Optimized Intrusion Detection System with Feature Extraction for effective Network Traffic Classification

Guided by:
Rajeev Azhuvath
Mentor
Tata Consultancy Services
&
Prof. Sumod Sundar
Centre for AI, TKMCE

Project Presentation
20.05.22

Presentation by:
Saranya T
TCS Intern ID: 2170618
M.Tech in AI
Centre for AI
TKMCE

Contents

- Introduction
- Problems under analysis
- Existing works
- Proposed techniques
- Dataset description
- Proposed system architecture
- Experiments and results
- Inference
- Limitation
- References

Introduction

- With increase in accessibility of internet it is possible to fetch and transfer data over multiple networks.
- **Cyber security:-** Security offered through online services to protect these information.
- From government to world's largest corporations and to a person as an individual cyber security plays a critical role.
- **Need of Cyber Security**

i. Spike in network related threats like viruses, eavesdropping and malicious attacks, etc.

ii. Helps in securing data from data theft, misuse, hacking.

- **A system which can properly monitor the network activity to detect threats that interrupt the security systems is necessary.**

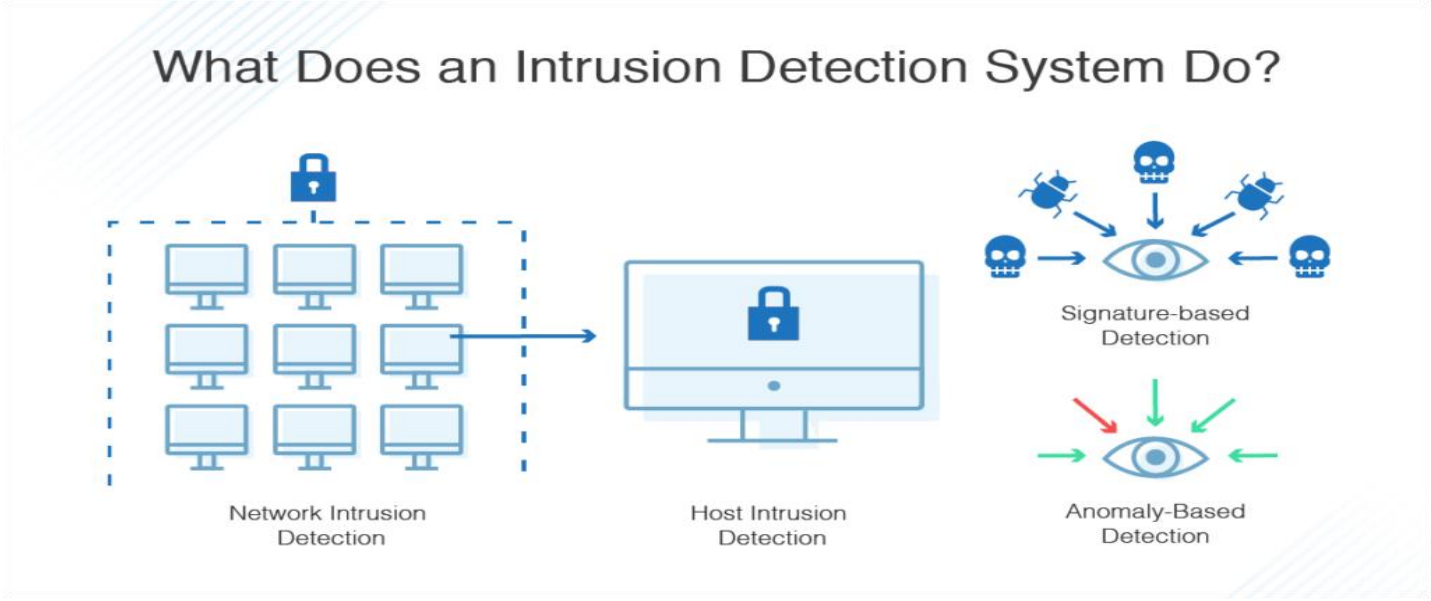
Intrusion Detection System (IDS)

- Monitor and study the pattern of network traffic under malicious activity.
- IDS types range in scope from single computers to large networks.
- Highest level IDS categorization:-

1. **Host based IDS (HIDS)**
2. **Network based IDS (NIDS)**

- Both HIDS and NIDS are classified as:-

1. **Signature-based IDS**:- Identify pre-defined patterns to flag an intrusion.
2. **Anomaly-based IDS**:- Scan the network and flags unusual behavior.



Challenges of IDS

- Traffic types in the network are increasing day by day.
- Network behavior characteristics are becoming increasingly complex.
- Disposed to false alarms:- IDS monitor networks for potentially malicious activity.
- Identify various malicious network traffics, especially unexpected malicious network traffics.

Fine tuning the IDS products is necessary i.e., proper setting up of IDS to recognize what normal traffic look like as compared to malicious activity

Problem under analysis

- An IDS can be established with a reliable feature selection technique.
- **Efficient detection of attacks requires identification of relevant features.**
- Features can be extracted using different machine learning and deep learning algorithms.
- Else, there is chance that some intrusion attacks are misclassified as normal, and some normal flows are misclassified as intrusions.
- Analyze **interpacket patterns in time domain**, as features in different packets are different.

Objective

Reduce the False Alarm Rate (FAR) and thereby improve the detection accuracy.

Existing works

Reference	Title	Technique	Advantage	Disadvantage
[1]	A novel feature extraction methodology using Siamese convolutional neural networks for intrusion detection	**Feature Selection**:- Siamese CNN **Classification** :- NB, AdaBoost, Random forest, KNN, Decision tree	1. Increased discrimination capability. 2. Potentially used as a risk indicator to identify cyber attacks.	Difficult to apply the model where decisions should be taken based on complex and heterogenous data.
[2]	A Deep Long Short-Term Memory based classifier for Wireless Intrusion Detection System	**Feature Selection**:- Information Gain (IG) **Classification**:- SVM, KNN, NB, RF, ANN, Feed Forward Dense Neural Network, DLSTM RNN	1. DLSTM RNN method outperforms ML methods and FFDNN. 2. Information gain discover non-linear relationship between variables in the dataset.	-

Existing works (cntd…)

Reference	Title	Technique	Advantage	Disadvantage
[3]	Adversarial machine learning in Network Intrusion Detection Systems	**Feature Selection:-** GA,PSO and GAN **Classification**:- SVM, Decision tree (DT), NB, KNN, Random Forest, MLP, Gradient Boosting, Linear Regression	Identified SVM as most vulnerable hence refrained from using them in NIDS.	Cannot analyze the internal operations of ML models and why some models are more robust than others.
[4]	Genetic convolutional neural network for intrusion detection systems	**Feature Selection:-**Genetic algorithm **Classification :-** CNN	1. High quality feature set. 2. Improved final detection performance. 3. Well-fitted in practical computer network environments.	1. Time consumption: implementation of GA and identify best fit CNN model. 2. Repetition of 5-fold cross-validation consumes more time and power.

Proposed techniques

❑ Feature extraction with XGBoost and Autoencoder

- XGBoost helps to enhance execution speed and improve model performance with parallel tree boosting approach and decision tree as the predictor.
- XGBoost is flexible because it supports classification, and ranking – Efficient for learning structured tabular data because of its sequential learning and combined prediction.
- Autoencoder provides learning with compressed data and helps to learn the non-linear relationship among the variables.

❑ Classification with TabNet

- The 2-stage feature selection helps to improve the use of only relevant attributes towards classification.

Dataset description

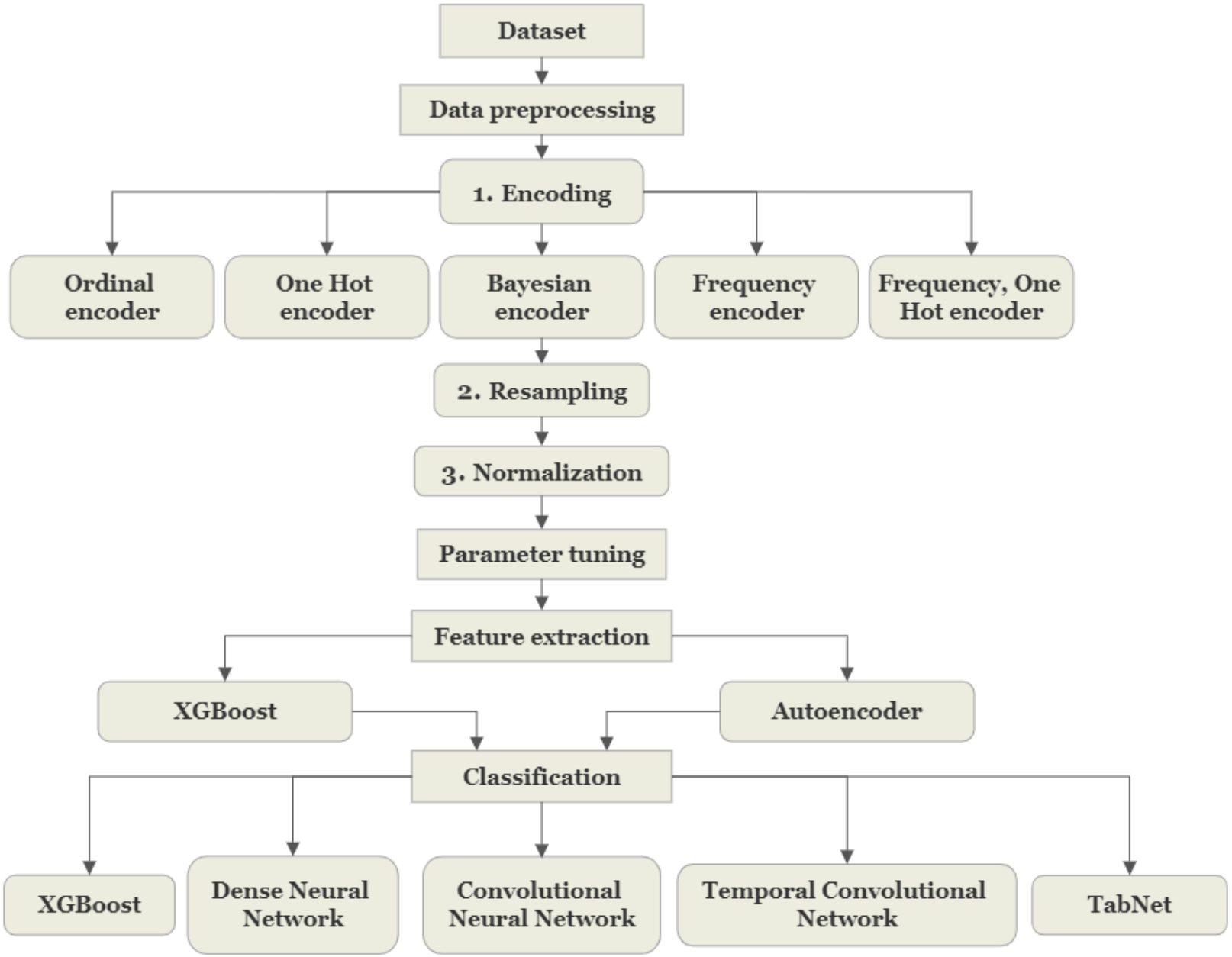
UNSW-NB15 dataset

- The number of samples in the training set is 175,341 and testing set is 82,332.
- Number of features:- 43
- This data set has 2 labels: 9 families of attacks all categorized as 'intrusive' and a 'normal' class.

NSL-KDD dataset

- The number of samples in the training set is 125,973 and testing set is 22,544.
- Number of features:- 41
- This data set has 2 labels: 39 families of attacks all categorized as 'intrusive' and a 'normal' class.

Proposed system architecture



1) Experiments on **Data Preprocessing**

- Import libraries and load both training and testing dataset.
- Create an extra column with name '**source**' to distinguish data as train and test for data concatenation.
- Concatenate both training and testing data.
- Identify the missing data, categorical data and count of categories.

a) <u>Encoding</u>

Encoded the original dataset with: -

1. **Ordinal Encoder**
2. **One Hot Encoder**
3. **Frequency Encoder**
4. **One Hot and Frequency Encoder**
5. **Bayesian Encoder**

Dataset	Category	Count of category
UNSW NB15	Proto	133
	Service	13
	State	11
NSLKDD	Protocol_type	3
	Service	70
	Flag	11

1) Experiments on Data Preprocessing

Experimental Results: - Encoded the original dataset with

1. Ordinal Encoder

UNSW NB15

Three categorical data ('service', 'state', 'proto') were ordinal encoded, to create additional 3 columns

state_code	proto_code	service_code
2.0	113.0	6.0
2.0	113.0	6.0
2.0	113.0	6.0
2.0	113.0	2.0
2.0	113.0	6.0
...	...	...
3.0	119.0	1.0
2.0	113.0	6.0
3.0	119.0	1.0
3.0	119.0	1.0
3.0	119.0	1.0

NSLKDD

Three categorical data ('service', 'flag', 'protocol_type') were ordinal encoded, to create additional 3 columns

protocol_type_code	service_code	flag_code
1.0	20.0	9.0
2.0	44.0	9.0
1.0	49.0	5.0
1.0	24.0	9.0
1.0	24.0	9.0
...	...	...
1.0	49.0	5.0
2.0	49.0	9.0
1.0	54.0	9.0

1) Experiments on Data Preprocessing

Experimental Results: - Encoded the original dataset with

2. One hot Encoder

UNSW NB15

Three categorical data ('service', 'state', 'proto') were one-hot encoded, to create additional 157 columns

state_ECO	state_FIN	state_INT	state_PAR	state_REQ	state_RST	state_URN	state_no
0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0
...	...	...	...	...	...	...	...
0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	0	0	0	0	0

NSLKDD

Three categorical data ('service', 'flag', 'protocol_type') were one-hot encoded, to create additional 84 columns

flag_RSTO	flag_RSTOS0	flag_RSTR	flag_S0	flag_S1	flag_S2	flag_S3	flag_SF
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0

1) Experiments on Data Preprocessing

Experimental Results: - Encoded the original dataset with

3. Frequency Encoder

UNSW NB15

Three categorical data ('service', 'state', 'proto') were frequency encoded, to create additional 3 columns.

proto_freq_encode	service_freq_encode	state_freq_encode
0.477508	0.548451	0.454700
0.477508	0.548451	0.454700
0.477508	0.548451	0.454700
0.477508	0.019327	0.454700
0.477508	0.548451	0.454700
...	...	...
0.359762	0.548451	0.451883
0.477508	0.548451	0.454700
0.014926	0.548451	0.451883
0.014926	0.548451	0.451883
0.359762	0.548451	0.451883

NSLKDD

Three categorical data ('service', 'flag', 'protocol_type') were frequency encoded, to create additional 3 columns

protocol_type_freq_encode	service_freq_encode	flag_freq_encode
0.818553	0.051920	0.604779
0.118599	0.034993	0.604779
0.818553	0.179286	0.248214
0.818553	0.324481	0.604779
0.818553	0.324481	0.604779
...	...	...
0.818553	0.055529	0.604779
0.818553	0.324481	0.604779
0.818553	0.324481	0.604779
0.118599	0.066908	0.604779
0.818553	0.003636	0.101557

1) Experiments on Data Preprocessing

Experimental Results: - Encoded the original dataset with

4. One hot and Frequency Encoder

UNSW NB15

Three categorical data ('service', 'state', 'proto') were encoded, to create additional 160 columns

state_ECO	state_FIN	state_INT	state_PAR
0	1	0	0
0	1	0	0
0	1	0	0
0	1	0	0
0	1	0	0
...	...	...	...
0	0	1	0
0	1	0	0
0	0	1	0
0	0	1	0
0	0	1	0

NSLKDD

Three categorical data ('service', 'flag', 'protocol_type') were encoded, to create additional 87 columns

flag_S2	flag_S3	flag_SF	flag_SH
0	0	1	0
0	0	1	0
0	0	0	0
0	0	1	0
0	0	1	0
...	...	...	...
0	0	1	0
0	0	1	0
0	0	1	0
0	0	1	0

1) Experiments on Data Preprocessing

Experimental Results: - Encoded the original dataset with

5. Bayesian Encoder

UNSW NB15

Three categorical data ('service', 'state', 'proto') were Bayesian encoded, to create additional 3 columns

proto	service	state
0.455718	0.547930	0.476401
0.455718	0.547930	0.476401
0.455718	0.547930	0.476401
0.455718	0.603213	0.476401
0.455718	0.547930	0.476401
...	...	...
0.762473	0.547930	0.912400
0.455718	0.547930	0.476401
0.000000	0.547930	0.912400
0.000000	0.547930	0.912400
0.762473	0.547930	0.912400

NSLKDD

Three categorical data ('service', 'flag', 'protocol_type') were Bayesian encoded, to create additional 3 columns

protocol_type	service	flag
0.494592	0.312151	0.187263
0.193255	0.486434	0.187263
0.494592	0.931123	0.990397
0.494592	0.071984	0.187263
0.494592	0.071984	0.187263
...	...	...
0.494592	0.072754	0.187263
0.494592	0.071984	0.187263
0.494592	0.071984	0.187263
0.193255	0.001107	0.187263

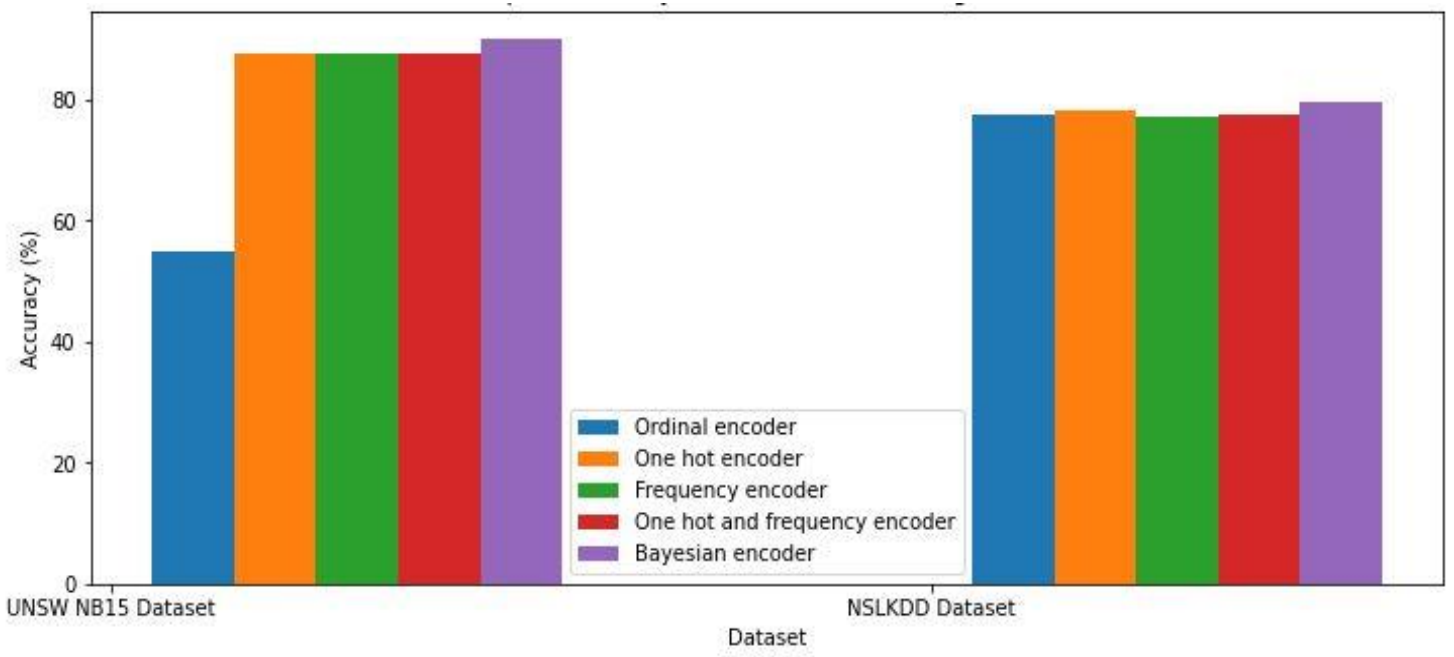
1) Experiments on Data Preprocessing

- Table shows the performance metric values obtained after encoding with five combinations.
- From this table, it is evident that the Bayesian encoder provides maximum accuracy.

Processing tool	Accuracy (%)		Precision		Recall		F1-Score	
	UNSW NB15 dataset	NSLKDD dataset	UNSW NB15 dataset	NSLKDD dataset	UNSW NB15 dataset	NSLKDD dataset	UNSW NB15 dataset	NSLKDD dataset
Ordinal Encoder	55.06	77.44	0.5721	0.8360	0.5506	0.7744	0.3551	0.7735
One Hot and Frequency encoder	87.59	77.59	0.8915	0.8371	0.8759	0.7759	0.8704	0.7750
One Hot Encoder	87.61	78.21	0.8914	0.8404	0.8761	0.7820	0.8706	0.7813
Frequency encoder	87.72	77.37	0.8928	0.8354	0.8772	0.7737	0.8717	0.7728
Bayesian encoder	**90.12**	**79.79**	**0.9071**	**0.8481**	**0.9012**	**0.7979**	**0.8984**	**0.7976**

1) Experiments on Data Preprocessing

- The Bayesian encoder provides good results because of its ability to study how a feature is dependent on the target data.
- Target or Bayesian encoding evaluates the mean value of a particular category based on its occurrence with the target.



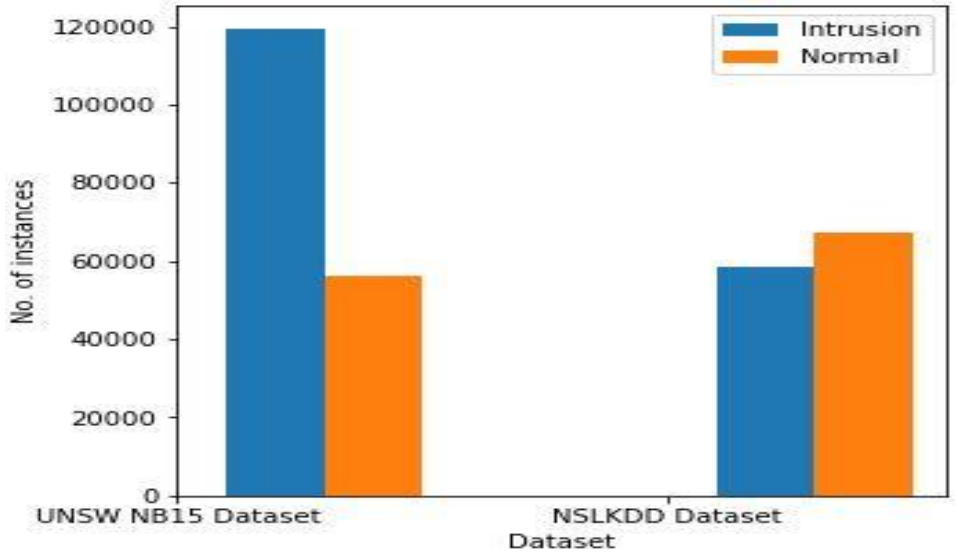
1) Experiments on **Data Preprocessing**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

b) <u>Resampling with SMOTE (Synthetic Minority Resampling TEchnique)</u>

- The imbalance in the data can lead to reduced detection accuracy.
- Oversampling the minority class is one way to deal with unbalanced datasets.
- Hence, to improve the detection rate SMOTE algorithm is used.
- It is data augmentation for the minority population.



1) Experiments on **Data Preprocessing**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

b) **Resampling with SMOTE (Synthetic Minority Resampling TEchnique)**

- The imbalance in the data can lead to reduced detection accuracy.
- Oversampling the minority class is one way to deal with unbalanced datasets.
- Hence, to improve the detection rate SMOTE algorithm is used.
- It is data augmentation for the minority population.

Result: - The number of training instances in the UNSW NB15 and NSLKDD datasets increased from 175,341 to 238,682 and 125,973 to 134,686, respectively with equal number of classes.

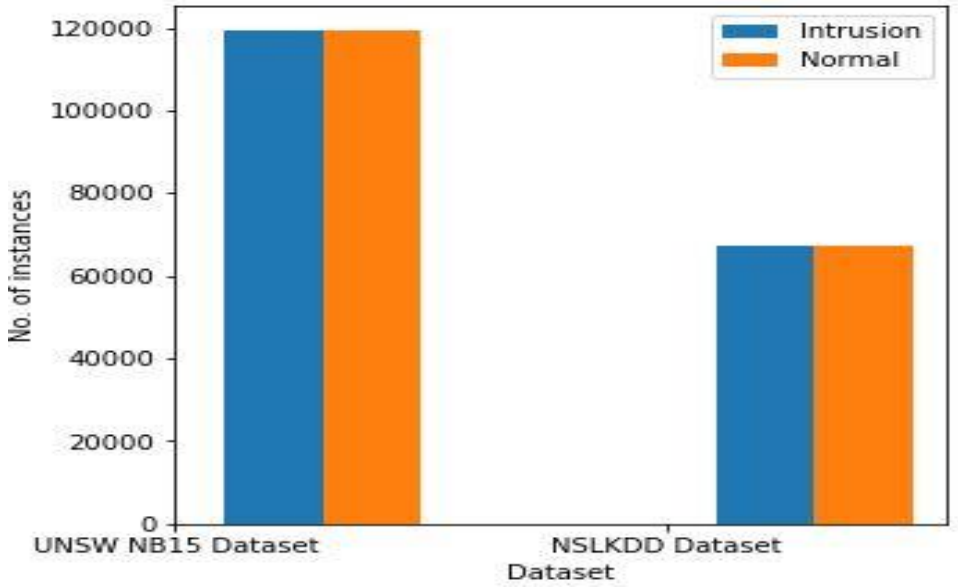
1) Experiments on **Data Preprocessing**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

b) <u>Resampling with SMOTE (Synthetic Minority Resampling TEchnique)</u>

- The imbalance in the data can lead to reduced detection accuracy.
- Oversampling the minority class is one way to deal with unbalanced datasets.
- Hence, to improve the detection rate SMOTE algorithm is used.
- It is data augmentation for the minority population.



1) Experiments on **Data Preprocessing**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

c) Normalization with Min-Max normalizer

- Both UNSW NB15 and NSLKDD dataset is normalized to improve the detection accuracy.
- Min-Max Scaler is used to normalize the data.
- The new range of normalization is fixed between 0 and 1.

$$\text{data\_norm} = \frac{(\text{data} - \text{data}_{\text{min}})(\text{range}_{\text{min}} - \text{range}_{\text{max}})}{\text{data}_{\text{max}} - \text{data}_{\text{min}}} + \text{range}_{\text{min}}$$

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

2) Experiments on **Parameter tuning**

- Parameters in XGBoost are broadly classified into 3: -

1. **Functioning / General parameters**
2. **Booster parameters**
3. **Optimization / Learning task parameters**

- XGBoost has some intrinsic characteristics, which makes it robust and efficient.
- But parameter tuning is done to identify the best parameters which will offer good detection rate.

Parameter	Value Obtained	
	UNSW NB15 Dataset	NSLKDD Dataset
n_estimators	1484	381
max_depth	4	5
min_child_weight	5	8
Gamma	0.4	0.0
Subsample	0.6	0.9
col_sample_bytree	0.6	0.6
Alpha	0.05	0.005

3) Experiments on **Feature extraction**

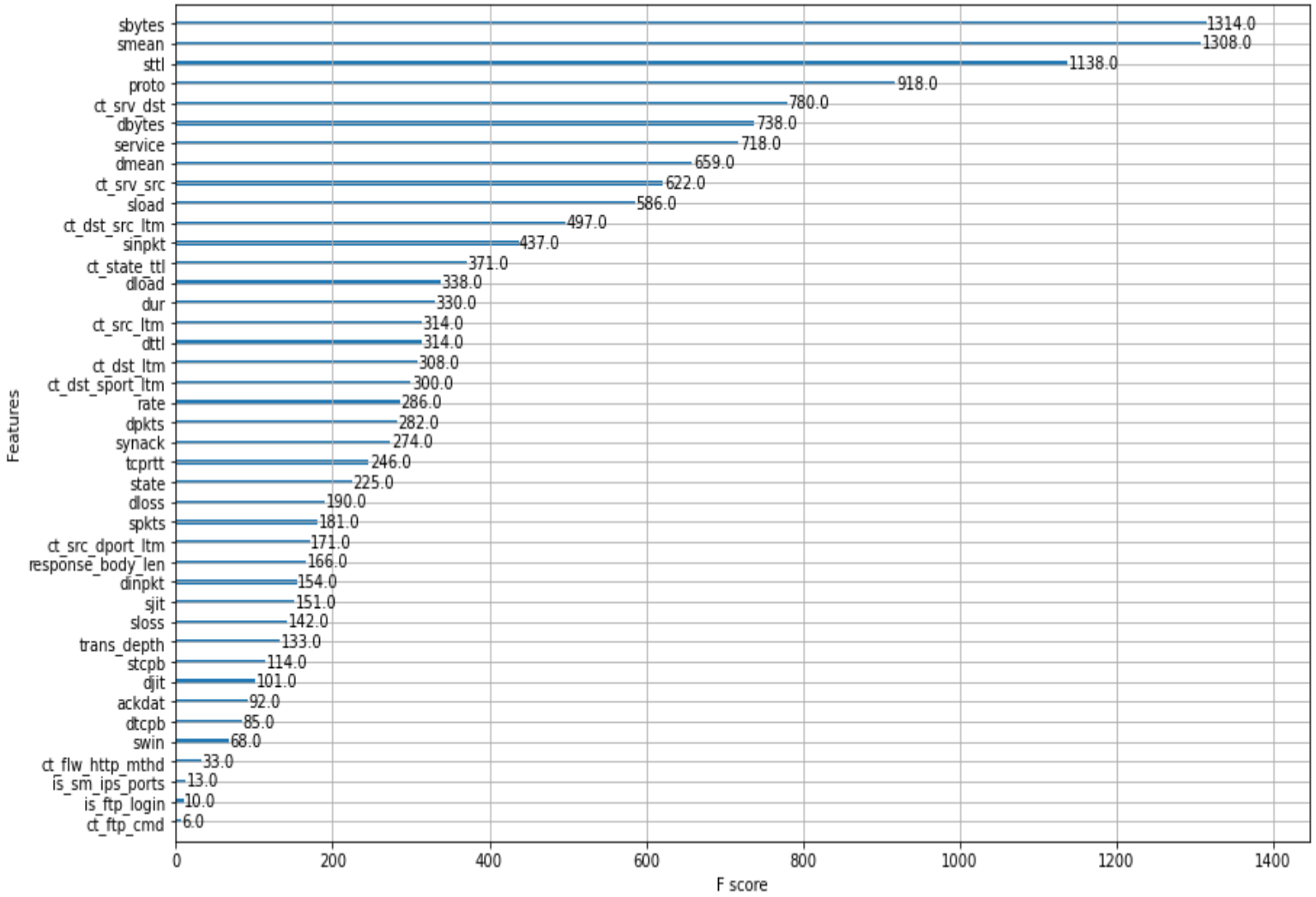
a) Feature selection using XGBoost

1. Feature importance graph

➢ Obtained the feature score graph and printed the f-score value with the plot_importance () and get_fscore () function.

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**



Feature importance graph in UNSW NB15 dataset

3) Experiments on **Feature extraction**

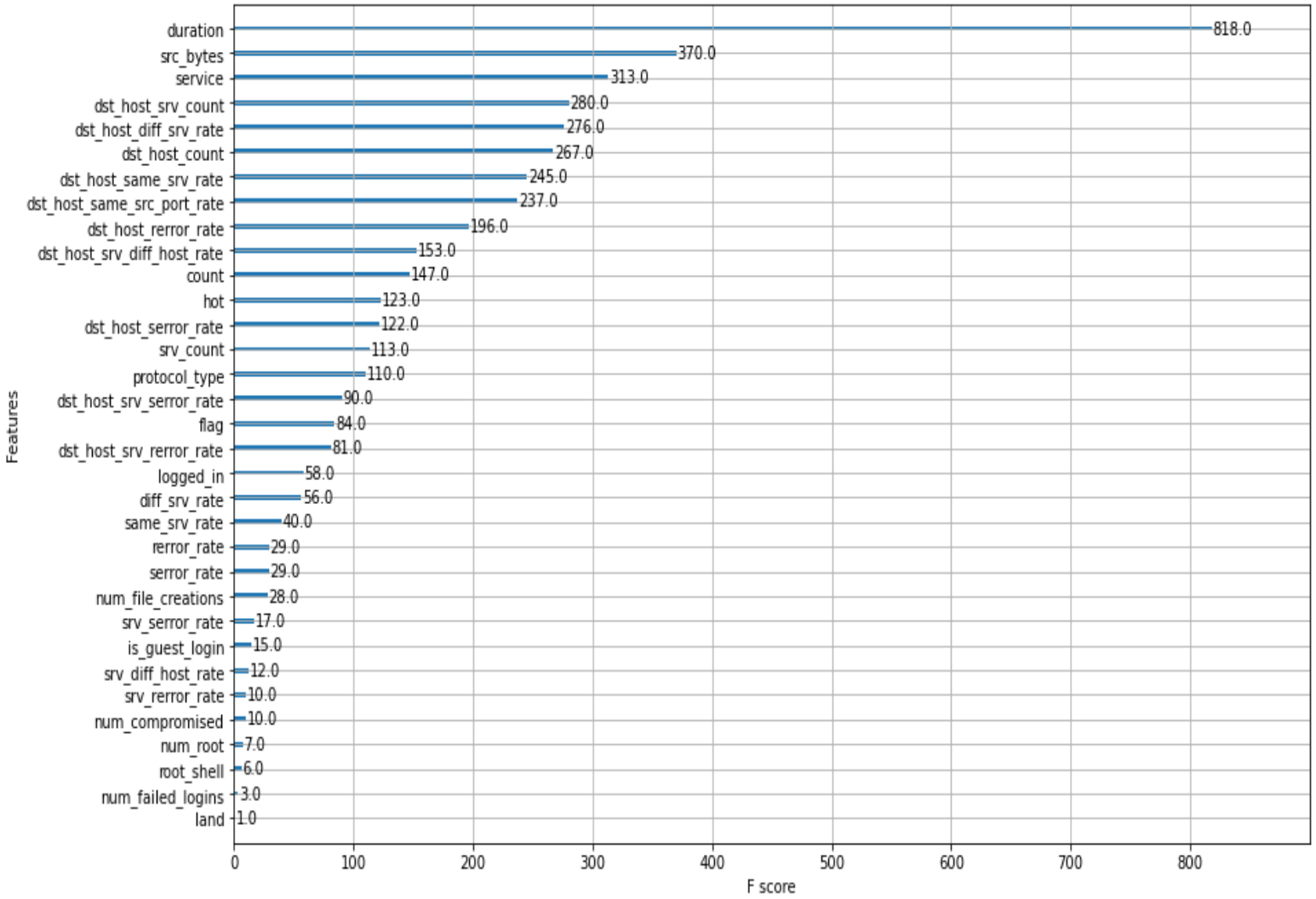
a) Feature selection using XGBoost

1. Feature importance graph

- Obtained the feature score graph and printed the f-score value with the plot_importance () and get_fscore () function.

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**



Feature importance graph in NSLKDD dataset

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

3) Experiments on **Feature extraction**

a) Feature extraction using XGBoost

2. Results: - Bayesian Encoder

➢ Highest accuracy of 90.53% with threshold = 0.016 and 15 selected features.

Threshold	Number of features	Accuracy (%)
0.000	42	90.20
0.001	40	90.22
0.002	38	90.20
0.003	33	90.39
0.004	26	90.48
0.004	24	90.34
0.004	20	90.18
0.016	**15**	**90.53**
0.016	11	86.91
0.016	6	81.03
0.016	2	80.57

Experimental result of XGBoost feature extraction with different thresholds in UNSW NB15 Dataset

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

3) Experiments on **Feature extraction**

a) Feature extraction using XGBoost

2. Results: - Bayesian Encoder

➢ Highest accuracy of 82.10% with threshold = 0.065 and 4 selected features.

Threshold	Number of features	Accuracy (%)
0.000	40	79.46
0.000	33	80.14
0.002	29	80.23
0.003	26	80.31
0.005	24	80.47
0.006	20	80.38
0.006	15	79.76
0.023	11	80.57
0.023	7	81.27
0.065	**4**	**82.10**
0.065	2	81.34

Experimental result of XGBoost feature extraction with different thresholds in NSLKDD Dataset

3) Experiments on **Feature extraction**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

b) Feature extraction using Autoencoder

- Input layers consist of units as the number of features in the UNSW NB15 and NSLKDD dataset.
- 2 encoder layers, each with a dense layer, batch normalization layer, and leaky ReLU layer.
- Created the bottleneck layer (dense layer).
- The decoder part has two layers: a dense layer, a batch normalization layer, and a leaky ReLU layer.
- Output layer with, optimizer = 'adam', loss='mse'.
- Trained the model and then saved the model with output as a bottleneck.
- Fit the saved autoencoder model data into the XGBoost classifier model and perform classification to identify the best feature extraction algorithm.

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

3) Experiments on **Feature extraction**

b) Feature extraction using Autoencoder

- Table shows Autoencoder and XGBoost feature selection model results in both datasets.
- Accuracy of 85.32% and 75.94% is obtained with Autoencoder feature extraction in UNSW NB15 and NSLKDD datasets, respectively.

Feature extraction	Accuracy (%)		Precision		Recall		F1-Score	
	UNSW NB15 dataset	NSLKDD dataset	UNSW NB15 dataset	NSLKDD dataset	UNSW NB15 dataset	NSLKDD dataset	UNSW NB15 dataset	NSLKDD dataset
Autoencoder	85.32	75.94	0.8620	0.8079	0.8532	0.7594	0.8477	0.7589
XGBoost	90.53	82.10	0.9116	0.8568	0.9053	0.8210	0.9025	0.8209

4) Experiments on **Classification**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- TabNet is a tabular deep learning model.
- Each decision step employs sequential attention to select a subset of relevant features and process them locally and globally.
- Hence, even though XGBoost provides good feature extraction results, in classification, TabNet outperforms.

Classifier	Accuracy (%)	Precision	Recall	F1 – Score	ROC AUC Score
XGBoost	90.53	0.9116	0.9053	0.9025	0.8975
DNN	78.75	0.8879	0.7875	0.7643	0.7636
CNN	80.94	0.8897	0.8094	0.7917	0.7880
TCN	89.75	0.9108	0.8975	0.8941	0.8886
TabNet	**93.02**	**0.9309**	**0.9302**	**0.9293**	**0.9279**

XGBoost feature extraction in UNSW NB15 dataset (Dimensionality-15)

4) Experiments on **Classification**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- TabNet is a tabular deep learning model.
- Each decision step employs sequential attention to select a subset of relevant features and process them locally and globally.
- Hence, even though XGBoost provides good feature extraction results, in classification, TabNet outperforms.

Classifier	Accuracy (%)	Precision	Recall	F1 – Score	ROC AUC Score
XGBoost	82.10	0.8568	0.8210	0.8209	0.8382
DNN	77.72	0.8064	0.7772	0.7771	0.7920
CNN	83.40	0.8405	0.8340	0.8332	0.8405
TCN	76.77	0.8011	0.7678	0.7675	0.7837
TabNet	**88.35**	**0.8845**	**0.8835**	**0.8807**	**0.8790**

XGBoost feature extraction in NSLKDD dataset (Dimensionality-4)

4) Experiments on **Classification**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- TabNet is a tabular deep learning model.
- Each decision step employs sequential attention to select a subset of relevant features and process them locally and globally.
- Hence, even though XGBoost provides good feature extraction results, in classification, TabNet outperforms.

Classifier	Accuracy (%)	Precision	Recall	F1 – Score	ROC AUC Score
XGBoost	85.32	0.8620	0.8532	0.8478	0.8427
DNN	81.03	0.8902	0.8103	0.7928	0.7890
CNN	85.10	0.8744	0.8510	0.8447	0.8392
TCN	89.06	0.8958	0.8906	0.8882	0.8848
TabNet	**89.25**	**0.8926**	**0.8925**	**0.8913**	**0.8909**

Autoencoder feature extraction in UNSW NB15 dataset (Dimensionality-14)

4) Experiments on **Classification**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- TabNet is a tabular deep learning model.
- Each decision step employs sequential attention to select a subset of relevant features and process them locally and globally.
- Hence, even though XGBoost provides good feature extraction results, in classification, TabNet outperforms.

Classifier	Accuracy (%)	Precision	Recall	F1 – Score	ROC AUC Score
XGBoost	75.94	0.8079	0.7594	0.7589	0.7802
DNN	71.77	0.8074	0.7177	0.7153	0.7447
CNN	78.67	0.8437	0.7867	0.7863	0.8083
TCN	77.44	0.8141	0.7744	0.7743	0.7920
TabNet	**80.84**	**0.8341**	**0.8084**	**0.8083**	**0.8225**

Autoencoder feature extraction in NSLKDD dataset (Dimensionality-13)

4) Experiments on **Classification**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- TabNet is a tabular deep learning model.
- Each decision step employs sequential attention to select a subset of relevant features and process them locally and globally.
- Hence, even though XGBoost provides good feature extraction results, in classification, TabNet outperforms.

Inference

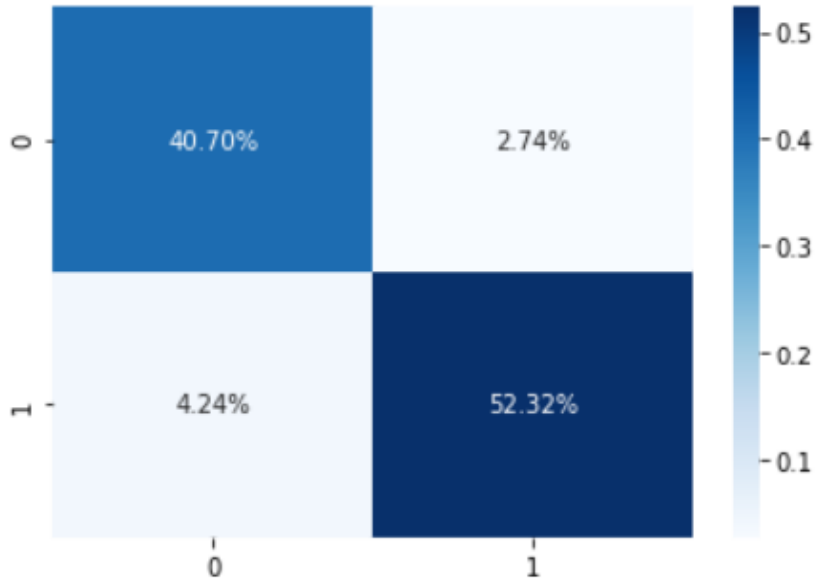
The detection rate with the proposed model XGBoost feature extraction and TabNet classification technique yields a maximum accuracy of 93.02% and 88.35% compared to all the models in the UNSW NB15 and NSLKDD dataset.

4) Experiments on **Classification**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- The obtained confusion matrix represents that the true positive rate, i.e., the rate of classifying intrusion as the intrusion, is high.
- The false positives and false negatives are fewer. Hence, the model improves detection accuracy by reducing the false alarm rate.



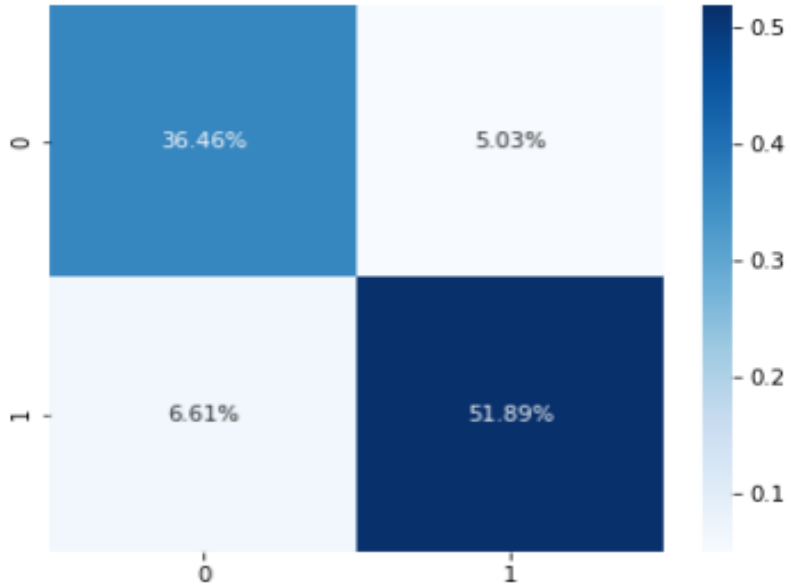
Confusion matrix of the proposed model in UNSW NB15 dataset

4) Experiments on **Classification**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- The obtained confusion matrix represents that the true positive rate, i.e., the rate of classifying intrusion as the intrusion, is high.
- The false positives and false negatives are fewer. Hence, the model improves detection accuracy by reducing the false alarm rate.



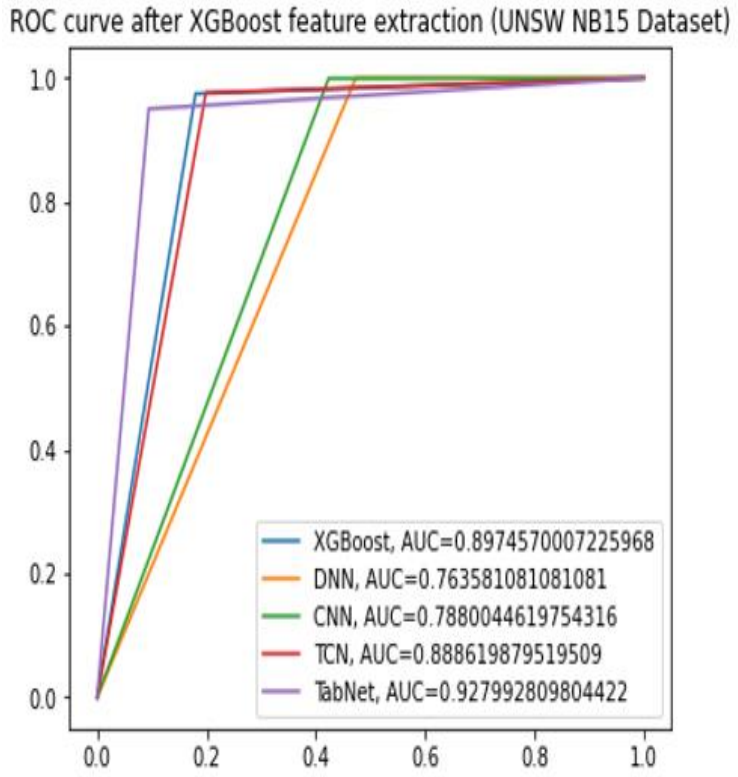
Confusion matrix of the proposed model in NSLKDD dataset

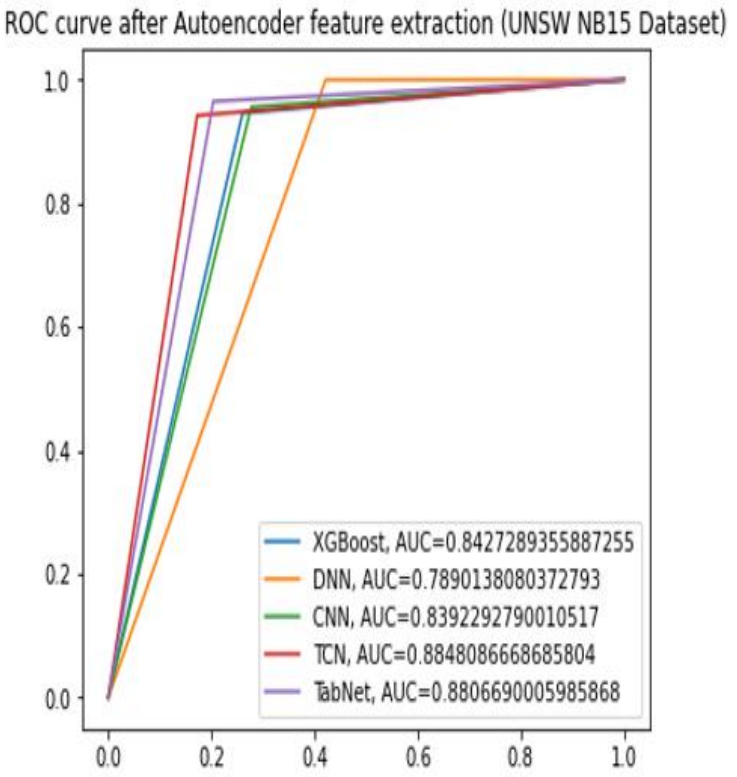
4) Experiments on **Classification**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- ROC Curves: - UNSW NB15 dataset



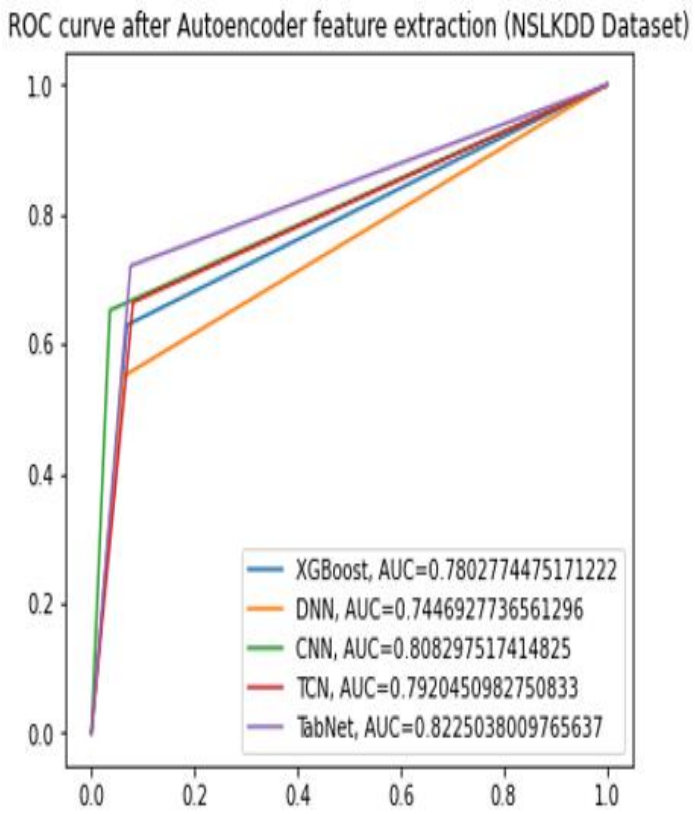


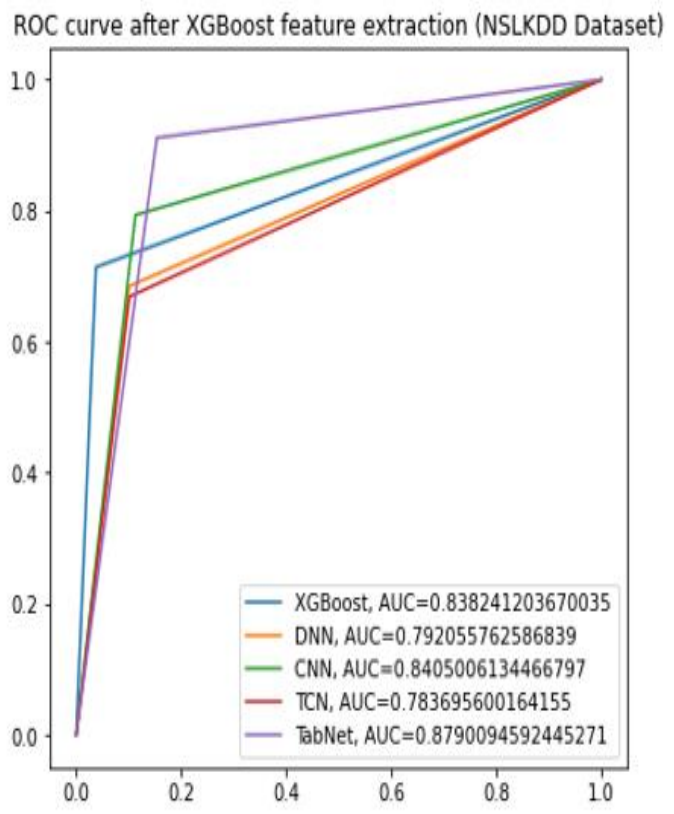
4) Experiments on **Classification**

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- ROC Curves: - NSLKDD dataset



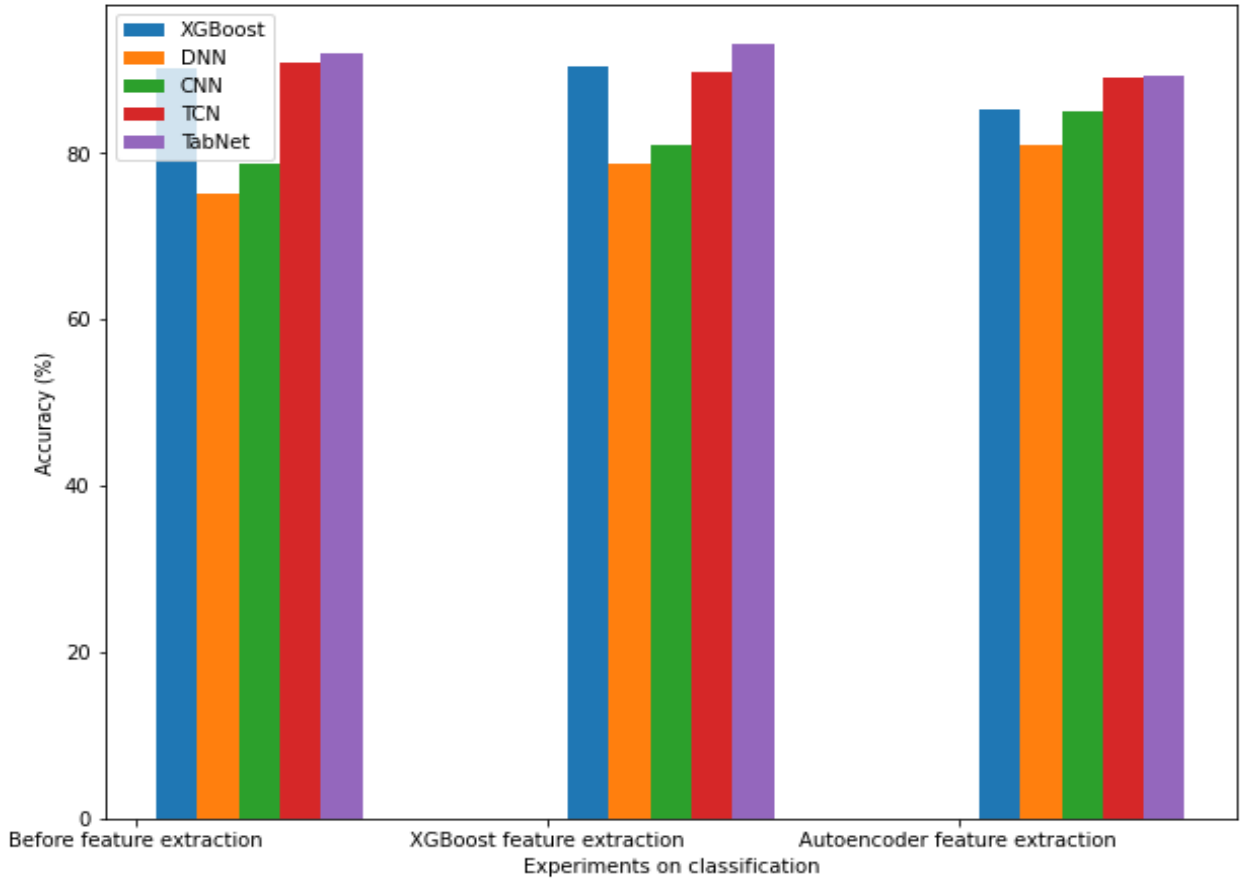


5) Comparative analysis

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- UNSW NB15 dataset

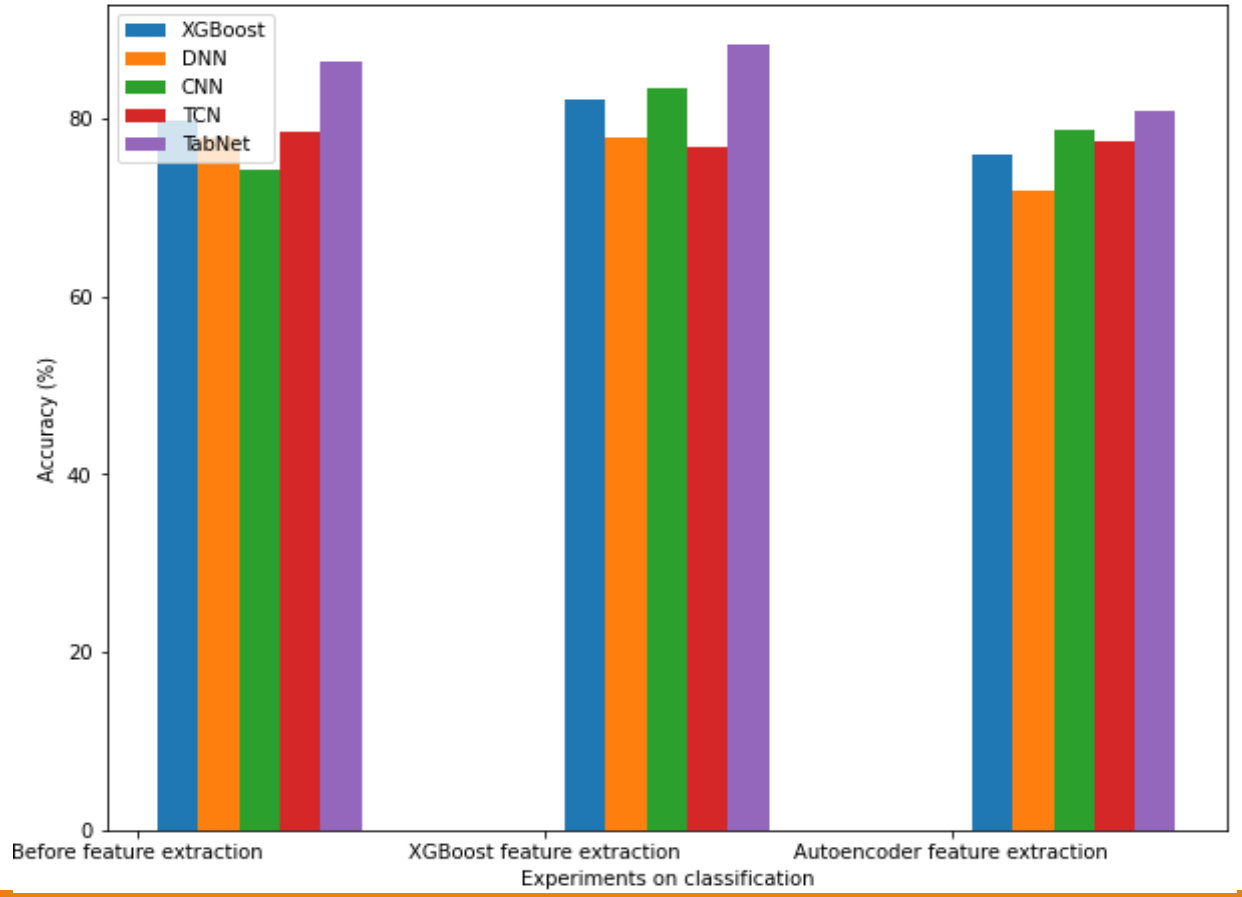


5) Comparative analysis

Encoded the original dataset with: -

1. One Hot Encoder
2. Frequency Encoder
3. One Hot Encoder and Frequency Encoder
4. Ordinal Encoder
5. **Bayesian Encoder**

- NSLKDD dataset



Inference

Experiments on Data-Preprocessing

- Best encoder: - Bayesian encoder
- Bayesian encoder could accurately map the relation between a categorical variable and a target variable which was not possible with other encoders.

Experiments on feature extraction

- Best feature extraction: - XGBoost (Dimensionality-15)
- In structured tabular data, XGBoost can minimize the error and thereby consider relevant features while passing through each subsequent tree sequentially.

Experiments on classification

- Best classifier: - TabNet
- TabNet models 2-stage feature transformer and mask ensures, only relevant features are passed to the subsequent blocks.
- Promising results from TCN also prove the relevance of temporal features in the data.

Limitation

- Even though XGBoost outperforms in feature extraction, it takes more time in extracting features corresponding to each threshold.
- In Autoencoder, training the encoder, decoder, and saving the model with output as ‘bottleneck’ gives the compressed features.
- TCN provided good performance results, but it was the only model which provided nearly same accuracy before and after feature extraction. This regards to a smaller number of temporal features after feature extraction. Hence to extract time domain features specifically, some other feature extraction techniques need to be used.

References

1. Moustakidis S, Karlsson P. A novel feature extraction methodology using Siamese convolutional neural networks for intrusion detection. Cybersecurity. 2020 Dec;3(1):1-3.
2. Kasongo SM, Sun Y. A deep long short-term memory-based classifier for wireless intrusion detection system. ICT Express. 2020 Jun 1;6(2):98-103.
3. Alhajjar E, Maxwell P, Bastian N. Adversarial machine learning in network intrusion detection systems. Expert Systems with Applications. 2021 Dec 30;186:115782.
4. Nguyen MT, Kim K. Genetic convolutional neural network for intrusion detection systems. Future Generation Computer Systems. 2020 Dec 1;113:418-27.
5. Devan P, Khare N. An efficient XGBoost–DNN-based classification model for network intrusion detection system. Neural Computing and Applications. 2020 Jan 19:1-6.
6. Su T, Sun H, Zhu J, Wang S, Li Y. BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset. IEEE Access. 2020 Feb 10;8:29575-85.

THANK YOU!