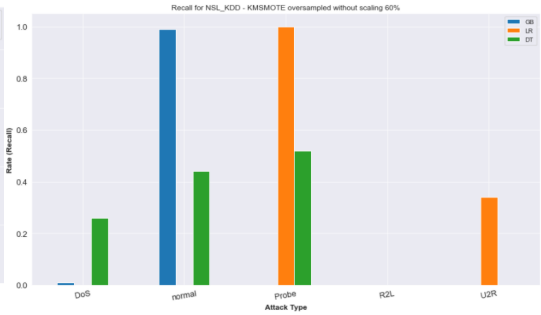
Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90

GB - ROS	78	19	***28***	***73***	75
GB - SMOTE	78	30	***28***	70	77
GB - BSMOTE	82	0	0	54	41
GB - ADASYN	82	1	13	63	67
GB - KSMOTE	***87***	59	27	72	4
LR - ROS	71	0	0	0	90
LR - SMOTE	72	0	0	0	90
LR - BSMOTE	71	0	0	0	90
LR - ADASYN	74	0	0	0	88
LR - KSMOTE	76	0	0	0	76
DT - ROS	83	65	15	13	97
DT - SMOTE	75	67	9	12	97
DT - BSMOTE	71	67	6	25	96
DT - ADASYN	79	64	17	12	96
DT - KSMOTE	76	59	3	18	97

xii. Oversampled 60% comparison results – non-scaled



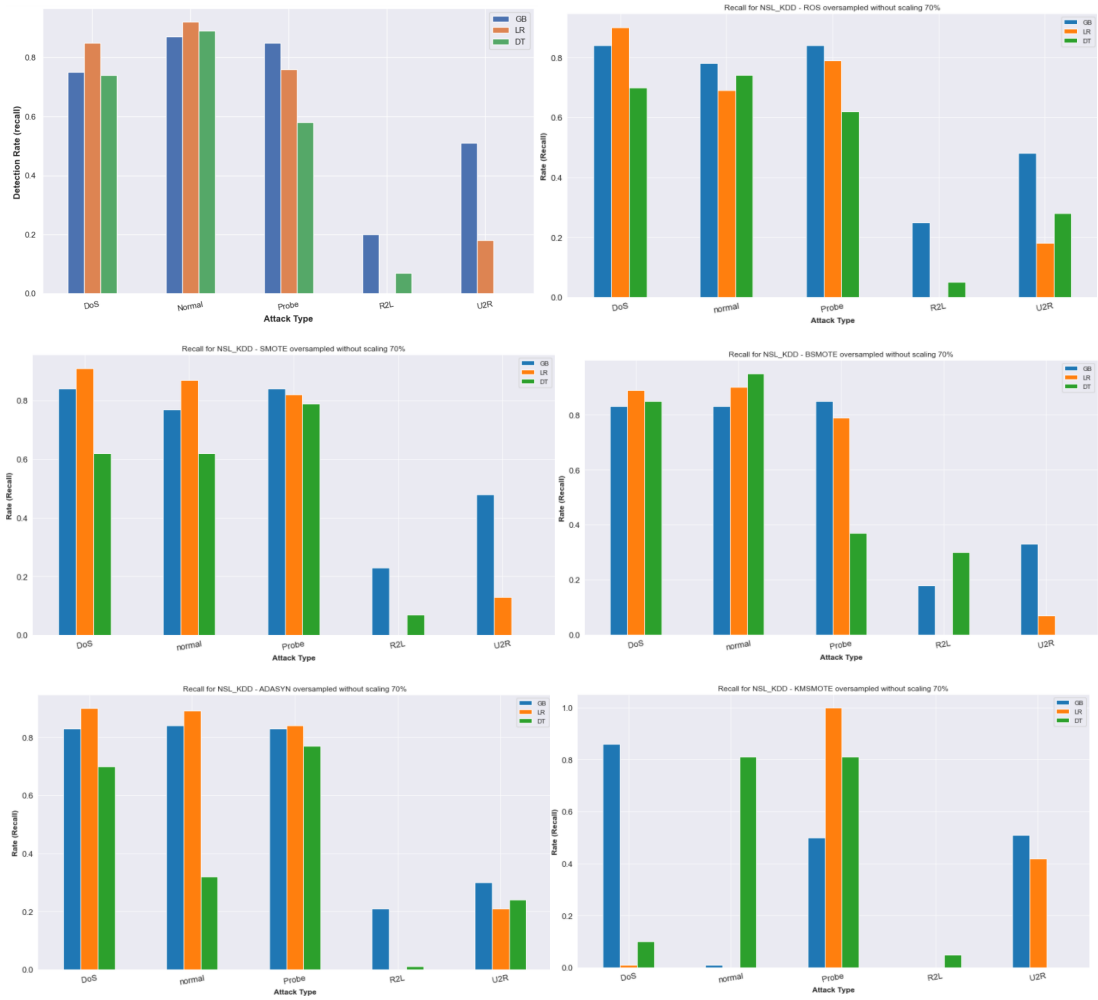




xiii. Oversampled 70% detection rate – non-scaled

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	79	17	***28***	64	75
GB - SMOTE	78	28	***28***	***70***	77
GB - BSMOTE	82	0	0	54	40
GB - ADASYN	82	1	12	63	67
GB - KSMOTE	74	63	***28***	69	84
LR - ROS	78	0	0	0	84
LR - SMOTE	75	0	0	0	88
LR - BSMOTE	73	0	0	0	90
LR - ADASYN	76	0	0	0	87
LR - KSMOTE	74	0	0	0	89
DT - ROS	76	67	16	0	97
DT - SMOTE	76	82	9	15	97
DT - BSMOTE	70	78	6	18	96
DT - ADASYN	70	78	***28***	12	96
DT - KSMOTE	76	64	10	16	97

xiv. Oversampled 70% comparison results – non-scaled

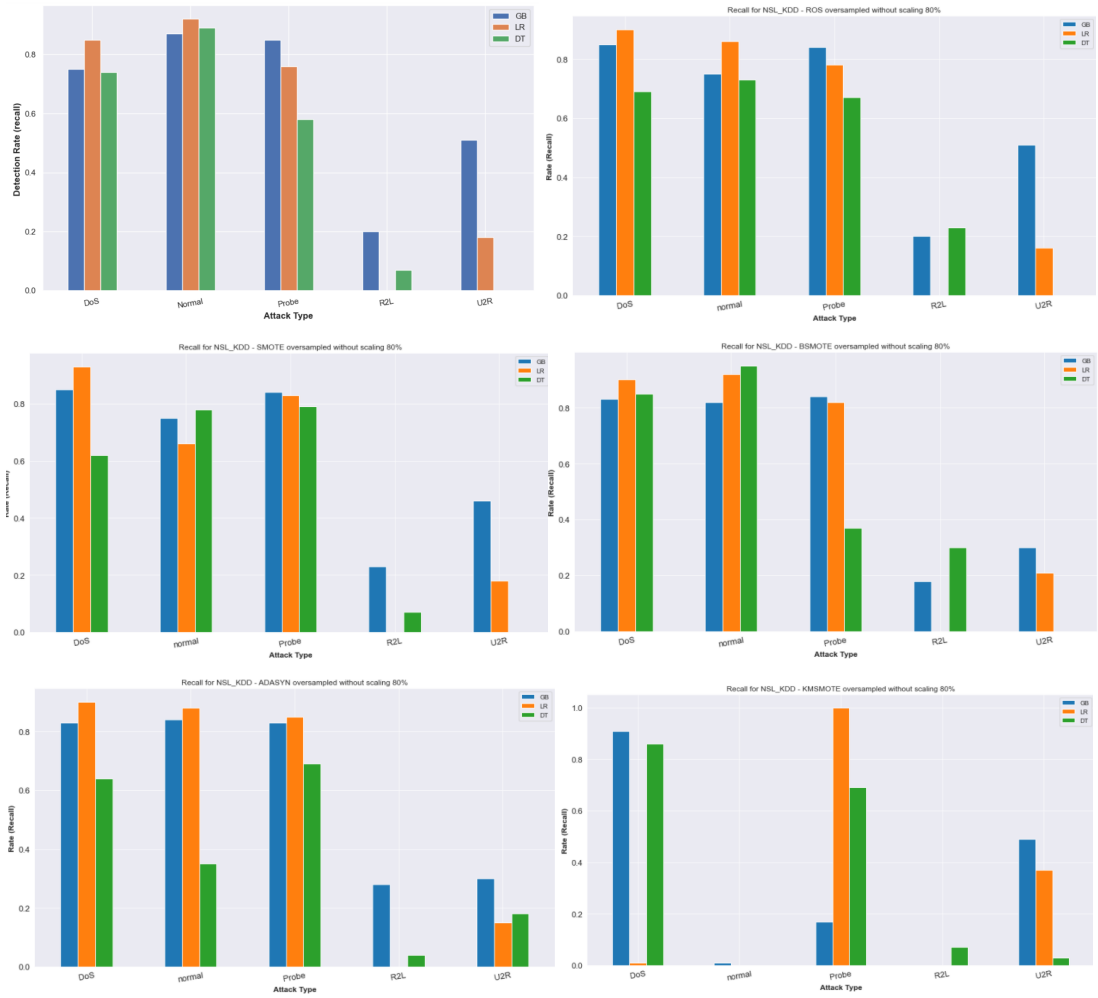


xv. Oversampled 80% detection rate – non-scaled

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	80	14	27	67	74
GB - SMOTE	79	17	***28***	69	74
GB - BSMOTE	82	0	0	54	39
GB - ADASYN	82	1	12	63	54
GB - KSMOTE	74	60	***28***	***73***	83
LR - ROS	75	0	0	0	86
LR - SMOTE	75	0	0	0	86
LR - BSMOTE	80	0	0	0	79

LR - ADASYN	77	0	0	0	86
LR - KSMOTE	76	0	0	0	86
DT - ROS	82	67	9	16	97
DT - SMOTE	76	64	6	7	97
DT - BSMOTE	70	78	9	10	96
DT - ADASYN	78	66	9	6	96
DT - KSMOTE	76	74	16	21	97

xvi. Oversampled 80% comparison results – non-scaled

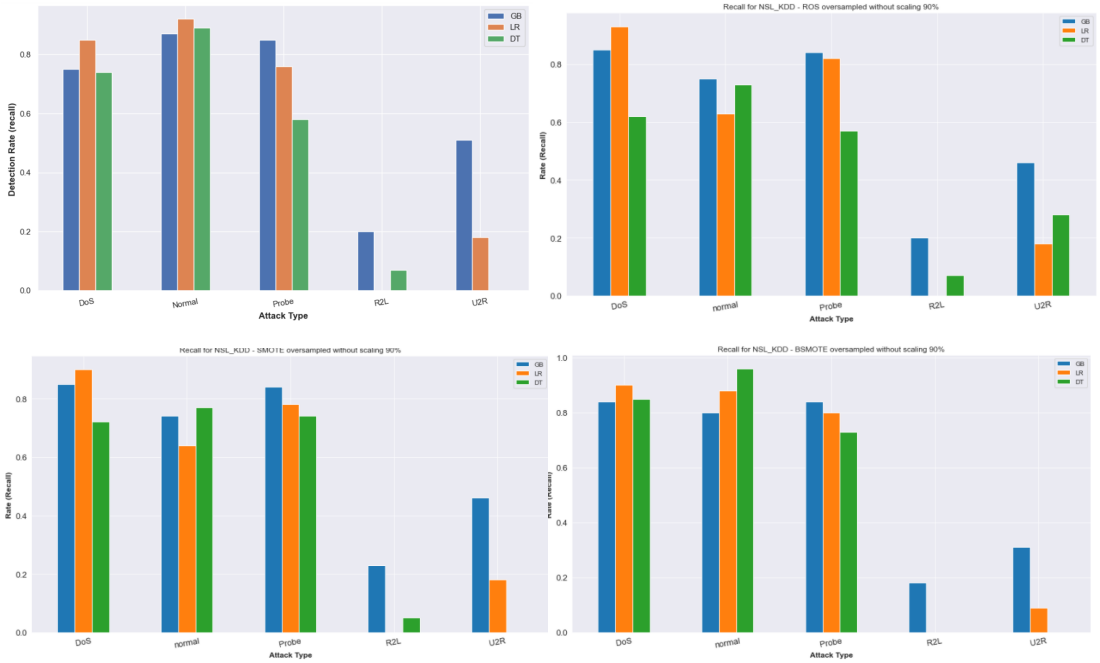


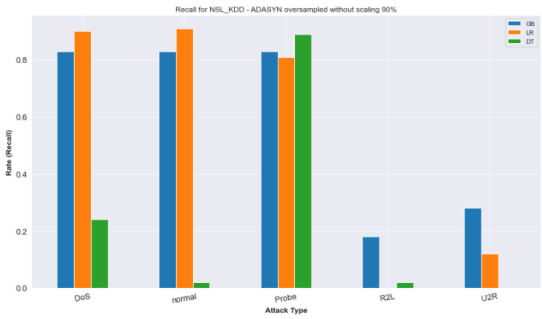
xvii. Oversampled 90% detection rate – non-scaled

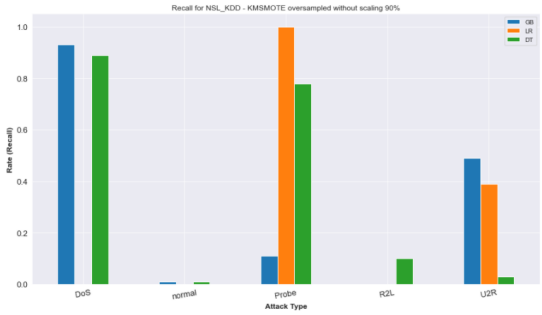
Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90

GB - ROS	80	14	27	67	73
GB - SMOTE	79	17	***28***	66	75
GB - BSMOTE	82	0	0	52	39
GB - ADASYN	82	1	10	63	51
GB - KSMOTE	76	58	27	***73***	80
LR - ROS	76	0	0	0	85
LR - SMOTE	78	0	0	0	82
LR - BSMOTE	77	0	0	0	78
LR - ADASYN	78	0	0	0	75
LR - KSMOTE	78	0	0	0	83
DT - ROS	81	67	16	15	97
DT - SMOTE	76	65	9	16	97
DT - BSMOTE	70	78	16	16	96
DT - ADASYN	72	81	2	12	96
DT - KSMOTE	77	74	16	40	97

xviii. Oversampled 90% comparison results – non-scaled



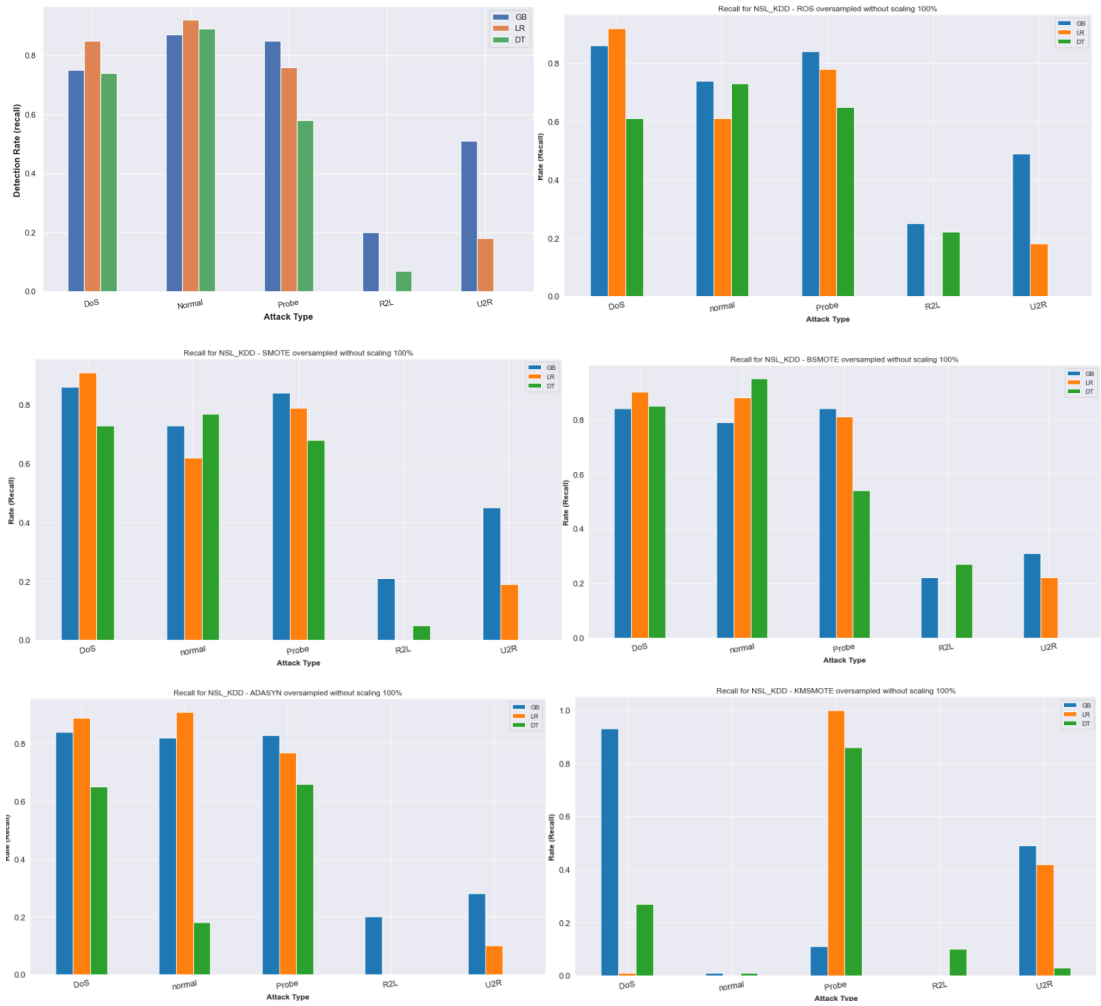




xix. Oversampled 100% detection rate – non-scaled

Result	DoS	Probe	R2L	U2R	Normal
GB	75	85	20	51	87
LR	85	76	0	18	92
DR	74	58	7	0	90
GB - ROS	79	17	***28***	67	74
GB - SMOTE	80	13	26	66	73
GB - BSMOTE	82	0	0	52	38
GB - ADASYN	82	1	10	63	54
GB - KSMOTE	76	58	27	***73***	80
LR - ROS	78	0	0	0	82
LR - SMOTE	78	0	0	0	80
LR - BSMOTE	78	0	0	0	83
LR - ADASYN	78	0	0	0	82
LR - KSMOTE	76	0	0	0	85
DT - ROS	77	66	17	13	96
DT - SMOTE	76	61	9	10	97
DT - BSMOTE	70	72	11	16	96
DT - ADASYN	72	75	8	12	96
DT - KSMOTE	77	73	7	18	97

xx. Oversampled 100% comparison results – non-scaled



APPENDIX L: Top 3 detection rate of each minority classes for NSL KDD

	DOS		PROBE		R2L		U2R	
	Combination	**DR**	**Combination**	**DR**	**Combination**	**DR**	**Combination**	**DR**
	Original	85	Original	85	Original	20	Original	51
S1	GB-ADASYN-100%	***94***	LR-KMSMOTE-30% to 100%	***100***	GB-ROS-30%	***90***	LR-KMSMOTE-10%	57
S2	LR-SMOTE-80% GB-KMSMOTE-90% LR-ROS-90%	93	LR-KSMOTE-20%	99	GB-ROS-100%	71		
S3	LR-SMOTE-70% GB-KMSMOTE-80%	91	LR-KSMOTE-10%	90	DT-BSMOTE-50%, 70%, 80%	30		
NS1	GB-KMSMOTE-50%, 60%	87	DT-BSMOTE-40% DT-SMOTE-70%	82	GB-ROS-10%, 30%, 40%, 50%, 60%. 70%. 100% GB-SMOTE-10%, 30%, 40%, 50%, 60%, 70%, 80% GB-ADASYN-10% GB-KMSMOTE-10%, 20%, 30%, 40%, 70%, 80%	28	GB-KMSMOTE-20%, 80%, 90%, 100% GB-ROS-40%, 60%	***73***
NS2			DT-ADASYN-90%	81	GB-KMSMOTE-100%	27	GB-ROS-30% GB-KMSMOTE-60%	72

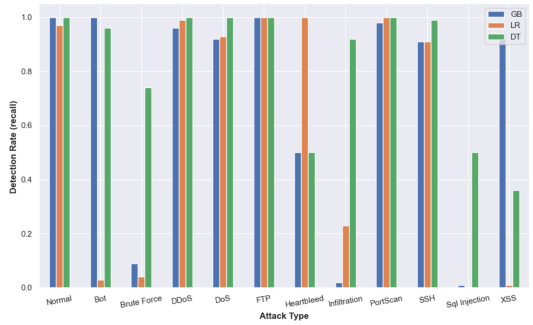
NS3			DT-ADASYN-50%, 70% DT-BSMOTE-70%	78	GB-SMOTE-100%	26	GB-KMSMOTE-10%, 30% GB-SMOTE-20%, 50%, 70%	70

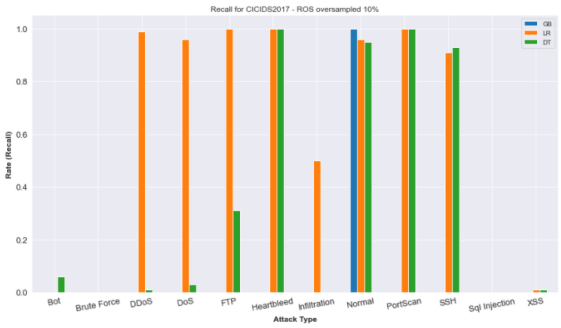
APPENDIX M: CICIDS 2017 oversampling individual DR on scaled data.

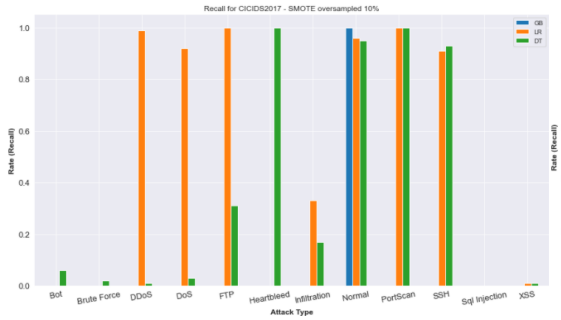
i. Oversampled 10% detection rate

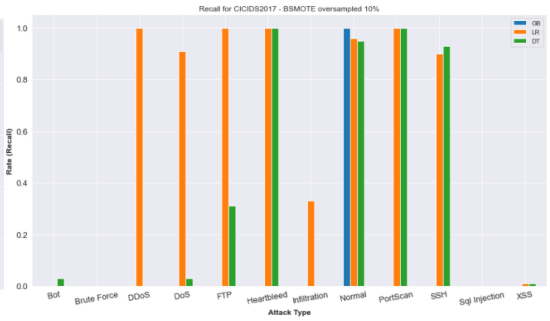
Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
LR ROS	0	0	99	96	100	***100***	***50***	96	100	91	0	1
LR SMOTE	0	0	99	92	100	0	33	96	100	91	0	1
LR BSMOTE	0	0	100	91	100	***100***	33	96	100	90	0	1
LR ADASYN	0	0	100	91	100	***100***	33	96	100	91	0	1
LR KSMOTE	0	0	98	96	100	0	33	96	100	90	0	1
DT ROS	6	0	1	3	31	***100***	0	95	100	93	0	1
DT SMOTE	6	2	1	3	31	***100***	17	95	100	93	0	1
DT BSMOTE	3	0	0	3	31	***100***	0	95	100	93	0	1
DT ADASYN	6	5	0	1	31	***100***	0	95	100	93	0	1
DT KSMOTE	6	0	1	1	31	***100***	0	94	100	93	0	1

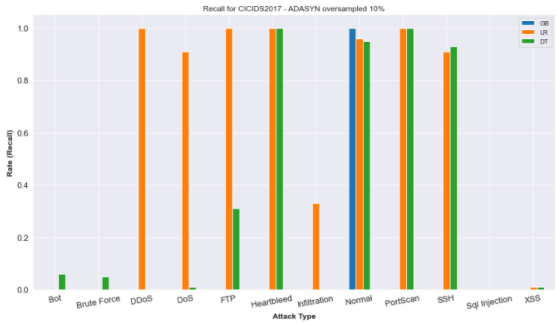
ii. Oversampled 10% comparison results

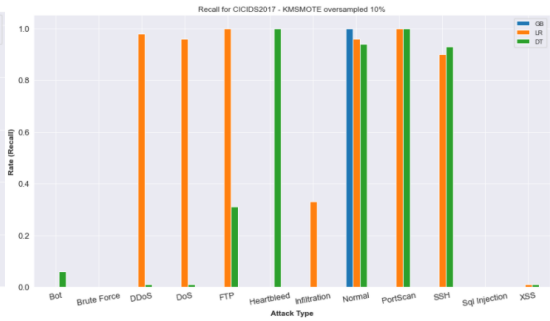










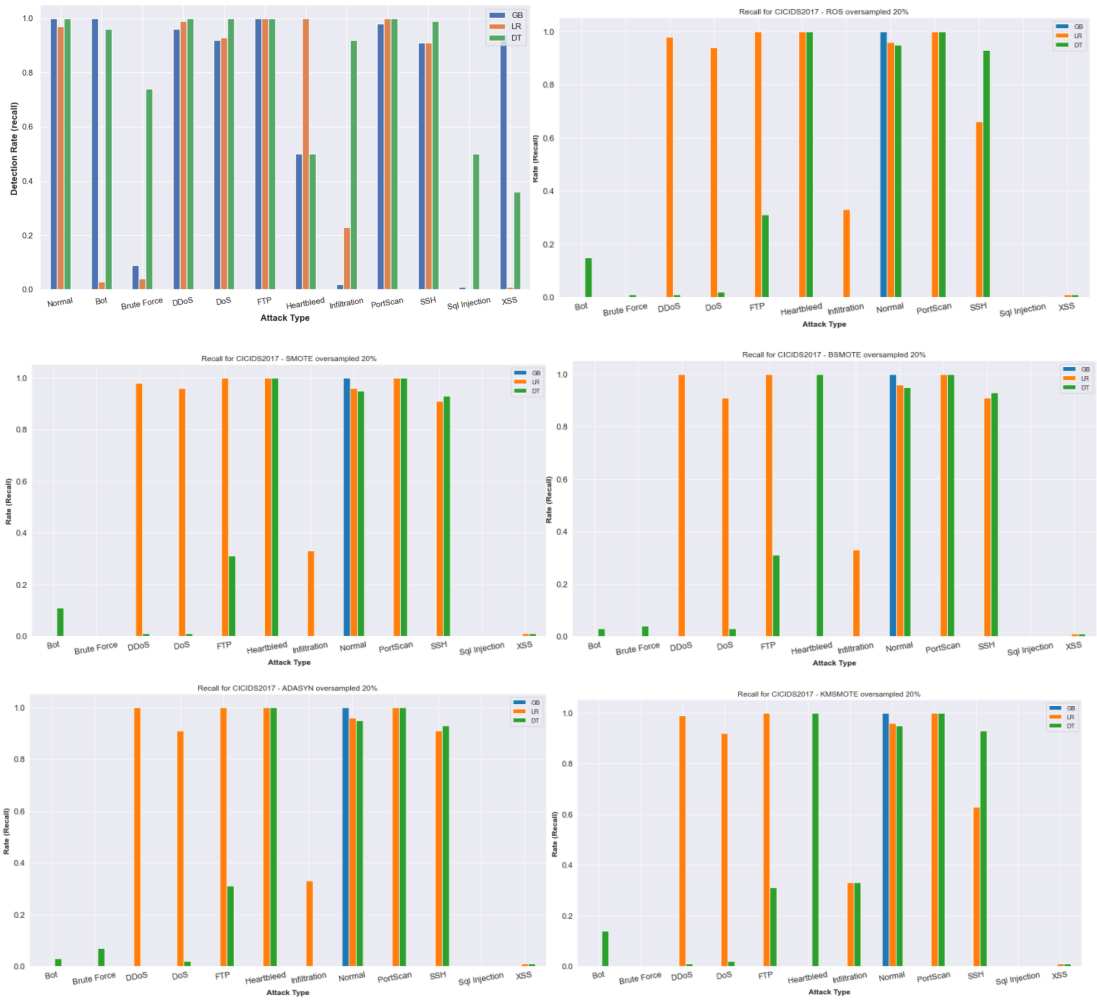


iii. Oversampled 20% detection rate

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
LR ROS	0	0	98	94	100	***100***	33	96	100	66	0	1
LR SMOTE	0	0	98	96	100	***100***	33	96	100	91	0	1

LR BSMOTE	0	0	100	91	100	0	33	96	100	91	0	1
LR ADASYN	0	0	100	91	100	***100***	33	96	100	91	0	1
LR KSMOTE	0	0	99	92	100	0	33	96	100	63	0	1
DT ROS	***15***	1	1	2	31	***100***	0	95	100	93	0	1
DT SMOTE	11	0	1	1	31	***100***	0	95	100	93	0	1
DT BSMOTE	3	4	0	3	31	***100***	0	95	100	93	0	1
DT ADASYN	3	7	0	2	31	***100***	0	95	100	93	0	1
DT KSMOTE	14	0	1	2	31	***100***	33	95	100	93	0	1

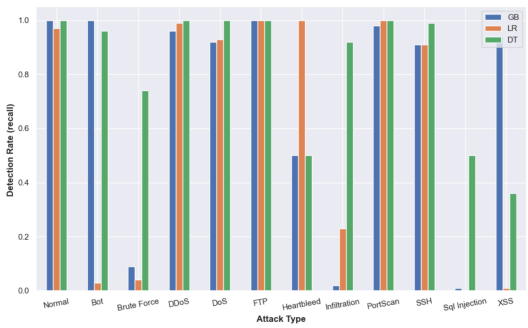
iv. Oversampled 20% comparison results

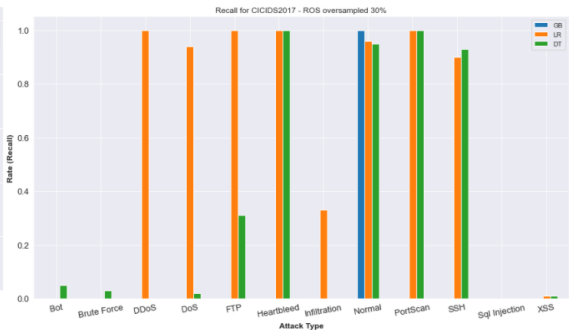


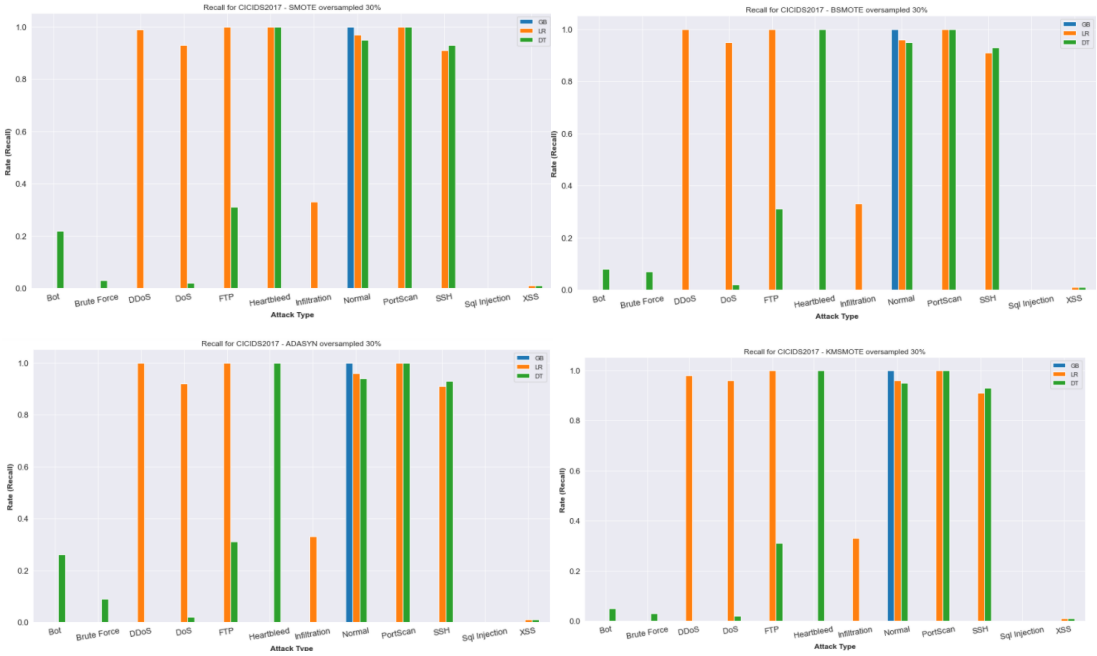
v. Oversampled 30% detection rate

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
LR ROS	0	0	100	94	100	***100***	33	96	100	90	0	1
LR SMOTE	0	0	99	93	100	***100***	33	97	100	91	0	1
LR BSMOTE	0	0	100	95	100	0	33	96	100	91	0	1
LR ADASYN	0	0	100	92	100	0	33	96	100	91	0	1
LR KSMOTE	0	0	98	96	100	0	33	96	100	91	0	1
DT ROS	5	3	0	2	31	***100***	0	95	100	93	0	1
DT SMOTE	22	3	0	2	31	***100***	0	95	100	93	0	1
DT BSMOTE	8	7	0	2	31	***100***	0	95	100	93	0	1
DT ADASYN	***26***	9	0	2	31	***100***	0	94	100	93	0	1
DT KSMOTE	5	3	0	2	31	***100***	0	95	100	93	0	1

vi. Oversampled 30% comparison results



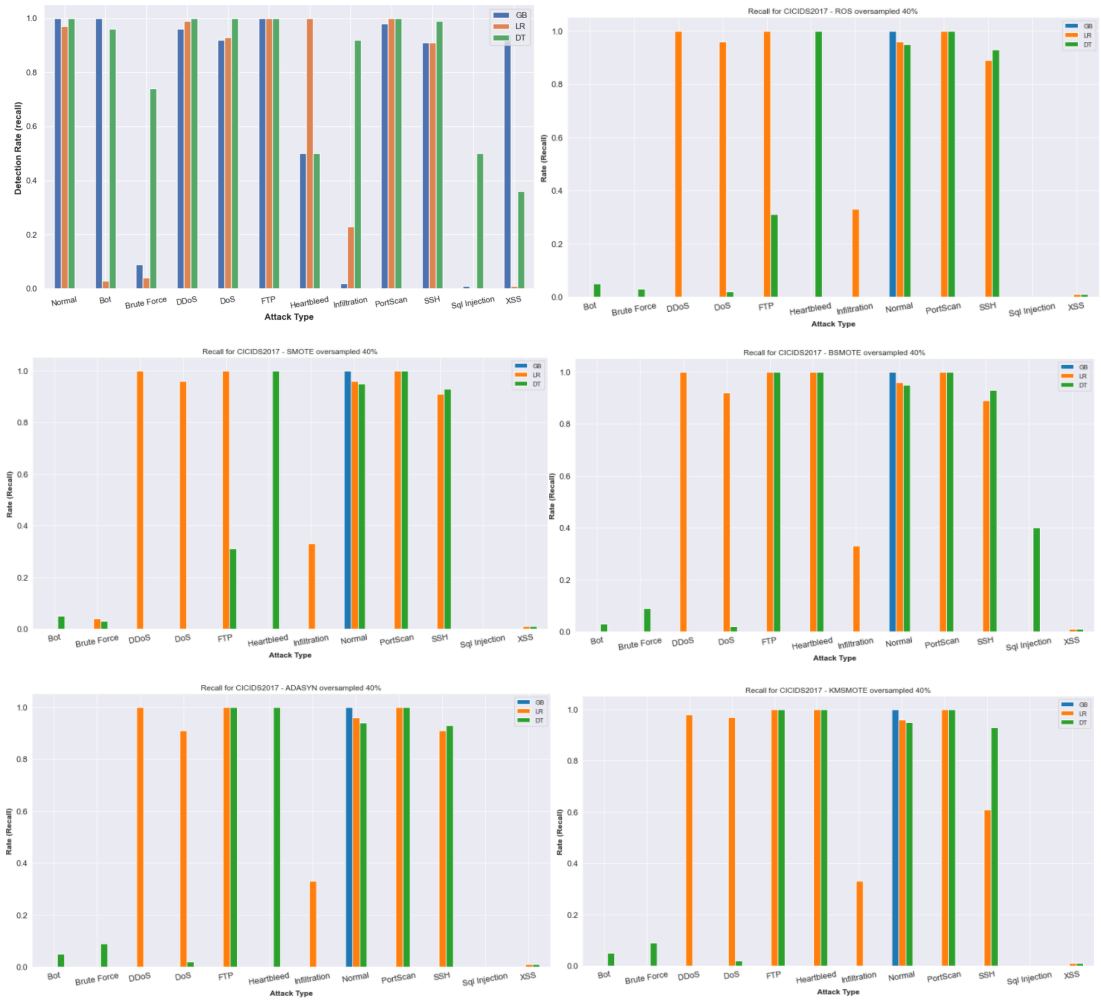




vii. Oversampled 40% detection rate

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
LR ROS	0	0	100	96	100	0	33	96	100	89	0	1
LR SMOTE	0	4	100	96	100	0	33	96	100	91	0	1
LR BSMOTE	0	0	100	92	100	***100***	33	96	100	89	0	1
LR ADASYN	0	0	100	91	100	0	33	96	100	91	0	1
LR KSMOTE	0	0	98	97	100	100	33	96	100	61	0	1
DT ROS	5	3	0	2	31	***100***	0	95	100	93	0	1
DT SMOTE	5	3	0	0	31	***100***	0	95	100	93	0	1
DT BSMOTE	3	9	0	2	100	***100***	0	95	100	93	***40***	1
DT ADASYN	5	9	0	2	100	***100***	0	94	100	93	0	1
DT KSMOTE	5	9	0	2	100	***100***	0	95	100	93	0	1

viii. Oversampled 40% comparison results

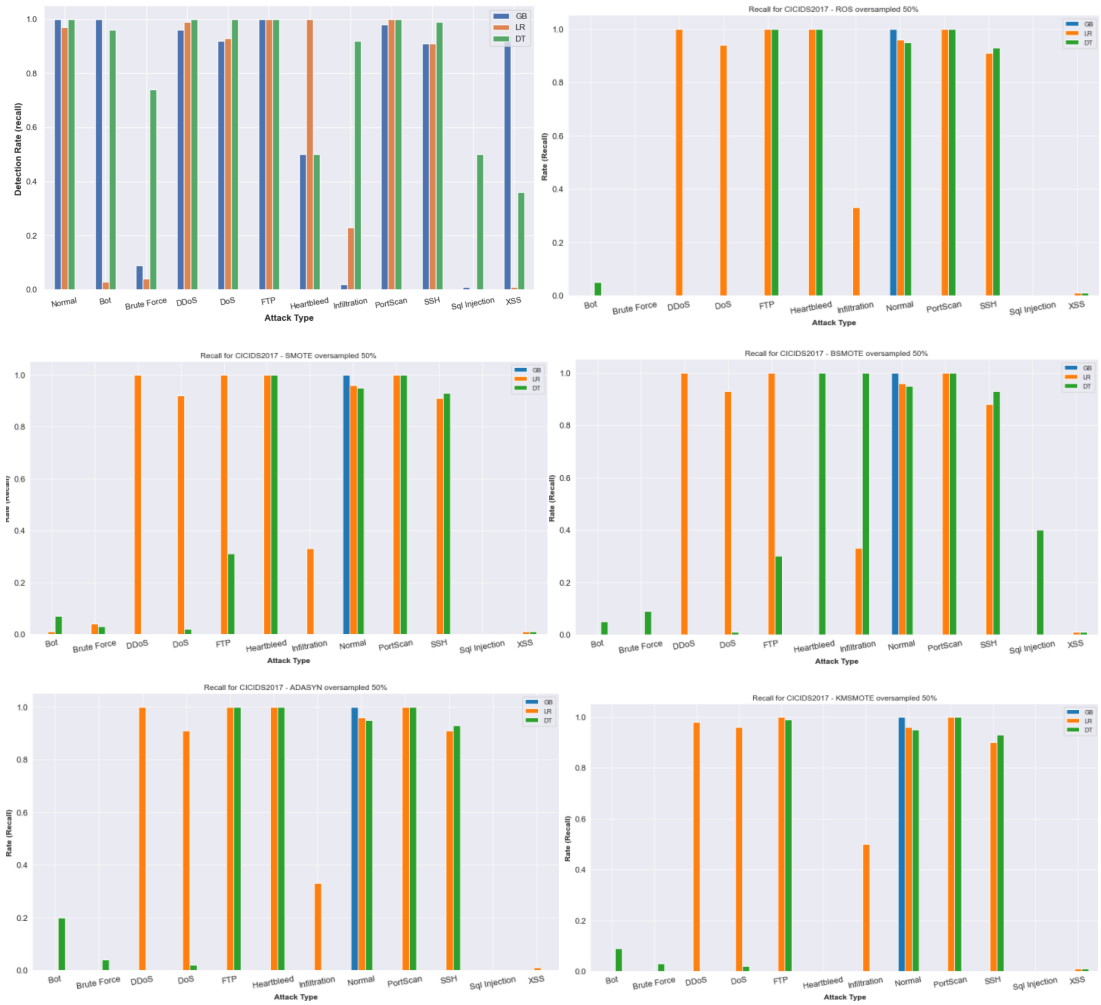


ix. Oversampled 50% detection rate

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
LR ROS	0	0	100	94	100	***100***	33	96	100	91	0	1

LR SMOTE	1	4	100	92	100	***100***	33	96	100	91	0	1
LR BSMOTE	0	0	100	93	100	0	33	96	100	88	0	1
LR ADASYN	0	0	100	91	100	***100***	33	96	100	91	0	1
LR KSMOTE	0	0	98	96	100	0	50	96	100	90	0	1
DT ROS	5	0	0	0	100	***100***	0	95	100	93	0	1
DT SMOTE	7	3	0	2	31	***100***	0	95	100	93	0	1
DT BSMOTE	5	9	0	1	30	***100***	***100***	95	100	93	***40***	1
DT ADASYN	***20***	4	0	2	100	***100***	0	95	100	93	0	0
DT KSMOTE	9	3	0	2	99	0	0	95	100	93	0	1

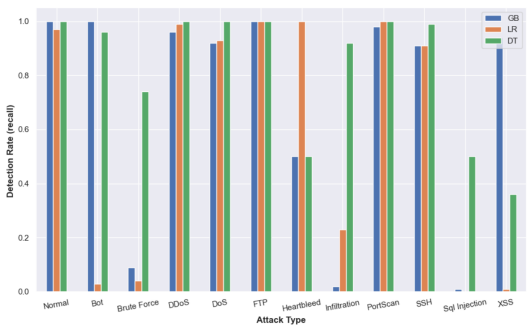
x. Oversampled 50% comparison results

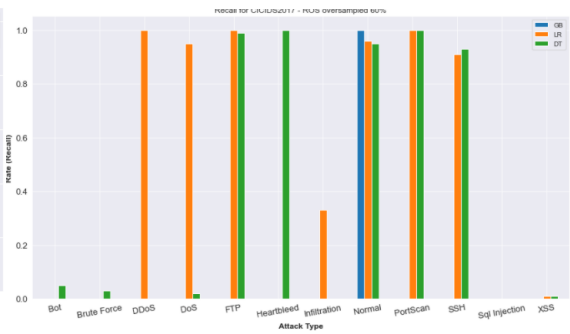


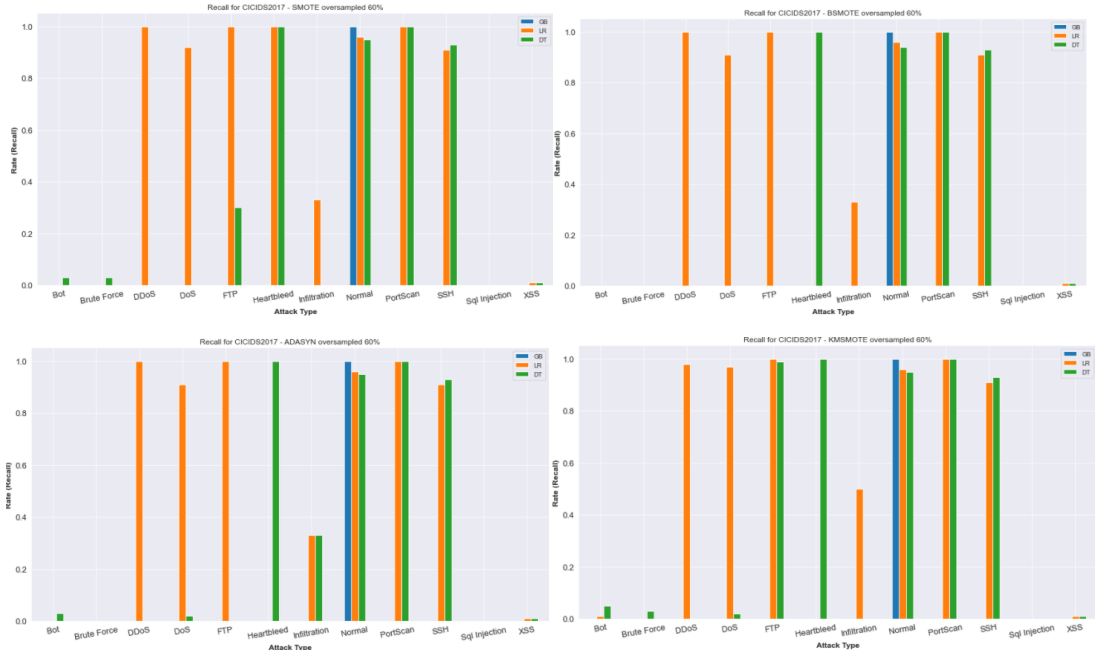
xi. Oversampled 60% detection rate

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
LR ROS	0	0	100	95	100	***100***	33	96	100	91	0	1
LR SMOTE	0	0	100	92	100	***100***	33	96	100	91	0	1
LR BSMOTE	0	0	100	91	100	0	33	96	100	91	0	1
LR ADASYN	0	0	100	91	100	0	33	96	100	91	0	1
LR KSMOTE	1	0	98	97	100	0	***50***	96	100	91	0	1
DT ROS	5	3	0	2	99	***100***	0	95	100	93	0	1
DT SMOTE	3	3	0	0	30	***100***	0	95	100	93	0	1
DT BSMOTE	0	0	0	0	0	***100***	0	94	100	93	0	1
DT ADASYN	3	0	0	2	0	***100***	33	95	100	93	0	1
DT KSMOTE	5	3	0	2	99	***100***	0	95	100	93	0	1

xii. Oversampled 60% comparison results



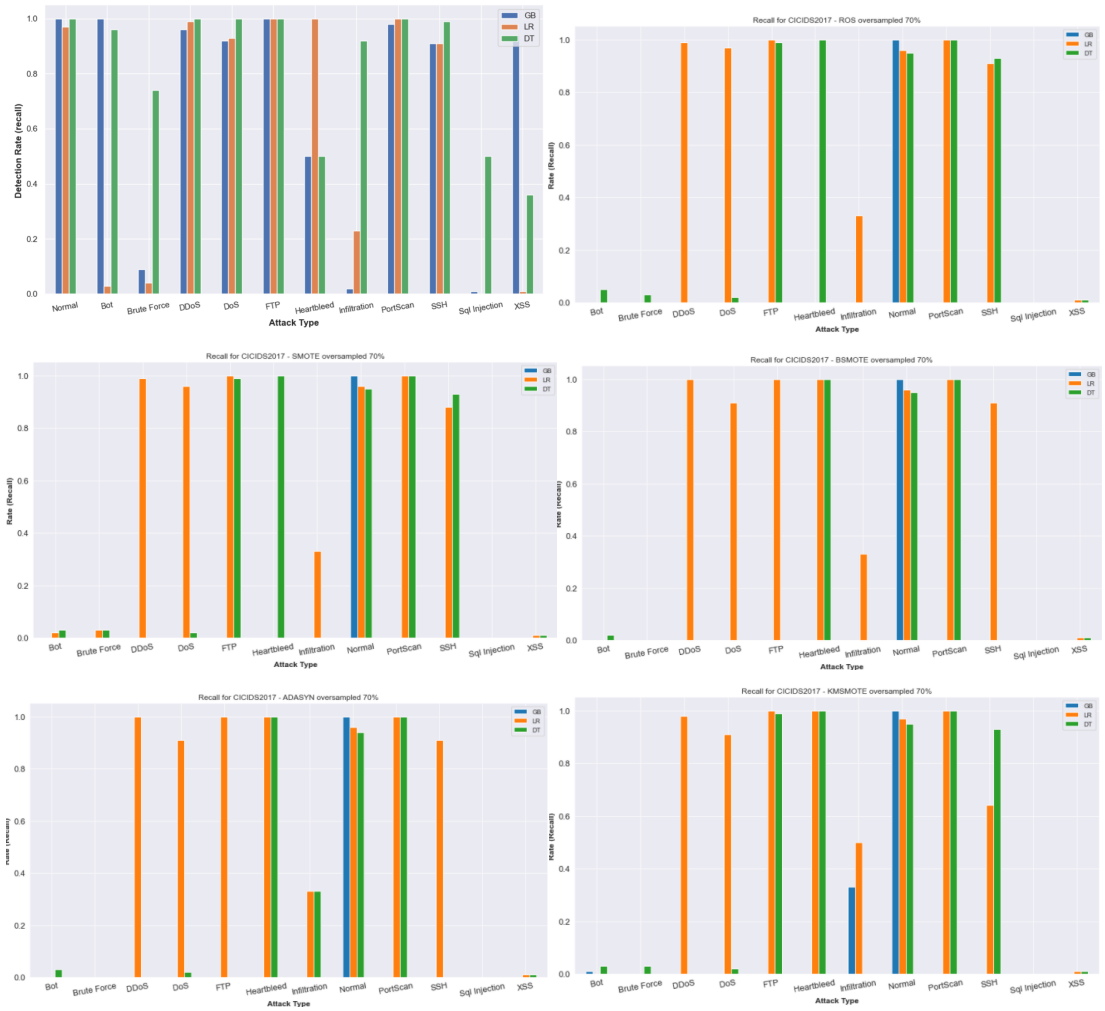




xiii. Oversampled 70% detection rate

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	33	100	0	0	0	0
LR ROS	0	0	99	97	100	0	33	96	100	91	0	1
LR SMOTE	2	3	99	96	100	0	33	96	100	88	0	1
LR BSMOTE	0	0	100	91	100	***100***	33	96	100	91	0	1
LR ADASYN	0	0	100	91	100	***100***	33	96	100	91	0	1
LR KSMOTE	0	0	98	91	100	***100***	***50***	97	100	64	0	1
DT ROS	5	3	0	2	99	***100***	0	95	100	93	0	1
DT SMOTE	3	3	0	2	99	***100***	0	95	100	93	0	1
DT BSMOTE	2	0	0	0	0	***100***	0	95	1	0	0	1
DT ADASYN	3	0	0	2	0	***100***	33	94	100	0	0	1
DT KSMOTE	3	3	0	2	99	***100***	0	95	100	93	0	1

xiv. Oversampled 70% comparison results

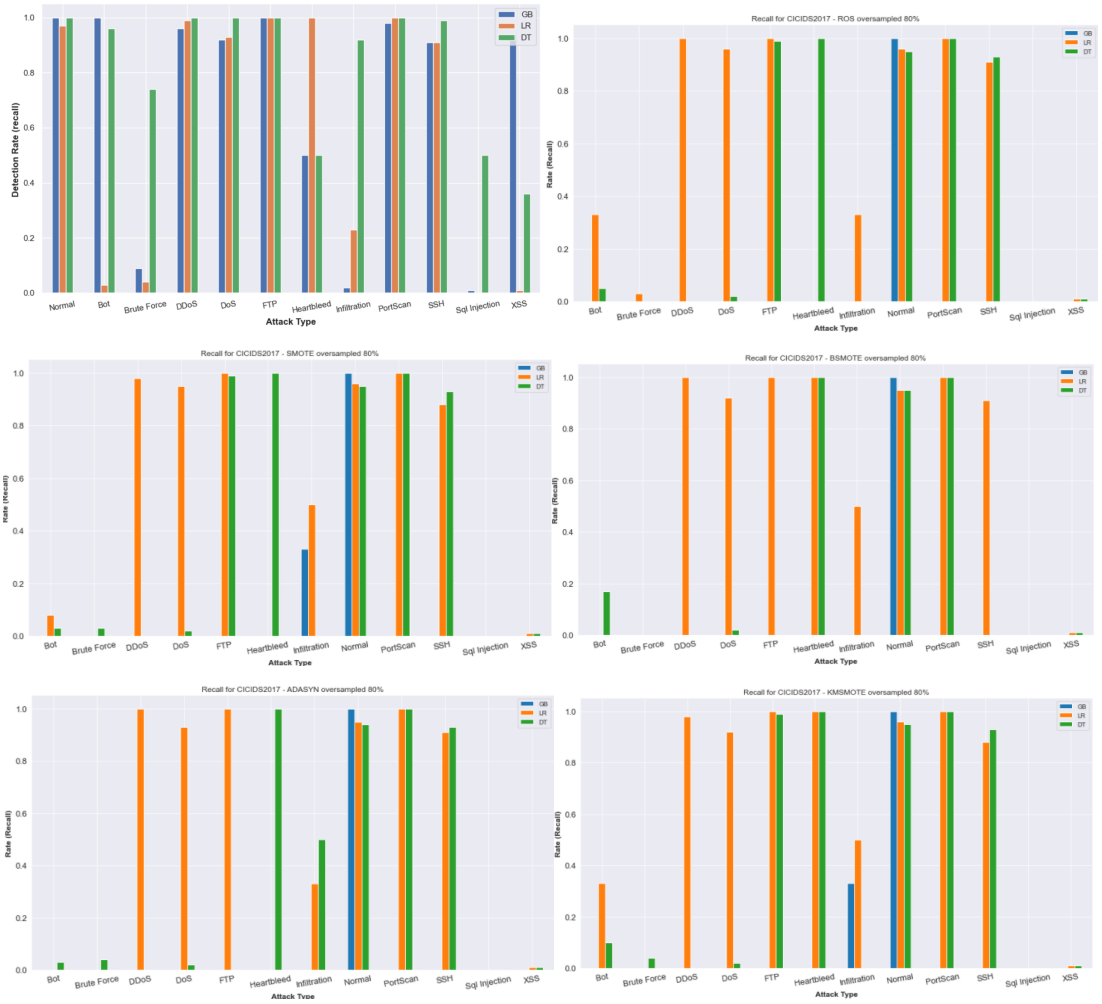


xv. Oversampled 80% detection rate

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	33	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	33	100	0	0	0	0
LR ROS	***33***	3	100	96	100	0	33	96	100	91	0	1

LR SMOTE	8	0	98	95	100	0	***50***	96	100	88	0	1
LR BSMOTE	0	0	100	92	100	***100***	***50***	95	100	91	0	1
LR ADASYN	0	0	100	93	100	0	33	95	100	91	0	1
LR KSMOTE	***33***	0	98	92	100	***100***	***50***	96	100	88	0	1
DT ROS	5	0	0	2	99	***100***	0	95	100	93	0	1
DT SMOTE	3	3	0	2	99	***100***	0	95	100	93	0	1
DT BSMOTE	17	0	0	2	0	***100***	0	95	100	0	0	1
DT ADASYN	3	4	0	2	0	***100***	***50***	94	100	93	0	1
DT KSMOTE	10	4	0	2	99	***100***	0	95	100	93	0	1

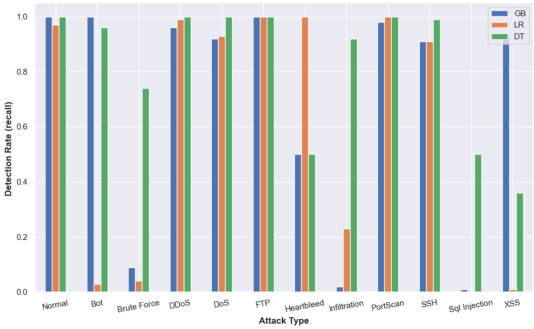
xvi. Oversampled 80% comparison results

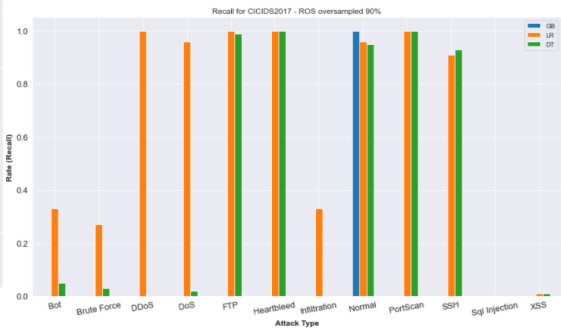


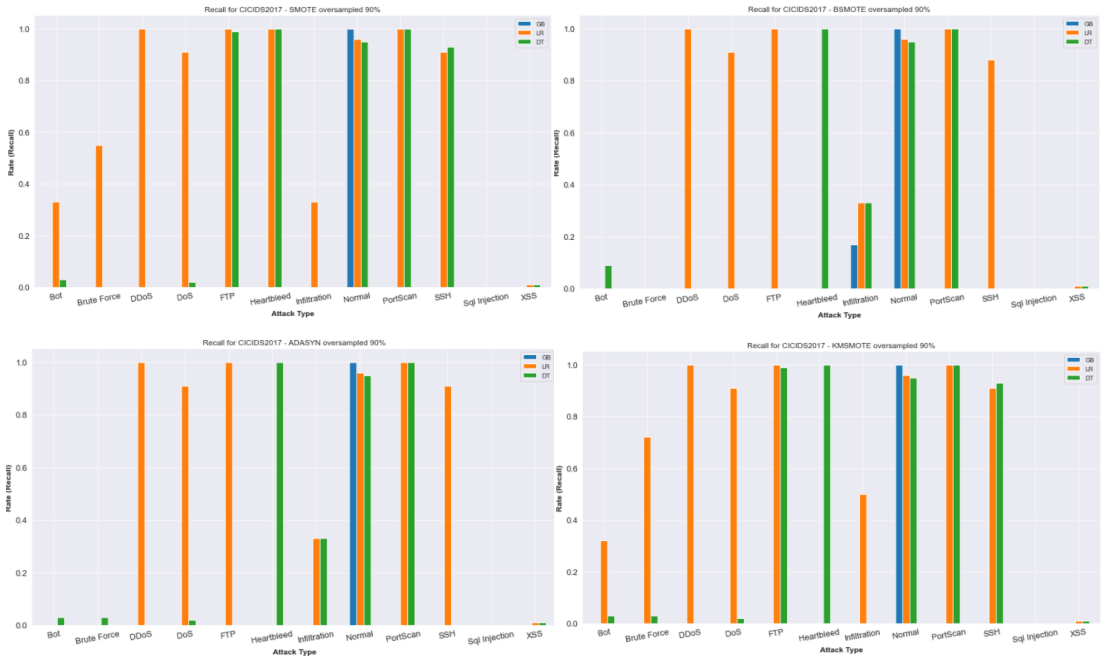
xvii. Oversampled 90% detection rate

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	17	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
LR ROS	***33***	27	100	96	100	***100***	33	96	100	91	0	1
LR SMOTE	***33***	55	100	91	100	***100***	33	96	100	91	0	1
LR BSMOTE	0	0	100	91	100	0	33	96	100	88	0	1
LR ADASYN	0	0	100	91	100	0	33	96	100	91	0	1
LR KSMOTE	32	***72***	100	91	100	0	***50***	96	100	91	0	1
DT ROS	5	3	0	2	99	***100***	0	95	100	93	0	1
DT SMOTE	3	0	0	2	99	***100***	0	95	100	93	0	1
DT BSMOTE	9	0	0	0	0	***100***	33	95	100	0	0	1
DT ADASYN	3	3	0	2	0	***100***	33	95	100	0	0	1
DT KSMOTE	3	3	0	2	99	***100***	0	95	100	93	0	1

xviii. Oversampled 90% comparison results



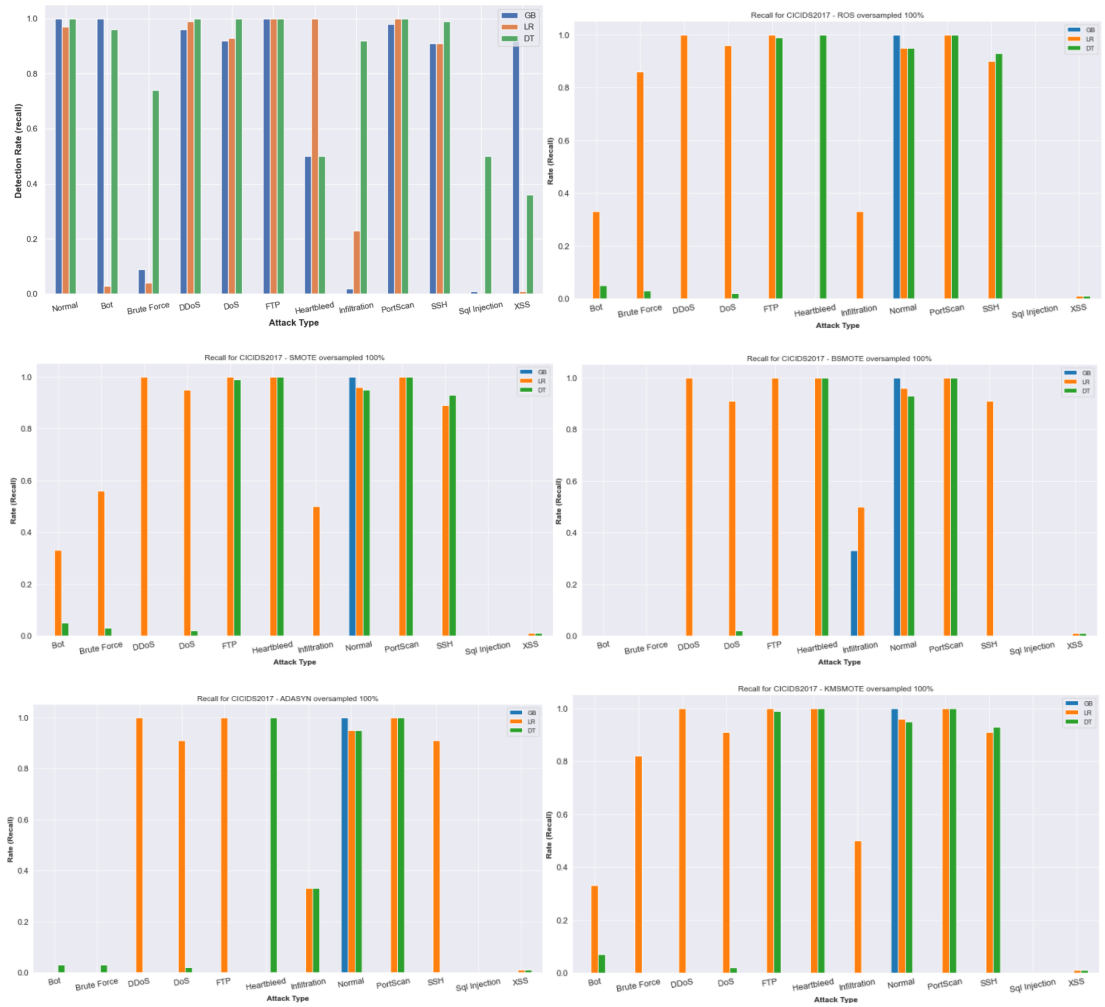




xix. Oversampled 100% detection rate

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	0	0	0	0	0	0	0	100	0	0	0	0
GB SMOTE	0	0	0	0	0	0	0	100	0	0	0	0
GB BSMOTE	0	0	0	0	0	0	33	100	0	0	0	0
GB ADASYN	0	0	0	0	0	0	0	100	0	0	0	0
GB KSMOTE	0	0	0	0	0	0	0	100	0	0	0	0
LR ROS	***33***	***86***	100	96	100	0	33	95	100	90	0	1
LR SMOTE	***33***	56	100	95	100	100	***50***	96	100	89	0	1
LR BSMOTE	0	0	100	91	100	100	***50***	96	100	91	0	1
LR ADASYN	0	0	100	91	100	0	33	95	100	91	0	1
LR KSMOTE	***33***	82	100	91	100	100	***50***	96	100	91	0	1
DT ROS	5	3	0	2	99	***100***	0	95	100	93	0	1
DT SMOTE	5	3	0	2	99	***100***	0	95	100	93	0	1
DT BSMOTE	0	0	0	2	0	***100***	0	93	100	0	0	1
DT ADASYN	3	3	0	2	0	***100***	33	95	100	0	0	1
DT KSMOTE	7	0	0	2	99	***100***	0	95	100	93	0	1

xx. Oversampled 100% comparison results



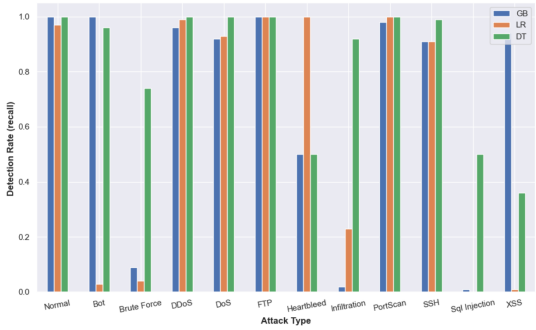
APPENDIX N: CICIDS 2017 oversampling results on non-scaled data

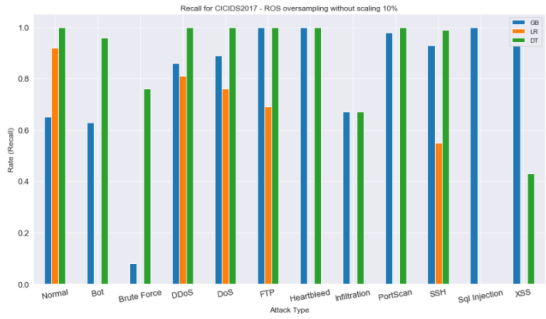
i. Oversampled 10% detection rate – non-scaled

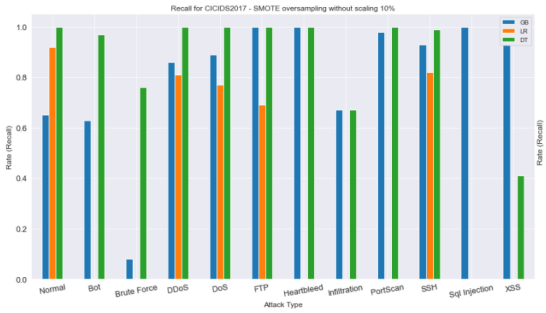
Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	63	8	86	89	100	***100***	67	65	98	93	***100***	93
GB SMOTE	63	8	86	89	100	***100***	67	65	98	93	***100***	93
GB BSMOTE	64	7	86	89	100	***100***	67	65	98	93	***100***	***94***
GB ADASYN	64	7	86	89	100	***100***	67	65	98	93	***100***	***94***

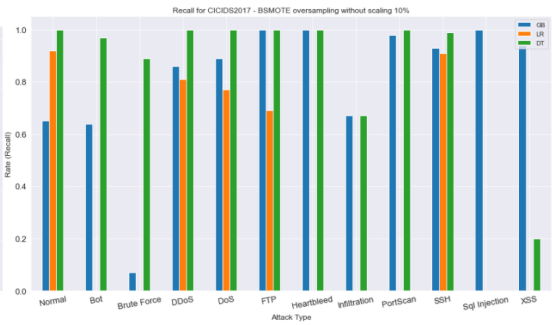
GB KSMOTE	64	63	86	89	100	***100***	67	65	98	93	***100***	39
LR ROS	0	0	81	76	69	0	0	92	0	55	0	0
LR SMOTE	0	0	81	77	69	0	0	92	0	82	0	0
LR BSMOTE	0	0	81	77	69	0	0	92	0	91	0	0
LR ADASYN	0	0	81	77	69	0	0	92	0	91	0	0
LR KSMOTE	0	0	81	77	69	0	0	92	0	86	0	0
DT ROS	96	76	100	100	100	***100***	67	100	100	99	0	43
DT SMOTE	***97***	76	100	100	100	***100***	67	100	100	99	0	41
DT BSMOTE	***97***	***89***	100	100	100	***100***	67	100	100	99	0	20
DT ADASYN	***97***	73	100	100	100	***100***	***100***	100	100	99	***40***	41
DT KSMOTE	***97***	75	100	100	100	***100***	17	100	100	99	0	42

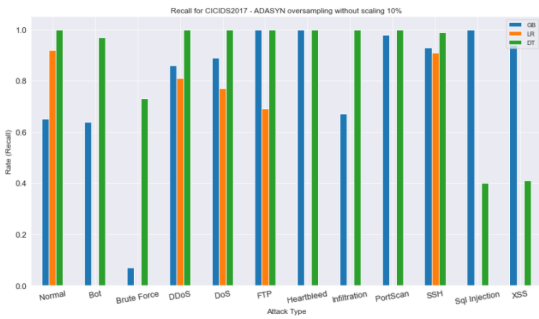
ii. Oversampled 10% comparison results – non-scaled

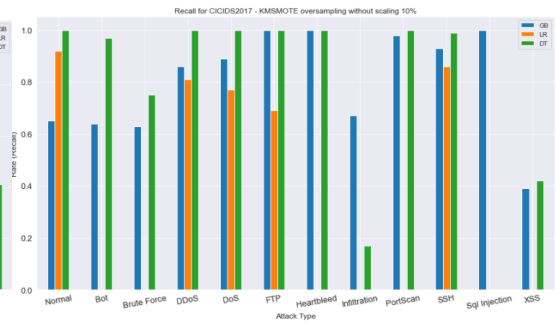








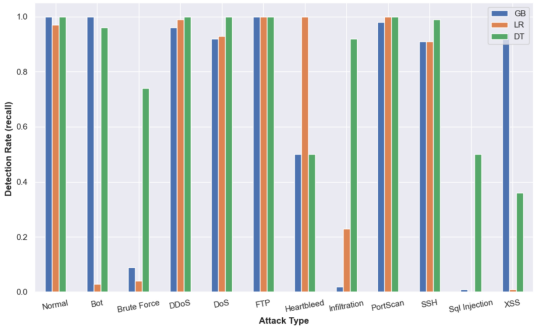


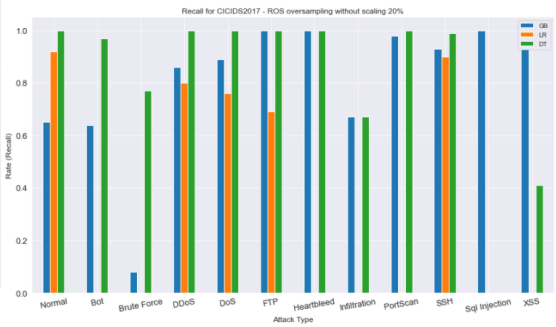


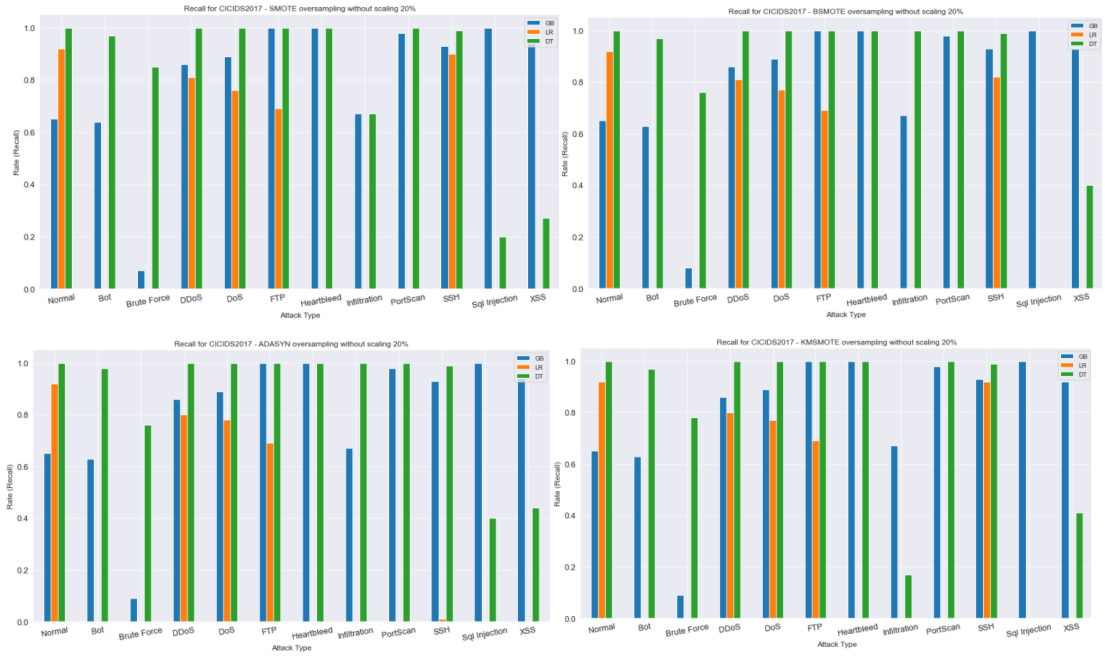
iii. Oversampled 20% detection rate – non-scaled

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	64	8	86	89	100	***100***	67	65	98	93	***100***	93
GB SMOTE	64	8	86	89	100	***100***	67	65	98	93	***100***	93
GB BSMOTE	63	8	86	89	100	***100***	67	65	98	93	***100***	93
GB ADASYN	63	9	86	89	100	***100***	67	65	98	93	***100***	*94*
GB KSMOTE	63	9	86	89	100	***100***	67	65	98	93	***100***	92
LR ROS	0	0	80	76	69	0	0	92	0	90	0	0
LR SMOTE	0	0	80	76	69	0	0	92	0	90	0	0
LR BSMOTE	0	0	81	77	69	0	0	92	0	82	0	0
LR ADASYN	0	0	80	78	69	0	0	92	0	1	0	0
LR KSMOTE	0	0	80	77	69	0	0	92	0	92	0	0
DT ROS	97	77	100	100	100	***100***	67	100	100	99	0	41
DT SMOTE	97	***85***	100	100	100	***100***	67	100	100	99	20	27
DT BSMOTE	97	76	100	100	100	***100***	***100***	100	100	99	0	40
DT ADASYN	***98***	76	100	100	100	***100***	***100***	100	100	99	40	44
DT KSMOTE	97	78	100	100	100	***100***	17	100	100	99	0	41

iv. Oversampled 20% comparison results – non-scaled



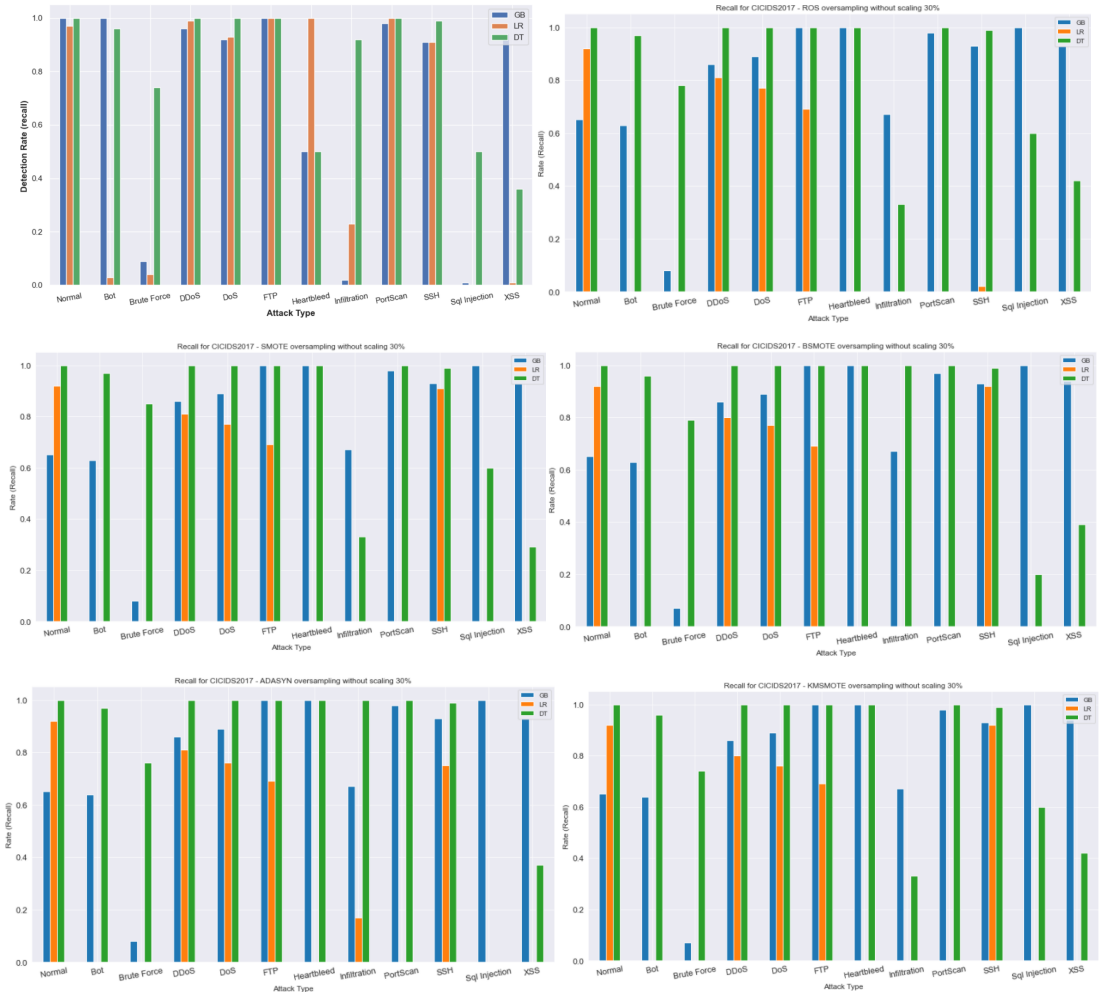




v. Oversampled 30% detection rate – non-scaled

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	63	8	86	89	100	***100***	67	65	98	93	***100***	93
GB SMOTE	63	8	86	89	100	***100***	67	65	98	93	***100***	93
GB BSMOTE	63	7	86	89	100	***100***	67	65	97	93	***100***	***94***
GB ADASYN	64	8	86	89	100	***100***	67	65	98	93	***100***	93
GB KSMOTE	64	7	86	89	100	***100***	67	65	98	93	***100***	***94***
LR ROS	0	0	81	77	69	0	0	92	0	2	0	0
LR SMOTE	0	0	81	77	69	0	0	92	0	91	0	0
LR BSMOTE	0	0	80	77	69	0	0	92	0	92	0	0
LR ADASYN	0	0	81	76	69	0	17	92	0	75	0	0
LR KSMOTE	0	0	80	76	69	0	0	92	0	92	0	0
DT ROS	***97***	78	100	100	100	***100***	33	100	100	99	60	42
DT SMOTE	***97***	***85***	100	100	100	***100***	33	100	100	99	60	29
DT BSMOTE	96	79	100	100	100	***100***	***100***	100	100	99	20	39
DT ADASYN	***97***	76	100	100	100	***100***	***100***	100	100	99	0	37
DT KSMOTE	96	74	100	100	100	***100***	33	100	100	99	60	42

vi. Oversampled 30% comparison results – non-scaled

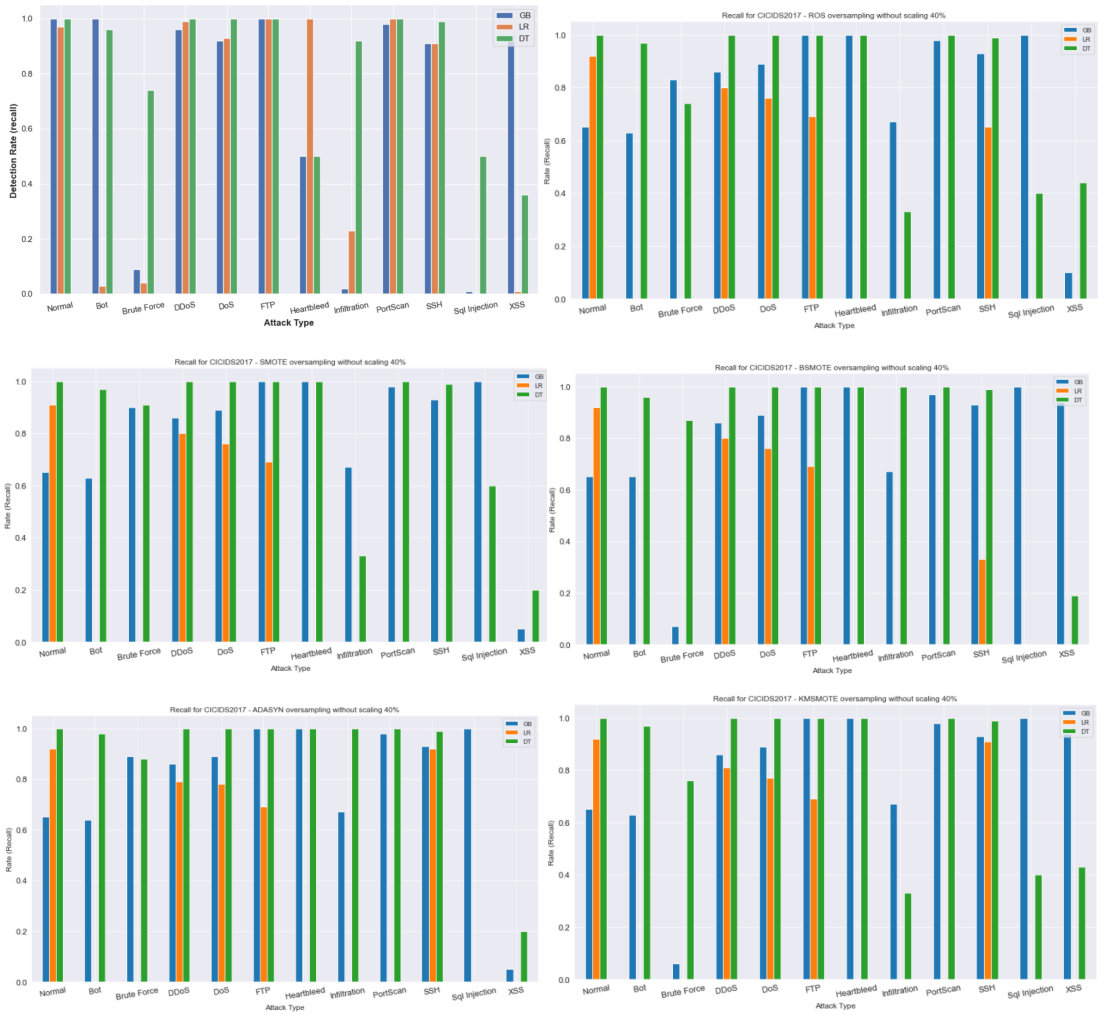


vii. Oversampled 40% detection rate – non-scaled

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	63	83	86	89	100	***100***	67	65	98	93	***100***	10
GB SMOTE	63	90	86	89	100	***100***	67	65	98	93	***100***	5
GB BSMOTE	65	7	86	89	100	***100***	67	65	97	93	***100***	***94***
GB ADASYN	64	89	86	89	100	***100***	67	65	98	93	***100***	5
GB KSMOTE	63	6	86	89	100	***100***	67	65	98	93	***100***	***94***
LR ROS	0	0	80	76	69	0	0	92	0	65	0	0
LR SMOTE	0	0	80	76	69	0	0	91	0	0	0	0

LR BSMOTE	0	0	80	76	69	0	0	92	0	33	0	0
LR ADASYN	0	0	79	78	69	0	0	92	0	92	0	0
LR KSMOTE	0	0	81	77	69	0	0	92	0	91	0	0
DT ROS	97	74	100	100	100	***100***	33	100	100	99	40	44
DT SMOTE	97	***91***	100	100	100	***100***	33	100	100	99	60	20
DT BSMOTE	96	87	100	100	100	***100***	***100***	100	100	99	0	19
DT ADASYN	***98***	88	100	100	100	***100***	***100***	100	100	99	0	20
DT KSMOTE	97	76	100	100	100	***100***	33	100	100	99	40	43

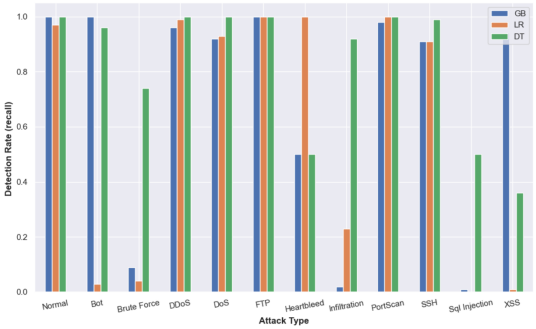
viii. Oversampled 40% comparison results – non-scaled

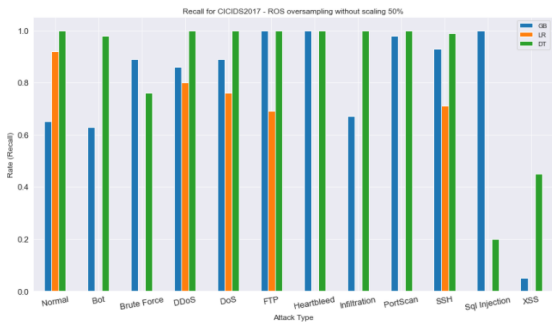


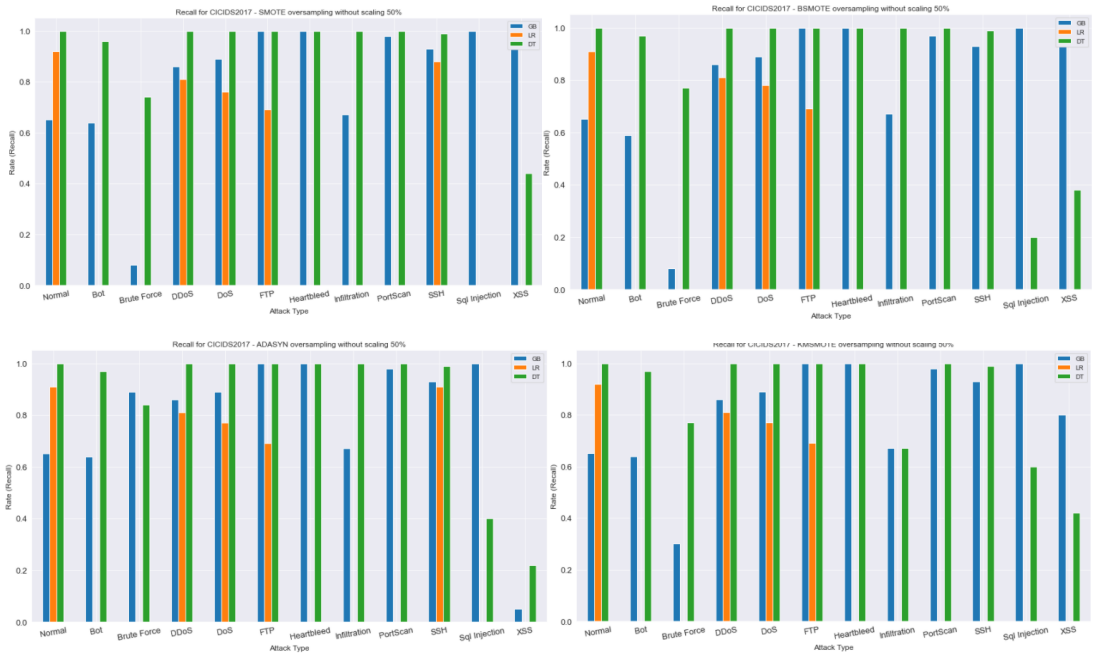
ix. Oversampled 50% detection rate – non-scaled

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	63	***89***	86	89	100	***100***	67	65	98	93	***100***	5
GB SMOTE	64	8	86	89	100	***100***	67	65	98	93	***100***	***93***
GB BSMOTE	59	8	86	89	100	***100***	67	65	97	93	***100***	***93***
GB ADASYN	64	***89***	86	89	100	***100***	67	65	98	93	***100***	5
GB KSMOTE	64	30	86	89	100	***100***	67	65	98	93	***100***	80
LR ROS	0	0	80	76	69	0	0	92	0	71	0	0
LR SMOTE	0	0	81	76	69	0	0	92	0	88	0	0
LR BSMOTE	0	0	81	78	69	0	0	91	0	0	0	0
LR ADASYN	0	0	81	77	69	0	0	91	0	91	0	0
LR KSMOTE	0	0	81	77	69	0	0	92	0	0	0	0
DT ROS	***98***	76	100	100	100	***100***	***100***	100	100	99	20	45
DT SMOTE	96	74	100	100	100	***100***	***100***	100	100	99	0	44
DT BSMOTE	97	77	100	100	100	***100***	***100***	100	100	99	20	38
DT ADASYN	97	84	100	100	100	***100***	***100***	100	100	99	40	22
DT KSMOTE	97	77	100	100	100	***100***	67	100	100	99	60	42

x. Oversampled 50% comparison results – non-scaled



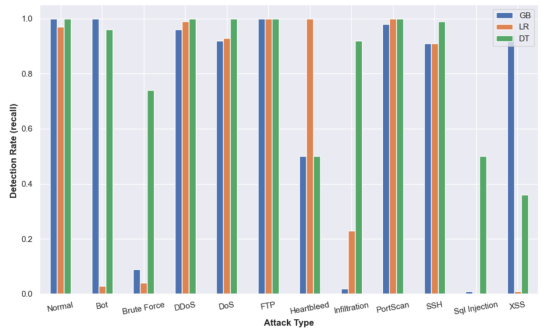


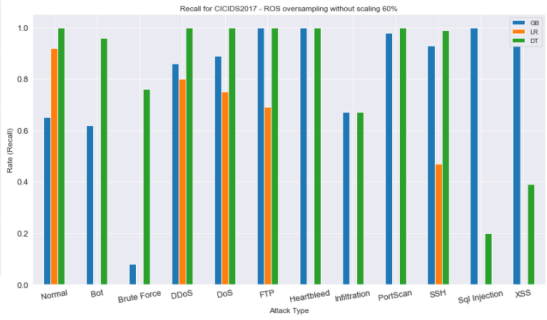


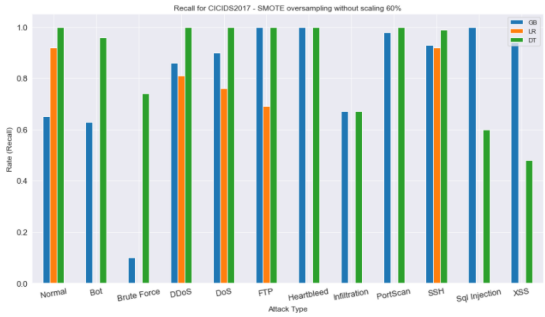
xi. Oversampled 60% detection rate – non-scaled

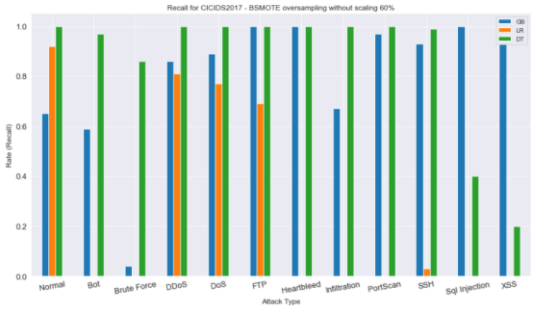
Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	62	8	86	89	100	***100***	67	65	98	93	***100***	93
GB SMOTE	63	10	86	90	100	***100***	67	65	98	93	***100***	***94***
GB BSMOTE	59	4	86	89	100	***100***	67	65	97	93	***100***	93
GB ADASYN	64	8	86	89	100	***100***	67	65	98	93	***100***	***94***
GB KSMOTE	63	6	86	89	100	***100***	67	65	98	93	***100***	***94***
LR ROS	0	0	80	75	69	0	0	92	0	47	0	0
LR SMOTE	0	0	81	76	69	0	0	92	0	92	0	0
LR BSMOTE	0	0	81	77	69	0	0	92	0	3	0	0
LR ADASYN	0	0	81	77	69	0	0	92	0	92	0	0
LR KSMOTE	0	0	80	76	69	0	0	92	0	92	0	0
DT ROS	96	76	100	100	100	***100***	67	100	100	99	20	39
DT SMOTE	96	74	100	100	100	***100***	67	100	100	99	60	48
DT BSMOTE	97	***86***	100	100	100	***100***	***100***	100	100	99	40	20
DT ADASYN	***99***	73	100	100	100	***100***	***100***	100	100	99	20	44
DT KSMOTE	98	75	100	100	100	***100***	67	100	100	99	20	38

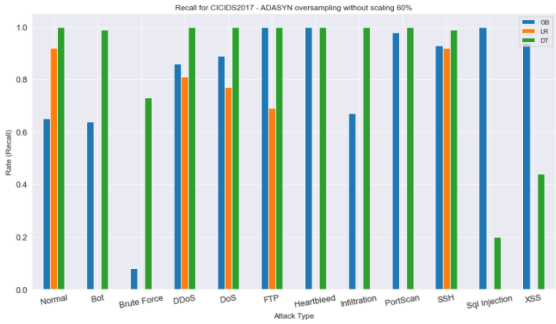
xii. Oversampled 60% comparison results – non-scaled

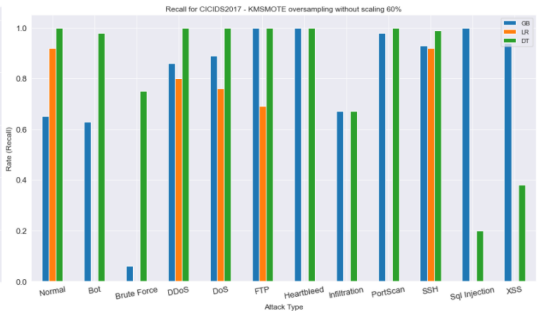










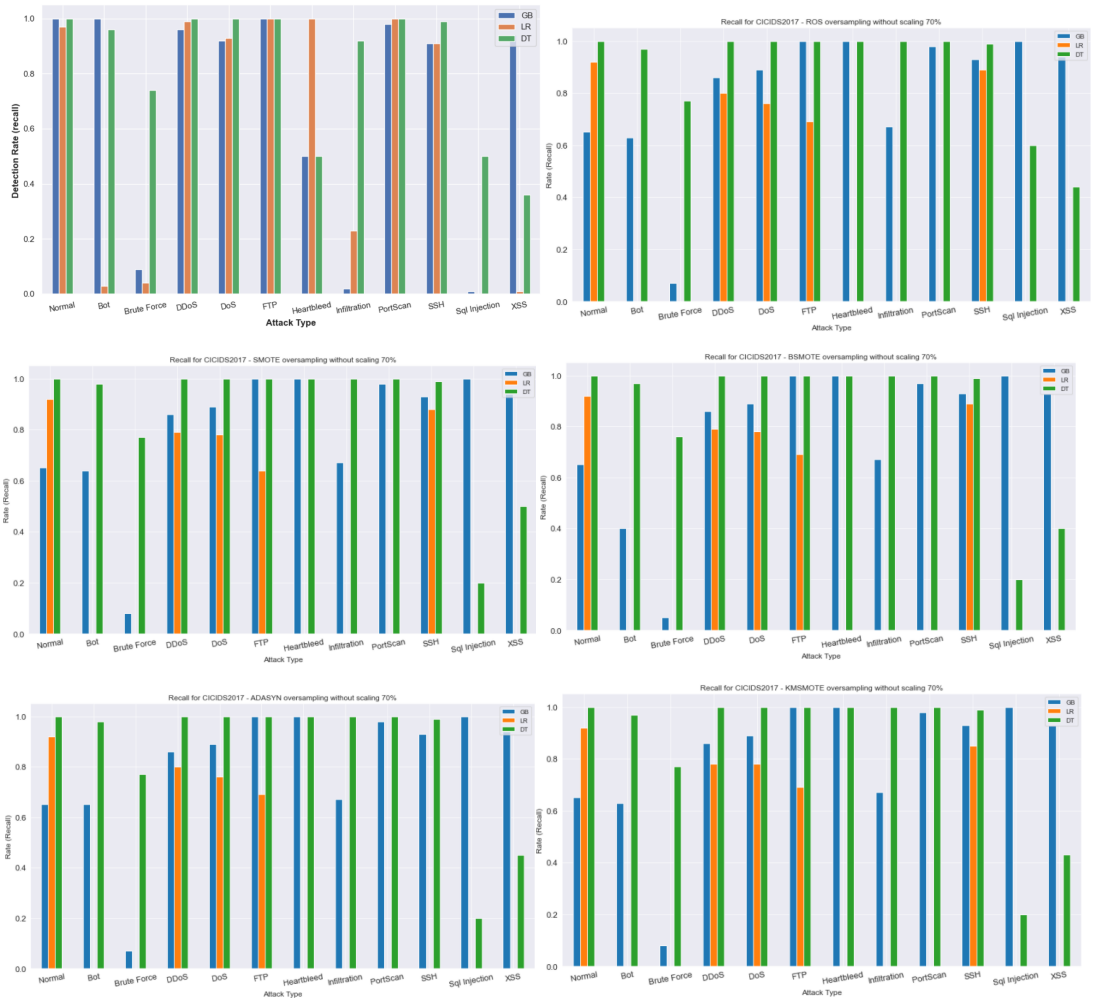


xiii. Oversampled 70% detection rate – non-scaled

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	63	7	86	89	100	***100***	67	65	98	93	***100***	***94***
GB SMOTE	64	8	86	89	100	***100***	67	65	98	93	***100***	***94***
GB BSMOTE	40	5	86	89	100	***100***	67	65	97	93	***100***	93
GB ADASYN	65	7	86	89	100	***100***	67	65	98	93	***100***	***94***
GB KSMOTE	63	8	86	89	100	***100***	67	65	98	93	***100***	93
LR ROS	0	0	80	76	69	0	0	92	0	89	0	0

LR SMOTE	0	0	79	78	64	0	0	92	0	88	0	0
LR BSMOTE	0	0	79	78	69	0	0	92	0	89	0	0
LR ADASYN	0	0	80	76	69	0	0	92	0	0	0	0
LR KSMOTE	0	0	78	78	69	0	0	92	0	85	0	0
DT ROS	97	***77***	100	100	100	***100***	***100***	100	100	99	60	44
DT SMOTE	0	0	79	78	64	0	0	92	0	88	0	0
DT BSMOTE	97	76	100	100	100	***100***	***100***	100	100	99	20	40
DT ADASYN	***98***	***77***	100	100	100	***100***	***100***	100	100	100	20	45
DT KSMOTE	97	***77***	100	100	100	***100***	***100***	100	100	99	20	43

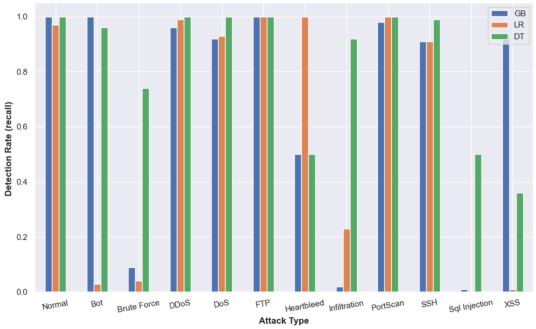
xiv. Oversampled 70% comparison results – non-scaled

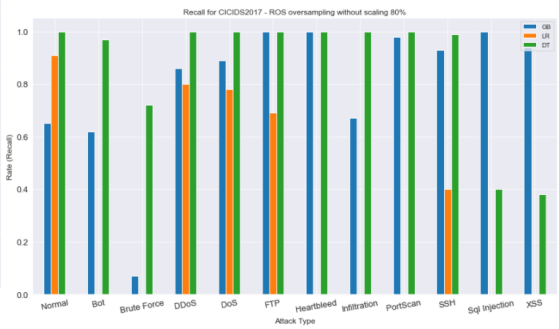


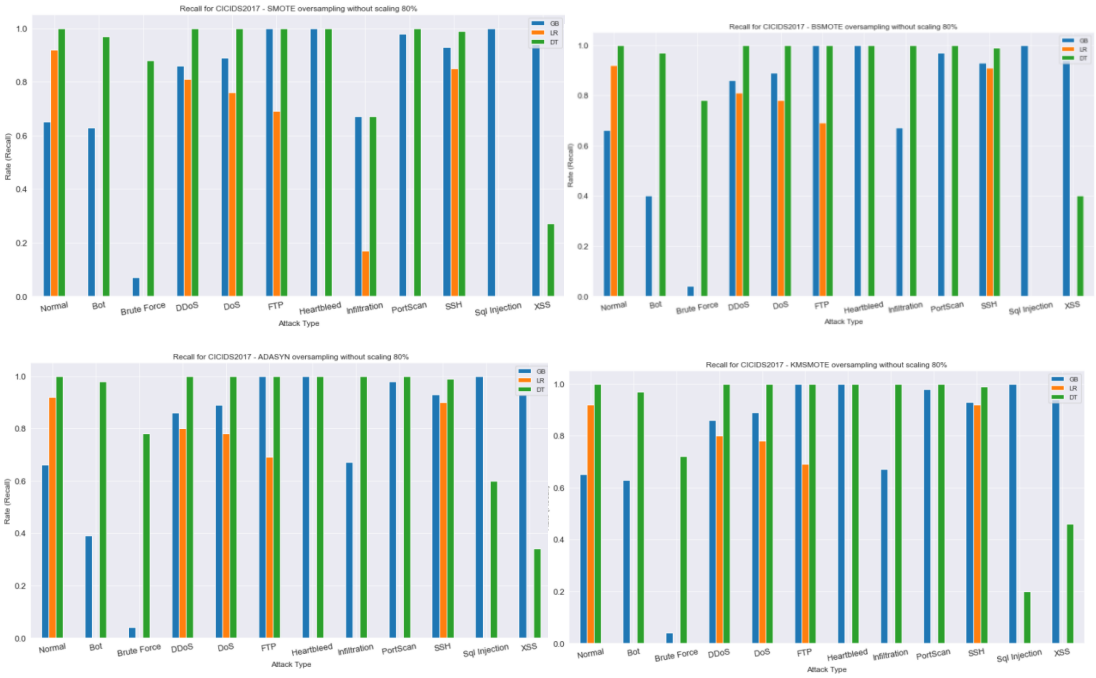
xv. Oversampled 80% detection rate – non-scaled

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	62	7	86	89	100	***100***	67	65	98	93	***100***	***94***
GB SMOTE	63	7	86	89	100	***100***	67	65	98	93	***100***	***94***
GB BSMOTE	40	4	86	89	100	***100***	67	66	97	93	***100***	***94***
GB ADASYN	39	4	86	89	100	***100***	67	66	98	93	***100***	93
GB KSMOTE	63	4	86	89	100	***100***	67	65	98	93	***100***	***94***
LR ROS	0	0	80	78	69	0	0	91	0	40	0	0
LR SMOTE	0	0	81	76	69	0	17	92	0	85	0	0
LR BSMOTE	0	0	81	78	69	0	0	92	0	91	0	0
LR ADASYN	0	0	80	78	69	0	0	92	0	90	0	0
LR KSMOTE	0	0	80	78	69	0	0	92	0	92	0	0
DT ROS	97	72	100	100	100	***100***	***100***	100	100	99	40	38
DT SMOTE	97	***88***	100	100	100	***100***	67	100	100	99	0	27
DT BSMOTE	97	78	100	100	100	***100***	***100***	100	100	99	0	40
DT ADASYN	***98***	78	100	100	100	***100***	***100***	100	100	99	60	34
DT KSMOTE	97	72	100	100	100	***100***	***100***	100	100	99	20	46

xvi. Oversampled 80% comparison results – non-scaled





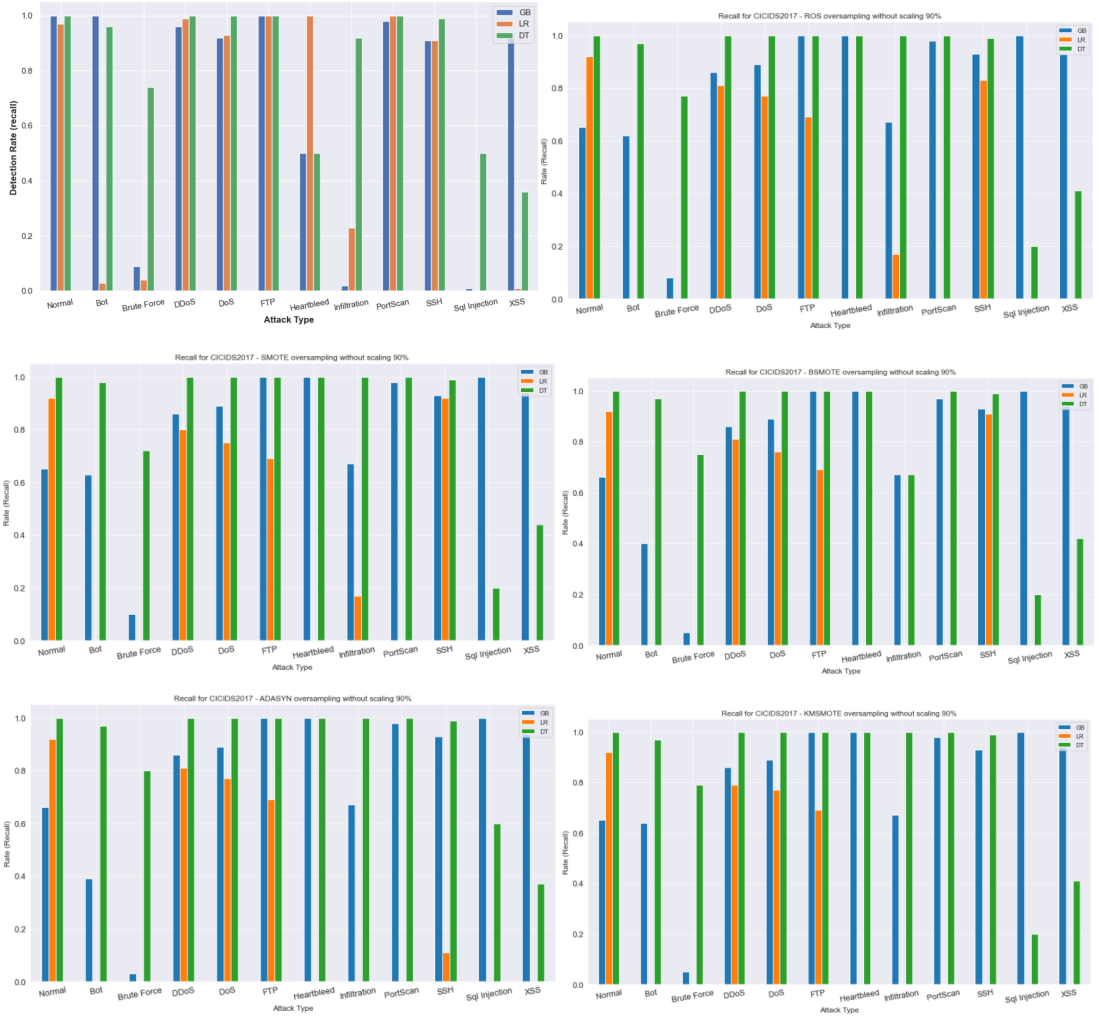


xvii. Oversampled 90% detection rate – non-scaled

Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	62	8	86	89	100	***100***	67	65	98	93	***100***	93
GB SMOTE	63	10	86	89	100	***100***	67	65	98	93	***100***	***94***
GB BSMOTE	40	5	86	89	100	***100***	67	66	97	93	***100***	***94***
GB ADASYN	39	3	86	89	100	***100***	67	66	98	93	***100***	***94***
GB KSMOTE	64	5	86	89	100	***100***	67	65	98	93	***100***	***94***
LR ROS	0	0	81	77	69	0	17	92	0	83	0	0
LR SMOTE	0	0	80	75	69	0	17	92	0	92	0	0
LR BSMOTE	0	0	81	76	69	0	0	92	0	91	0	0
LR ADASYN	0	0	81	77	69	0	0	92	0	11	0	0
LR KSMOTE	0	0	79	77	69	0	0	92	0	0	0	0
DT ROS	97	77	100	100	100	***100***	***100***	100	100	99	20	41
DT SMOTE	***98***	72	100	100	100	***100***	***100***	100	100	99	20	44
DT BSMOTE	97	75	100	100	100	***100***	67	100	100	99	20	42
DT ADASYN	97	***80***	100	100	100	***100***	***100***	100	100	99	60	37

DT KSMOTE	97	79	100	100	100	***100***	***100***	100	100	99	20	41

xviii. Oversampled 90% comparison results – non-scaled

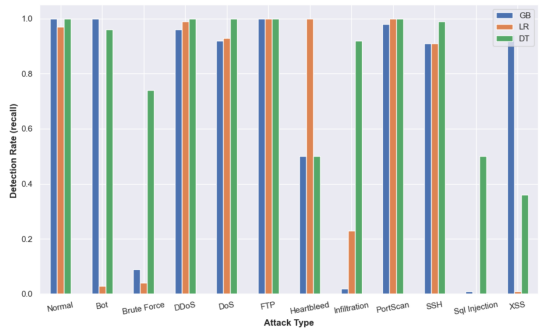


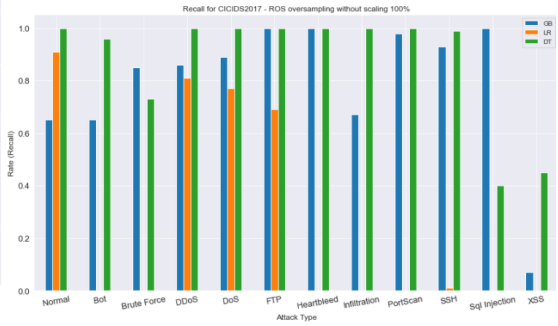
xix. Oversampled 100% detection rate – non-scaled

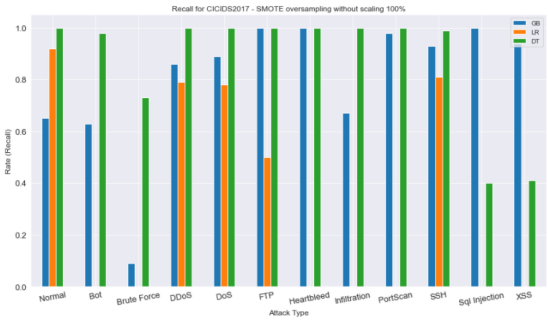
Result	Bot	Brute Force	DDoS	DoS	FTP	Heart bleed	Infiltration	Norm	Port Scan	SSH	SQL	XSS
GB	100	9	96	92	100	50	2	100	98	91	1	94
LR	3	4	99	93	100	100	23	97	100	91	0	1
DR	95	75	100	100	100	50	92	100	100	99	50	36
GB ROS	65	85	86	89	100	***100***	67	65	98	93	***100***	7
GB SMOTE	63	9	86	89	100	***100***	67	65	98	93	***100***	***94***
GB BSMOTE	40	4	86	89	100	***100***	67	66	97	93	***100***	***94***
GB ADASYN	38	5	86	89	100	***100***	67	66	98	93	***100***	93
GB KSMOTE	63	7	86	89	100	***100***	67	65	98	93	***100***	***94***

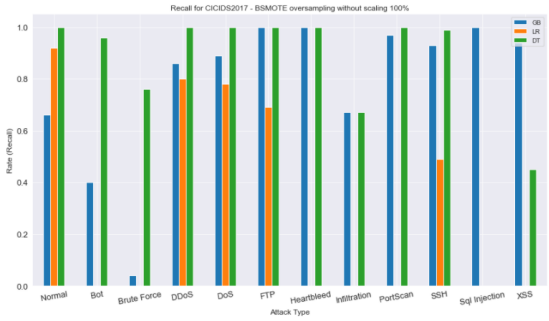
LR ROS	0	0	81	77	69	0	0	91	0	1	0	0
LR SMOTE	0	0	79	78	50	0	0	92	0	81	0	0
LR BSMOTE	0	0	80	78	69	0	0	92	0	49	0	0
LR ADASYN	0	0	81	78	69	0	0	92	0	2	0	0
LR KSMOTE	0	0	80	76	69	0	0	92	0	82	0	0
DT ROS	96	73	100	100	100	***100***	***100***	100	100	99	40	45
DT SMOTE	***98***	73	100	100	100	***100***	***100***	100	100	99	40	41
DT BSMOTE	96	76	100	100	100	***100***	67	100	100	99	0	45
DT ADASYN	***98***	***91***	100	100	100	***100***	***100***	100	100	99	60	16
DT KSMOTE	97	79	100	100	100	***100***	67	100	100	99	20	39

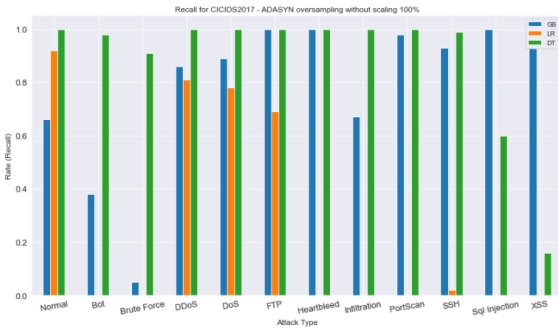
xx. Oversampled 100% comparison results – non-scaled

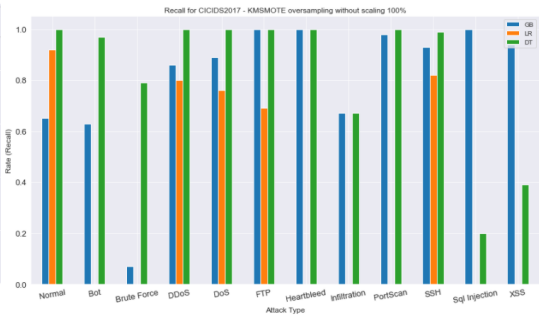










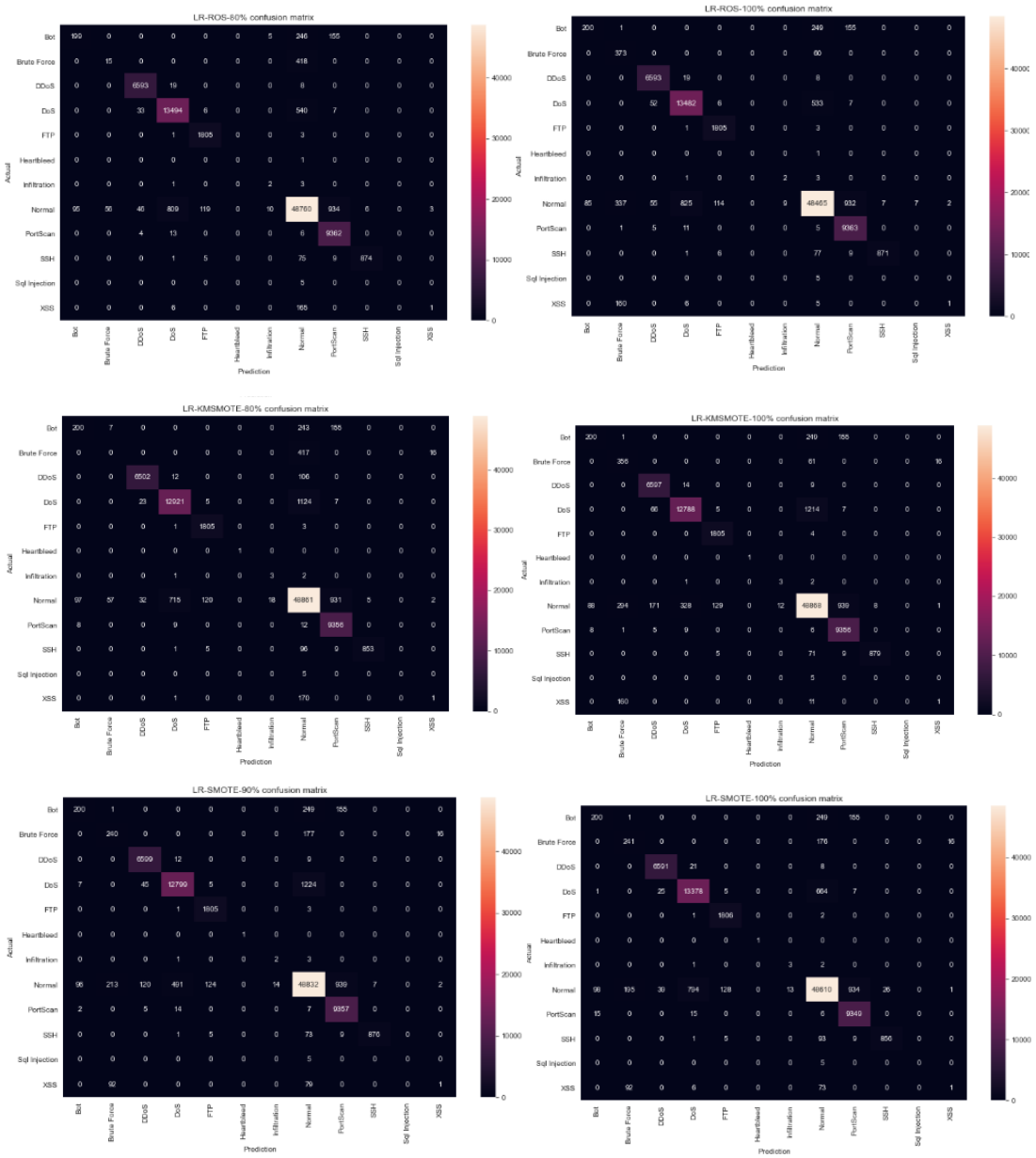


APPENDIX O: Top 3 detection rate among all observations for CICIDS 2017

	BOT		BRUTE FORCE		HEARTBLEED	
	Combination	**DR**	**Combination**	**DR**	**Combination**	**DR**
	Original	3	Original	9	Original	50
S1	LR-ROS-80%, 90%, 100% LR-KMSMOTE-80%, 100% LR-SMOTE-90%, 100%	33	LR-ROS-100%	86	DT-All-10% to 40%, 70% to 100%	***100***
S2	DT-ADASYN-30%	26	LR-KMSMOTE-90%	72		
S3	DT-ADASYN-50%	20	LR-KMSMOTE-100%	56		
NS1	DT-ADASYN-60%	***99***	DT-SMOTE-40% DT-ADASYN-100%	***91***	GB-All-10% to 100%	***100***
NS2	DT-ADASYN-20%, 40%, 70%, 80%, 100% DT-ROS-50% DT-KMSMOTE-60% DT-SMOTE-90%, 100%	98	DT-BSMOTE-10% GB-ROS-50% GB-ADASYN-50%	89		
NS3	DT-All-10% no ROS DT-All-20% no ADY DT-ROS-30%, 40% DT-SMOTE-30%, 40% DT-ADASYN-30% DT-KMSMOTE-40% DT-All-50% no ROS	97	DT-SMOTE-80%	88		

	INFILTRATION		SQL INJECTION		XSS	
	Combination	**DR**	**Combination**	**DR**	**Combination**	**DR**
	Original	23	Original	50	Original	36
S1	DT-BSMOTE-50%	***100***	LR-KMSMOTE-60%	50		
S2	LR-KMSMOTE-70%, 80%, 90%, 100% LR-SMOTE-80%, 100% LR-BSMOTE-80%, 100% DT-ADASYN-80%	50	DT-BSMOTE-40%, 50%	40		
S3	LR-All-20% DT-KMSMOTE-20% LR-All-30% LR-All-70% (x KSMOTE) LR-ROS-100% LR-ADASYN-100%	33				
NS1	DT-ADASYN-10% to 100% DT-BSMOTE-20%, 30%, 40%, 60% DT-All-50% no KMSMOTE DT-All-70%, 80% no SMOTE DT-All-90% no BSMOTE DT-ROS-100% DT-SMOTE-100%	***100***	GB-All-10% to 100%	***100***	GB-BSMOTE-10%, 30%, 40%, 100% GB-ADASYN-10%, 20%, 60%, 70% GB-KMSMOTE-30%, 40% , 60%, 100% GB-SMOTE-60%, 70%, 100% GB-ROS-70% GB-All-80% no ROS	***94***
NS2			DT-ROS-30%, 70% DT-SMOTE-30%, 60% DT-KMSMOTE-30%, 50% DT-ADASYN-80%, 90%	60	GB-ROS-10%, 30% GB-SMOTE-10%, 20%, 50% GB-BSMOTE-20%, 50%	93
NS3			DT-ADASYN-20%, 50% DT-BSMOTE-60% DT-ROS-80%	40	GB-KMSMOTE-20%	92

APPENDIX P: CICIDS 2017 reference confusion matrix



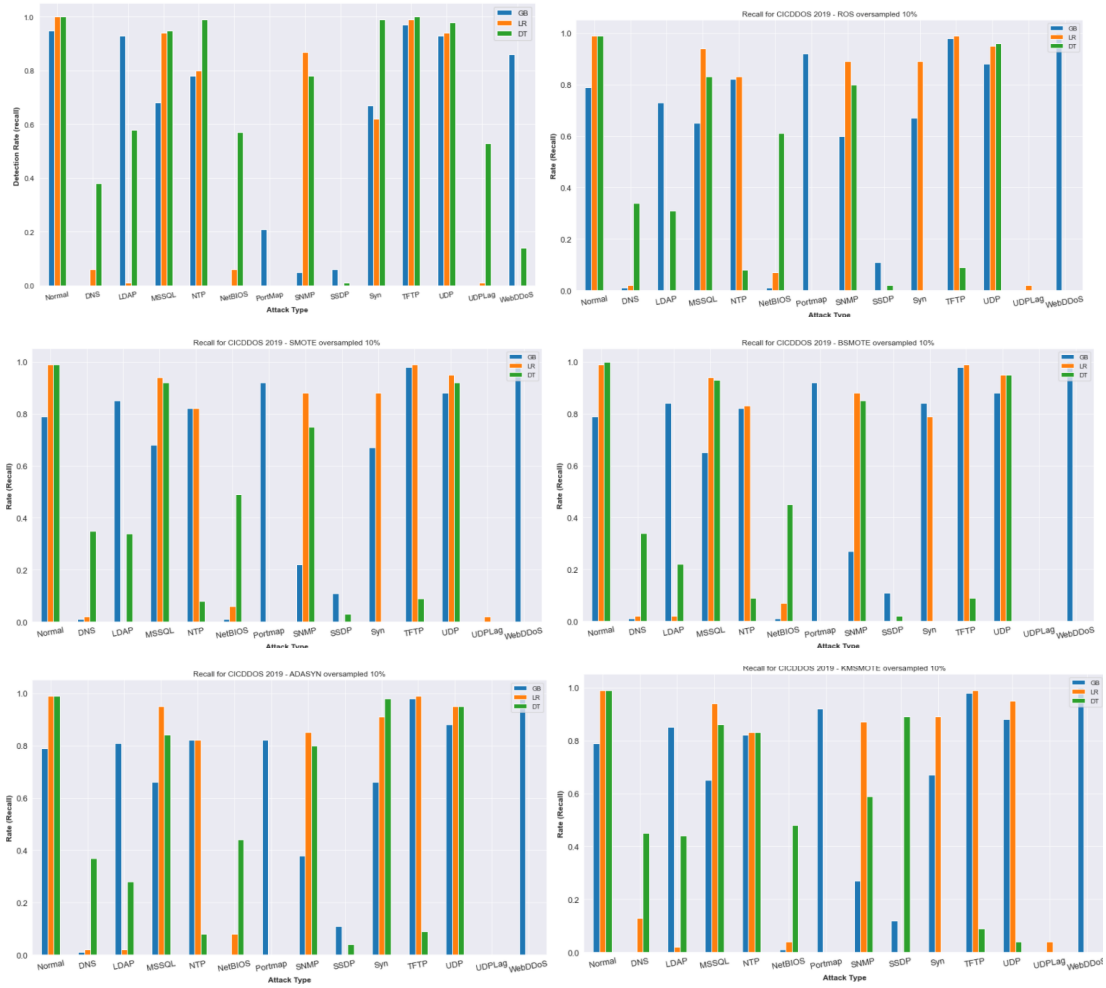
APPENDIX Q: CICDDOS 2019 oversampling individual DR on scaled data

i. Oversampled 10% detection rate

Result	DNS	LDAP	MSSQL	NTP	NetBIOS	Nor	PortMap	SNMP	SSDP	SYN	TFTP	UDP	ULag	WDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	1	73	65	82	1	79	***92***	6	11	67	98	88	0	***100***
GB SMOTE	1	***85***	68	82	1	79	***92***	22	11	67	98	88	0	***100***

GB BSMOTE	1	84	65	82	1	79	***92***	27	11	84	98	88	0	***100***
GB ADASYN	1	81	66	82	0	79	***92***	38	11	66	98	88	0	***100***
GB KSMOTE	0	***85***	65	82	1	79	***92***	27	12	67	98	88	0	***100***
LR ROS	2	0	94	83	7	99	0	***89***	0	89	99	95	2	0
LR SMOTE	2	0	94	82	6	99	0	88	0	88	99	95	2	0
LR BSMOTE	2	2	94	83	7	99	0	88	0	79	99	95	0	0
LR ADASYN	2	2	95	82	8	99	0	85	0	91	99	95	0	0
LR KSMOTE	13	2	94	83	4	99	0	87	0	89	99	95	4	0
DT ROS	34	31	83	8	***61***	99	0	80	2	0	9	96	0	0
DT SMOTE	35	34	92	8	49	99	0	75	3	0	9	92	0	0
DT BSMOTE	34	22	93	9	45	100	0	85	2	0	9	95	0	0
DT ADASYN	37	28	84	8	44	99	0	80	4	***98***	9	95	0	0
DT KSMOTE	***45***	44	86	83	48	99	0	59	***89***	0	9	4	0	0

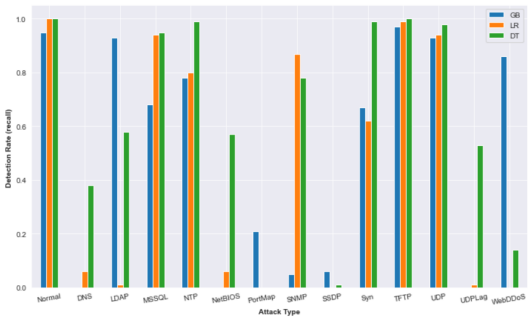
ii. Oversampled 10% comparison results

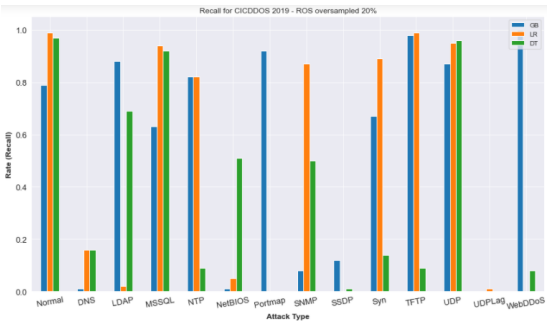


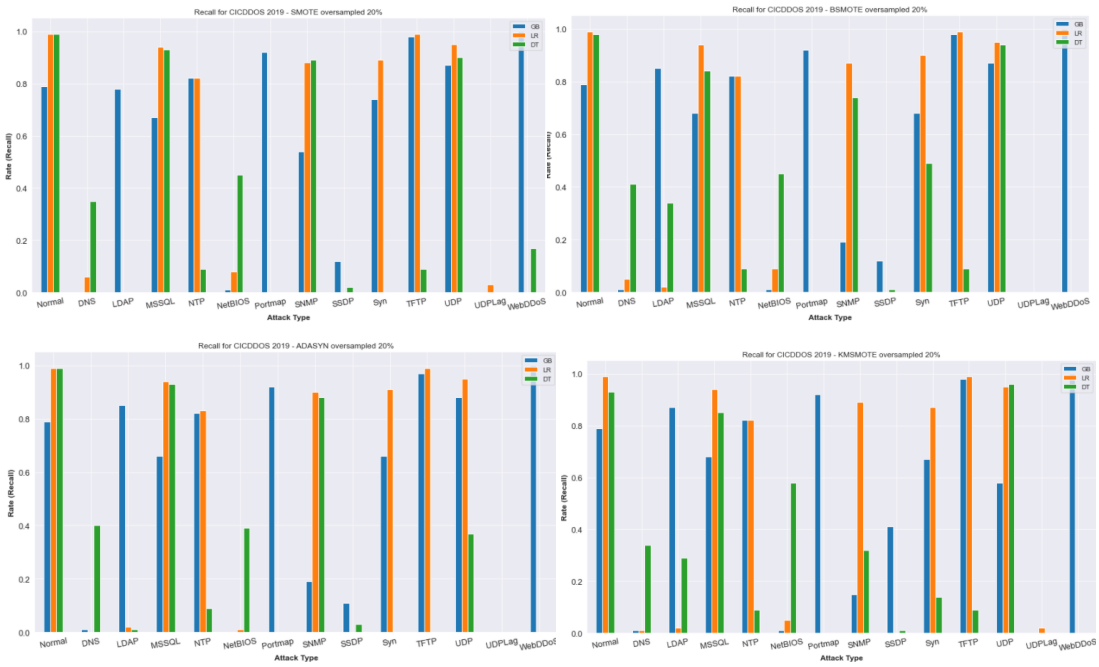
iii. Oversampled 20% detection rate

Result	DNS	LDAP	MSSQL	NTP	NetBIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	ULag	WDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	1	***88***	63	82	1	79	***92***	8	12	67	98	87	0	***100***
GB SMOTE	0	78	67	82	1	79	***92***	54	12	74	98	87	0	***100***
GB BSMOTE	1	85	68	82	1	79	***92***	19	12	68	98	87	0	***100***
GB ADASYN	1	85	66	82	0	79	***92***	19	11	66	97	88	0	***100***
GB KSMOTE	1	87	68	82	1	79	***92***	15	***41***	67	98	58	0	***100***
LR ROS	16	2	94	82	5	99	0	87	0	89	99	95	1	0
LR SMOTE	6	0	94	82	0	99	0	88	0	89	99	95	3	0
LR BSMOTE	5	2	94	82	9	99	0	87	0	9	99	95	0	0
LR ADASYN	0	2	94	83	1	99	0	***90***	0	***91***	99	95	0	0
LR KSMOTE	1	2	94	82	5	99	0	89	0	87	99	95	2	0
DT ROS	16	69	92	9	51	97	0	50	1	14	9	96	0	8
DT SMOTE	35	0	93	9	45	99	0	89	2	0	9	90	0	17
DT BSMOTE	***41***	34	84	9	45	98	0	74	1	49	9	94	0	0
DT ADASYN	40	1	93	9	39	99	0	88	3	0	0	37	0	0
DT KSMOTE	34	29	85	9	***58***	93	0	32	1	14	9	96	0	0

iv. Oversampled 20% comparison results



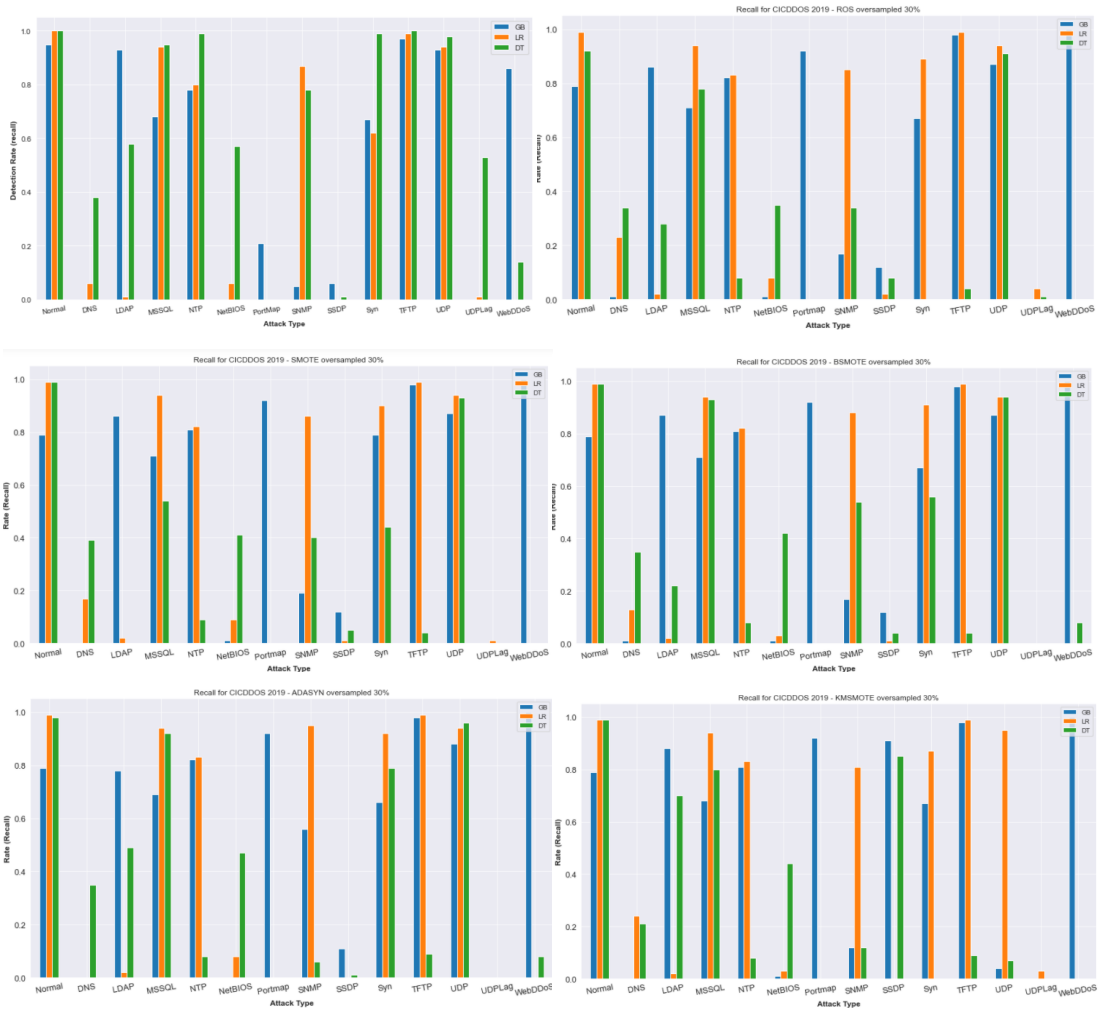




v. Oversampled 30% detection rate

Result	DNS	LDAP	MSSQL	NTP	NetBIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	ULag	WDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	1	86	71	82	1	79	***92***	17	12	67	98	87	0	***100***
GB SMOTE	0	86	71	81	1	79	***92***	19	12	79	98	87	0	***100***
GB BSMOTE	1	87	71	81	1	79	***92***	17	12	67	98	87	0	***100***
GB ADASYN	0	78	69	82	0	79	***92***	56	11	66	98	88	0	***100***
GB KSMOTE	0	***88***	68	81	1	79	***92***	12	***91***	67	98	4	0	***100***
LR ROS	23	2	94	83	8	99	0	85	2	89	99	94	4	0
LR SMOTE	17	2	94	82	9	99	0	86	1	90	99	94	1	0
LR BSMOTE	13	2	94	82	3	99	0	88	1	91	99	94	0	0
LR ADASYN	0	2	94	83	8	99	0	***95***	0	***92***	99	94	0	0
LR KSMOTE	24	2	94	83	3	99	0	81	0	87	99	95	3	0
DT ROS	34	28	78	8	35	92	0	34	8	0	4	91	1	0
DT SMOTE	***39***	0	54	9	41	99	0	40	5	44	4	93	0	0
DT BSMOTE	35	22	93	8	42	99	0	54	4	56	4	94	0	8
DT ADASYN	35	49	92	8	47	98	0	6	1	79	9	96	0	8
DT KSMOTE	21	70	80	8	44	99	0	12	85	0	9	7	0	0

vi. Oversampled 30% comparison results

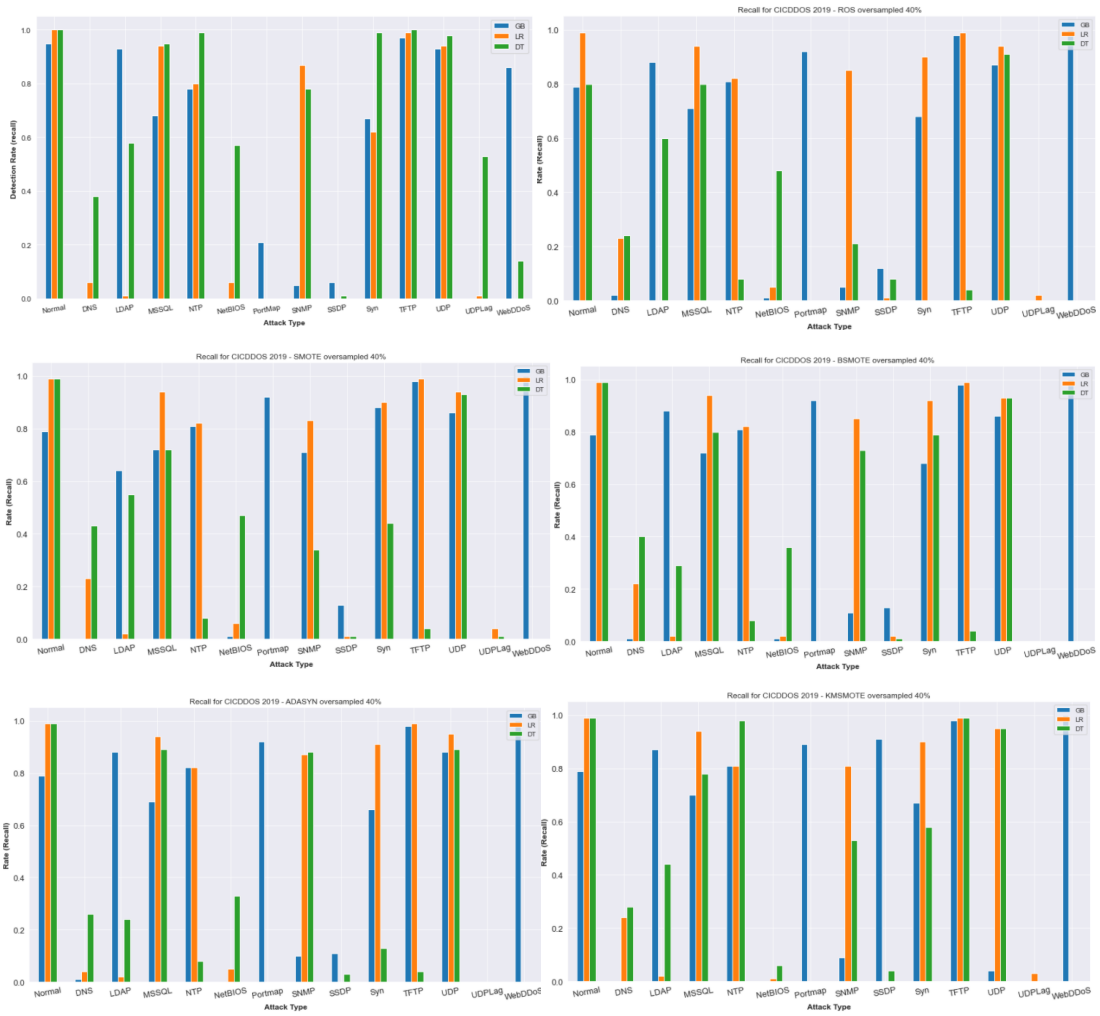


vii. Oversampled 40% detection rate

Result	DNS	LDAP	MSSQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	ULag	WDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	2	***88***	71	81	1	79	***92***	5	12	68	98	87	0	***100***
GB SMOTE	0	64	72	81	1	79	***92***	71	13	88	98	86	0	***100***
GB BSMOTE	1	***88***	72	81	1	79	***92***	11	13	68	98	86	0	***100***
GB ADASYN	1	***88***	69	82	0	79	82	10	11	66	98	88	0	***100***
GB KSMOTE	0	87	70	81	0	79	89	9	***91***	67	98	4	0	***100***
LR ROS	23	0	94	82	5	99	0	85	1	90	99	94	2	0
LR SMOTE	23	2	94	82	6	99	0	83	1	90	99	94	4	0

LR BSMOTE	22	2	94	82	2	99	0	85	2	92	99	93	0	0
LR ADASYN	4	2	94	82	5	99	0	87	0	***91***	99	95	0	0
LR KSMOTE	24	2	94	81	1	99	0	81	0	90	99	95	3	0
DT ROS	24	60	80	8	48	80	0	21	8	0	4	91	0	0
DT SMOTE	***43***	55	72	8	47	99	0	34	1	44	4	93	1	0
DT BSMOTE	40	29	80	8	36	99	0	72	1	79	4	93	0	0
DT ADASYN	26	24	89	8	33	99	0	***88***	3	13	4	89	0	0
DT KSMOTE	28	44	78	98	6	99	0	53	4	58	99	95	0	58

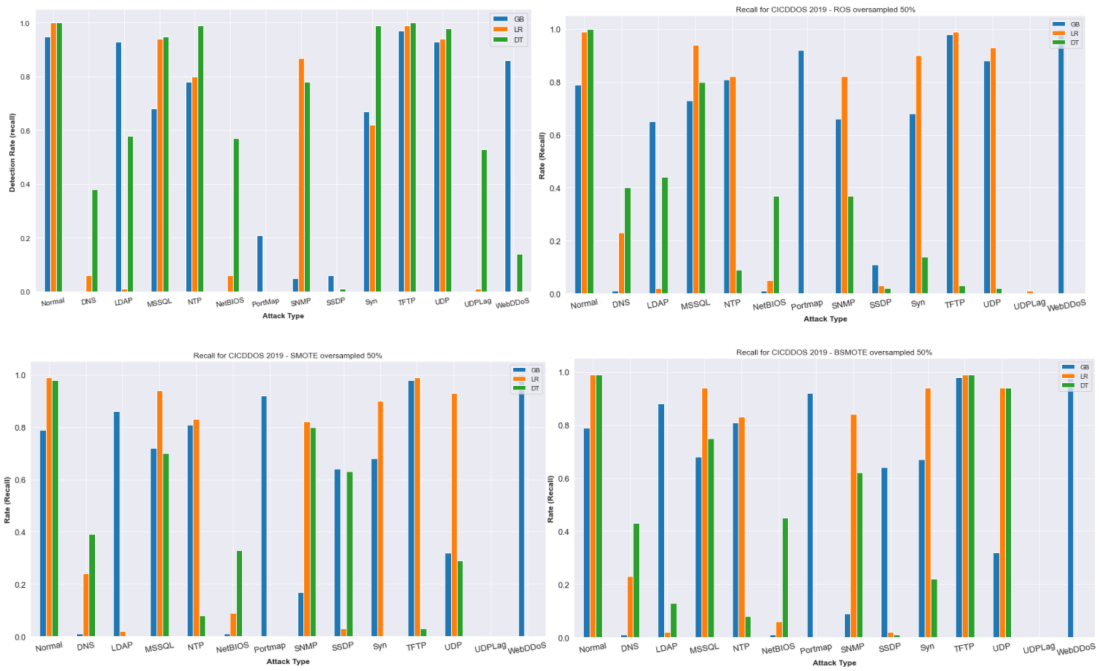
viii. Oversampled 40% comparison results

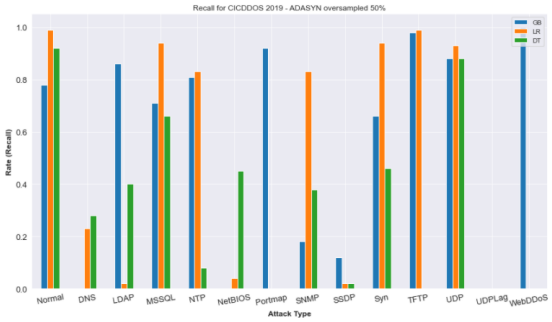


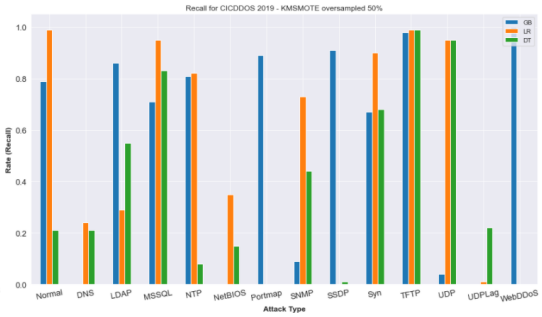
ix. Oversampled 50% detection rate

Result	DNS	LDAP	MSSQL	NTP	NetBIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	UDPLag	WebDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	1	65	73	81	1	79	***92***	66	11	68	98	88	0	***100***
GB SMOTE	1	86	72	81	1	79	***92***	17	64	68	98	32	0	***100***
GB BSMOTE	1	***88***	68	81	1	79	***92***	9	64	67	98	32	0	***100***
GB ADASYN	0	86	71	81	0	78	***92***	18	12	66	98	88	0	***100***
GB KSMOTE	0	86	71	81	0	79	89	9	***91***	67	98	4	0	***100***
LR ROS	23	2	94	82	5	99	0	82	3	90	99	93	1	0
LR SMOTE	24	2	94	83	9	99	0	82	3	90	99	93	0	0
LR BSMOTE	23	2	94	83	6	99	0	84	2	94	99	94	0	0
LR ADASYN	23	2	94	83	4	99	0	83	2	***94***	99	93	0	0
LR KSMOTE	24	29	95	82	35	99	0	73	0	90	99	95	1	0
DT ROS	40	44	80	9	37	100	0	37	2	14	3	2	0	0
DT SMOTE	39	0	70	8	33	98	0	80	63	0	3	29	0	0
DT BSMOTE	***43***	13	75	8	45	99	0	62	1	22	99	94	0	0
DT ADASYN	28	40	66	8	45	92	0	38	2	46	0	88	0	0
DT KSMOTE	21	55	83	8	15	21	0	44	1	68	99	95	22	0

x. Oversampled 50% comparison results



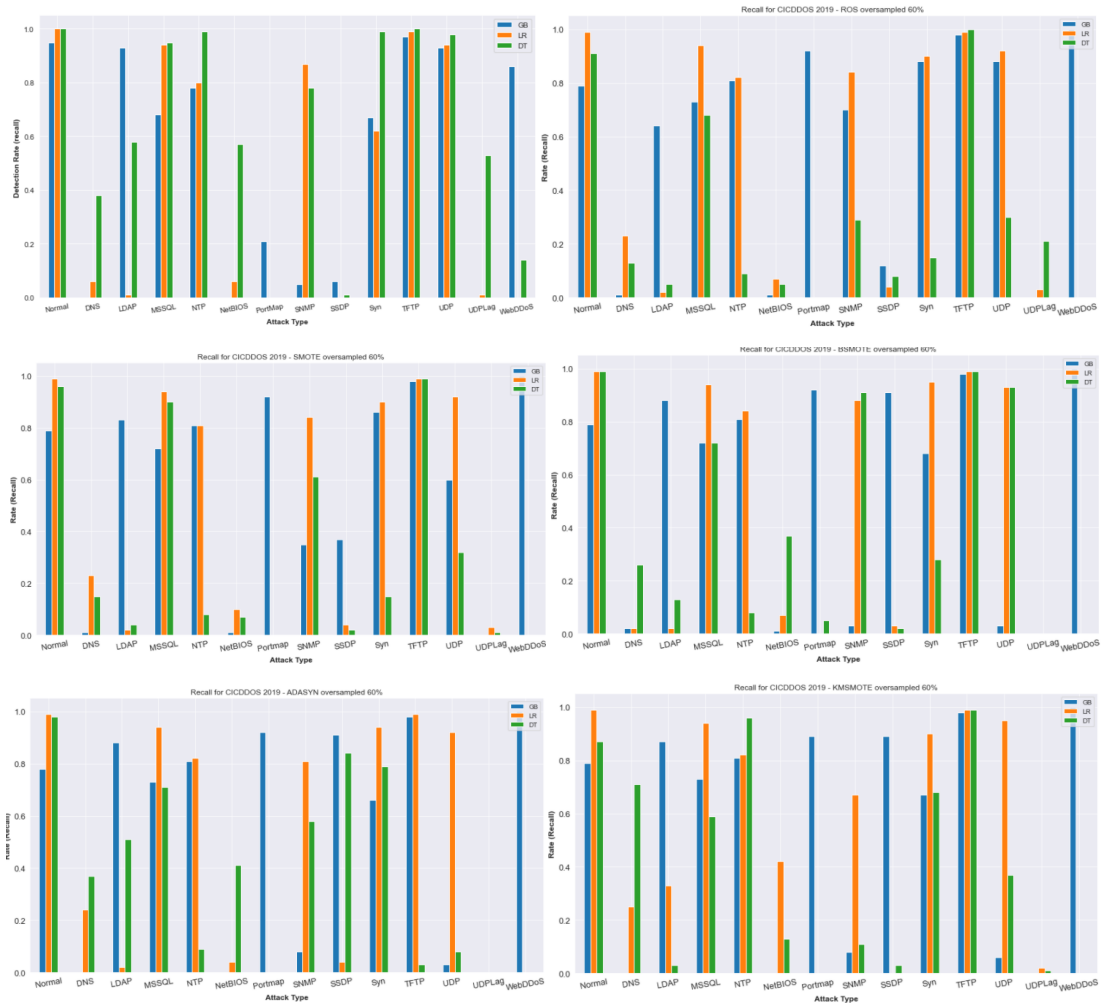




xi. Oversampled 60% detection rate

Result	DNS	LDAP	MSSQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	ULag	WDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	1	64	73	81	1	79	***92***	70	12	88	98	88	0	***100***
GB SMOTE	1	83	72	81	1	79	***92***	35	37	86	98	60	0	***100***
GB BSMOTE	2	***88***	72	81	1	79	***92***	3	***91***	68	98	3	0	***100***
GB ADASYN	0	***88***	73	81	0	78	***92***	8	***91***	66	98	3	0	***100***
GB KSMOTE	0	87	73	81	0	79	89	8	89	67	98	6	0	***100***
LR ROS	23	2	94	82	7	99	0	84	4	90	99	922	3	0
LR SMOTE	23	2	94	81	10	99	0	84	4	90	99	92	3	0
LR BSMOTE	2	2	94	81	7	99	0	88	3	95	99	93	0	0
LR ADASYN	24	2	94	82	4	99	0	81	4	***94***	99	92	0	0
LR KSMOTE	25	33	94	82	42	99	0	67	0	90	99	95	2	0
DT ROS	13	5	68	9	5	91	0	29	8	15	100	30	21	0
DT SMOTE	15	4	90	8	7	96	0	61	2	15	99	32	1	0
DT BSMOTE	26	13	72	8	37	99	5	***91***	2	28	99	93	0	0
DT ADASYN	37	51	71	9	41	98	0	58	84	79	3	8	0	0
DT KSMOTE	***71***	3	59	96	13	87	0	11	3	68	99	37	1	0

xii. Oversampled 60% comparison results

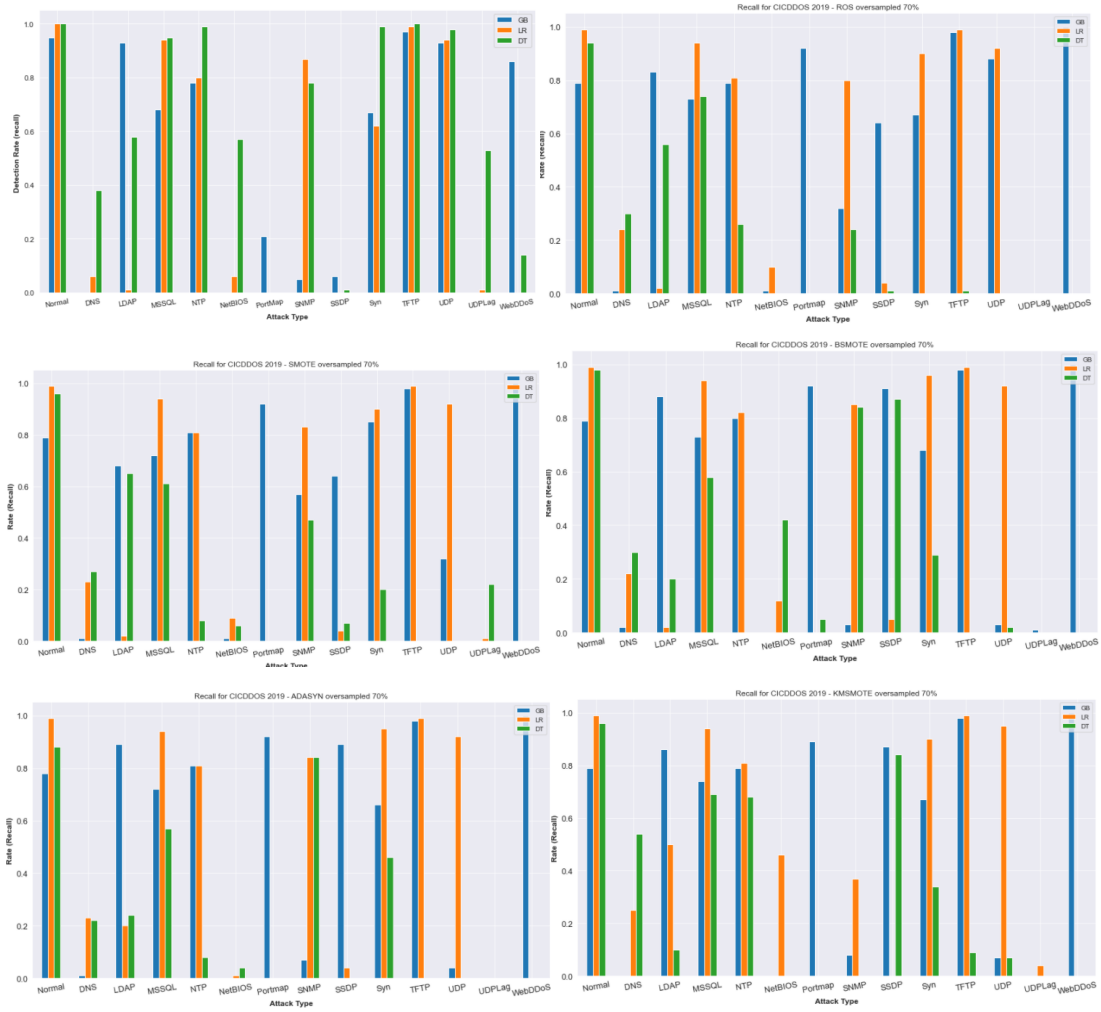


xiii. Oversampled 70% detection rate

Result	DNS	LD AP	MS SQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TF TP	UDP	U Lag	W DDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	1	83	73	79	1	79	***92***	32	64	67	98	33	0	***100***
GB SMOTE	1	68	72	81	1	79	***92***	57	64	85	98	32	0	***100***
GB BSMOTE	2	88	73	80	0	79	***92***	3	***91***	68	98	3	1	***100***
GB ADASYN	1	***89***	72	81	0	78	***92***	7	89	66	98	4	0	***100***
GB KSMOTE	0	86	74	79	0	79	89	8	87	67	98	7	0	***100***
LR ROS	24	2	94	81	10	99	0	80	4	90	99	92	0	0
LR SMOTE	23	2	94	81	9	99	0	83	4	90	99	92	1	0
LR BSMOTE	22	2	94	82	12	99	0	85	5	***96***	99	92	0	0

LR ADASYN	23	2	94	81	1	99	0	84	4	95	99	92	0	0
LR KSMOTE	25	50	94	81	46	99	0	37	0	90	99	95	4	0
DT ROS	30	56	74	26	0	94	0	24	1	0	1	0	0	0
DT SMOTE	27	65	61	8	6	96	0	47	7	20	0	0	22	0
DT BSMOTE	30	20	58	0	42	98	5	84	87	29	0	2	0	0
DT ADASYN	22	24	57	8	4	88	0	84	0	46	0	0	0	0
DT KSMOTE	***54***	10	69	68	0	96	0	0	84	34	9	7	0	58

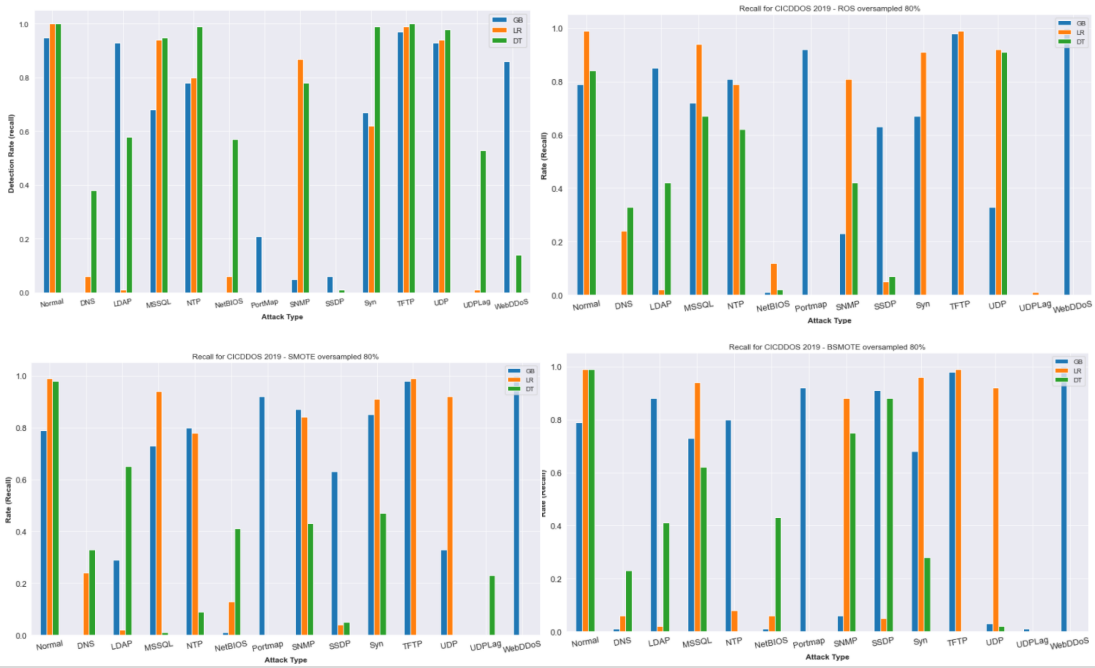
xiv. Oversampled 70% comparison results

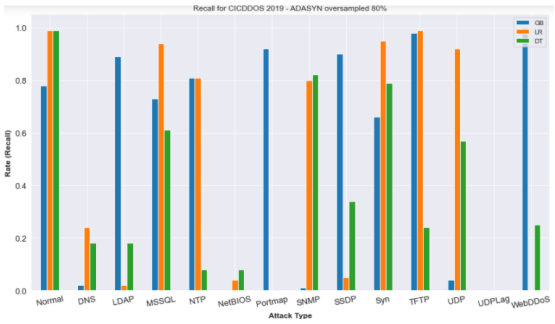


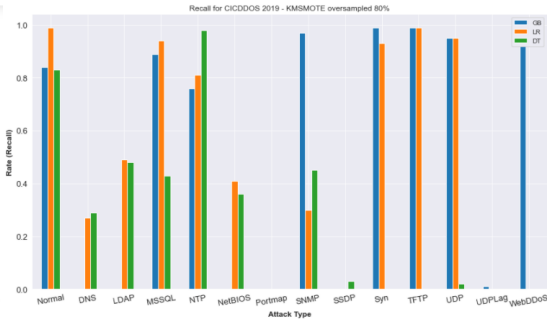
xv. Oversampled 80% detection rate

Result	DNS	LDAP	MSSQL	NTP	NetBIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	ULag	WDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	0	85	72	81	1	79	***92***	23	63	67	98	33	0	***100***
GB SMOTE	0	29	73	80	1	79	***92***	87	63	85	98	33	0	***100***
GB BSMOTE	1	88	73	80	1	79	***92***	6	***91***	68	98	3	1	***100***
GB ADASYN	2	***89***	73	81	0	78	***92***	1	90	66	98	4	0	***100***
GB KSMOTE	0	0	89	76	0	84	0	***97***	0	***99***	99	95	1	92
LR ROS	24	2	94	79	12	99	0	81	5	91	99	92	1	0
LR SMOTE	24	2	94	78	13	99	0	84	4	91	99	92	0	0
LR BSMOTE	6	2	94	80	6	99	0	88	5	96	99	92	0	0
LR ADASYN	24	2	94	81	4	99	0	80	5	96	99	92	0	0
LR KSMOTE	27	49	94	81	41	99	0	30	0	93	99	95	0	0
DT ROS	33	42	67	62	2	84	0	42	7	0	0	91	0	0
DT SMOTE	33	65	1	9	41	98	0	43	5	47	0	0	23	0
DT BSMOTE	23	41	62	0	43	99	0	75	88	28	0	2	0	0
DT ADASYN	18	18	61	8	8	99	0	82	34	79	24	57	0	25
DT KSMOTE	29	48	43	98	36	83	0	45	3	0	0	2	0	58

xvi. Oversampled 80% comparison results



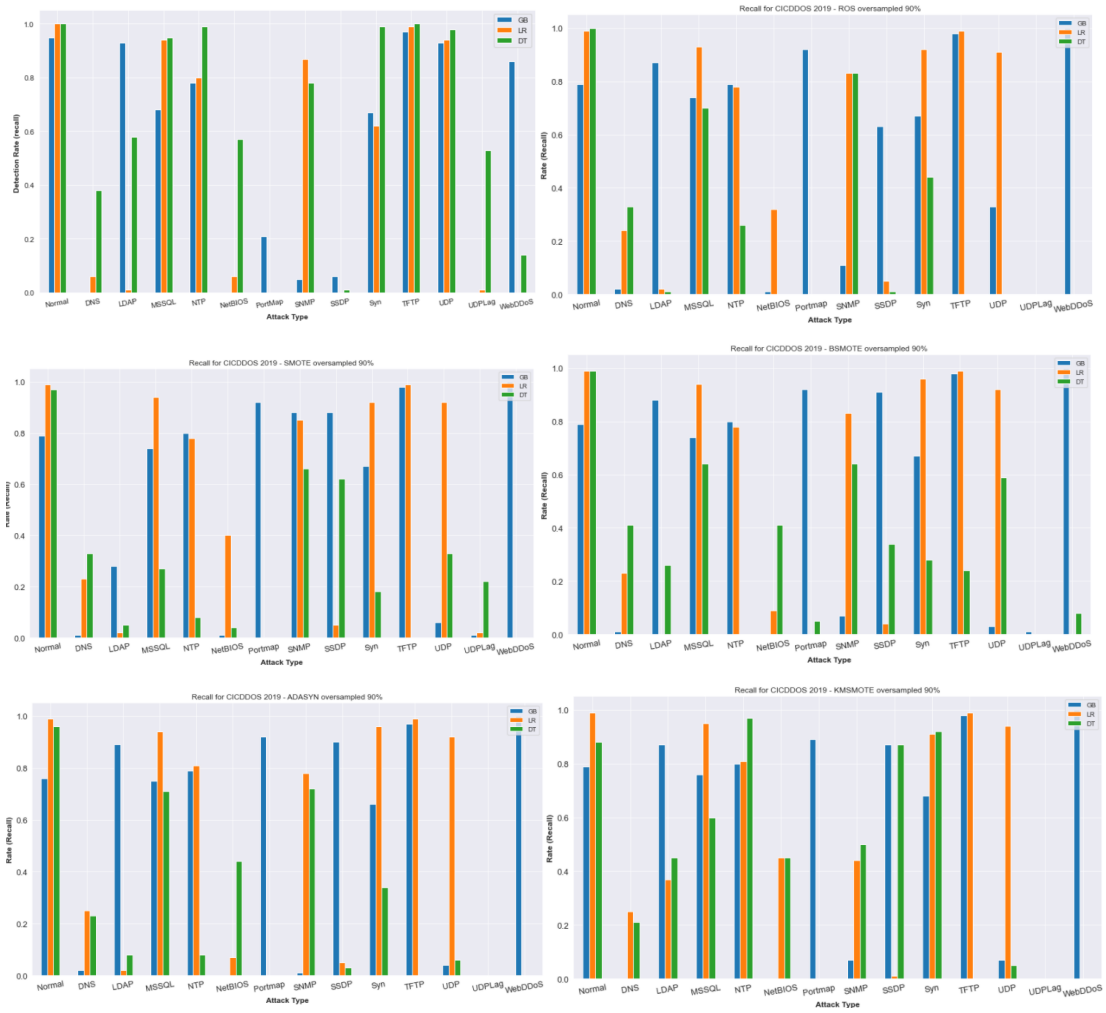




xvii. Oversampled 90% detection rate

Result	DNS	LDAP	MSSQL	NTP	NetBIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	ULag	WDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	2	87	74	79	1	79	***92***	11	63	67	98	33	0	***100***
GB SMOTE	1	28	74	80	1	79	***92***	***88***	88	67	98	6	1	***100***
GB BSMOTE	1	88	74	80	0	79	***92***	7	***91***	67	98	3	1	***100***
GB ADASYN	2	***89***	75	79	0	76	***92***	1	90	66	97	4	0	***100***
GB KSMOTE	0	87	76	80	0	79	89	7	87	68	98	7	0	***100***
LR ROS	24	3	93	78	32	99	0	83	5	92	99	91	0	0
LR SMOTE	23	2	94	78	40	99	0	85	5	92	99	92	2	0
LR BSMOTE	23	0	94	78	9	99	0	83	4	***96***	99	92	0	0
LR ADASYN	25	2	94	81	7	99	0	78	5	***96***	99	92	0	0
LR KSMOTE	25	37	95	81	45	99	0	44	1	91	99	94	0	0
DT ROS	33	1	70	26	0	100	0	83	1	44	0	0	0	0
DT SMOTE	33	5	27	8	4	97	0	66	62	18	0	33	22	0
DT BSMOTE	41	26	64	0	41	99	5	64	34	28	24	59	0	8
DT ADASYN	23	8	71	8	44	96	0	72	3	34	0	6	0	0
DT KSMOTE	21	45	60	97	45	88	0	50	87	92	0	5	0	0

xviii. Oversampled 90% comparison results

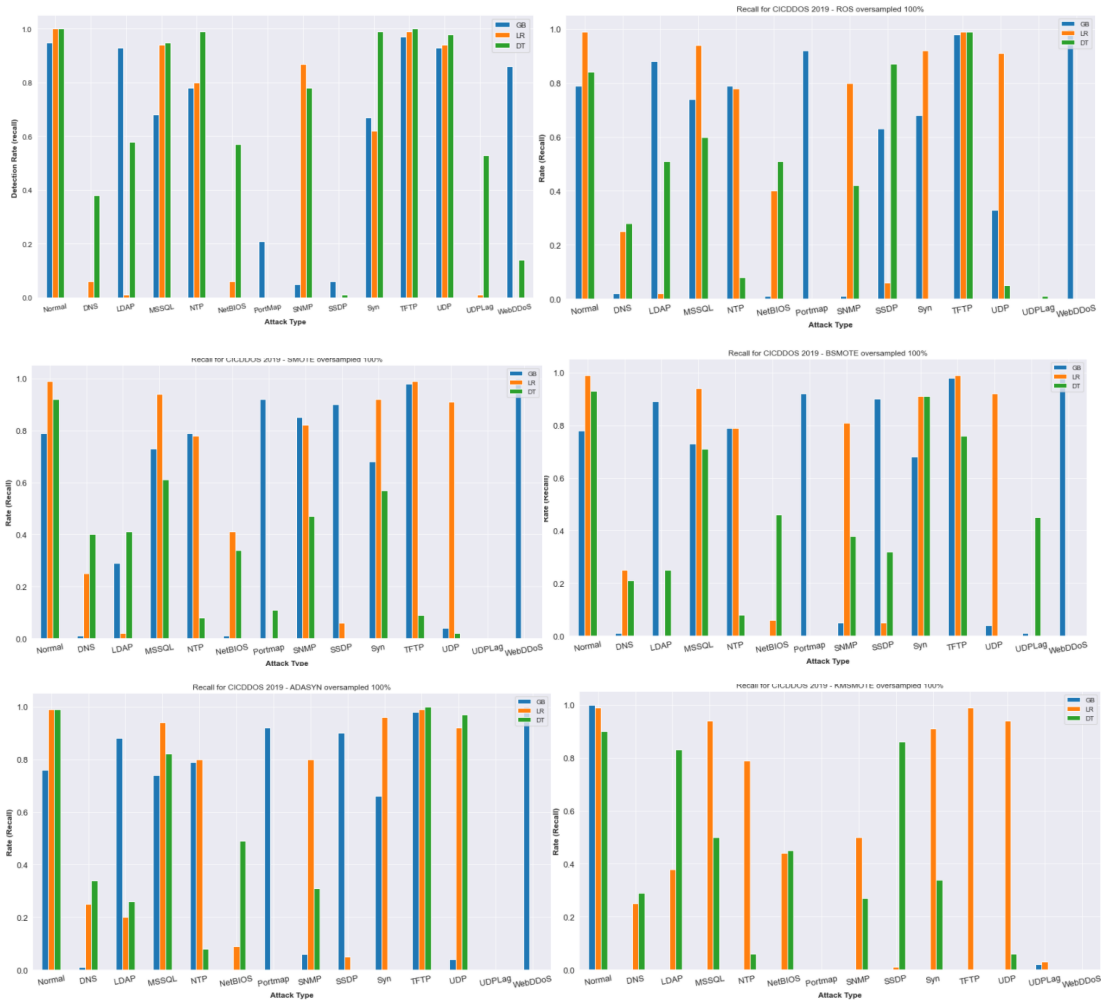


xix. Oversampled 100% detection rate

Result	DNS	LD AP	MS SQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TF TP	UDP	U Lag	W DDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	2	88	74	79	1	79	***92***	1	63	68	98	33	0	***100***
GB SMOTE	1	29	73	79	1	79	***92***	85	***90***	68	98	4	0	***100***
GB BSMOTE	1	***89***	73	79	0	78	***92***	5	***90***	68	98	4	1	***100***
GB ADASYN	1	88	74	79	0	76	***92***	6	***90***	66	98	4	0	***100***
GB KSMOTE	0	0	0	0	0	100	0	0	0	0	0	0	2	0
LR ROS	25	2	94	78	40	99	0	80	6	92	99	91	0	0
LR SMOTE	25	2	94	78	41	99	0	82	6	92	99	91	0	0
LR BSMOTE	25	0	94	79	6	99	0	81	5	***97***	99	92	0	0

LR ADASYN	25	2	94	80	9	99	0	80	5	96	99	92	0	0
LR KSMOTE	25	38	94	79	44	99	0	50	1	91	99	94	3	0
DT ROS	28	51	60	8	51	84	0	42	87	0	99	5	1	0
DT SMOTE	***40***	41	61	8	34	92	11	47	0	57	9	2	0	0
DT BSMOTE	21	25	71	8	46	93	0	38	32	91	76	0	45	0
DT ADASYN	34	26	82	8	49	99	0	31	0	0	100	97	0	0
DT KSMOTE	29	83	50	6	45	90	0	27	86	34	0	6	0	0

xx. Oversampled 100% comparison results

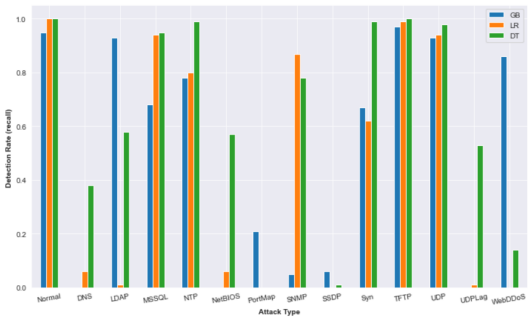


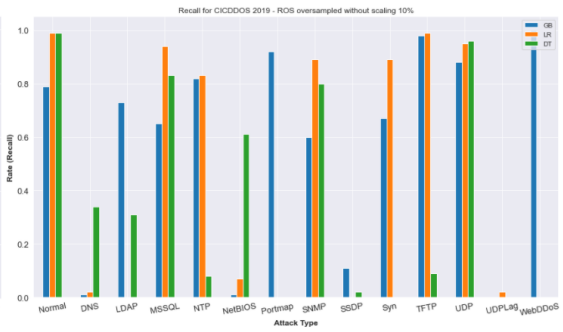
APPENDIX R: CICDDOS 2019 oversampling individual DR on non-scaled data

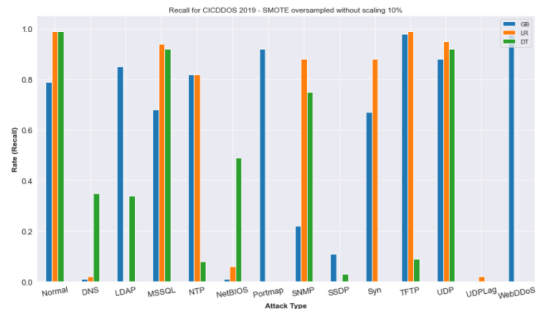
i. Oversampled 10% detection rate – non-scaled

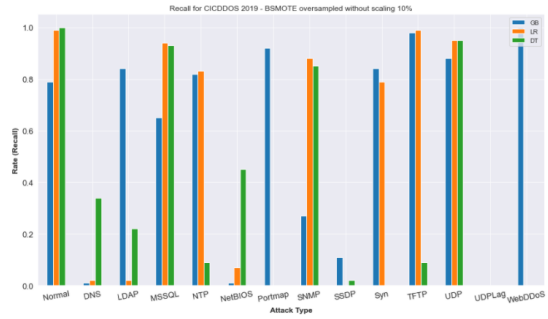
Result	DNS	LDAP	MSSQL	NTP	NetBIOS	Nor	Portmap	SNMP	SSDP	SYN	TFTP	UDP	UDPLag	WebDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	0	0	62	0	0	100	0	36	3	0	0	0	0	0
GB SMOTE	0	0	61	0	0	100	0	36	3	0	0	0	0	0
GB BSMOTE	0	0	62	0	0	100	0	36	3	0	0	0	0	0
GB ADASYN	0	0	62	0	0	100	0	36	3	0	0	0	0	0
GB KSMOTE	0	30	63	0	0	100	0	28	3	0	0	0	0	0
LR ROS	0	0	85	0	0	34	0	0	0	38	92	1	0	0
LR SMOTE	0	0	96	0	0	28	0	0	2	49	96	0	0	0
LR BSMOTE	0	0	95	0	0	28	0	0	0	49	32	1	0	0
LR ADASYN	0	0	96	0	0	38	0	0	0	36	91	1	0	0
LR KSMOTE	0	0	96	0	0	58	0	0	2	26	78	2	0	0
DT ROS	44	35	95	98	50	100	0	88	3	***99***	100	97	46	8
DT SMOTE	***46***	50	96	98	54	100	20	***89***	30	***99***	100	98	46	50
DT BSMOTE	43	41	96	98	51	100	0	85	1	***99***	100	98	47	0
DT ADASYN	43	47	95	98	53	100	20	83	30	***99***	100	98	46	***25***
DT KSMOTE	43	36	96	98	51	100	0	88	1	***99***	100	98	45	8

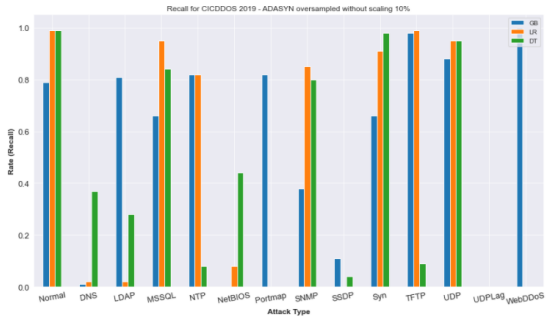
ii. Oversampled 10% comparison results – non-scaled

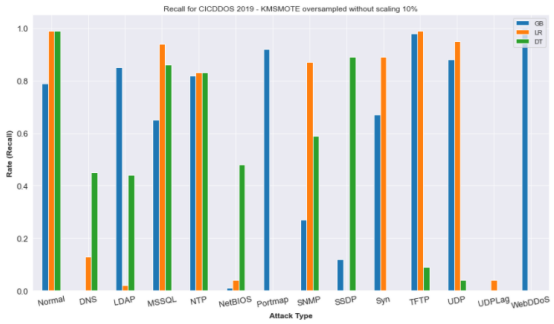








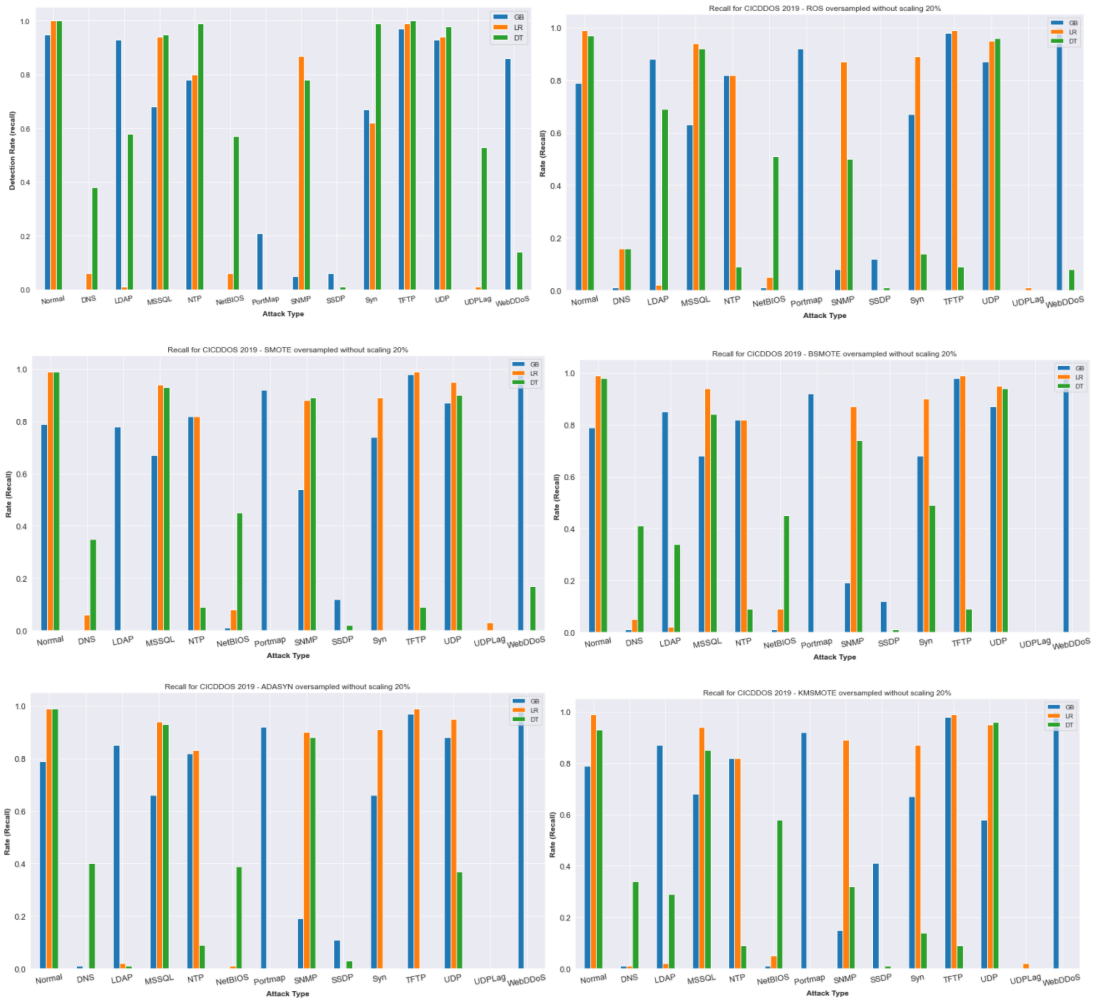




iii. Oversampled 20% detection rate – non-scaled

Result	DNS	LDAP	MSSQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	U Lag	W DDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	0	0	61	0	0	100	0	36	3	0	0	0	0	0
GB SMOTE	0	0	61	0	0	100	0	36	3	0	0	0	0	0
GB BSMOTE	0	0	62	0	0	100	0	36	3	0	0	0	0	0
GB ADASYN	0	0	62	0	0	100	***36***	3	0	0	0	0	0	0
GB KSMOTE	0	36	63	0	0	100	0	10	3	0	0	0	0	0
LR ROS	0	0	96	0	0	38	0	0	1	46	78	1	0	0
LR SMOTE	0	0	96	0	0	30	0	0	2	49	78	3	0	0
LR BSMOTE	0	0	96	0	0	31	0	0	0	44	91	1	0	0
LR ADASYN	0	0	95	0	0	29	0	0	3	43	92	1	0	0
LR KSMOTE	0	0	97	0	0	31	0	0	0	50	78	1	0	0
DT ROS	42	36	95	98	51	100	0	***89***	3	***99***	100	97	46	25
DT SMOTE	44	42	96	98	52	100	0	85	3	***99***	100	96	46	0
DT BSMOTE	44	37	95	98	51	100	0	***89***	2	***99***	100	97	46	0
DT ADASYN	***45***	46	95	98	52	100	0	84	1	***99***	100	97	46	***25***
DT KSMOTE	42	45	95	98	50	100	0	85	1	***99***	100	98	46	0

iv. Oversampled 20% comparison results – non-scaled

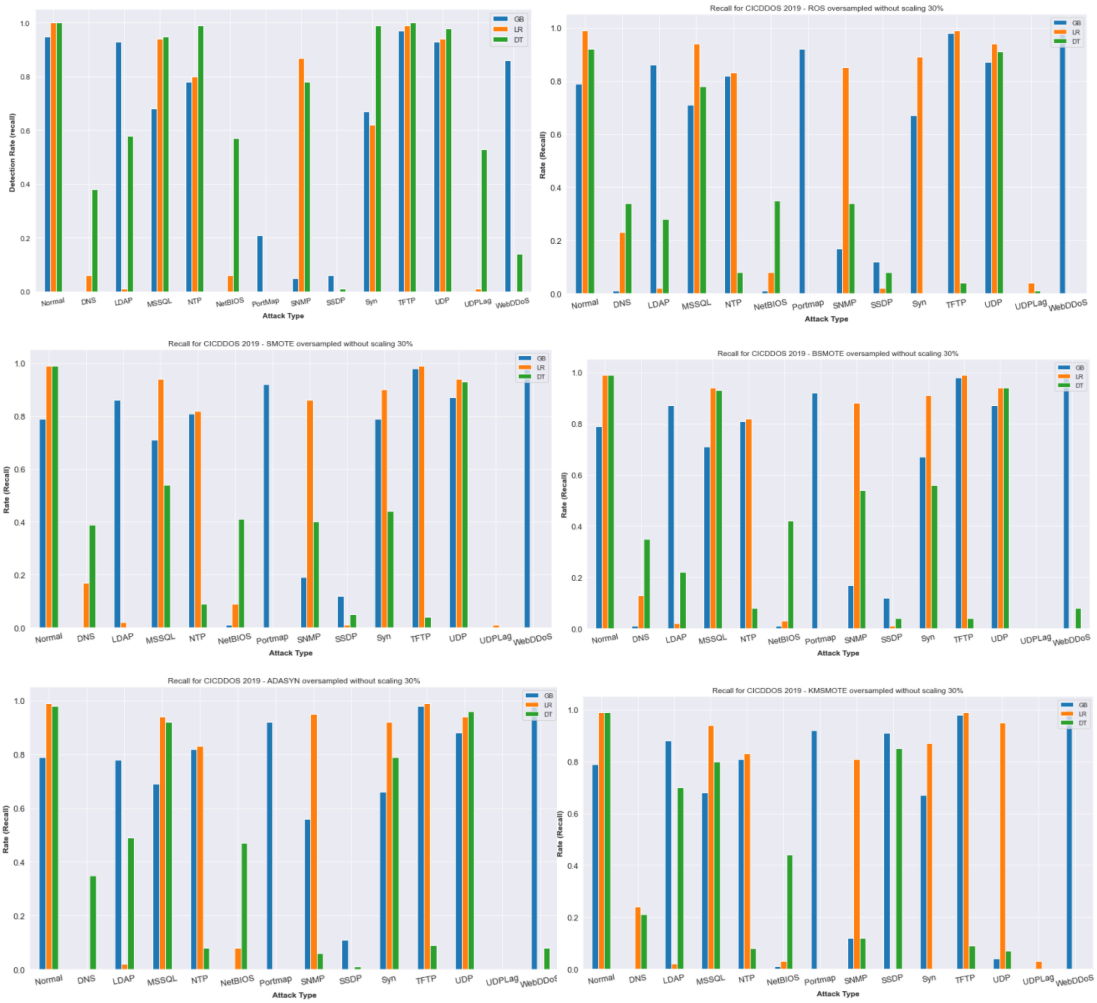


v. Oversampled 30% detection rate – non-scaled

Result	DNS	LD AP	MS SQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TF TP	UDP	U Lag	W DDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	0	0	60	0	0	100	0	36	3	0	0	0	0	0
GB SMOTE	0	0	61	0	0	100	0	36	3	0	0	0	0	0
GB BSMOTE	0	0	61	0	0	100	0	36	3	0	0	0	0	0
GB ADASYN	0	0	62	0	0	100	0	36	3	0	0	0	0	0
GB KSMOTE	0	41	66	0	0	99	0	0	0	0	0	0	0	0
LR ROS	0	0	96	1	0	37	0	0	2	50	78	0	0	0
LR SMOTE	0	0	96	0	6	39	0	0	2	47	78	0	0	0

LR BSMOTE	0	0	6	0	0	37	0	90	0	42	78	1	0	0
LR ADASYN	0	0	96	0	0	53	0	0	0	36	78	1	0	0
LR KSMOTE	0	0	99	0	0	48	0	0	2	53	5	0	0	0
DT ROS	45	39	95	98	53	100	0	87	2	***99***	100	97	46	0
DT SMOTE	***49***	47	94	98	52	100	0	80	***13***	***99***	100	90	50	***25***
DT BSMOTE	43	34	96	98	53	100	0	***89***	2	***99***	100	97	46	0
DT ADASYN	44	35	95	98	52	100	0	***89***	1	***99***	100	98	46	0
DT KSMOTE	46	37	95	98	51	100	0	83	1	***99***	100	98	46	***25***

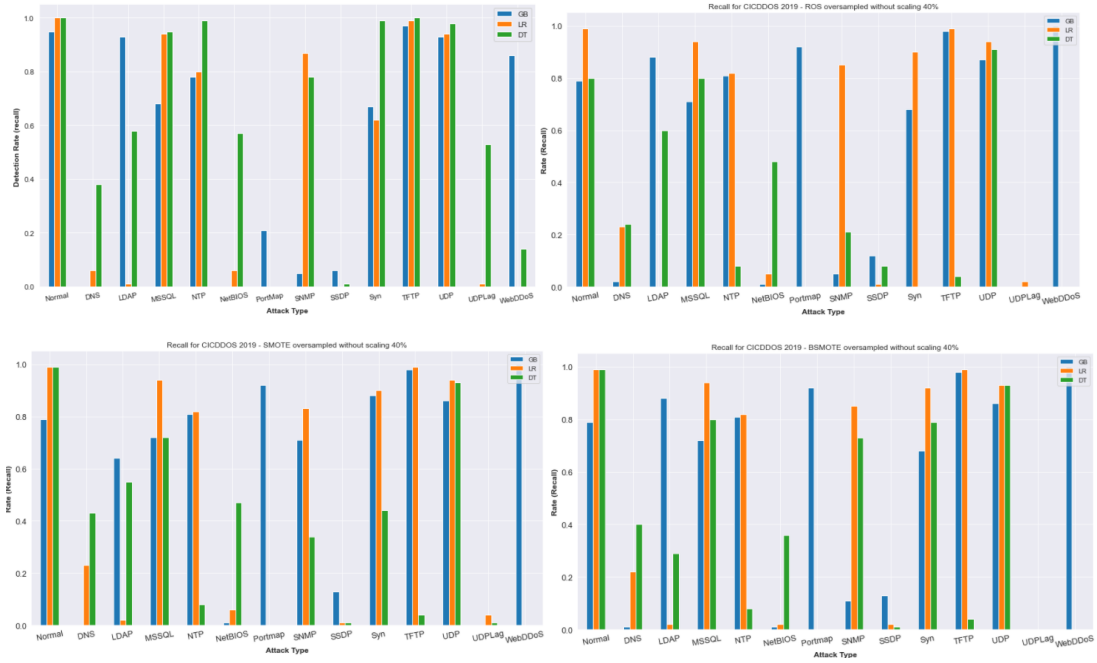
vi. Oversampled 30% comparison results – non-scaled

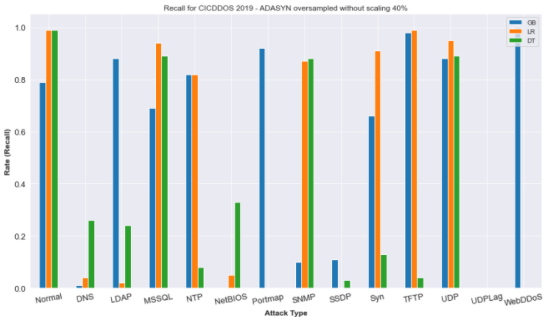


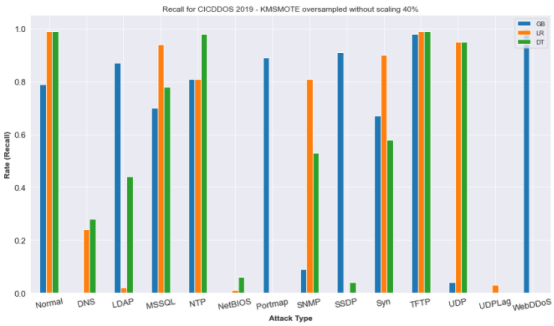
vii. Oversampled 40% detection rate – non-scaled

Result	DNS	LDAP	MSSQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	U Lag	W DDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	0	0	60	0	0	100	0	36	3	0	0	0	0	0
GB SMOTE	0	0	60	0	0	100	0	36	3	0	0	0	0	0
GB BSMOTE	0	0	61	0	0	100	0	36	3	0	0	0	1	0
GB ADASYN	0	0	61	0	0	100	0	36	3	0	0	0	0	0
GB KSMOTE	0	41	53	0	0	100	0	0	0	0	0	3	0	0
LR ROS	0	0	96	0	0	33	0	0	0	53	78	1	0	0
LR SMOTE	0	0	96	0	0	44	0	0	2	53	5	0	0	0
LR BSMOTE	0	0	6	0	0	33	0	92	1	45	78	1	0	0
LR ADASYN	0	0	96	0	0	36	0	0	0	46	78	1	0	0
LR KSMOTE	0	0	99	0	0	50	0	0	0	54	5	1	0	0
DT ROS	***50***	48	92	99	52	100	0	75	***21***	98	100	84	51	25
DT SMOTE	48	44	95	98	54	100	0	80	14	***99***	100	90	49	50
DT BSMOTE	47	41	94	98	51	100	3	78	14	98	100	89	49	0
DT ADASYN	43	33	95	98	51	100	0	***89***	2	***99***	100	97	46	0
DT KSMOTE	48	51	94	98	55	100	0	76	13	***99***	100	90	47	***58***

viii. Oversampled 40% comparison results – non-scaled



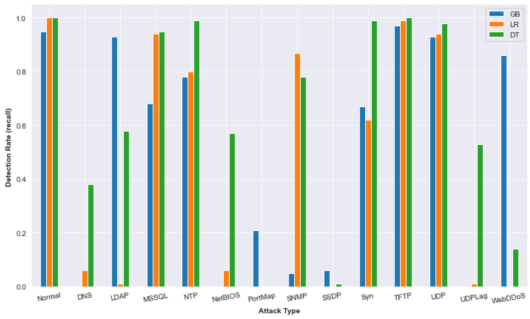


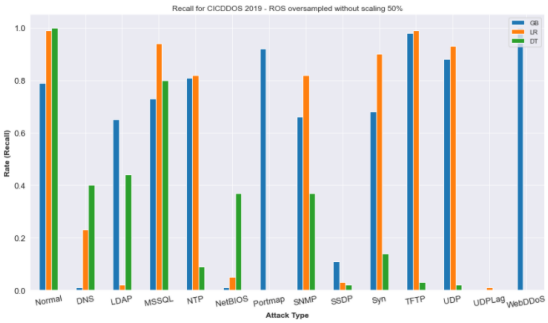


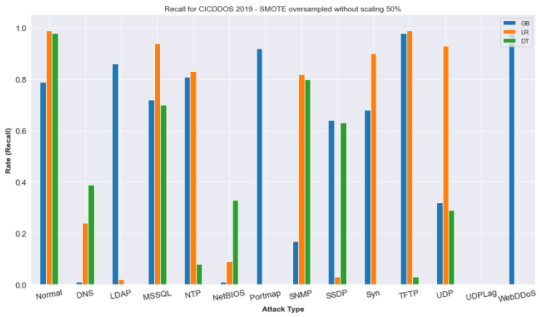
ix. Oversampled 50% detection rate – non-scaled

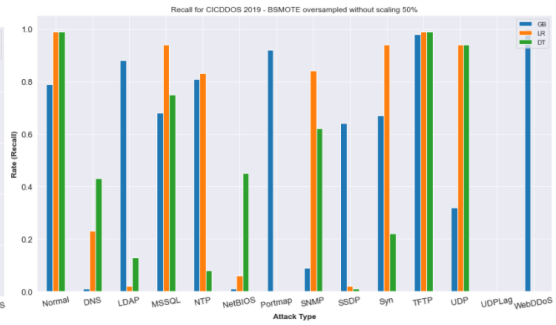
Result	DNS	LDAP	MSSQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	U Lag	W DDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	0	0	59	0	0	100	0	36	3	0	0	0	0	0
GB SMOTE	0	0	60	0	0	100	0	36	3	0	0	0	0	0
GB BSMOTE	0	0	60	0	0	100	0	36	3	0	0	0	2	0
GB ADASYN	0	0	61	0	0	100	0	36	3	0	0	0	0	0
GB KSMOTE	0	41	54	0	0	100	0	0	0	0	0	3	0	0
LR ROS	0	0	6	0	0	35	0	92	0	55	5	1	0	0
LR SMOTE	0	0	99	0	0	35	0	0	2	55	5	0	0	0
LR BSMOTE	0	0	96	0	0	30	0	0	0	49	78	1	0	0
LR ADASYN	0	0	7	0	0	35	0	92	2	47	78	0	0	0
LR KSMOTE	0	0	96	0	0	50	0	0	1	53	5	1	0	0
DT ROS	50	42	90	98	53	100	3	71	34	98	100	70	50	33
DT SMOTE	48	47	94	98	58	100	0	78	14	99	100	89	50	8
DT BSMOTE	50	40	93	98	51	100	0	82	13	99	100	90	51	0
DT ADASYN	50	41	92	98	48	100	8	78	14	99	100	88	50	8
DT KSMOTE	47	48	95	98	53	100	0	76	14	99	100	89	50	33

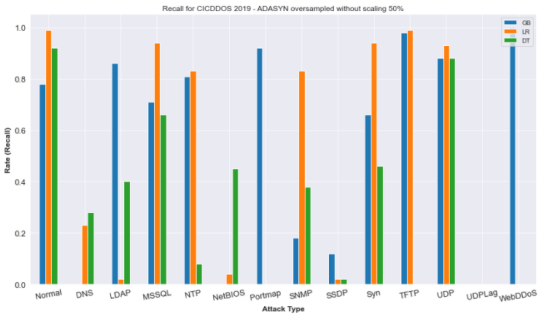
x. Oversampled 50% comparison results– non-scaled

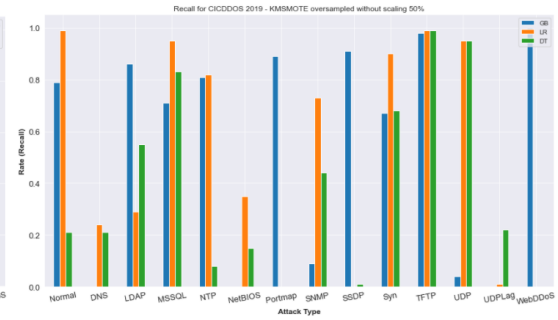










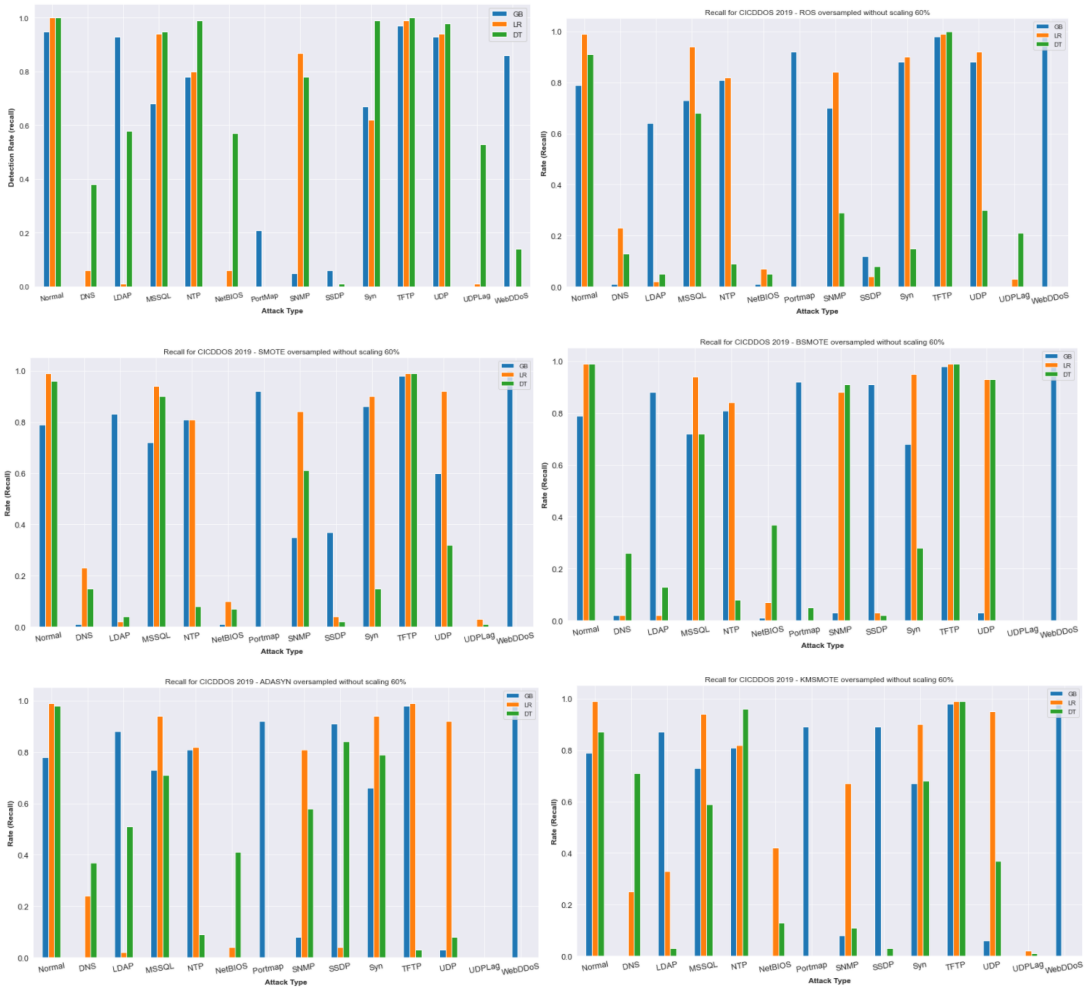


xi. Oversampled 60% detection rate – non-scaled

Result	DNS	LDAP	MSSQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	U Lag	W DDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	0	0	59	0	0	100	0	36	3	0	0	0	0	0
GB SMOTE	0	0	59	0	0	100	0	36	3	0	0	0	0	0
GB BSMOTE	0	0	60	0	0	100	0	36	3	0	0	0	2	0
GB ADASYN	0	0	60	0	0	100	0	36	3	0	0	0	0	0
GB KSMOTE	0	41	52	0	0	100	0	0	0	0	0	3	0	0
LR ROS	0	0	96	0	0	31	0	0	2	59	5	0	0	0
LR SMOTE	0	0	6	0	0	37	0	92	0	53	5	1	0	0
LR BSMOTE	0	0	0	0	0	54	0	99	0	47	5	0	0	0

LR ADASYN	0	0	5	0	0	36	0	91	2	48	78	0	0	0
LR KSMOTE	0	0	99	0	0	38	0	0	1	54	5	1	0	0
DT ROS	49	44	90	98	52	100	11	71	***35***	98	100	69	50	***33***
DT SMOTE	***50***	43	93	98	48	100	8	76	19	98	100	86	51	8
DT BSMOTE	48	40	93	98	52	100	11	76	14	98	100	88	50	8
DT ADASYN	48	44	93	98	52	100	0	78	20	98	100	85	51	33
DT KSMOTE	48	48	95	98	53	100	0	76	17	***99***	100	87	51	25

xii. Oversampled 60% comparison results – non-scaled

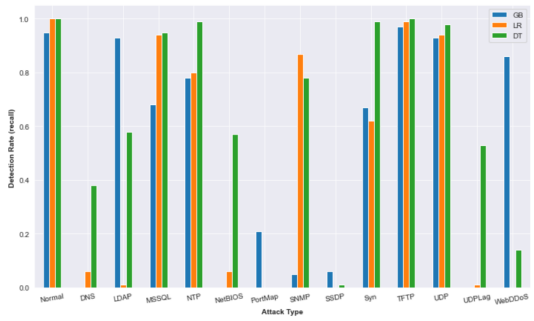


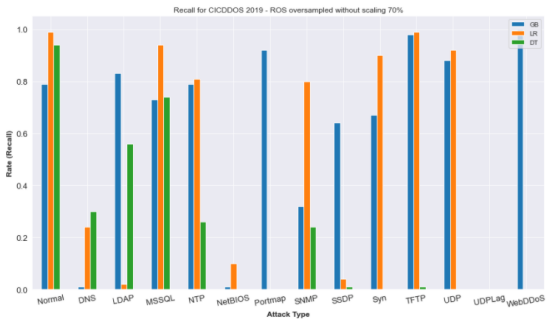
xiii. Oversampled 70% detection rate – non-scaled

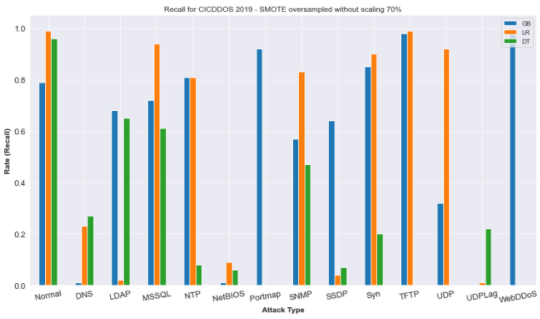
Result	DNS	LDAP	MSSQL	NTP	NetBIOS	Nor	Portmap	SNMP	SSDP	SYN	TFTP	UDP	ULag	WDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14

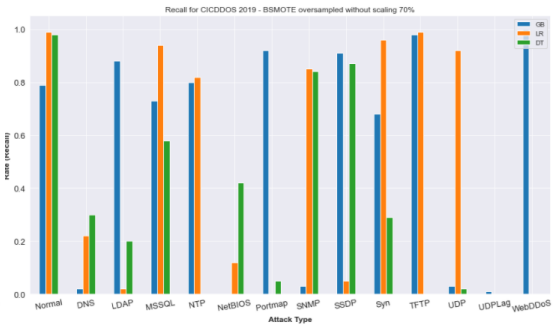
GB ROS	0	0	59	0	0	100	0	36	3	0	0	0	0	0
GB SMOTE	0	0	59	0	0	100	0	36	3	0	0	0	0	0
GB BSMOTE	0	0	59	0	0	100	0	36	3	0	0	0	3	0
GB ADASYN	0	0	60	0	0	100	0	36	3	0	0	0	0	0
GB KSMOTE	0	41	52	0	0	100	0	0	0	0	0	3	0	0
LR ROS	0	0	6	0	0	31	0	***92***	1	55	5	1	0	0
LR SMOTE	0	0	5	0	0	30	0	***92***	1	58	5	1	0	0
LR BSMOTE	0	0	6	0	0	51	0	***92***	0	47	5	1	0	0
LR ADASYN	0	0	96	0	0	33	0	0	2	52	5	0	0	0
LR KSMOTE	0	0	98	0	0	57	0	0	0	51	78	1	0	0
DT ROS	***50***	42	90	98	53	100	3	71	***36***	***98***	100	69	***54***	8
DT SMOTE	49	45	94	98	51	100	3	77	20	***98***	100	85	49	17
DT BSMOTE	***50***	36	93	98	53	100	3	80	19	***98***	100	86	51	8
DT ADASYN	49	40	93	98	51	100	0	82	20	***98***	100	86	51	50
DT KSMOTE	49	54	93	98	57	100	0	73	29	***98***	100	77	48	***58***

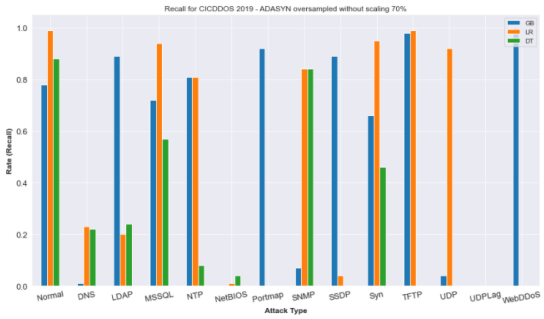
xiv. Oversampled 70% comparison results – non-scaled

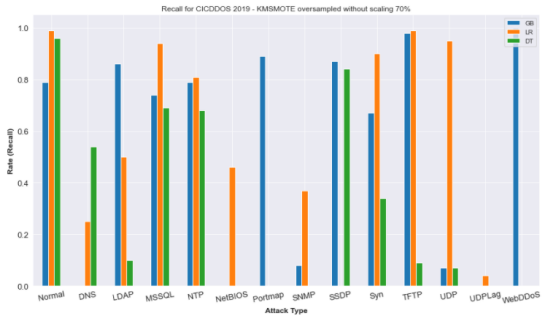








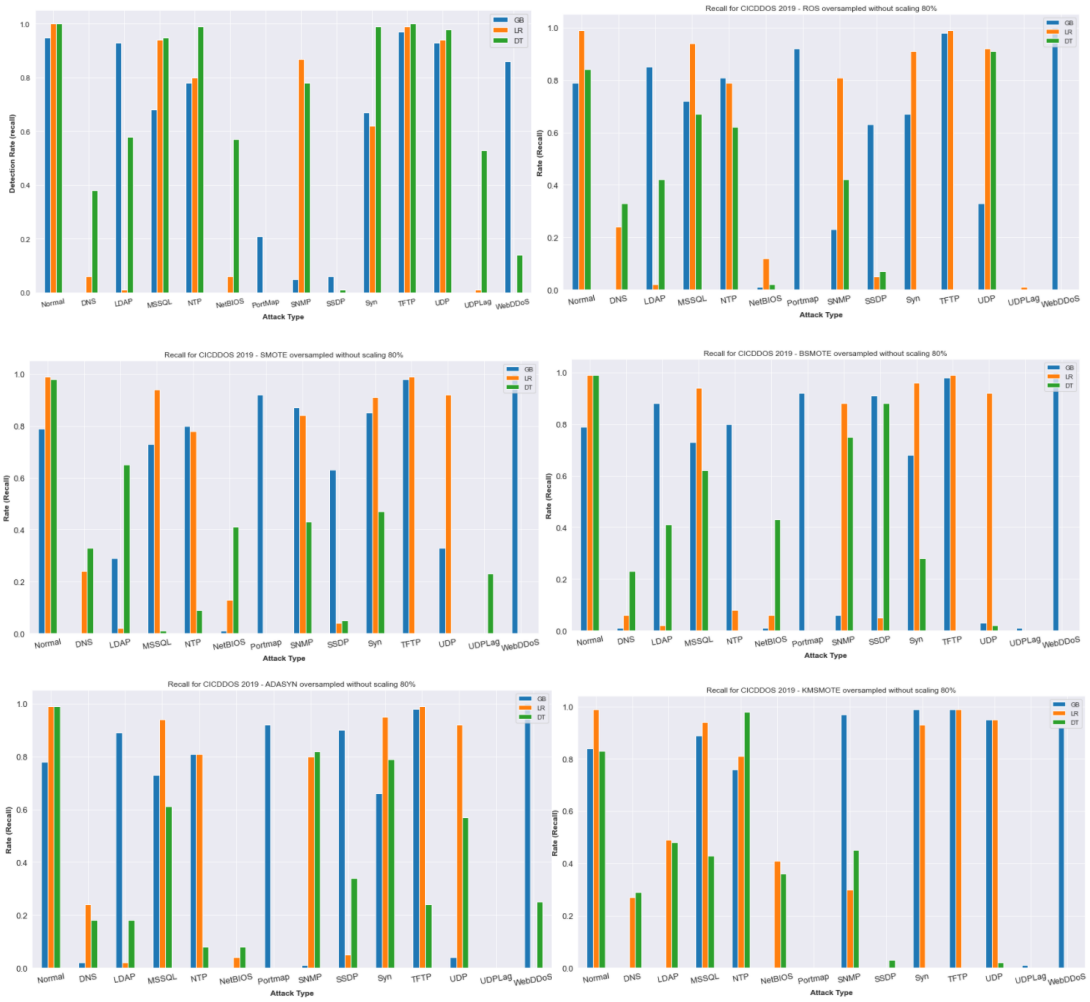




xv. Oversampled 80% detection rate – non-scaled

Result	DNS	LDAP	MSSQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	U Lag	W DDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	0	0	58	0	0	100	0	51	3	0	0	0	1	0
GB SMOTE	0	0	58	0	0	100	0	49	3	0	0	0	1	0
GB BSMOTE	0	0	58	0	0	100	0	36	3	0	0	0	3	0
GB ADASYN	0	36	60	0	0	100	0	11	3	0	0	0	0	0
GB KSMOTE	0	41	52	0	0	100	0	0	0	0	0	3	0	0
LR ROS	0	0	7	0	0	35	0	92	1	53	5	0	0	0
LR SMOTE	0	0	7	0	0	27	0	93	2	58	5	0	0	0
LR BSMOTE	0	0	6	0	0	42	0	92	2	48	5	0	0	0
LR ADASYN	0	0	6	0	0	30	0	92	0	54	5	1	0	0
LR KSMOTE	0	0	96	0	0	46	0	0	0	50	5	1	0	0
DT ROS	***51***	42	89	98	52	100	5	73	***36***	97	100	68	***62***	17
DT SMOTE	50	46	93	98	53	100	0	78	20	98	100	85	51	8
DT BSMOTE	49	40	93	98	53	100	3	77	22	98	100	84	50	8
DT ADASYN	49	40	93	98	54	100	5	80	19	98	100	86	50	25
DT KSMOTE	48	48	95	98	***58***	100	0	74	28	***99***	100	77	48	***58***

xvi. Oversampled 80% comparison results – non-scaled

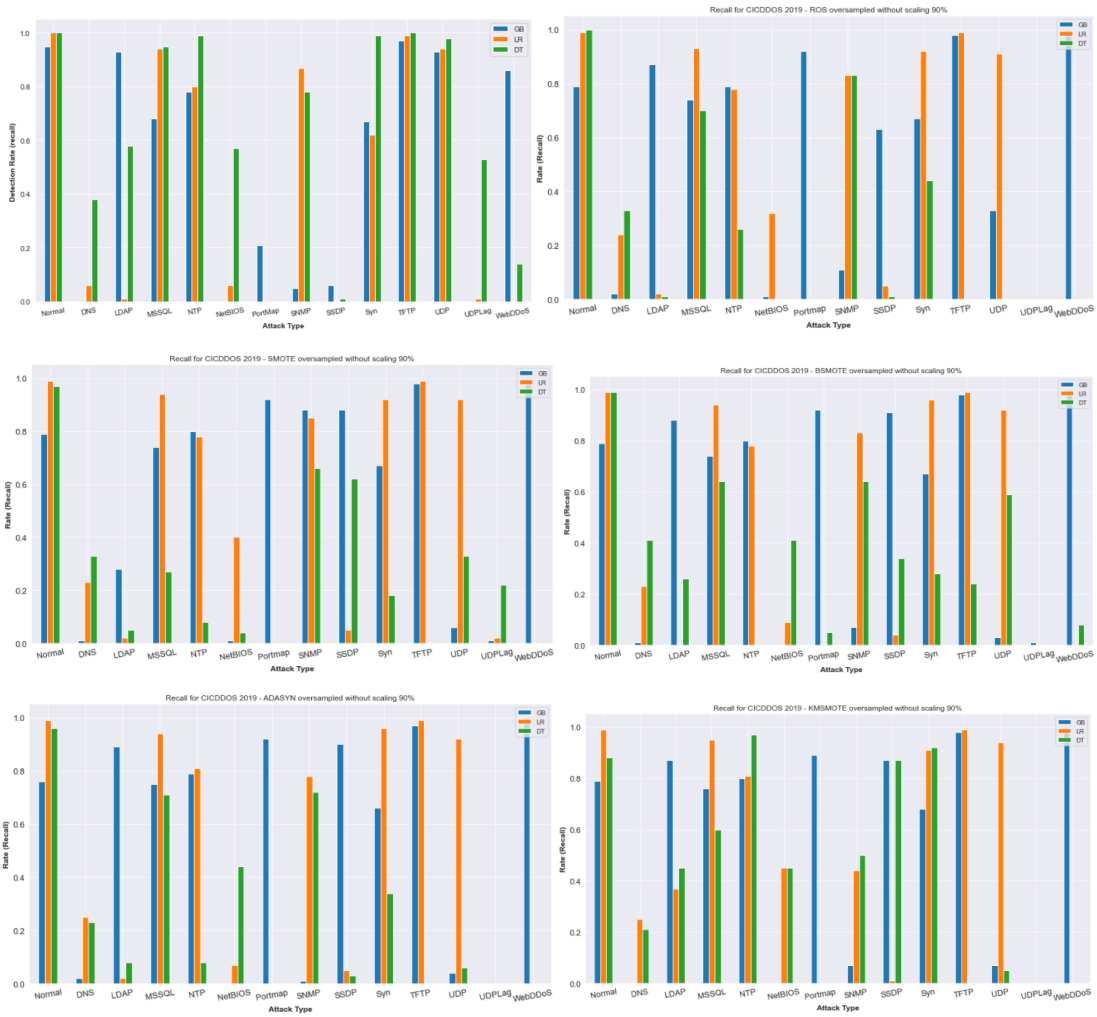


xvii. Oversampled 90% detection rate – non-scaled

Result	DNS	LD AP	MS SQL	NTP	Net BIOS	Nor	Port Map	SNMP	SSDP	SYN	TF TP	UDP	U Lag	W DDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14
GB ROS	0	0	58	79	0	100	0	47	3	70	0	0	0	0
GB SMOTE	49	25	58	80	40	100	15	52	3	70	0	3	47	10
GB BSMOTE	30	0	58	77	0	100	0	36	3	0	0	0	3	0
GB ADASYN	1	0	59	79	0	100	15	36	3	0	0	0	0	0
GB KSMOTE	0	54	51	80	40	100	0	0	0	0	0	3	47	0
LR ROS	0	0	6	0	0	48	0	93	2	49	5	0	0	0
LR SMOTE	0	0	6	0	0	28	0	92	2	59	5	0	0	0
LR BSMOTE	0	0	5	0	0	27	0	92	1	55	5	1	0	0

LR ADASYN	0	0	6	0	0	56	0	***93***	2	44	5	0	0	0
LR KSMOTE	0	0	99	0	0	64	0	0	0	44	5	1	0	0
DT ROS	***52***	38	89	98	50	100	3	73	***36***	97	100	68	***60***	***33***
DT SMOTE	47	47	93	98	53	100	5	77	24	***98***	100	82	50	17
DT BSMOTE	50	36	92	98	52	100	0	75	33	***98***	100	73	49	8
DT ADASYN	48	39	92	98	50	100	8	82	28	***98***	100	80	49	17
DT KSMOTE	49	50	94	98	52	100	0	71	29	***98***	100	73	52	17

xviii. Oversampled 90% comparison results – non-scaled

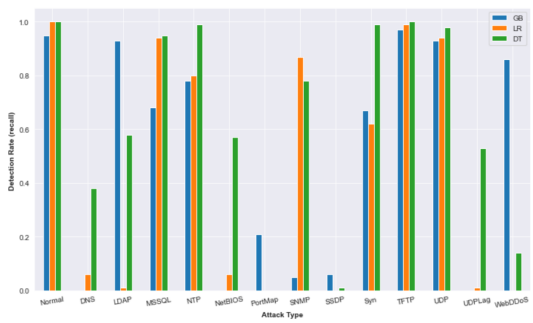


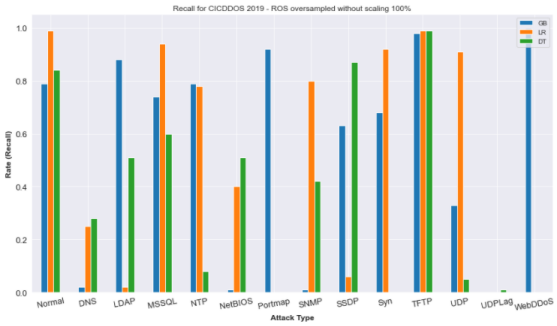
xix. Oversampled 100% detection rate – non-scaled

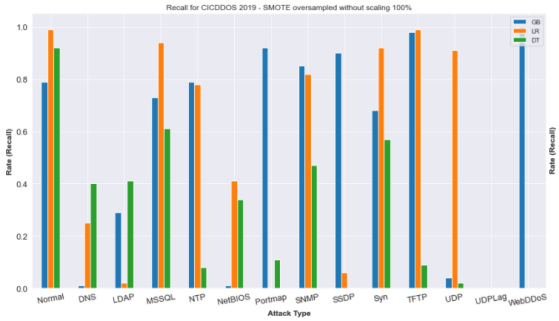
Result	DNS	LDAP	MSSQL	NTP	NetBIOS	Nor	Port Map	SNMP	SSDP	SYN	TFTP	UDP	ULag	WDDoS
GB	0	93	68	78	0	95	21	5	6	67	97	93	0	86
LR	6	7	94	80	6	100	0	87	0	62	99	94	1	0
DR	38	58	95	99	57	100	0	78	1	99	100	98	53	14

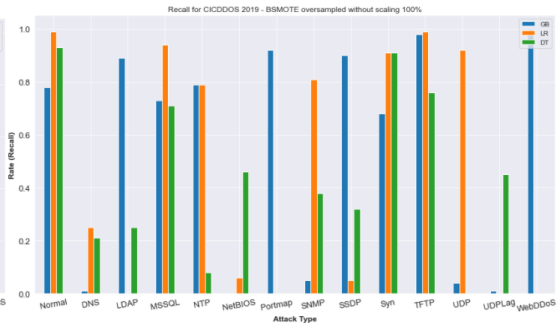
GB ROS	0	0	56	0	0	100	0	57	3	0	0	0	1	0
GB SMOTE	0	0	57	0	0	100	0	55	3	0	0	0	1	0
GB BSMOTE	0	36	59	0	0	100	0	12	3	0	0	0	3	0
GB ADASYN	1	0	59	0	0	100	0	36	3	0	0	0	1	0
GB KSMOTE	0	56	50	0	0	100	0	0	0	0	0	3	0	0
LR ROS	0	0	8	0	0	30	0	***93***	0	55	5	1	0	0
LR SMOTE	0	0	4	0	0	29	0	***93***	0	56	5	1	0	0
LR BSMOTE	0	0	12	0	0	29	0	90	2	50	5	0	0	0
LR ADASYN	0	0	6	0	0	26	0	***93***	1	58	5	0	0	0
LR KSMOTE	0	0	6	0	0	50	0	***93***	0	57	5	0	0	0
DT ROS	***52***	39	88	98	52	100	5	73	***36***	97	100	68	52	17
DT SMOTE	49	43	91	98	54	100	3	75	32	***98***	100	72	51	***33***
DT BSMOTE	50	38	92	98	48	100	5	76	34	***98***	100	71	50	0
DT ADASYN	48	39	92	98	52	100	0	79	21	***98***	100	84	51	25
DT KSMOTE	49	50	93	98	56	100	0	72	31	***98***	100	73	50	***33***

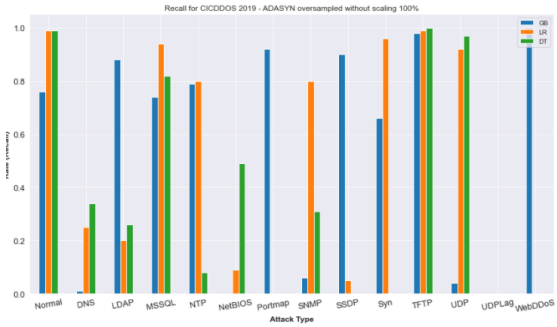
xx. Oversampled 100% comparison results – non-scaled

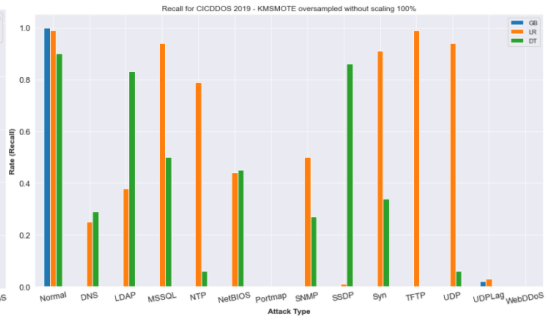












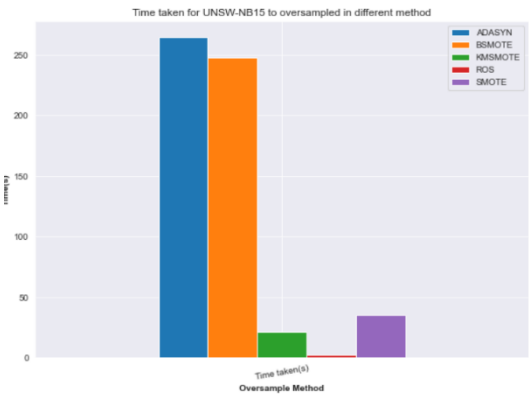
APPENDIX S: Top 3 detection rate among all observations for CICDDOS 2019

	DNS		LDAP		NETBIOS	
	Combination	**DR**	**Combination**	**DR**	**Combination**	**DR**
	Original	38	Original	58	Original	57
S1	DT-KMSMOTE60%	***71***	GB-ADASYN-70% to 100%	***89***	DT-ROS-10%	***61***
S2	DT-KMSMOTE-70%	54	GB-ROS-20%, 40%, 100% GB-KMSMOTE-30% GB-BSMOTE-40% to 90% GB-ADASYN-40%, 60%, 100%	88	DT-KMSMOTE-20%	58
S3	DT-KMSMOTE-10%	45	GB-SMOTE-10% GB-KMSMOTE-10% GB-ROS-80%	85		
NS1	DT-ROS-90%, 100%	52			DT-SMOTE-50% DT-KMSMOTE-80%	58
NS2	DT-ROS-80%	51				
NS3	DT-ROS-40%, 50%, 70% DT-BSMOTE-50%, 70%, 90% DT-ADASYN-50% DT-SMOTE-60%	50				

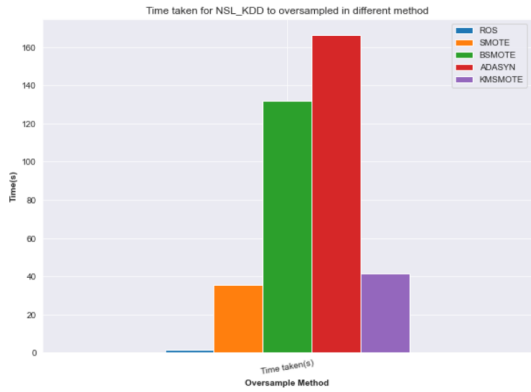
	PORTMAP		SNMP		SSDP	
	Combination	**DR**	**Combination**	**DR**	**Combination**	**DR**
	Original	21	Original	87	Original	2
S1	GB-ROS-10%, 40% GB-SMOTE-10%, 40% GB-KMSMOTE-10% GB-ADASYN-10% GB-All-20%, 30%, 50% - 100% GB-BSMOTE-40%	***92***	GB-KMSMOTE-80%	97	GB-KMSMOTE-30%, 40%, 50% GB-BSMOTE-60%, 70% to 90% GB-ADASYN-60%	***91***
S2			LR-ADASYN-30%	95	GB-ADASYN-80%, 100% GB-SMOTE-100% GB-BSMOTE-100%	90
S3			LR-ADASYN-20%	90	DT-KMSMOTE-10% GB-ADASYN-70%	89
NS1	GB-ADASYN-20%	36	LR-BSMOTE-60%	***99***	DT-ROS-70% to 100%	36
NS2			LR-SMOTE-80% LR-All-100%	93	DT-ROS-60%	35
NS3			LR-BSMOTE-40%, 70%, 80% LR-ROS-50%, 70%, 80% LR-ADASYN-50%, 80% LR-SMOTE-60%, 70%	92	DT-ROS-50%	34

	SYN		UDP LAG		WEB DDOS	
	Combination	**DR**	**Combination**	**DR**	**Combination**	**DR**
	Original	23	Original	50	Original	36
S1	GB-KMSMOTE-80%	***99***			GB-All-10% to 100%	***100***
S2	DT-ADASYN-10%	98				
S3	LR-BSMOTE-100%	97				
NS1	DT-All-10% to 30%, 50% DT-SMOTE-40% DT-ADASYN-40% DT-KMSMOTE-40%, 60%, 80%	***99***	DT-ROS-80%	***62***	DT-KMSMOTE-40%, 70%, 80%	58
NS2	DT-ROS-40% DT-BSMOTE-40% DT-All-60% no KMSMOTE DT-All-70% to 100%	98	DT-ROS-90%	60	DT-SMOTE-40%	50
NS3			DT-ROS-70%	54	DT-ROS-50%, 60%, 90% DT-KMSMOTE-50%, 100% DT-ADASYN-60% DT-SMOTE-100%	33

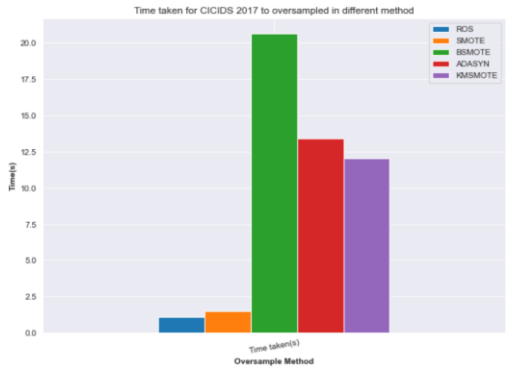
APPENDIX T: Runtime Performance for all oversampling methods in each data set



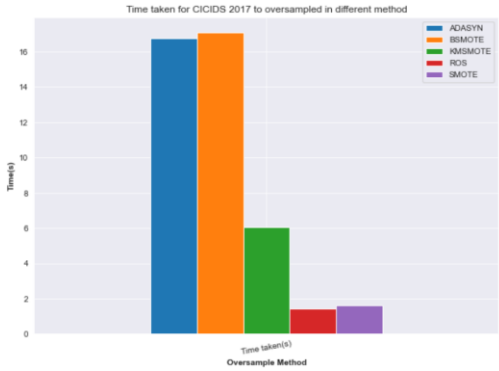
(a) UBSW-NB15 scaled & non-scaled



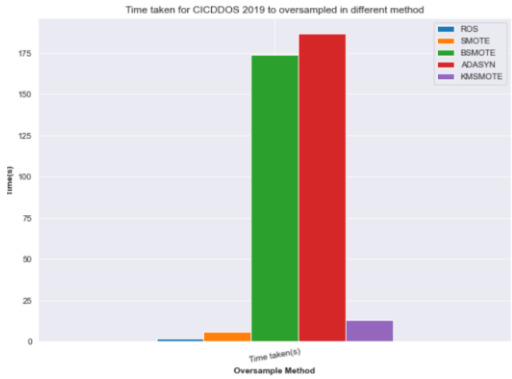
(b) NSL KDD scaled & non-scaled



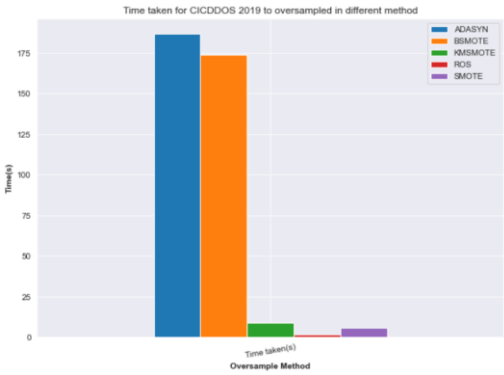
(c) CICIDS 2017 scaled



(d) CICIDS 2017 non-scaled



(e) CICDDOS 2019 scaled



(f) CICDDOS 2019 non-scaled

APPENDIX U: Overall attack performance metrics in scaled UNSW-NB15.

i. Oversampled 10% overall attacks performance metrics - scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
10% Oversampled				
ROS	GB	24.69	32.73	26.65
	LR	30.41	29.66	26.03
	DT	11.51	10.87	8.88
SMOTE	GB	24.04	32.52	25.97
	LR	30.06	30.38	26.07
	DT	15.37	22.95	10.06
BSMOTE	GB	24.25	**32.61**	26.10
	LR	30.67	26.77	26.29
	DT	13.98	11.35	8.50
ADASYN	GB	10.53	7.56	5.89
	LR	15.05	3.87	5.98
	DT	11.85	3.60	5.00
KMSMOTE	GB	**34.17**	22.81	21.59
	LR	30.72	24.13	**27.25**
	DT	4.72	9.84	2.55

ii. Oversampled 20% overall attacks performance metrics - scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		**36.85**	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
20% Oversampled				

ROS	GB	25.58	**32.74**	**26.79**
	LR	30.75	28.40	26.83
	DT	11.22	6.40	5.40
SMOTE	GB	23.69	32.02	25.69
	LR	30.80	27.04	26.80
	DT	13.97	7.36	6.26
BSMOTE	GB	22.64	31.56	25.19
	LR	30.42	29.59	26.70
	DT	13.74	5.33	5.79
ADASYN	GB	10.55	7.55	5.90
	LR	14.72	5.37	5.72
	DT	12.05	3.33	4.93
KMSMOTE	GB	20.62	31.59	24.31
	LR	30.16	30.84	26.81
	DT	15.86	15.69	9.76

iii. Oversampled 30% overall attacks performance metrics - scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		**36.85**	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
30% Oversampled				
ROS	GB	25.63	**32.39**	26.59
	LR	30.37	27.29	26.52
	DT	14.45	14.07	9.35
SMOTE	GB	22.63	31.33	24.81
	LR	30.11	26.91	25.73
	DT	16.40	8.24	8.84
BSMOTE	GB	22.60	31.22	24.65
	LR	30.23	27.78	26.63
	DT	13.53	5.59	5.54

ADASYN	GB	13.64	7.30	7.64
	LR	14.69	5.23	5.63
	DT	12.51	2.64	4.27
KMSMOTE	GB	25.21	31.16	26.00
	LR	30.46	29.47	**27.73**
	DT	15.48	17.24	10.00

iv. Oversampled 40% overall attacks performance metrics - scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		**36.85**	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
40% Oversampled				
ROS	GB	30.07	21.83	22.25
	LR	29.93	27.99	25.53
	DT	11.56	7.11	7.27
SMOTE	GB	22.20	30.74	24.60
	LR	29.41	28.29	25.50
	DT	17.13	12.90	10.09
BSMOTE	GB	22.31	30.39	24.51
	LR	29.48	30.19	25.64
	DT	16.97	8.73	9.15
ADASYN	GB	13.86	7.35	7.81
	LR	16.00	4.18	6.20
	DT	13.01	2.80	4.43
KMSMOTE	GB	21.06	**31.28**	24.10
	LR	30.15	29.09	**27.09**
	DT	15.20	8.04	10.40

v. **Oversampled 50% overall attacks performance metrics - scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		**36.85**	23.25	20.88
LR		31.08	26.66	**26.34**
DT		8.19	5.40	4.30
50% Oversampled				
ROS	GB	25.04	**31.83**	26.28
	LR	29.76	27.50	25.31
	DT	13.43	11.41	8.37
SMOTE	GB	22.23	30.25	24.48
	LR	29.48	29.92	24.94
	DT	15.95	8.51	9.90
BSMOTE	GB	25.52	30.62	25.44
	LR	28.80	30.62	25.91
	DT	14.66	10.13	7.39
ADASYN	GB	21.36	29.15	23.05
	LR	29.47	28.58	24.94
	DT	14.30	11.54	7.78
KMSMOTE	GB	30.72	22.51	22.93
	LR	29.68	30.73	25.82
	DT	8.84	4.36	5.50

vi. **Oversampled 60% overall attacks performance metrics - scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		**36.85**	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
60% Oversampled				
ROS	GB	30.90	22.67	22.89
	LR	29.85	27.15	25.86

	DT	8.86	9.21	6.22
SMOTE	GB	21.47	29.44	23.30
	LR	29.46	27.15	24.90
	DT	17.24	**32.64**	12.61
BSMOTE	GB	18.85	29.84	21.80
	LR	29.41	30.13	25.28
	DT	18.01	11.50	11.75
ADASYN	GB	21.28	29.30	23.30
	LR	29.41	28.32	25.65
	DT	14.69	23.85	9.70
KMSMOTE	GB	25.32	30.11	25.20
	LR	29.30	31.13	**26.82**
	DT	11.42	9.16	4.33

vii. Oversampled 70% overall attacks performance metrics - scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		**36.85**	23.25	20.88
LR		31.08	26.66	**26.34**
DT		8.19	5.40	4.30
70% Oversampled				
ROS	GB	31.62	23.66	24.00
	LR	29.41	30.39	25.85
	DT	9.50	5.61	6.06
SMOTE	GB	19.36	29.57	22.67
	LR	29.80	25.51	25.51
	DT	15.51	20.37	9.08
BSMOTE	GB	30.68	22.04	23.76
	LR	29.14	29.37	25.09
	DT	14.56	11.78	7.95
ADASYN	GB	21.37	29.18	23.37
	LR	29.52	30.39	25.80

	DT	13.38	19.59	8.55
KMSMOTE	GB	22.04	**30.63**	24.10
	LR	29.06	30.54	26.12
	DT	14.58	8.40	8.41

viii. Oversampled 80% overall attacks performance metrics - scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		**36.85**	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
80% Oversampled				
ROS	GB	31.89	24.52	24.55
	LR	29.93	27.96	25.16
	DT	9.94	5.30	5.83
SMOTE	GB	32.54	24.07	24.99
	LR	29.30	29.24	24.48
	DT	17.15	30.41	11.33
BSMOTE	GB	18.22	29.64	21.47
	LR	29.54	29.80	25.13
	DT	14.58	19.51	7.54
ADASYN	GB	21.96	29.44	23.84
	LR	29.67	28.16	24.83
	DT	15.01	12.17	5.91
KMSMOTE	GB	21.78	**30.77**	24.44
	LR	29.18	29.73	**26.71**
	DT	14.98	8.07	8.47

ix. Oversampled 90% overall attacks performance metrics - scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88

LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
90% Oversampled				
ROS	GB	32.12	24.23	25.45
	LR	29.18	29.30	25.03
	DT	8.97	5.53	5.19
SMOTE	GB	**32.70**	25.96	25.50
	LR	29.52	29.28	24.47
	DT	15.40	5.64	6.99
BSMOTE	GB	30.90	24.99	25.46
	LR	29.69	30.69	25.22
	DT	14.03	21.25	7.05
ADASYN	GB	32.06	20.57	23.78
	LR	28.72	28.15	24.71
	DT	14.95	24.29	8.04
KMSMOTE	GB	22.10	**31.02**	24.00
	LR	29.18	29.33	**26.70**
	DT	8.65	6.63	4.56

x. Oversampled 100% overall attacks performance metrics - scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		**36.85**	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
100% Oversampled				
ROS	GB	32.46	24.55	25.64
	LR	29.33	27.82	24.63
	DT	9.94	7.26	6.96
SMOTE	GB	32.58	25.06	26.11
	LR	29.58	**29.55**	24.96
	DT	12.96	27.11	9.57

BSMOTE	GB	31.64	24.86	26.35
	LR	29.40	31.07	25.93
	DT	15.04	9.43	8.99
ADASYN	GB	20.85	28.95	22.88
	LR	29.56	31.14	25.53
	DT	16.03	15.94	6.41
KMSMOTE	GB	31.17	24.13	**27.25**
	LR	28.38	29.21	26.27
	DT	11.60	9.50	7.19

APPENDIX V: Overall attack performance metrics in non-scaled UNSW-NB15.

i. Oversampled 10% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
10% Oversampled				
ROS	GB	26.20	22.05	17.89
	LR	22.31	13.43	15.47
	DT	41.72	38.31	**39.22**
SMOTE	GB	26.55	21.92	17.88
	LR	22.31	13.86	15.50
	DT	41.94	37.96	38.73
BSMOTE	GB	26.41	22.31	17.85
	LR	22.31	13.54	15.46
	DT	41.72	38.31	39.22
ADASYN	GB	8.81	3.59	4.27
	LR	23.60	24.65	21.50
	DT	42.86	**38.86**	38.31
	GB	27.27	31.71	17.79

KMSMOTE	LR	22.31	13.50	15.46
	DT	**43.00**	37.59	38.46

ii. Oversampled 20% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
20% Oversampled				
ROS	GB	26.74	21.83	17.89
	LR	22.31	13.52	15.47
	DT	**42.46**	**38.67**	38.38
SMOTE	GB	26.82	21.93	17.91
	LR	22.30	13.53	15.51
	DT	41.26	38.00	38.90
BSMOTE	GB	26.52	21.62	17.87
	LR	22.30	13.42	15.46
	DT	17.87	5.78	8.28
ADASYN	GB	8.81	3.60	4.27
	LR	3.66	4.82	1.66
	DT	41.64	37.76	39.07
KMSMOTE	GB	26.51	21.62	18.34
	LR	22.30	13.47	15.45
	DT	41.46	37.98	**39.39**

iii. Oversampled 30% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30

30% Oversampled				
ROS	GB	26.58	21.68	17.91
	LR	22.31	13.53	15.52
	DT	40.57	**38.62**	39.17
SMOTE	GB	26.89	21.55	17.95
	LR	22.31	13.50	15.50
	DT	**42.64**	38.13	38.37
BSMOTE	GB	26.40	21.29	18.45
	LR	22.30	13.46	15.47
	DT	17.50	5.29	8.26
ADASYN	GB	12.83	5.14	6.53
	LR	3.66	4.82	1.66
	DT	42.61	37.60	37.36
KMSMOTE	GB	31.50	15.48	19.16
	LR	25.11	14.50	17.88
	DT	41.20	38.62	**39.67**

iv. Oversampled 40% overall attacks performance metrics – non-scaled

Method	**Machine Learning**	**Recall**	**Precision**	**F-measure**
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
40% Oversampled				
ROS	GB	26.52	21.86	17.87
	LR	22.31	13.58	15.54
	DT	40.95	**38.50**	**39.24**
SMOTE	GB	26.70	21.00	18.57
	LR	22.31	13.50	15.52
	DT	**42.14**	36.76	37.88
BSMOTE	GB	27.20	20.84	18.40
	LR	22.30	13.39	15.48

	DT	41.63	38.35	**39.24**
ADASYN	GB	4.53	12.77	1.96
	LR	3.65	4.83	1.64
	DT	17.31	5.21	7.67
KMSMOTE	GB	27.73	18.56	19.54
	LR	22.79	13.74	16.10
	DT	39.87	37.99	38.43

v. Oversampled 50% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
50% Oversampled				
ROS	GB	31.71	17.27	19.47
	LR	22.31	13.55	15.57
	DT	40.46	38.03	38.65
SMOTE	GB	26.73	21.14	18.60
	LR	22.30	13.51	15.53
	DT	38.76	33.94	35.43
BSMOTE	GB	26.58	21.16	18.57
	LR	22.29	13.31	15.48
	DT	39.20	34.18	35.81
ADASYN	GB	31.64	17.21	19.30
	LR	22.34	14.81	15.66
	DT	**41.96**	36.11	37.12
KMSMOTE	GB	26.78	20.33	18.36
	LR	23.11	13.75	16.05
	DT	41.07	**38.68**	**39.37**

vi. Oversampled 60% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
60% Oversampled				
ROS	GB	27.59	20.41	18.55
	LR	22.32	13.65	15.71
	DT	40.33	**37.94**	38.77
SMOTE	GB	26.89	21.15	18.56
	LR	22.40	15.61	15.82
	DT	39.94	37.48	37.87
BSMOTE	GB	27.22	20.64	18.60
	LR	22.29	13.43	15.51
	DT	**40.85**	35.56	37.17
ADASYN	GB	31.70	17.13	19.44
	LR	22.27	13.30	15.48
	DT	40.69	36.02	37.49
KMSMOTE	GB	28.69	17.15	20.41
	LR	31.30	14.42	19.33
	DT	40.60	37.70	**38.91**

vii. Oversampled 70% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
70% Oversampled				
ROS	GB	27.60	20.25	18.65
	LR	2.77	4.01	1.25

	DT	40.16	38.14	38.94
SMOTE	GB	31.67	17.28	19.32
	LR	22.68	16.70	16.41
	DT	40.55	38.13	39.03
BSMOTE	GB	27.42	20.48	18.56
	LR	22.39	15.49	15.78
	DT	**40.94**	38.56	39.37
ADASYN	GB	31.66	17.06	19.37
	LR	22.34	15.07	15.70
	DT	40.78	**38.74**	**39.41**
KMSMOTE	GB	28.89	17.32	20.14
	LR	23.66	13.70	16.25
	DT	39.68	36.38	37.76

viii. Oversampled 80% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
80% Oversampled				
ROS	GB	31.66	17.31	19.36
	LR	2.95	4.51	1.45
	DT	39.91	38.33	38.76
SMOTE	GB	31.65	17.35	19.34
	LR	2.84	3.91	1.27
	DT	40.89	35.61	36.58
BSMOTE	GB	27.56	20.46	18.54
	LR	22.69	14.90	15.92
	DT	**40.91**	38.20	38.86
ADASYN	GB	31.59	16.64	19.22
	LR	5.25	2.24	2.35

	DT	38.39	35.04	34.79
KMSMOTE	GB	29.58	16.95	20.96
	LR	31.54	14.59	19.17
	DT	39.93	**38.94**	**39.13**

ix. Oversampled 90% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
90% Oversampled				
ROS	GB	31.63	16.81	19.31
	LR	4.18	3.54	2.51
	DT	40.68	38.20	38.81
SMOTE	GB	31.64	16.94	19.31
	LR	2.77	3.53	1.21
	DT	**41.19**	38.00	**39.07**
BSMOTE	GB	27.74	20.50	18.37
	LR	29.78	14.25	18.82
	DT	40.61	37.57	38.80
ADASYN	GB	31.56	15.37	19.20
	LR	13.54	2.58	3.80
	DT	40.91	38.11	38.92
KMSMOTE	GB	29.95	16.95	20.88
	LR	31.50	14.45	19.06
	DT	39.92	**39.25**	38.92

x. **Oversampled 100% overall attacks performance metrics – non-scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.85	23.25	20.88
LR		31.08	26.66	26.34
DT		8.19	5.40	4.30
100% Oversampled				
ROS	GB	31.62	16.85	19.27
	LR	12.45	3.72	5.32
	DT	40.24	38.07	39.00
SMOTE	GB	31.61	17.08	19.27
	LR	11.70	3.39	4.89
	DT	40.78	38.61	**39.01**
BSMOTE	GB	27.54	20.56	18.51
	LR	13.31	2.48	3.66
	DT	**41.55**	**39.27**	39.51
ADASYN	GB	31.56	15.73	19.20
	LR	11.57	2.46	3.24
	DT	40.82	37.68	38.63
KMSMOTE	GB	30.02	16.78	20.90
	LR	31.53	14.28	19.16
	DT	39.75	38.12	38.86

APPENDIX W: Overall attack performance metrics in scaled NSL KDD.

i. **Oversampled 10% overall attacks performance metrics –scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53

10% Oversampled				
ROS	GB	37.25	44.66	37.78
	LR	36.72	41.01	36.82
	DT	32.05	33.20	30.96
SMOTE	GB	37.29	44.68	37.82
	LR	36.93	42.98	37.55
	DT	29.97	37.52	30.73
BSMOTE	GB	**38.28**	45.12	38.32
	LR	37.74	42.85	37.90
	DT	31.78	32.62	31.64
ADASYN	GB	38.06	**45.67**	**39.18**
	LR	38.19	41.66	37.83
	DT	29.38	32.18	28.81
KMSMOTE	GB	31.79	16.13	21.31
	LR	9.88	8.95	2.42
	DT	31.24	32.08	30.17

ii. Oversampled 20% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
20% Oversampled				
ROS	GB	38.72	44.34	38.48
	LR	37.74	42.05	37.40
	DT	30.23	36.01	31.32
SMOTE	GB	38.74	44.71	38.46
	LR	37.49	41.12	37.60
	DT	29.46	31.93	29.41
BSMOTE	GB	38.76	44.11	38.46
	LR	38.11	40.71	37.76

	DT	32.04	37.60	31.73
ADASYN	GB	**39.31**	**45.32**	**39.84**
	LR	38.56	42.18	38.23
	DT	21.73	35.05	21.96
KMSMOTE	GB	32.56	20.28	24.70
	LR	10.90	9.47	2.32
	DT	11.11	23.59	6.27

iii. Oversampled 30% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	**45.12**	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
30% Oversampled				
ROS	GB	38.89	44.05	38.16
	LR	38.25	41.23	37.89
	DT	32.71	34.31	32.67
SMOTE	GB	39.66	44.50	39.34
	LR	37.74	40.54	37.41
	DT	29.74	37.64	30.43
BSMOTE	GB	39.51	44.41	39.72
	LR	38.43	40.39	37.63
	DT	32.13	32.04	29.95
ADASYN	GB	**39.76**	44.84	**39.94**
	LR	38.42	40.56	37.63
	DT	32.86	36.20	32.61
KMSMOTE	GB	32.32	20.45	24.48
	LR	10.96	10.44	2.41
	DT	9.88	18.15	6.31

iv. **Oversampled 40% overall attacks performance metrics –scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	**45.12**	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
40% Oversampled				
ROS	GB	39.04	43.73	38.41
	LR	37.87	40.32	37.04
	DT	32.48	34.55	32.53
SMOTE	GB	**39.84**	43.90	39.09
	LR	38.09	39.65	36.93
	DT	31.28	31.87	30.00
BSMOTE	GB	38.71	43.62	38.23
	LR	38.28	41.49	37.82
	DT	31.55	34.22	29.27
ADASYN	GB	39.15	44.07	**39.27**
	LR	38.77	42.77	38.08
	DT	33.71	34.06	27.46
KMSMOTE	GB	32.23	17.15	22.43
	LR	11.04	14.06	2.60
	DT	35.00	34.31	33.42

v. **Oversampled 50% overall attacks performance metrics –scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	**45.12**	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
50% Oversampled				
ROS	GB	39.16	43.20	37.72
	LR	38.25	41.26	37.08

	DT	31.77	33.94	31.61
SMOTE	GB	39.30	43.24	38.43
	LR	38.50	39.65	37.27
	DT	28.99	35.34	29.31
BSMOTE	GB	**39.36**	43.65	**38.58**
	LR	38.91	42.92	38.43
	DT	36.04	40.59	34.98
ADASYN	GB	38.88	43.08	38.06
	LR	38.19	39.86	37.31
	DT	34.10	35.89	32.93
KMSMOTE	GB	3.33	17.61	2.75
	LR	10.90	5.95	2.32
	DT	35.53	19.67	25.41

vi. Oversampled 60% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
60% Oversampled				
ROS	GB	**40.06**	42.89	38.43
	LR	38.28	34.13	34.26
	DT	32.09	32.60	31.67
SMOTE	GB	39.80	42.46	38.20
	LR	38.01	37.08	35.61
	DT	31.49	31.81	30.36
BSMOTE	GB	38.77	43.16	38.30
	LR	37.89	39.34	36.66
	DT	35.82	45.01	36.78
ADASYN	GB	39.60	43.79	**39.31**
	LR	38.46	39.31	37.17

	DT	29.73	**45.41**	28.76
KMSMOTE	GB	0.00072	15.82	0.031
	LR	10.90	7.97	2.35
	DT	14.37	23.52	11.68

vii. Oversampled 70% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
70% Oversampled				
ROS	GB	**40.20**	42.28	**38.66**
	LR	38.44	33.18	33.58
	DT	30.31	32.24	29.87
SMOTE	GB	40.12	42.05	38.08
	LR	38.70	39.42	37.01
	DT	29.65	35.57	29.29
BSMOTE	GB	38.84	42.47	37.79
	LR	37.86	39.43	36.59
	DT	36.01	**46.06**	38.64
ADASYN	GB	39.16	42.67	38.11
	LR	38.76	39.78	37.12
	DT	31.73	33.96	26.02
KMSMOTE	GB	33.76	21.49	25.84
	LR	11.12	15.49	2.80
	DT	12.52	33.86	10.14

viii. Oversampled 80% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47

LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
80% Oversampled				
ROS	GB	39.92	40.54	36.51
	LR	38.38	36.92	35.97
	DT	33.08	33.27	32.52
SMOTE	GB	**40.19**	40.45	37.03
	LR	39.90	33.20	33.60
	DT	29.89	35.23	29.97
BSMOTE	GB	38.90	42.18	37.66
	LR	38.73	40.23	37.49
	DT	35.96	**45.85**	38.67
ADASYN	GB	40.03	43.41	**39.50**
	LR	39.08	39.04	37.00
	DT	29.28	41.90	29.02
KMSMOTE	GB	32.15	13.71	18.84
	LR	11.04	14.41	2.61
	DT	36.95	20.39	24.38

ix. Oversampled 90% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
90% Oversampled				
ROS	GB	**40.50**	39.83	37.00
	LR	39.70	30.85	32.92
	DT	27.58	35.44	29.08
SMOTE	GB	39.71	39.90	36.38
	LR	38.32	31.27	32.64
	DT	32.56	29.94	29.60

BSMOTE	GB	39.21	41.75	37.58
	LR	38.42	38.35	36.34
	DT	36.01	45.85	37.26
ADASYN	GB	38.89	41.91	37.61
	LR	38.92	39.73	37.21
	DT	17.98	34.88	15.33
KMSMOTE	GB	32.35	13.43	18.48
	LR	10.94	10.29	2.42
	DT	39.09	24.23	26.53

x. Oversampled 100% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
100% Oversampled				
ROS	GB	**40.66**	39.81	37.20
	LR	38.84	29.30	31.49
	DT	29.67	35.42	30.81
SMOTE	GB	40.42	39.47	36.36
	LR	38.70	30.63	32.12
	DT	32.19	29.48	29.42
BSMOTE	GB	39.73	41.48	37.85
	LR	38.54	38.30	36.49
	DT	37.45	49.59	40.92
ADASYN	GB	39.37	41.75	37.90
	LR	37.75	39.47	36.50
	DT	28.78	28.56	26.52
KMSMOTE	GB	32.34	13.44	18.46
	LR	11.04	14.67	2.61
	DT	19.56	14.14	12.83

APPENDIX X: Overall attack performance metrics in non-scaled NSL KDD.

i. Oversampled 10% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
10% Oversampled				
ROS	GB	32.71	37.69	32.56
	LR	21.89	23.96	22.82
	DT	32.79	**52.00**	37.43
SMOTE	GB	32.52	37.39	32.27
	LR	21.90	23.98	22.84
	DT	33.27	50.15	36.30
BSMOTE	GB	29.40	28.03	23.83
	LR	20.98	24.23	22.58
	DT	**34.16**	50.77	**37.91**
ADASYN	GB	31.19	33.15	28.94
	LR	21.79	24.36	23.14
	DT	33.28	50.16	36.92
KMSMOTE	GB	32.38	37.47	32.10
	LR	21.85	24.04	22.83
	DT	32.67	52.56	37.32

ii. Oversampled 20% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53

20% Oversampled				
ROS	GB	31.69	35.47	30.12
	LR	22.28	23.76	22.85
	DT	33.07	51.85	37.30
SMOTE	GB	**33.95**	37.18	34.16
	LR	22.30	23.76	22.87
	DT	33.19	51.87	37.38
BSMOTE	GB	29.11	25.35	21.78
	LR	22.21	23.68	23.19
	DT	31.09	48.39	34.37
ADASYN	GB	30.70	31.51	27.07
	LR	22.29	30.18	22.82
	DT	31.64	47.96	35.41
KMSMOTE	GB	32.82	37.88	32.63
	LR	22.93	21.06	21.84
	DT	33.54	**52.47**	**37.78**

iii. Oversampled 30% overall attacks performance metrics – non-scaled

Method	**Machine Learning**	**Recall**	**Precision**	**F-measure**
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
30% Oversampled				
ROS	GB	32.16	36.54	30.70
	LR	22.66	22.37	22.60
	DT	35.35	48.79	37.76
SMOTE	GB	33.91	38.58	33.57
	LR	22.89	22.60	22.80
	DT	32.92	51.63	**37.11**
BSMOTE	GB	28.74	23.82	21.08
	LR	22.38	23.77	22.94

	DT	30.57	43.06	33.80
ADASYN	GB	30.44	30.16	26.50
	LR	22.75	28.09	22.18
	DT	33.90	46.89	38.49
KMSMOTE	GB	34.15	40.61	35.20
	LR	22.72	22.23	22.52
	DT	33.23	**52.17**	37.91

iv. Oversampled 40% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
40% Oversampled				
ROS	GB	34.46	39.44	34.44
	LR	23.79	21.24	22.43
	DT	34.20	52.36	38.41
SMOTE	GB	32.39	37.03	31.72
	LR	23.98	24.19	22.40
	DT	**35.91**	52.55	39.17
BSMOTE	GB	28.45	24.09	20.80
	LR	22.93	22.93	22.71
	DT	33.89	**53.46**	39.28
ADASYN	GB	29.60	29.58	24.75
	LR	22.69	22.32	22.62
	DT	35.82	50.40	39.95
KMSMOTE	GB	34.81	39.69	34.62
	LR	22.88	30.86	22.73
	DT	32.61	50.83	36.64

v. **Oversampled 50% overall attacks performance metrics – non-scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
50% Oversampled				
ROS	GB	**34.93**	39.80	35.26
	LR	24.40	20.77	22.44
	DT	32.83	**50.47**	36.48
SMOTE	GB	34.45	39.61	34.76
	LR	24.14	21.56	22.93
	DT	34.64	51.71	**38.37**
BSMOTE	GB	27.06	13.95	18.14
	LR	23.07	21.89	22.66
	DT	32.04	50.25	35.94
ADASYN	GB	29.05	28.28	23.73
	LR	23.66	21.57	22.48
	DT	34.27	49.06	37.90
KMSMOTE	GB	28.92	10.48	15.60
	LR	24.50	21.45	21.72
	DT	32.86	41.37	36.32

vi. **Oversampled 60% overall attacks performance metrics – non-scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
60% Oversampled				
ROS	GB	31.86	35.31	30.04
	LR	23.22	21.24	22.41

	DT	36.13	**52.63**	**40.14**
SMOTE	GB	32.61	37.23	32.01
	LR	23.85	22.13	23.02
	DT	33.29	51.45	37.51
BSMOTE	GB	27.11	14.03	17.67
	LR	23.61	21.83	22.72
	DT	31.56	48.69	35.22
ADASYN	GB	29.08	27.81	23.74
	LR	24.65	21.02	22.56
	DT	35.31	44.6	38.21
KMSMOTE	GB	**38.70**	31.26	29.11
	LR	24.84	26.19	21.95
	DT	31.89	49.48	35.09

vii. Oversampled 70% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
70% Oversampled				
ROS	GB	35.02	42.39	36.49
	LR	25.85	19.82	22.28
	DT	34.17	51.46	38.68
SMOTE	GB	32.41	37.17	31.79
	LR	24.58	21.11	22.97
	DT	35.07	51.02	38.3
BSMOTE	GB	27.21	14.15	17.69
	LR	24.12	21.8	22.78
	DT	32.27	48.58	35.41
ADASYN	GB	28.86	27.7	23.49
	LR	25.20	21.12	22.87

	DT	**35.40**	51.31	**40.10**
KMSMOTE	GB	31.94	34.75	29.86
	LR	24.46	30.21	23.15
	DT	33.30	**52.25**	37.69

viii. Oversampled 80% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
80% Oversampled				
ROS	GB	34.80	42.79	36.74
	LR	25.35	20.08	22.39
	DT	**35.37**	50.95	38.50
SMOTE	GB	31.59	34.6	29.56
	LR	24.82	20.52	22.63
	DT	32.83	51.37	37.10
BSMOTE	GB	27.34	13.91	17.73
	LR	26.43	17.48	21.10
	DT	32.71	49.49	35.88
ADASYN	GB	28.87	25.99	21.98
	LR	25.06	33.28	22.29
	DT	34.10	42.55	36.52
KMSMOTE	GB	31.31	33.75	28.80
	LR	25.17	20.41	22.53
	DT	35.33	**52.84**	**39.94**

ix. Oversampled 90% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
90% Oversampled				
ROS	GB	31.71	33.2	28.74
	LR	26.78	31.27	24.27
	DT	**35.99**	52.47	40.06
SMOTE	GB	31.52	34.46	29.63
	LR	25.71	18.81	21.96
	DT	33.11	50.96	37.31
BSMOTE	GB	27.11	14.8	17.45
	LR	25.27	20.26	22.39
	DT	33.48	50.88	38.08
ADASYN	GB	28.59	25.11	21.54
	LR	25.98	19.64	22.48
	DT	32.91	45.19	36.12
KMSMOTE	GB	34.81	41.7	35.65
	LR	25.24	19.97	22.09
	DT	35.72	**53.25**	**40.36**

x. Oversampled 100% overall attacks performance metrics – non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		32.52	45.12	34.47
LR		30.37	43.27	36.97
DT		28.67	34.98	29.53
100% Oversampled				
ROS	GB	31.80	33.89	29.54
	LR	25.78	19.96	22.36

	DT	34.84	49.64	**39.12**
SMOTE	GB	31.27	33.23	28.48
	LR	25.75	17.82	21.21
	DT	32.84	**51.66**	37.02
BSMOTE	GB	27.29	14.66	17.60
	LR	25.89	19.56	21.97
	DT	32.33	48.32	36.84
ADASYN	GB	28.76	25.49	21.74
	LR	25.85	18.77	21.55
	DT	32.84	47.97	35.8
KMSMOTE	GB	**34.99**	41.56	36.12
	LR	25.21	26.33	22.33
	DT	34.19	46.68	37.93

APPENDIX Y: Overall attack performance metrics in scaled CICIDS 2017.

i. Oversampled 10% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	**37.11**	37.10
DT		36.70	36.70	36.70
10% Oversampled				
ROS	GB	0	0.08	0.06
	LR	**37.90**	36.11	**37.26**
	DT	13.43	16.05	12.46
SMOTE	GB	0.00281	0.08	0.06
	LR	37.31	36.15	36.69
	DT	13.32	16.51	12.40
BSMOTE	GB	0	0.08	0.06
	LR	37.25	36.73	36.63

	DT	13.16	14.15	12.07
ADASYN	GB	0	0.08	0.06
	LR	37.17	36	36.52
	DT	13.04	22.97	11.62
KMSMOTE	GB	0	0.08	0.06
	LR	37.78	36.05	37.17
	DT	13.31	15.75	11.75

ii. Oversampled 20% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	**37.11**	**37.10**
DT		36.70	36.70	36.70
20% Oversampled				
ROS	GB	0	0.08	0.06
	LR	37.29	36.15	36.65
	DT	13.38	30.37	12.08
SMOTE	GB	0	0.08	0.06
	LR	**37.81**	36.07	37.2
	DT	12.98	27.85	11.48
BSMOTE	GB	0	0.08	0.06
	LR	37.22	36.09	36.6
	DT	13.16	23.66	12.5
ADASYN	GB	0	0.08	0.06
	LR	37.21	36.04	36.56
	DT	13.14	27.07	12
KMSMOTE	GB	0	0.08	0.06
	LR	36.91	36.38	36.26
	DT	13.5	29.27	12.62

iii. Oversampled 30% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	**37.11**	37.10
DT		36.70	36.70	36.70
30% Oversampled				
ROS	GB	0	0.08	0.06
	LR	37.57	36.39	36.94
	DT	13.03	23.74	12.03
SMOTE	GB	0.00281	0.08	0.06
	LR	37.23	36.71	**37.21**
	DT	13.41	28.48	12.46
BSMOTE	GB	0	0.08	0.06
	LR	**37.80**	36.08	37.20
	DT	13.23	25.46	12.21
ADASYN	GB	0	0.08	0.06
	LR	37.25	36.1	36.62
	DT	13.63	21.6	11.95
KMSMOTE	GB	0	0.08	0.06
	LR	37.79	36.06	37.18
	DT	13.32	28.17	12.26

iv. Oversampled 40% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	**37.11**	37.10
DT		36.70	36.70	36.70
40% Oversampled				
ROS	GB	0	0.08	0.06
	LR	**37.88**	36.17	**37.28**

	DT	13.05	23.73	12.04
SMOTE	GB	0.00281	0.08	0.06
	LR	37.86	36.51	37.27
	DT	12.77	23.29	10.93
BSMOTE	GB	0	0.08	0.06
	LR	37.48	36.34	36.84
	DT	14.42	25.45	13.45
ADASYN	GB	0.000281	0.08	0.06
	LR	37.28	36.14	36.64
	DT	14.86	21.87	13.05
KMSMOTE	GB	0	0.08	0.06
	LR	37.47	36.38	36.83
	DT	14.71	23.43	13.76

v. Oversampled 50% overall attacks performance metrics –scaled

Method	**Machine Learning**	**Recall**	**Precision**	**F-measure**
Original				
GB		30.39	34.70	35.55
LR		32.50	**37.11**	37.10
DT		36.70	36.70	36.70
50% Oversampled				
ROS	GB	0	0.08	0.06
	LR	37.52	36.36	36.89
	DT	14.46	28.18	12.66
SMOTE	GB	0.00281	0.08	0.06
	LR	37.5	36.66	36.91
	DT	13.1	27.6	11.6
BSMOTE	GB	0	0.08	0.06
	LR	37.33	36.24	36.71
	DT	12.73	20.32	11.46
ADASYN	GB	0	0.08	0.06
	LR	37.16	36.01	36.52

	DT	14.95	26.11	13.4
KMSMOTE	GB	0	0.08	0.06
	LR	**37.86**	36.16	**37.26**
	DT	14.52	28.08	13.09

vi. Oversampled 60% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	**37.11**	37.10
DT		36.70	36.70	36.70
60% Oversampled				
ROS	GB	0	0.08	0.06
	LR	37.73	36.02	37.13
	DT	14.75	28.2	13.52
SMOTE	GB	0.0281	0.08	0.06
	LR	37.5	36.34	36.86
	DT	12.88	21.08	10.94
BSMOTE	GB	0	0.08	0.06
	LR	37.00	35.94	36.38
	DT	12.34	24.62	10.02
ADASYN	GB	0.000281	0.08	0.06
	LR	36.93	35.91	36.32
	DT	12.17	25.2	10.33
KMSMOTE	GB	0. 000471	0.08	0.06
	LR	**37.87**	36.24	**37.29**
	DT	14.51	27.9	13.04

vii. Oversampled 70% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	**37.11**	37.10
DT		36.70	36.70	36.70
70% Oversampled				
ROS	GB	0	0.08	0.06
	LR	**37.79**	36.12	**37.21**
	DT	14.78	28.44	13.3
SMOTE	GB	0	0.08	0.06
	LR	**37.79**	36.49	37.2
	DT	14.23	21.55	12.96
BSMOTE	GB	0	0.08	0.06
	LR	37.00	35.95	36.39
	DT	11.32	20.97	9.51
ADASYN	GB	0	0.08	0.06
	LR	36.96	35.92	36.34
	DT	11.32	24.81	9.78
KMSMOTE	GB	0	0.44	0.05
	LR	36.49	36.71	36.5
	DT	14.46	27.69	12.98

viii. Oversampled 80% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	**37.11**	37.10
DT		36.70	36.70	36.70
80% Oversampled				
ROS	GB	0	0.08	0.06
	LR	**38.04**	36.98	**37.67**

	DT	14.3	19.02	11.5
SMOTE	GB	0	0.08	0.06
	LR	37.53	36.68	37.02
	DT	14.7	25.67	11.77
BSMOTE	GB	0	0.08	0.06
	LR	37.43	35.88	36.27
	DT	11.16	25.11	9.9
ADASYN	GB	0.000281	0.08	0.06
	LR	37.56	35.38	36.38
	DT	12.37	24.07	10.39
KMSMOTE	GB	0	0.08	0.06
	LR	37.33	36.74	36.95
	DT	14.54	25.21	13.22

ix. Oversampled 90% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
90% Oversampled				
ROS	GB	0	0.08	0.06
	LR	**38.08**	37.23	37.27
	DT	14.54	25.49	11.55
SMOTE	GB	0	0.08	0.06
	LR	37.57	37.47	37.43
	DT	14.43	25.51	11.48
BSMOTE	GB	0	0.08	0.06
	LR	36.74	35.82	36.18
	DT	11.3	25.14	9.66
ADASYN	GB	0	0.08	0.06
	LR	36.8	35.83	36.21

	DT	11.51	24.84	10.19
KMSMOTE	GB	0	0.08	0.06
	LR	37.77	**37.34**	**37.64**
	DT	14.77	26.48	11.77

x. Oversampled 100% overall attacks performance metrics –scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
100% Oversampled				
ROS	GB	0	0.08	0.06
	LR	**38.69**	37.09	37.43
	DT	14.52	19.47	12.41
SMOTE	GB	0.0281	0.08	0.06
	LR	37.96	**37.25**	37.22
	DT	14.72	26.31	12
BSMOTE	GB	0	0.08	0.06
	LR	36.86	35.92	36.3
	DT	11.41	19.47	9.48
ADASYN	GB	0	0.08	0.06
	LR	37.32	35.82	36.15
	DT	11.43	20.18	10.04
KMSMOTE	GB	0	0.08	0.06
	LR	37.74	37.22	**37.65**
	DT	14.62	26	11.74

APPENDIX Z: Overall attack performance metrics in non-scaled CICIDS 2017.

i. Oversampled 10% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
10% Oversampled				
ROS	GB	36.3	30.25	32.08
	LR	20.84	21.05	21.04
	DT	39.71	39.71	**39.71**
SMOTE	GB	36.32	30.17	32.02
	LR	21.27	20.88	21.51
	DT	39.71	**39.72**	39.72
BSMOTE	GB	36.26	30.17	31.99
	LR	21.49	21.7	21.71
	DT	**39.72**	39.68	39.69
ADASYN	GB	36.27	30.05	31.88
	LR	21.54	21.79	21.16
	DT	39.68	39.69	39.68
KMSMOTE	GB	36.44	30.24	32.07
	LR	21.64	21.32	21.26
	DT	39.69	39.70	39.69

ii. Oversampled 20% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70

20% Oversampled				
ROS	GB	36.24	30.32	32.11
	LR	21.37	21.57	21.58
	DT	39.71	**39.70**	**39.70**
SMOTE	GB	36.28	30.2	32.03
	LR	21.59	21.80	21.19
	DT	**39.73**	39.69	**39.70**
BSMOTE	GB	36.27	30.13	31.97
	LR	21.12	21.26	21.34
	DT	39.67	39.67	39.67
ADASYN	GB	36.25	30.03	31.83
	LR	20.38	19.54	20.26
	DT	39.7	39.71	**39.70**
KMSMOTE	GB	36.28	30.28	32.08
	LR	21.41	21.63	21.62
	DT	39.71	**39.70**	**39.70**

iii. Oversampled 30% overall attacks performance metrics –non-scaled

Method	**Machine Learning**	**Recall**	**Precision**	**F-measure**
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
30% Oversampled				
ROS	GB	36.28	30.3	32.11
	LR	20.31	19.58	20.19
	DT	39.72	39.72	39.72
SMOTE	GB	36.31	30.3	32.12
	LR	21.38	21.58	21.59
	DT	**40.52**	**40.12**	**40.33**
BSMOTE	GB	36.23	30.11	31.94
	LR	21.52	21.75	21.13

	DT	39.70	39.69	39.70
ADASYN	GB	36.28	30.07	31.88
	LR	21.21	21.46	21.44
	DT	39.67	39.67	39.67
KMSMOTE	GB	36.29	30.23	32.06
	LR	21.19	21.39	21.41
	DT	39.69	39.70	39.70

iv. Oversampled 40% overall attacks performance metrics –non-scaled

Method	**Machine Learning**	**Recall**	**Precision**	**F-measure**
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
40% Oversampled				
ROS	GB	36.55	30.16	32.06
	LR	21.06	21.38	21.29
	DT	39.7	39.70	**39.70**
SMOTE	GB	36.65	30.13	32.06
	LR	20.44	19.5	19.77
	DT	**42.74**	**39.71**	40.21
BSMOTE	GB	36.18	30.23	32.01
	LR	20.67	21.06	20.78
	DT	39.68	39.65	39.66
ADASYN	GB	36.55	30.10	31.99
	LR	21.55	21.16	21.17
	DT	39.71	39.68	39.68
KMSMOTE	GB	36.32	30.28	32.12
	LR	21.68	21.36	21.31
	DT	39.69	39.70	39.69

v. **Oversampled 50% overall attacks performance metrics –non-scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
50% Oversampled				
ROS	GB	36.25	30.14	32.05
	LR	21.03	21.34	21.25
	DT	**40.68**	**39.74**	**39.74**
SMOTE	GB	36.60	30.30	32.10
	LR	21.63	21.91	21.14
	DT	39.70	39.71	39.70
BSMOTE	GB	36.22	30.17	32.00
	LR	20.92	19.25	20.22
	DT	39.68	39.68	39.68
ADASYN	GB	36.50	30.07	31.92
	LR	21.50	21.19	21.28
	DT	39.66	39.64	39.64
KMSMOTE	GB	36.35	30.33	32.15
	LR	20.43	19.84	20.21
	DT	39.71	39.72	39.72

vi. **Oversampled 60% overall attacks performance metrics –non-scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
60% Oversampled				
ROS	GB	36.3	30.32	32.13
	LR	20.8	21.12	20.96

	DT	39.70	39.70	39.70
SMOTE	GB	36.44	30.09	32.01
	LR	21.35	21.54	21.57
	DT	**39.71**	**39.72**	**39.71**
BSMOTE	GB	36.21	30.39	32.22
	LR	20.67	20.79	20.51
	DT	39.69	39.67	39.67
ADASYN	GB	36.4	30.05	31.98
	LR	21.48	21.72	21.7
	DT	39.68	39.69	39.68
KMSMOTE	GB	36.33	30.18	32.05
	LR	21.17	21.3	21.39
	DT	39.69	39.70	39.70

vii. Oversampled 70% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
70% Oversampled				
ROS	GB	35.95	30.31	32.48
	LR	21.15	21.28	21.37
	DT	39.73	**39.74**	39.73
SMOTE	GB	36.22	30.39	32.13
	LR	21.56	21.12	21.17
	DT	39.73	39.74	**39.74**
BSMOTE	GB	36.10	30.24	32.11
	LR	21.49	21.07	21.09
	DT	**40.52**	39.67	39.67
ADASYN	GB	36.33	30.07	31.95
	LR	20.27	19.75	20.11

	DT	39.69	39.69	39.69
KMSMOTE	GB	36.29	30.34	32.16
	LR	21.22	20.74	21.44
	DT	39.69	39.70	39.70

viii. Oversampled 80% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
80% Oversampled				
ROS	GB	36.35	30.11	31.99
	LR	21.25	20.89	20.83
	DT	39.68	39.69	39.69
SMOTE	GB	36.32	30.32	32.15
	LR	21.26	21.49	21.48
	DT	**41.73**	39.71	**39.71**
BSMOTE	GB	35.79	30.4	32.43
	LR	21.49	21.06	21.71
	DT	39.69	39.68	39.69
ADASYN	GB	35.81	29.88	31.97
	LR	21.60	21.24	21.23
	DT	39.69	**39.70**	39.69
KMSMOTE	GB	36.37	30.25	32.12
	LR	21.56	21.13	21.17
	DT	39.68	**39.70**	39.69

ix. Oversampled 90% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
90% Oversampled				
ROS	GB	36.17	30.11	31.86
	LR	21.22	21.42	21.45
	DT	39.70	39.72	39.72
SMOTE	GB	36.30	30.3	32.12
	LR	21.19	21.39	21.40
	DT	39.68	39.69	39.69
BSMOTE	GB	35.68	30.29	32.27
	LR	21.24	21.43	21.46
	DT	39.68	39.68	39.68
ADASYN	GB	35.89	29.89	32.04
	LR	20.53	20.83	20.49
	DT	39.69	39.69	39.69
KMSMOTE	GB	36.28	30.37	32.19
	LR	20.35	20.34	20.13
	DT	**39.72**	**39.73**	**39.72**

x. Oversampled 100% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		30.39	34.70	35.55
LR		32.50	37.11	37.10
DT		36.70	36.70	36.70
100% Oversampled				
ROS	GB	36.50	30.45	32.25
	LR	20.78	20.01	20.08

	DT	39.70	**39.71**	39.70
SMOTE	GB	36.31	30.24	32.06
	LR	21.09	21.43	21.24
	DT	39.69	39.71	39.7
BSMOTE	GB	35.68	30.41	32.38
	LR	21.20	21.40	21.36
	DT	39.67	39.68	39.68
ADASYN	GB	36.05	29.86	32.12
	LR	20.42	19.65	20.31
	DT	**39.72**	39.68	39.68
KMSMOTE	GB	36.43	30.34	32.22
	LR	21.30	21.58	21.50
	DT	39.71	**39.71**	**39.71**

APPENDIX AA: Overall attack performance metrics in scaled CICDDOS 2019.

i. Oversampled 10% overall attacks performance metrics – scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
10% Oversampled				
ROS	GB	37.61	38.61	37.9
	LR	**40.14**	38.14	**38.14**
	DT	17.14	35.23	16.18
SMOTE	GB	36.61	37.61	36.9
	LR	**40.14**	38.14	**38.14**
	DT	17.14	25.23	16.18
BSMOTE	GB	37.61	38.61	37.9
	LR	**40.14**	38.14	**38.14**
	DT	16.61	22.23	14.18

ADASYN	GB	36.61	37.61	36.90
	LR	**40.14**	38.14	**38.14**
	DT	19.14	24.61	16.14
KMSMOTE	GB	36.61	37.61	36.90
	LR	**40.14**	**39.14**	**38.14**
	DT	14.14	32.23	14.33

ii. Oversampled 20% overall attacks performance metrics – scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
20% Oversampled				
ROS	GB	36.61	37.61	36.90
	LR	**40.14**	38.14	38.14
	DT	17.18	23.18	14.18
SMOTE	GB	37.61	**38.61**	37.90
	LR	**40.14**	38.14	**42.85**
	DT	16.14	31.23	13.18
BSMOTE	GB	36.61	37.61	36.9
	LR	**40.14**	38.14	38.14
	DT	17.66	25.18	17.66
ADASYN	GB	36.61	37.61	36.90
	LR	**40.16**	38.14	38.14
	DT	10.14	10.71	8.18
KMSMOTE	GB	35.61	**38.61**	35.90
	LR	**40.20**	38.14	38.14
	DT	16.28	31.76	12.76

iii. **Oversampled 30% overall attacks performance metrics – scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
30% Oversampled				
ROS	GB	36.64	38.61	36.9
	LR	40.15	**39.14**	**38.14**
	DT	13.8	16.28	13.8
SMOTE	GB	37.61	37.61	36.9
	LR	**40.16**	38.14	**38.14**
	DT	14.14	18.23	12.18
BSMOTE	GB	36.62	37.61	36.9
	LR	40.15	**39.14**	**38.14**
	DT	17.14	23.71	16.18
ADASYN	GB	37.61	37.61	37.9
	LR	38.14	40.14	**38.14**
	DT	17.66	32.71	15.18
KMSMOTE	GB	32.61	38.61	31.9
	LR	40.14	**39.14**	**38.14**
	DT	11.14	31.28	8.71

iv. **Oversampled 40% overall attacks performance metrics – scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
40% Oversampled				
ROS	GB	36.61	37.61	36.90
	LR	**40.14**	38.14	**38.14**

	DT	15.09	33.61	13.37
SMOTE	GB	38.66	38.61	37.90
	LR	**40.14**	38.14	**38.14**
	DT	15.14	22.18	15.66
BSMOTE	GB	36.61	37.61	36.90
	LR	**40.14**	38.14	**38.14**
	DT	17.14	21.66	16.14
ADASYN	GB	32.61	37.61	36.9
	LR	**40.14**	38.14	**38.14**
	DT	15.14	19.23	12.18
KMSMOTE	GB	36.63	**41.61**	31.90
	LR	**40.14**	39.14	**38.14**
	DT	39.14	39.18	37.66

v. Oversampled 50% overall attacks performance metrics – scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
50% Oversampled				
ROS	GB	37.61	38.61	37.90
	LR	40.14	38.14	38.14
	DT	6.61	28.23	5.18
SMOTE	GB	33.61	38.61	34.90
	LR	**40.18**	38.14	38.14
	DT	10.66	25.71	11.18
BSMOTE	GB	33.61	37.61	34.90
	LR	40.14	38.14	38.14
	DT	35.14	35.71	34.18
ADASYN	GB	37.14	37.61	36.90
	LR	40.14	38.14	38.14

	DT	13.80	11.18	10.76
KMSMOTE	GB	32.61	38.61	31.90
	LR	40.16	**39.14**	**39.14**
	DT	37.00	36.85	35.19

vi. Oversampled 60% overall attacks performance metrics – scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
60% Oversampled				
ROS	GB	38.61	38.61	37.90
	LR	**40.16**	38.14	**38.14**
	DT	29.33	32.8	28.80
SMOTE	GB	36.61	38.61	36.90
	LR	40.14	38.14	**38.14**
	DT	29.71	35.23	30.23
BSMOTE	GB	32.61	38.61	30.90
	LR	**40.16**	38.14	**38.14**
	DT	35.14	35.71	34.66
ADASYN	GB	32.14	37.61	30.90
	LR	**40.16**	38.14	**38.14**
	DT	11.66	31.14	11.14
KMSMOTE	GB	32.61	38.61	31.90
	LR	**40.16**	**39.14**	**38.14**
	DT	32.42	37.18	32.80

vii. **Oversampled 70% overall attacks performance metrics – scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
70% Oversampled				
ROS	GB	34.61	**38.61**	34.90
	LR	**40.14**	38.14	**38.14**
	DT	5.76	3.76	2.76
SMOTE	GB	34.61	38.61	35.90
	LR	**40.14**	38.14	**38.14**
	DT	5.71	12.23	5.71
BSMOTE	GB	32.61	37.61	30.90
	LR	**40.14**	37.61	**38.14**
	DT	8.66	3.71	5.18
ADASYN	GB	32.14	37.61	30.90
	LR	**40.14**	38.14	**38.14**
	DT	5.90	6.71	4.33
KMSMOTE	GB	32.61	**38.61**	31.90
	LR	**40.14**	38.14	**38.14**
	DT	11.71	22.23	10.23

viii. **Oversampled 80% overall attacks performance metrics – scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
80% Oversampled				
ROS	GB	34.61	**38.61**	34.90
	LR	**40.14**	38.14	**38.14**

	DT	14.99	26.61	11.33
SMOTE	GB	35.61	38.61	35.90
	LR	**40.14**	38.14	**38.14**
	DT	3.66	11.38	3.66
BSMOTE	GB	32.61	**38.61**	30.9
	LR	**40.14**	37.61	**38.14**
	DT	9.14	4.71	5.18
ADASYN	GB	32.14	37.61	30.90
	LR	**40.14**	37.61	**38.14**
	DT	18.14	30.66	18.66
KMSMOTE	GB	39.99	36.61	37.33
	LR	**40.14**	38.14	**38.14**
	DT	6.52	12.76	4.90

ix. Oversampled 90% overall attacks performance metrics – scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
90% Oversampled				
ROS	GB	33.61	**37.61**	34.90
	LR	**40.14**	38.14	**38.14**
	DT	6.61	0.82	4.66
SMOTE	GB	33.61	**38.61**	32.90
	LR	**40.14**	38.14	**38.14**
	DT	8.18	16.23	8.71
BSMOTE	GB	32.61	**38.61**	30.90
	LR	**40.14**	37.61	**38.14**
	DT	17.14	29.71	18.18
ADASYN	GB	32.19	36.61	30.42
	LR	**40.14**	37.61	**38.14**

	DT	6.71	22.71	7.71
KMSMOTE	GB	32.61	**38.61**	31.90
	LR	**40.14**	38.14	**38.14**
	DT	12.9	8.14	9.28

x. Oversampled 100% overall attacks performance metrics – scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
100% Oversampled				
ROS	GB	33.61	37.61	34.90
	LR	**40.14**	38.14	**38.14**
	DT	28.99	34.61	28.33
SMOTE	GB	33.61	**38.61**	32.90
	LR	**40.14**	38.14	**38.14**
	DT	8.80	20.66	8.23
BSMOTE	GB	32.14	37.61	30.90
	LR	**40.14**	37.61	**38.14**
	DT	24.28	24.66	29.71
ADASYN	GB	32.19	37.61	30.95
	LR	**40.14**	37.61	**38.14**
	DT	35.14	34.28	33.71
KMSMOTE	GB	0.39	2.66	0.28
	LR	**40.14**	38.14	**38.14**
	DT	8.85	8.66	6.76

APPENDIX BB: Overall attack performance metrics in non-scaled CICDDOS 2019.

i. Oversampled 10% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
10% Oversampled				
ROS	GB	3.77	4.55	3.77
	LR	25.55	16.26	14.43
	DT	42.01	41.08	**40.67**
SMOTE	GB	3.76	4.55	3.76
	LR	26.53	13.07	14.51
	DT	**43.41**	41.06	40.55
BSMOTE	GB	3.77	4.55	3.76
	LR	25.7	14.33	14.21
	DT	42.01	41.21	40.53
ADASYN	GB	3.79	6.17	3.79
	LR	25.22	14.66	14.59
	DT	42.00	**41.25**	40.59
KMSMOTE	GB	3.88	4.91	3.96
	LR	22.46	19.19	16.25
	DT	42.04	41.05	40.55

ii. Oversampled 20% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
20% Oversampled				

ROS	GB	3.72	6.41	3.73
	LR	22.71	15.58	14.3
	DT	41.98	40.74	**40.65**
SMOTE	GB	3.74	6.45	3.75
	LR	22.92	16.09	13.35
	DT	41.99	**41.08**	**40.65**
BSMOTE	GB	3.78	5.96	3.77
	LR	25.46	14.64	13.94
	DT	41.98	40.71	40.59
ADASYN	GB	3.79	6.17	3.79
	LR	25.86	16.15	14.24
	DT	41.98	40.95	40.62
KMSMOTE	GB	3.66	6.52	3.66
	LR	22.89	14.81	13.25
	DT	**42.02**	40.95	40.62

iii. Oversampled 30% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
30% Oversampled				
ROS	GB	3.71	4.60	3.70
	LR	22.74	13.66	14.29
	DT	41.68	40.79	**40.94**
SMOTE	GB	3.72	4.52	3.72
	LR	22.45	13.82	14.16
	DT	42.01	40.76	40.67
BSMOTE	GB	3.73	4.08	3.73
	LR	19.45	14.28	12.40
	DT	41.97	40.72	40.61

ADASYN	GB	3.77	6.02	3.78
	LR	22.63	18.12	15.06
	DT	**42.02**	**41.21**	40.57
KMSMOTE	GB	3.86	5.81	2.86
	LR	7.39	23.28	3.75
	DT	42.00	40.95	40.51

iv. Oversampled 40% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
40% Oversampled				
ROS	GB	3.69	5.92	3.7
	LR	22.81	14.43	13.59
	DT	41.32	40.77	40.94
SMOTE	GB	3.68	6.42	3.7
	LR	7.23	20.29	3.83
	DT	41.75	40.94	**41.04**
BSMOTE	GB	3.74	4.09	3.74
	LR	19.59	16.25	12.56
	DT	41.52	40.64	40.84
ADASYN	GB	3.77	6.17	3.77
	LR	22.52	17.5	13.75
	DT	41.35	**41.1**	40.56
KMSMOTE	GB	3.23	19.31	3.75
	LR	7.50	24.3	3.8
	DT	**42.01**	40.84	40.93

v. **Oversampled 50% overall attacks performance metrics –non-scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
50% Oversampled				
ROS	GB	3.66	4.28	3.67
	LR	4.33	20.98	2.53
	DT	40.34	40.47	40.40
SMOTE	GB	3.68	4.01	3.68
	LR	7.54	20.50	3.87
	DT	41.67	**40.86**	**41.00**
BSMOTE	GB	3.71	4.09	3.73
	LR	22.74	15.46	13.00
	DT	**41.68**	40.83	40.94
ADASYN	GB	3.77	6.7	3.79
	LR	19.37	15.74	12.72
	DT	41.48	40.68	40.84
KMSMOTE	GB	3.24	19.11	3.74
	LR	7.36	22.9	3.73
	DT	41.66	40.85	40.96

vi. **Oversampled 60% overall attacks performance metrics –non-scaled**

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
60% Oversampled				
ROS	GB	3.63	4.93	3.66
	LR	7.84	20.83	4.05

	DT	40.33	40.50	40.40
SMOTE	GB	3.60	6.58	3.65
	LR	4.05	21.41	2.87
	DT	41.45	40.79	40.97
BSMOTE	GB	3.68	4.07	3.71
	LR	3.9	16.16	2.24
	DT	41.44	40.63	40.81
ADASYN	GB	3.72	6.83	3.77
	LR	19.16	12.56	12.82
	DT	41.35	40.73	40.92
KMSMOTE	GB	3.18	18.97	3.69
	LR	7.76	24.11	3.75
	DT	**41.61**	**40.86**	**41.02**

vii. Oversampled 70% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
70% Oversampled				
ROS	GB	3.61	6.10	3.64
	LR	4.48	21.65	2.96
	DT	40.32	40.52	40.4
SMOTE	GB	3.62	6.56	3.67
	LR	4.12	21.55	2.92
	DT	**41.46**	40.82	**40.99**
BSMOTE	GB	3.66	4.12	3.72
	LR	4.30	20.38	2.91
	DT	41.44	40.77	40.93
ADASYN	GB	3.70	5.97	3.75
	LR	7.39	20.89	3.76

	DT	41.45	**40.83**	40.98
KMSMOTE	GB	3.16	18.81	3.68
	LR	22.69	20.48	17.01
	DT	40.99	40.76	40.83

viii. Oversampled 80% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
80% Oversampled				
ROS	GB	3.78	4.32	3.85
	LR	4.20	19.88	2.91
	DT	40.18	40.46	40.3
SMOTE	GB	3.77	4.28	3.84
	LR	4.61	18.32	2.6
	DT	41.4	40.77	**40.95**
BSMOTE	GB	3.62	4.11	3.69
	LR	3.93	21.35	2.81
	DT	41.32	40.74	40.93
ADASYN	GB	3.57	4.16	3.63
	LR	3.99	21.67	2.82
	DT	**41.43**	40.77	40.94
KMSMOTE	GB	3.14	18.77	3.66
	LR	7.54	20.2	3.61
	DT	41.07	**40.78**	40.87

ix. Oversampled 90% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
90% Oversampled				
ROS	GB	3.76	4.24	3.81
	LR	3.89	18.06	2.66
	DT	40.15	40.45	40.28
SMOTE	GB	3.79	4.31	3.86
	LR	4.11	17.53	2.67
	DT	**41.31**	**40.83**	**40.98**
BSMOTE	GB	3.61	4.12	3.69
	LR	4.44	21.27	2.55
	DT	40.71	40.63	40.65
ADASYN	GB	3.64	4.07	3.69
	LR	4.15	19.58	2.86
	DT	41.19	40.82	40.94
KMSMOTE	GB	3.17	19.00	3.73
	LR	7.16	23.92	3.72
	DT	40.70	40.54	40.61

x. Oversampled 100% overall attacks performance metrics –non-scaled

Method	Machine Learning	Recall	Precision	F-measure
Original				
GB		36.47	37.71	36.58
LR		38.72	38.76	37.24
DT		38.97	38.72	38.72
100% Oversampled				
ROS	GB	3.80	4.40	3.88
	LR	4.58	23.40	2.95

	DT	40.14	40.44	40.27
SMOTE	GB	3.78	4.37	3.87
	LR	4.03	20.48	2.81
	DT	40.59	40.60	40.58
BSMOTE	GB	3.52	4.13	3.6
	LR	4.58	18.12	2.84
	DT	40.56	40.57	40.55
ADASYN	GB	3.64	4.08	3.70
	LR	4.51	16.89	2.89
	DT	**41.26**	**40.69**	**40.87**
KMSMOTE	GB	3.16	19.07	3.74
	LR	4.40	20.27	2.79
	DT	40.78	40.68	40.71