Cyber Security - NSL-KDD

30-September-2019 | Version 1.0

—

Dai Yirui, Dong Meirong, Lim Chong Seng Hermann, Yam Gui Peng David

Prepared for NUS-ISS Master of Technology in Intelligent Systems Pattern Recognition CA1

Table of Contents

1.0 Executive Summary

For this assignment, the team decided to work on the NSL-KDD dataset (network security related) as it is widely documented with a large number of samples that mimic real scenarios. With the sizable dataset, the team was able to apply different algorithms to analyze their effectiveness in classifying the data. We utilized techniques for feature selection such as Principal Component Analysis to prepare the data, and then applied classification algorithms such as Decision Tree, Random Forest, K-Nearest Neighbors, Isolation Forest. Through this project, the team learned how to practically prepare and partition data, and also on how to build ensemble/hybrid models.

2.0 Introduction into Dataset

The team uses NSL-KDD dataset in this assignment to construct data models to classify and evaluate network anomalies. NSL-KDD dataset is a dataset derived from KDD-99 dataset which includes a wide variety of intrusions simulated in a military network environment. The dataset is chosen because it is one of the only few labeled data available for the public to study network-based intrusion patterns and is said to resolve some of the inherent problems of KDD-99 dataset to improve the performance of evaluated systems.

These are the inherent problems resolved from KDD-99 to NSL-KDD:

- NSL-KDD removes a huge number of duplicated data in KDD-99 training and test sets to improve data quality. This prevents learning algorithms to be biased towards the more frequent records and puts the learning of infrequent records (such as User-to-Root attacks) at a disadvantage.

- the number of selected records from each difficult level group is inversely proportional to the percentage of records in the KDD-99 data set to enable easier comparisons of evaluation results.

- the number of records in NSL-KDD is reasonable which allows the team to run experiments on complete set with decent computing power and time without the need to select a random portion. As a result, evaluation results of different experiments will be consistent and comparable.

3.0 Description of Dataset

3.1 By Categories

There are 125,973 records in the training set (in KDDTrain+.txt) and 22,544 records in test set (in KDDTest+.txt). Each of the record contains 41 features and is labeled as either "normal" or "attack". All labeled attacks fall in one of the following four categories:

1. Denial of Service Attack (DOS): DOS exhausts computing resources to prevent victimized system from performing legitimate activities.

2. User to Root Attack (U2R): U2R begins by gaining control of normal user accounts via means like dictionary attack or social engineering. Then the normal user accounts will be used to exploit some vulnerability to get root access to the system.

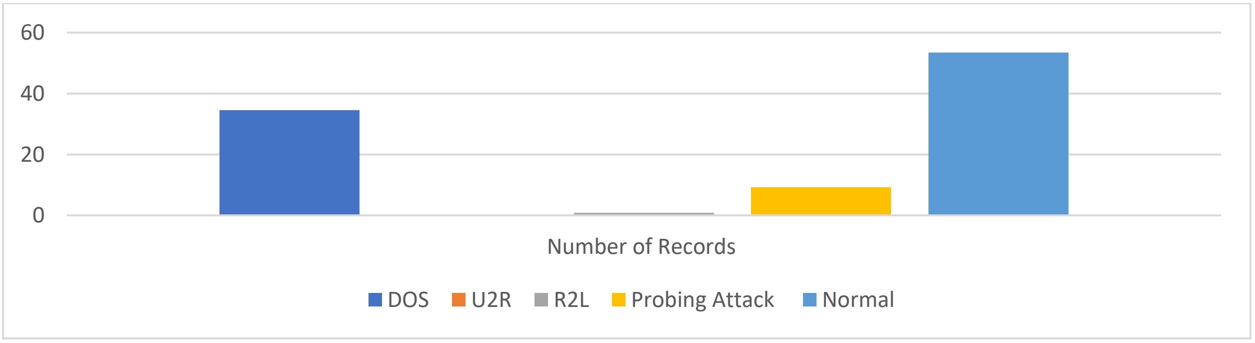
3. Remote to Local Attack (R2L): R2L is employed to exploit system vulnerability to gain unauthorized local access as a user of the environment.

4. Probing Attack: Probing Attack is an attempt to gather information about a network of computers for the purpose of circumventing its security controls.

The table below summarizes the training datasets by their categories.

Attack	Types (in KDDTrain+.txt)	Percentage	Summary
DOS	'neptune', 'back', 'smurf', 'pod', 'land', and 'teardrop'	36.46%	46.54%
U2R	'buffer_overflow', 'loadmodule', 'rootkit' and 'perl'	0.04%	
R2L	'warezclient', ' multihop', ' ftp_write', 'imap', 'guess_passwd', 'warezmaster', 'spy' and 'phf'	0.79%	
Probing Attack	'portsweep', 'satan', 'nmap', and 'ipsweep'	9.25%	
Normal	'normal'	53.46%	53.46%

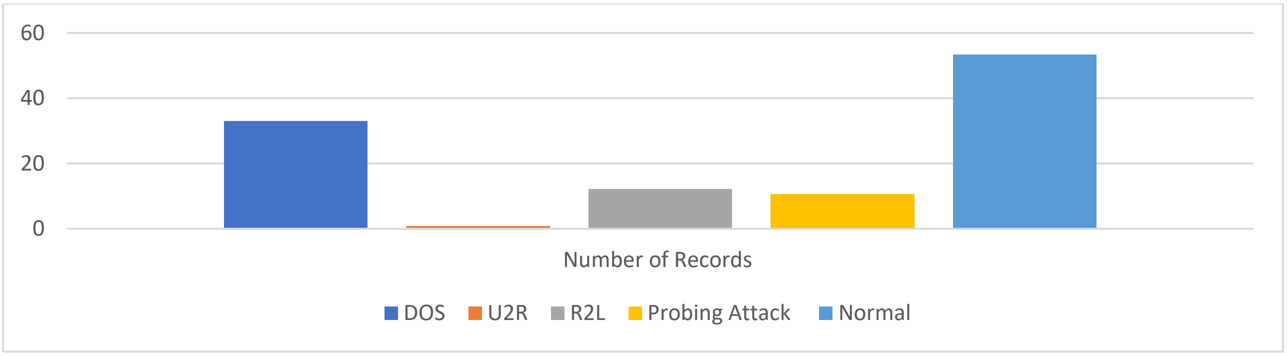
The graph represents training data by categories.



The table below summarizes the test dataset by their categories.

Attack	Types (in KDDTest+.txt)	Percentage	Summary
DOS	'neptune', 'back', 'smurf', 'pod', 'land', and 'teardrop'	33.08%	56.93%
U2R	'buffer_overflow', 'loadmodule', 'rootkit' and 'perl'	0.89%	
R2L	'warezclient', ' multihop', ' ftp_write', 'imap', 'guess_passwd', 'warezmaster', 'spy' and 'phf'	12.22%	
Probing Attack	'portsweep', 'satan', 'nmap', and 'ipsweep'	10.74%	
Normal	‘normal’	43.08 %	43.08%

The graph represents training data by categories.



3.2 By Protocols

On top of that, NSL-KDD dataset is analyzed by network protocols to examine attack patterns with more details. The protocols included in this dataset are Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Internet Control Message Protocol (ICMP).

1. TCP is a reliable connection-oriented protocol which implies that data sent from one side is sure to reach the destination in the same order.

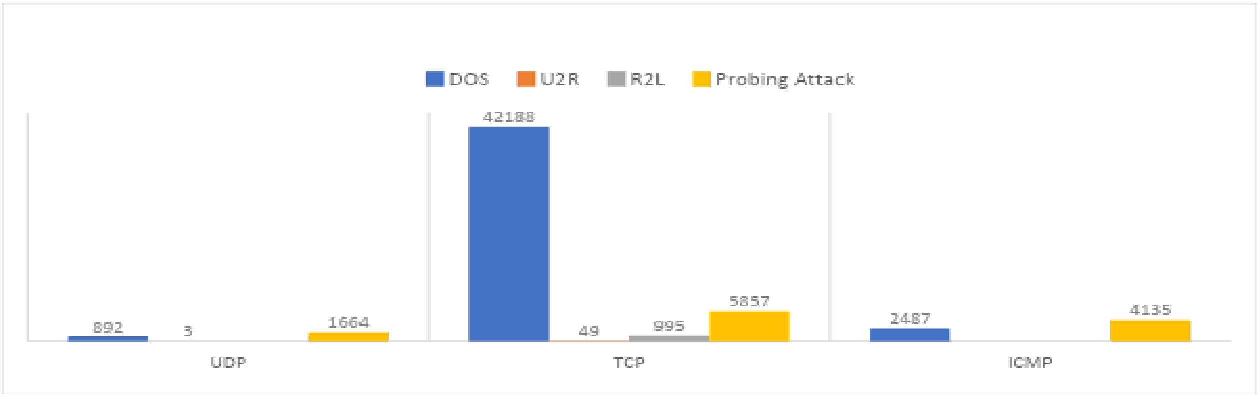
2. UDP is similar in behavior to TCP except that it is unreliable and connectionless protocol which is used primarily for establishing low-latency and loss-tolerating connections between applications on the internet.

3. ICMP is used by network devices, including routers, to send error messages and operational information indicating, for example, that a requested service is not available or that a host or router could not be reached.

The table below summarizes the training datasets by their network protocols.

Protocol_Type	Attack_name	Count
UDP	normal	14,993
	teardrop, satan, nmap, rootkit	
TCP	normal	102,689
	neptune, guess_passwd, land, portsweep, buffer_overflow, phf, warezmaster, ipsweep, multihop, warezclient, perl, back, ftp_write, loadmodule, satan, spy, imap, rootkit	
ICMP	normal	8,291
	portsweep, ipsweep, smurf, satan, pod, nmap	

The graph represents protocols used by various attacks.



3.3 Observations

Based on the tables and graphs above, if the team simply differentiates the datasets by “normal” or “attack”, there will be enough records to train each category.

However, if the team zooms in multiclass classification, the categories “normal “and “DOS” will have noticeably more records than U2R and R2L in training sets and the protocol TCP has significantly more records than UDP and ICMP. And if the team trains a multiclass classification model without fixing the imbalanced datasets, the model will be completely biased. Hence, the training sets have to be pre-processed before training to avoid overfit or underfit.

The team will elaborate more on the issues highlighted above in sections below.

4.0 Data Processing

4.1 One Hot Encoding

There are 3 types of data in NSL-KDD dataset which are nominal, numeric and binary.

Nominal (symbolic data)	protocol_type, service,flag
Numeric (continuous data)	duration,src_bytes,dst_bytes,wrong_fragment,urgent,hot,num_failed_logins,num_compromised,num_root,num_file_creations,num_shells,num_access_files,num_outbound_cmds,count,srv_count,serror_rate,srv_serror_rate,rerror_rate,srv_rerror_rate,same_srv_rate,diff_srv_rate,srv_diff_host_rate,dst_host_count,dst_host_srv_count,dst_host_same_srv_rate,dst_host_diff_srv_rate,dst_host_same_src_port_rate,dst_host_srv_diff_host_rate,dst_host_serror_rate,dst_host_srv_serror_rate,dst_host_rerror_rate,dst_host_srv_rerror_rate
Binary (symbolic data)	land, logged_in,root_shell,su_attempted,is_host_login,is_guest_login

The team converts nominal data into a series of zeros and ones via one-hot encoding for machine learning algorithms to do a better job in prediction.

After the conversion of nominal data to binary data, the training set has 122 features data while the test has 116 features. The team compared all features from both datasets and appends those missing features to respective datasets with zeros assigned for machine learning algorithms to perform proper prediction.

4.2 Normalization

The numeric data in NSL-KDD dataset has a wide range of values. When we do further analysis like multilayer perceptron (mlp), the attributes with wide range features will intrinsically influence the result more due to its larger value. But this does not necessarily mean it is more important as a predictor.

Therefore, the team applies normalization in this assignment to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values.

For the variables that always get positive numbers and that show much more variation with higher values (heteroscedasticity), a log-normal distribution (i.e., normal after log-transformation) is applied to better describe the data. Because log-transforming to these variables makes the distributions more normally distributed, stabilizes the variances. However, log-transforming also makes the models multiplicative on the raw scale instead of additive. (the composition of the data are skewed towards one side, log-transformation is applied to make the data more balanced).

Log-transformation Features	duration, src_bytes, dst_bytes, hot, num_failed_logins, num_compromised, num_root, num_file_creations, num_shells, num_access_files

The team uses standard scaler in this assignment:

```
Normlised_tr_values = (tr_values - mean(tr_values))/standard_deviation(tr_values)
```

The test data accuracy generally improves after normalization.

Model (without hyperparameter tuning)	Accuracy	
	Before Normalization	After Normalization
Decision Tree	0.79	0.78
Multilayer Perceptron (mlp)	0.72	0.81
K-Nearest Neighbors (KNN)	0.77	0.78
Support Vector Machine (SVM)	0.71	0.79

**Tree-based decision algorithms like decision trees or random forests seek for the best split point in each feature. The split point is determined by the percentage of labels correctly classified using a feature, which is resilient to feature scaling. Therefore, standardization doesn't have any significant impact on this type of ML models.

4.3 Feature Selection and Elimination

The dimensionality of a data set is the number of attributes or features that the data possesses. High dimensional data with a staggeringly high number of dimensions may lead to these problems.

- increase the computational complexity of calculations and storage space for machine
- the number of samples required to generalize accurately grows exponentially
- model overfitting
- slower learning cycles
- poor prediction performance

The team tries these dimensionality reduction techniques to address the curse of dimensionality.

- **Variance Threshold**

The technique removes features with variation below a certain cutoff with the assumption that when a feature does not vary much within itself, it generally has very little predictive power.

There is one problem with the assumption, because some features might be extremely powerful in explaining the target variable, although they have very low variance. In this assignment, for example, after one hot encoding, the features flag_S1" and "flag_S0" used to predict if a connection is normal have very low variance but very high importance.

Nevertheless, the team still explored this technique in this assignment for learning purpose. After the team set f cut-off to 90%, only 31 features remain.

- **F-Test**

The technique does a hypothesis testing model X and Y where X is a model created by just a constant and Y is the model created by a constant and a feature. Then, the least square errors in both models are compared and checks if the difference in errors between model X and Y are significant or introduced by chance to select K important features.

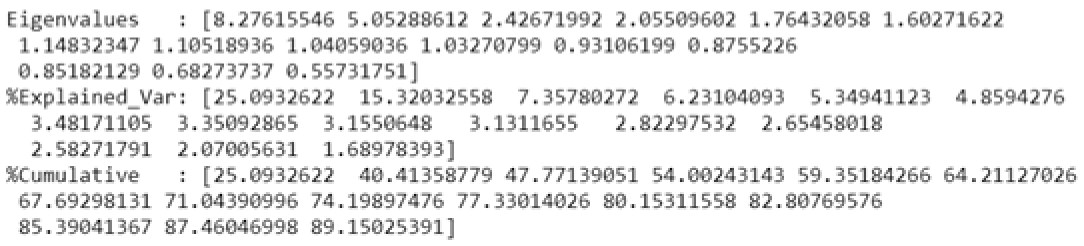
- **Recursive feature elimination (RFE)**

This technique eliminates worst performing features on a particular model one after the other until the best subset of features is known.

- **Principal Component Analysis (PCA)**

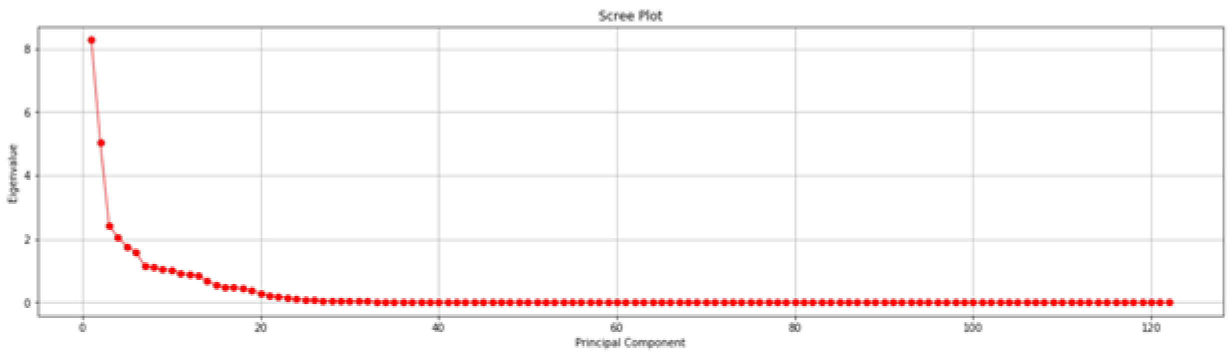
This technique is used for feature extraction and is based on linear correlations. It examines the correlations among the features and uses this information to construct the appropriate principal components.

The screenshot shows eigenvalues, explained variances and cumulative variances of 15 principal components:



```
Eigenvalues   : [8.27615546 5.05288612 2.42671992 2.05509602 1.76432058 1.60271622
 1.14832347 1.10518936 1.04059036 1.03270799 0.93106199 0.8755226
 0.85182129 0.68273737 0.55731751]
%Explained_Var: [25.0932622  15.32032558  7.35780272  6.23104093  5.34941123  4.8594276
  3.48171105  3.35092865  3.1550648   3.1311655   2.82297532  2.65458018
  2.58271791  2.07005631  1.68978393]
%Cumulative   : [25.0932622  40.41358779 47.77139051 54.00243143 59.35184266 64.21127026
 67.69298131 71.04390996 74.19897476 77.33014026 80.15311558 82.80769576
 85.39041367 87.46046998 89.15025391]
```

For this dataset, 10 components should be shortlisted for consideration based on eigenvalues because Kaiser rule calls for 1.0 cut-off. The first 10 components explains 77% of the variability with 23% loss of information.



The result tallies with the scree plot for the dataset which has “elbows" at around 10 principal components and eigenvalue cut-off at around 1.04.

Therefore, the team set PCA(n_components=10) in this assignment.

- **Observations**

The table provides a simple illustration on how the models will perform before and after dimensionality reduction.

Model (without hyperparameter tuning)	Dimensionality Reduction Accuracy			
	Variance Threshold (threshold=.90)	F-Test (percentile=10)	rfe (n_features_ to_select=15)	PCA (n_components=10)
Decision Tree	Same	Higher (insignificantly)	Higher (significantly)	Lower (insignificantly)
Multilayer Perceptron (mlp)	Lower (insignificantly)	Lower (insignificantly)	-	Lower (insignificantly)
K-Nearest Neighbors (KNN)	Lower (insignificantly)	Same	-	Lower (insignificantly)
Support Vector Machine (SVM)	Higher (insignificantly)	Higher (insignificantly)	-	Lower (insignificantly)

(Note: The team obtained these results by training the models with different sets of fixed parameters before and after dimensionality reduction with no hyperparameter tuning. Hence, these results cannot conclude the overall performance of the models on the datasets with reduced dimensionality.)

These are the observations derived from the table.

- Decision tree performs significantly better after rfe (n_features_ to_select=15) is performed. This could be because the reduction technique uses model accuracy to identify which attributes (and combination of attributes) contribute the most to predicting the target attribute and effectively reduces overfitting.

- Dimensionality reduction techniques do not have significant impacts on MLP, KNN and SVM.

- The team understands that the primary goal of these techniques is to reduce the dimensionality of the data where the effect can be a reduction in computation, reduced sparsity, and in general a dataset easier to handle and these techniques do not necessarily improve the model accuracy.

- The team will only use the table above as one of the bases to select reduction technique performs to shorten computing power and reduce overfitting if required.

4.3 Imbalanced Dataset

From the trial training results, the team notices that “normal-DOS” classifications generally performs better than “normal - attack” classifications. Therefore, the team decides to train the models based on “normal-DOS” classifications.

However, as mentioned above, the dataset is imbalanced because some categories have significantly more records than other categories (e.g. “normal” as compared to “R2L”).Machine learning algorithms have trouble learning when one class dominates the other, because the algorithms could easily classify the data under the majority group and yet achieve high accuracy.

There are 4 ways of addressing class imbalance problems like these:

- Synthesis of new minority class instances
- Oversampling of minority class
- Under-sampling of majority class
- Tweak the cost function to make misclassification of minority instances more important than misclassification of majority instances

The team chose oversampling SMOTE (Synthetic Minority Over-sampling Technique) to generate synthetic data by randomly creating samples of the attributes based on the observations in the minority classes.

The point to note is after is sampling is used, the correlation in dataset will increase. Hence, it is always recommended to do oversampling before dimensionality reduction.

5.0 Binary Classification

The team uses these machine learning algorithms and ensemble learnings methods in this assignment for binary classification of the connections into normal vs DOS attack, normal vs U2R attack, normal vs R2L attack and normal vs Probing attack.

Machine Learning Algorithms

- Decision Tree Classifier (DTC)
- KNeighbors Classifier (KNN)
- Support Vector Machine (SVM)
- Isolation Forest (ISF), only for R2L and U2R

Ensemble Algorithms

- Voting Classifier (VTC)
- Random Forest Classifier (RFC)
- XGBoost Classifier (XGB)

Binary Classification Results - Summary of ROC Curve	**Model**				**Ensemble**		
	DTC	**KNN**	**SVM**	**ISF**	**VTC**	**RFC**	**XGB**
Normal vs DOS Attack	**93.99%**	**91.56%**	**95.24%**	**-**	**98.10%**	**97.67%**	**96.85%**
Normal vs U2R Attack	**62.75%**	**56.18%**	**87.17%**	**94.95%**	**90.86%**	**84.40%**	**90.46%**
Normal vs R2L Attack	**71.90%**	**61.24%**	**67.69%**	**85.51%**	**81.36%**	**77.19%**	**72.41%**
Normal vs Probing Attack	**95.08%**	**86.64%**	**97.02%**	**-**	**97.64%**	**98.01%**	**95.58%**

** ROC curve: Receiver Operating Characteristic curve

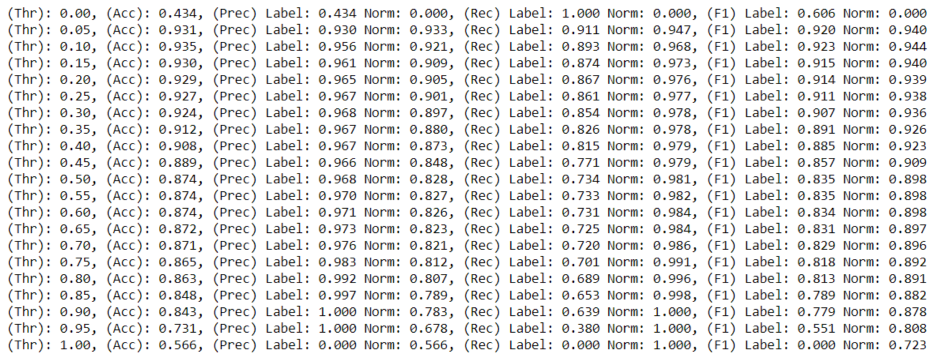
Based on the training result, the team observes that:

- Normal vs DOS Classification has best performance when modeled using Voting Classifier (of 3 Random Forest Classifiers).
- Normal vs U2R Classification has best performance when modeled using Isolation Forest Classifier.
- Normal vs R2L Classification has best performance when modeled using Isolation Forest Classifier.
- Normal vs Probing Attack Classification has best performance when modeled using Random Forest Classifier.
- Ensemble methods generally have better performance as compared to machine learning algorithms like DTC, KNN and SVM. This is because are ensemble methods are meta-algorithms that combine several machine learning techniques into one predictive model in order to decrease variance/bias or to improve predictions.
- However, ISF did better for U2R and R2L likely due to the large mismatch in sample sizes between the positive and negative classes.

Based on the observations, the team then trained the best-performing algorithm with grid search method to fine tune hyperparameters to evaluate the accuracy on the test set. The approach is taken to reduce the computational complexity of calculations and storage space for machine and shorten the training time.

- **Normal - DOS Classification with Voting Classifier (of 3 Random Forest Classifiers)**

F1- Score Report



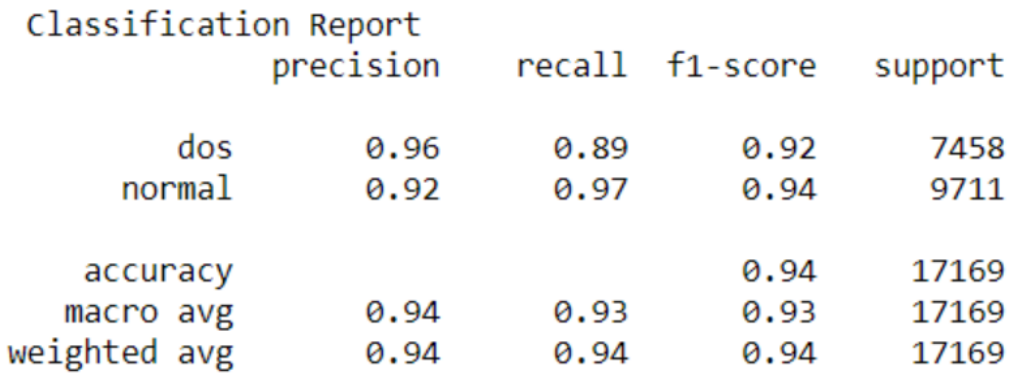
```
(Thr): 0.00, (Acc): 0.434, (Prec) Label: 0.434 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.606 Norm: 0.000
(Thr): 0.05, (Acc): 0.931, (Prec) Label: 0.930 Norm: 0.933, (Rec) Label: 0.911 Norm: 0.947, (F1) Label: 0.920 Norm: 0.940
(Thr): 0.10, (Acc): 0.935, (Prec) Label: 0.956 Norm: 0.921, (Rec) Label: 0.893 Norm: 0.968, (F1) Label: 0.923 Norm: 0.944
(Thr): 0.15, (Acc): 0.930, (Prec) Label: 0.961 Norm: 0.909, (Rec) Label: 0.874 Norm: 0.973, (F1) Label: 0.915 Norm: 0.940
(Thr): 0.20, (Acc): 0.929, (Prec) Label: 0.965 Norm: 0.905, (Rec) Label: 0.867 Norm: 0.976, (F1) Label: 0.914 Norm: 0.939
(Thr): 0.25, (Acc): 0.927, (Prec) Label: 0.967 Norm: 0.901, (Rec) Label: 0.861 Norm: 0.977, (F1) Label: 0.911 Norm: 0.938
(Thr): 0.30, (Acc): 0.924, (Prec) Label: 0.968 Norm: 0.897, (Rec) Label: 0.854 Norm: 0.978, (F1) Label: 0.907 Norm: 0.936
(Thr): 0.35, (Acc): 0.912, (Prec) Label: 0.967 Norm: 0.880, (Rec) Label: 0.826 Norm: 0.978, (F1) Label: 0.891 Norm: 0.926
(Thr): 0.40, (Acc): 0.908, (Prec) Label: 0.967 Norm: 0.873, (Rec) Label: 0.815 Norm: 0.979, (F1) Label: 0.885 Norm: 0.923
(Thr): 0.45, (Acc): 0.889, (Prec) Label: 0.966 Norm: 0.848, (Rec) Label: 0.771 Norm: 0.979, (F1) Label: 0.857 Norm: 0.909
(Thr): 0.50, (Acc): 0.874, (Prec) Label: 0.968 Norm: 0.828, (Rec) Label: 0.734 Norm: 0.981, (F1) Label: 0.835 Norm: 0.898
(Thr): 0.55, (Acc): 0.874, (Prec) Label: 0.970 Norm: 0.827, (Rec) Label: 0.733 Norm: 0.982, (F1) Label: 0.835 Norm: 0.898
(Thr): 0.60, (Acc): 0.874, (Prec) Label: 0.971 Norm: 0.826, (Rec) Label: 0.731 Norm: 0.984, (F1) Label: 0.834 Norm: 0.898
(Thr): 0.65, (Acc): 0.872, (Prec) Label: 0.973 Norm: 0.823, (Rec) Label: 0.725 Norm: 0.984, (F1) Label: 0.831 Norm: 0.897
(Thr): 0.70, (Acc): 0.871, (Prec) Label: 0.976 Norm: 0.821, (Rec) Label: 0.720 Norm: 0.986, (F1) Label: 0.829 Norm: 0.896
(Thr): 0.75, (Acc): 0.865, (Prec) Label: 0.983 Norm: 0.812, (Rec) Label: 0.701 Norm: 0.991, (F1) Label: 0.818 Norm: 0.892
(Thr): 0.80, (Acc): 0.863, (Prec) Label: 0.992 Norm: 0.807, (Rec) Label: 0.689 Norm: 0.996, (F1) Label: 0.813 Norm: 0.891
(Thr): 0.85, (Acc): 0.848, (Prec) Label: 0.997 Norm: 0.789, (Rec) Label: 0.653 Norm: 0.998, (F1) Label: 0.789 Norm: 0.882
(Thr): 0.90, (Acc): 0.843, (Prec) Label: 1.000 Norm: 0.783, (Rec) Label: 0.639 Norm: 1.000, (F1) Label: 0.779 Norm: 0.878
(Thr): 0.95, (Acc): 0.731, (Prec) Label: 1.000 Norm: 0.678, (Rec) Label: 0.380 Norm: 1.000, (F1) Label: 0.551 Norm: 0.808
(Thr): 1.00, (Acc): 0.566, (Prec) Label: 0.000 Norm: 0.566, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.723
```

Based on the finding, a threshold value of 0.1 will give the highest F1-score. The accuracy on test data achieved using the threshold is 93.35%.

Confusion Matrix

	Predicted Normal	Predicted Attack	Total
Actual Normal	9402	309	9711
Actual DOS	801	6657	7458
Total	10203	6966	17169

Classification Report



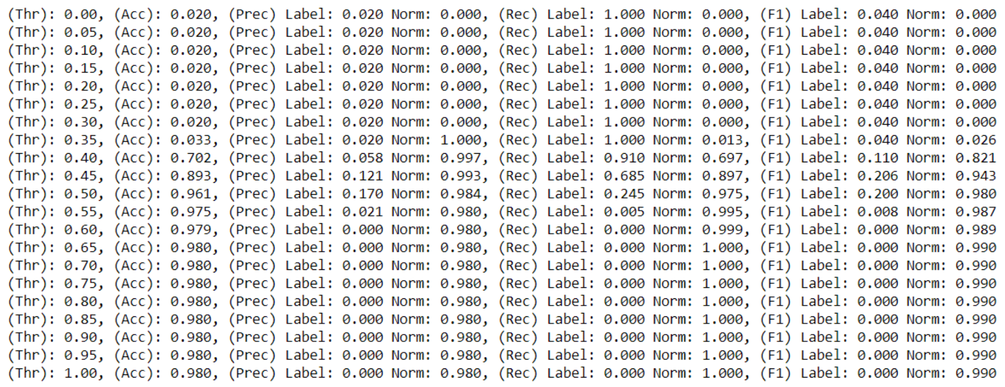
```
 Classification Report
              precision    recall  f1-score   support

         dos       0.96      0.89      0.92      7458
      normal       0.92      0.97      0.94      9711

    accuracy                           0.94     17169
   macro avg       0.94      0.93      0.93     17169
weighted avg       0.94      0.94      0.94     17169
```

Although the accuracy is high in prediction, the team also notices that the accuracy is mainly contributed by true negative rate. This is not ideal. It means the team has limited insights about the model's ability to predict true attacks.

Another point to note is false negative rate is not considered low. In this assignment, because it is security related, high false negative will cause more losses to an entity than false positive rate. Therefore, one of the improvements the team can make is to lower the false positive rate.

- **Normal - U2R Classification with Isolation Forest Classifier**

F1-Score Report



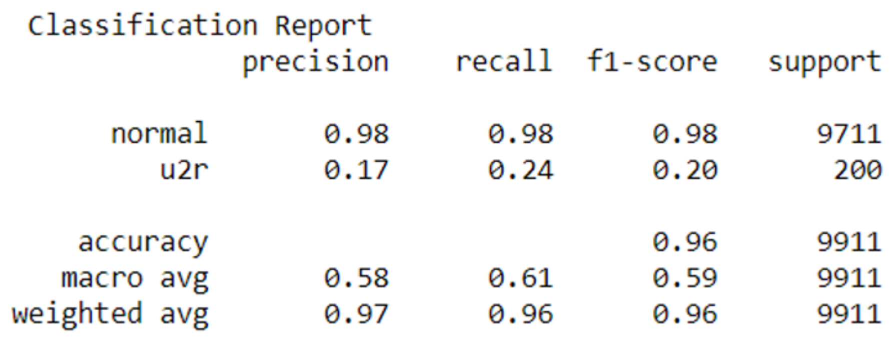
```
(Thr): 0.00, (Acc): 0.020, (Prec) Label: 0.020 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.040 Norm: 0.000
(Thr): 0.05, (Acc): 0.020, (Prec) Label: 0.020 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.040 Norm: 0.000
(Thr): 0.10, (Acc): 0.020, (Prec) Label: 0.020 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.040 Norm: 0.000
(Thr): 0.15, (Acc): 0.020, (Prec) Label: 0.020 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.040 Norm: 0.000
(Thr): 0.20, (Acc): 0.020, (Prec) Label: 0.020 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.040 Norm: 0.000
(Thr): 0.25, (Acc): 0.020, (Prec) Label: 0.020 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.040 Norm: 0.000
(Thr): 0.30, (Acc): 0.020, (Prec) Label: 0.020 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.040 Norm: 0.000
(Thr): 0.35, (Acc): 0.033, (Prec) Label: 0.020 Norm: 1.000, (Rec) Label: 1.000 Norm: 0.013, (F1) Label: 0.040 Norm: 0.026
(Thr): 0.40, (Acc): 0.702, (Prec) Label: 0.058 Norm: 0.997, (Rec) Label: 0.910 Norm: 0.697, (F1) Label: 0.110 Norm: 0.821
(Thr): 0.45, (Acc): 0.893, (Prec) Label: 0.121 Norm: 0.993, (Rec) Label: 0.685 Norm: 0.897, (F1) Label: 0.206 Norm: 0.943
(Thr): 0.50, (Acc): 0.961, (Prec) Label: 0.170 Norm: 0.984, (Rec) Label: 0.245 Norm: 0.975, (F1) Label: 0.200 Norm: 0.980
(Thr): 0.55, (Acc): 0.975, (Prec) Label: 0.021 Norm: 0.980, (Rec) Label: 0.005 Norm: 0.995, (F1) Label: 0.008 Norm: 0.987
(Thr): 0.60, (Acc): 0.979, (Prec) Label: 0.000 Norm: 0.980, (Rec) Label: 0.000 Norm: 0.999, (F1) Label: 0.000 Norm: 0.989
(Thr): 0.65, (Acc): 0.980, (Prec) Label: 0.000 Norm: 0.980, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.990
(Thr): 0.70, (Acc): 0.980, (Prec) Label: 0.000 Norm: 0.980, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.990
(Thr): 0.75, (Acc): 0.980, (Prec) Label: 0.000 Norm: 0.980, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.990
(Thr): 0.80, (Acc): 0.980, (Prec) Label: 0.000 Norm: 0.980, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.990
(Thr): 0.85, (Acc): 0.980, (Prec) Label: 0.000 Norm: 0.980, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.990
(Thr): 0.90, (Acc): 0.980, (Prec) Label: 0.000 Norm: 0.980, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.990
(Thr): 0.95, (Acc): 0.980, (Prec) Label: 0.000 Norm: 0.980, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.990
(Thr): 1.00, (Acc): 0.980, (Prec) Label: 0.000 Norm: 0.980, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.990
```

A threshold value of 0.5 was chosen given a relatively high recall and highest F1-score for u2r. The accuracy on test data achieved using the threshold is 96.05%.

Confusion Matrix

	Predicted Normal	Predicted Attack	Total
Actual Normal	9471	240	9711
Actual DOS	151	49	200
Total	9622	289	9911

Classification Report



```
Classification Report
              precision    recall  f1-score   support

      normal       0.98      0.98      0.98      9711
         u2r       0.17      0.24      0.20       200

    accuracy                           0.96      9911
   macro avg       0.58      0.61      0.59      9911
weighted avg       0.97      0.96      0.96      9911
```

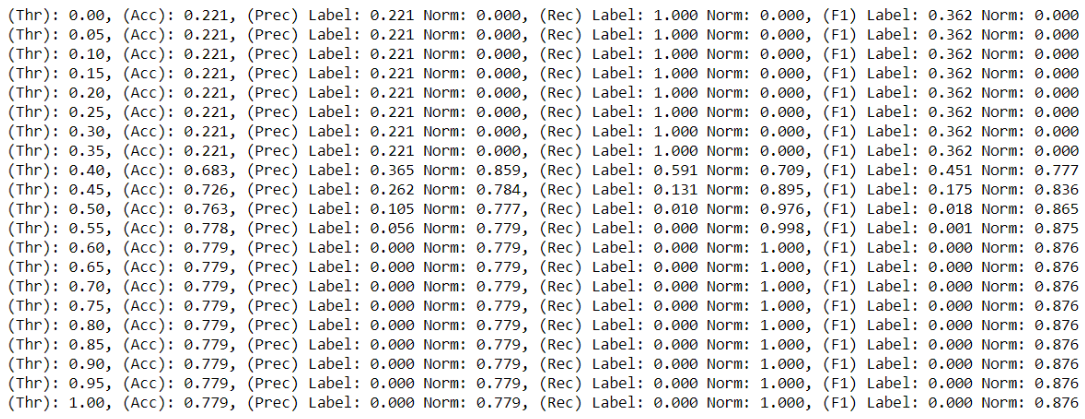
This classification has high true negative rate also. The problem here is even worse due very unevenly distributed data (many more normal connections than U2R connections). Although the team has tried oversampling to generate more U2R records for training, the problem is still significant.

One of the ways to address this issue is to gather or simulated more real U2R data for training. But it is a challenging task to capture meaningful cyber-security related data, so this problem is not easy to overcome.

- **Normal - R2L Classification with Isolation Forest Classifier**

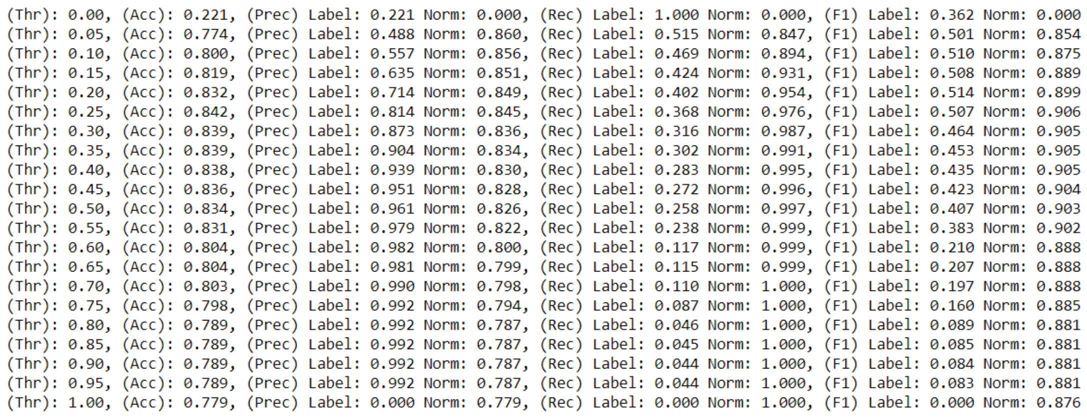
F1-Score Report

Isolation Forest Classifier where the threshold at 0.4



```
(Thr): 0.00, (Acc): 0.221, (Prec) Label: 0.221 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.362 Norm: 0.000
(Thr): 0.05, (Acc): 0.221, (Prec) Label: 0.221 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.362 Norm: 0.000
(Thr): 0.10, (Acc): 0.221, (Prec) Label: 0.221 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.362 Norm: 0.000
(Thr): 0.15, (Acc): 0.221, (Prec) Label: 0.221 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.362 Norm: 0.000
(Thr): 0.20, (Acc): 0.221, (Prec) Label: 0.221 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.362 Norm: 0.000
(Thr): 0.25, (Acc): 0.221, (Prec) Label: 0.221 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.362 Norm: 0.000
(Thr): 0.30, (Acc): 0.221, (Prec) Label: 0.221 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.362 Norm: 0.000
(Thr): 0.35, (Acc): 0.221, (Prec) Label: 0.221 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.362 Norm: 0.000
(Thr): 0.40, (Acc): 0.683, (Prec) Label: 0.365 Norm: 0.859, (Rec) Label: 0.591 Norm: 0.709, (F1) Label: 0.451 Norm: 0.777
(Thr): 0.45, (Acc): 0.726, (Prec) Label: 0.262 Norm: 0.784, (Rec) Label: 0.131 Norm: 0.895, (F1) Label: 0.175 Norm: 0.836
(Thr): 0.50, (Acc): 0.763, (Prec) Label: 0.105 Norm: 0.777, (Rec) Label: 0.010 Norm: 0.976, (F1) Label: 0.018 Norm: 0.865
(Thr): 0.55, (Acc): 0.778, (Prec) Label: 0.056 Norm: 0.779, (Rec) Label: 0.000 Norm: 0.998, (F1) Label: 0.001 Norm: 0.875
(Thr): 0.60, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
(Thr): 0.65, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
(Thr): 0.70, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
(Thr): 0.75, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
(Thr): 0.80, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
(Thr): 0.85, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
(Thr): 0.90, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
(Thr): 0.95, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
(Thr): 1.00, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
```

Voting Classifier (of 3 Random Forest Classifiers) where the threshold of 0.25



```
(Thr): 0.00, (Acc): 0.221, (Prec) Label: 0.221 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.362 Norm: 0.000
(Thr): 0.05, (Acc): 0.774, (Prec) Label: 0.488 Norm: 0.860, (Rec) Label: 0.515 Norm: 0.847, (F1) Label: 0.501 Norm: 0.854
(Thr): 0.10, (Acc): 0.800, (Prec) Label: 0.557 Norm: 0.856, (Rec) Label: 0.469 Norm: 0.894, (F1) Label: 0.510 Norm: 0.875
(Thr): 0.15, (Acc): 0.819, (Prec) Label: 0.635 Norm: 0.851, (Rec) Label: 0.424 Norm: 0.931, (F1) Label: 0.508 Norm: 0.889
(Thr): 0.20, (Acc): 0.832, (Prec) Label: 0.714 Norm: 0.849, (Rec) Label: 0.402 Norm: 0.954, (F1) Label: 0.514 Norm: 0.899
(Thr): 0.25, (Acc): 0.842, (Prec) Label: 0.814 Norm: 0.845, (Rec) Label: 0.368 Norm: 0.976, (F1) Label: 0.507 Norm: 0.906
(Thr): 0.30, (Acc): 0.839, (Prec) Label: 0.873 Norm: 0.836, (Rec) Label: 0.316 Norm: 0.987, (F1) Label: 0.464 Norm: 0.905
(Thr): 0.35, (Acc): 0.839, (Prec) Label: 0.904 Norm: 0.834, (Rec) Label: 0.302 Norm: 0.991, (F1) Label: 0.453 Norm: 0.905
(Thr): 0.40, (Acc): 0.838, (Prec) Label: 0.939 Norm: 0.830, (Rec) Label: 0.283 Norm: 0.995, (F1) Label: 0.435 Norm: 0.905
(Thr): 0.45, (Acc): 0.836, (Prec) Label: 0.951 Norm: 0.828, (Rec) Label: 0.272 Norm: 0.996, (F1) Label: 0.423 Norm: 0.904
(Thr): 0.50, (Acc): 0.834, (Prec) Label: 0.961 Norm: 0.826, (Rec) Label: 0.258 Norm: 0.997, (F1) Label: 0.407 Norm: 0.903
(Thr): 0.55, (Acc): 0.831, (Prec) Label: 0.979 Norm: 0.822, (Rec) Label: 0.238 Norm: 0.999, (F1) Label: 0.383 Norm: 0.902
(Thr): 0.60, (Acc): 0.804, (Prec) Label: 0.982 Norm: 0.800, (Rec) Label: 0.117 Norm: 0.999, (F1) Label: 0.210 Norm: 0.888
(Thr): 0.65, (Acc): 0.804, (Prec) Label: 0.981 Norm: 0.799, (Rec) Label: 0.115 Norm: 0.999, (F1) Label: 0.207 Norm: 0.888
(Thr): 0.70, (Acc): 0.803, (Prec) Label: 0.990 Norm: 0.798, (Rec) Label: 0.110 Norm: 1.000, (F1) Label: 0.197 Norm: 0.888
(Thr): 0.75, (Acc): 0.798, (Prec) Label: 0.992 Norm: 0.794, (Rec) Label: 0.087 Norm: 1.000, (F1) Label: 0.160 Norm: 0.885
(Thr): 0.80, (Acc): 0.789, (Prec) Label: 0.992 Norm: 0.787, (Rec) Label: 0.046 Norm: 1.000, (F1) Label: 0.089 Norm: 0.881
(Thr): 0.85, (Acc): 0.789, (Prec) Label: 0.992 Norm: 0.787, (Rec) Label: 0.045 Norm: 1.000, (F1) Label: 0.085 Norm: 0.881
(Thr): 0.90, (Acc): 0.789, (Prec) Label: 0.992 Norm: 0.787, (Rec) Label: 0.044 Norm: 1.000, (F1) Label: 0.084 Norm: 0.881
(Thr): 0.95, (Acc): 0.789, (Prec) Label: 0.992 Norm: 0.787, (Rec) Label: 0.044 Norm: 1.000, (F1) Label: 0.083 Norm: 0.881
(Thr): 1.00, (Acc): 0.779, (Prec) Label: 0.000 Norm: 0.779, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.876
```

For the Isolation Forest Classifier, a threshold value of 0.4 was chosen given a relatively high recall and highest F1-score for R2L. Another close contender is the Voting Classifier with a threshold of 0.25 due to the highest accuracy. The accuracy on test data achieved using threshold value 0.4 is 68.26%, at threshold value 0.25 is 84.17%

Confusion Matrix

Isolation Forest Classifier with threshold at 0.4

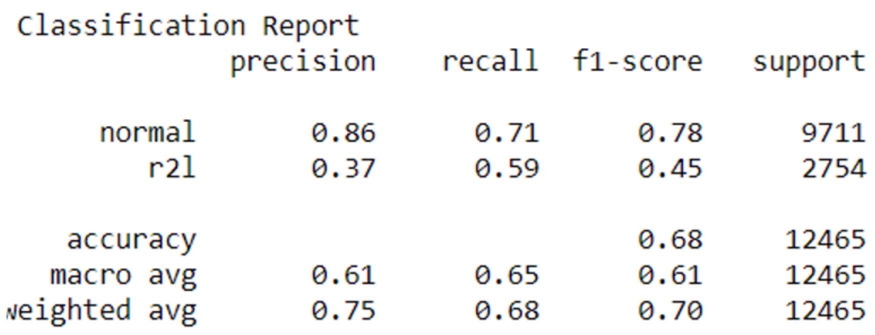
	Predicted Normal	Predicted Attack	Total
Actual Normal	6881	2830	9711
Actual Attack	1126	1628	2754
Total	8007	4458	12465

Voting Classifier w threshold at 0.25

	Predicted Normal	Predicted Attack	Total
Actual Normal	9479	232	9711
Actual Attack	1741	1013	2754
Total	11220	1245	12465

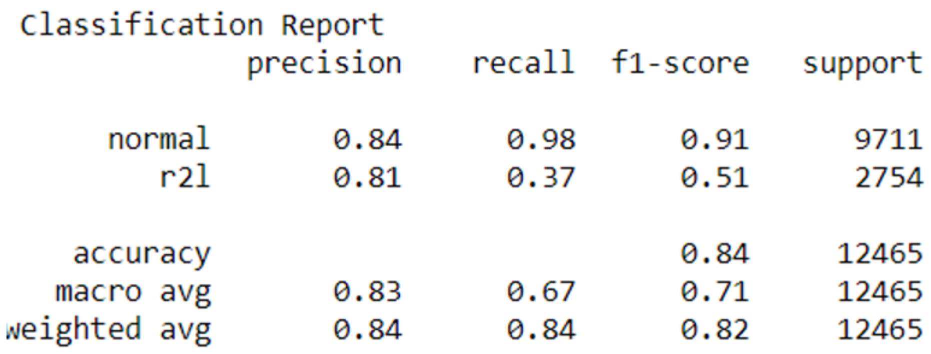
Classification Report

Threshold at 0.4



```
Classification Report
              precision    recall  f1-score   support

      normal       0.86      0.71      0.78      9711
         r2l       0.37      0.59      0.45      2754

    accuracy                           0.68     12465
   macro avg       0.61      0.65      0.61     12465
weighted avg       0.75      0.68      0.70     12465
```

Threshold at 0.25



```
Classification Report
              precision    recall  f1-score   support

      normal       0.84      0.98      0.91      9711
         r2l       0.81      0.37      0.51      2754

    accuracy                           0.84     12465
   macro avg       0.83      0.67      0.71     12465
weighted avg       0.84      0.84      0.82     12465
```

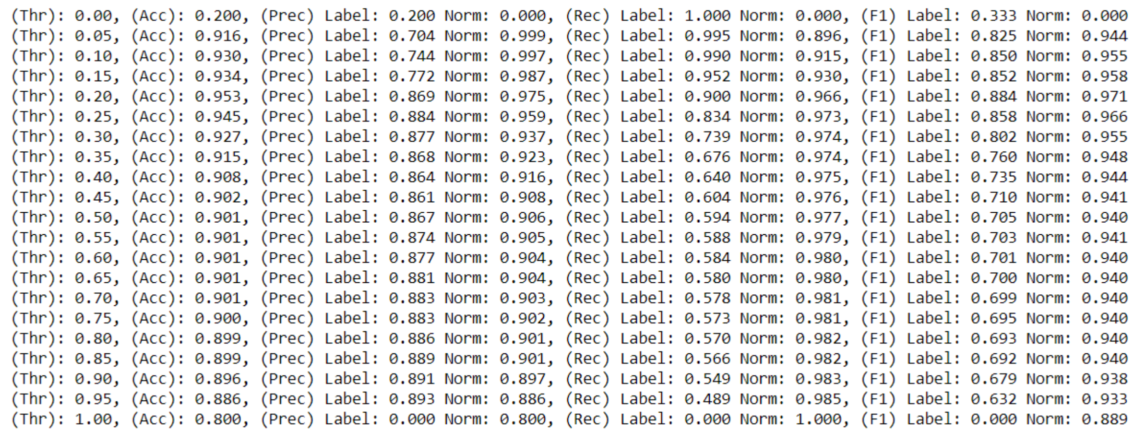
Although threshold 0.25 gives a better accuracy score. The team still prefers to use threshold 0.4 for prediction given that it can better predict true attacks. As the ability to find out attacks using the model is way more important than finding out normal connections.

Overall, the results obtained are not desirable because of the poor balance between accuracy and low false positive rate.

The poor result is also caused by a lack of meaningful data for training.

- **Normal - Probe Attack Classification with Random Forest Classifier**

F1-Score Report



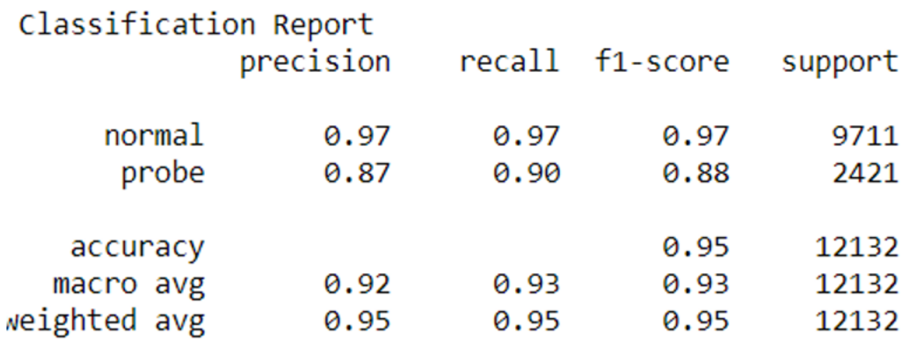
```
(Thr): 0.00, (Acc): 0.200, (Prec) Label: 0.200 Norm: 0.000, (Rec) Label: 1.000 Norm: 0.000, (F1) Label: 0.333 Norm: 0.000
(Thr): 0.05, (Acc): 0.916, (Prec) Label: 0.704 Norm: 0.999, (Rec) Label: 0.995 Norm: 0.896, (F1) Label: 0.825 Norm: 0.944
(Thr): 0.10, (Acc): 0.930, (Prec) Label: 0.744 Norm: 0.997, (Rec) Label: 0.990 Norm: 0.915, (F1) Label: 0.850 Norm: 0.955
(Thr): 0.15, (Acc): 0.934, (Prec) Label: 0.772 Norm: 0.987, (Rec) Label: 0.952 Norm: 0.930, (F1) Label: 0.852 Norm: 0.958
(Thr): 0.20, (Acc): 0.953, (Prec) Label: 0.869 Norm: 0.975, (Rec) Label: 0.900 Norm: 0.966, (F1) Label: 0.884 Norm: 0.971
(Thr): 0.25, (Acc): 0.945, (Prec) Label: 0.884 Norm: 0.959, (Rec) Label: 0.834 Norm: 0.973, (F1) Label: 0.858 Norm: 0.966
(Thr): 0.30, (Acc): 0.927, (Prec) Label: 0.877 Norm: 0.937, (Rec) Label: 0.739 Norm: 0.974, (F1) Label: 0.802 Norm: 0.955
(Thr): 0.35, (Acc): 0.915, (Prec) Label: 0.868 Norm: 0.923, (Rec) Label: 0.676 Norm: 0.974, (F1) Label: 0.760 Norm: 0.948
(Thr): 0.40, (Acc): 0.908, (Prec) Label: 0.864 Norm: 0.916, (Rec) Label: 0.640 Norm: 0.975, (F1) Label: 0.735 Norm: 0.944
(Thr): 0.45, (Acc): 0.902, (Prec) Label: 0.861 Norm: 0.908, (Rec) Label: 0.604 Norm: 0.976, (F1) Label: 0.710 Norm: 0.941
(Thr): 0.50, (Acc): 0.901, (Prec) Label: 0.867 Norm: 0.906, (Rec) Label: 0.594 Norm: 0.977, (F1) Label: 0.705 Norm: 0.940
(Thr): 0.55, (Acc): 0.901, (Prec) Label: 0.874 Norm: 0.905, (Rec) Label: 0.588 Norm: 0.979, (F1) Label: 0.703 Norm: 0.941
(Thr): 0.60, (Acc): 0.901, (Prec) Label: 0.877 Norm: 0.904, (Rec) Label: 0.584 Norm: 0.980, (F1) Label: 0.701 Norm: 0.940
(Thr): 0.65, (Acc): 0.901, (Prec) Label: 0.881 Norm: 0.904, (Rec) Label: 0.580 Norm: 0.980, (F1) Label: 0.700 Norm: 0.940
(Thr): 0.70, (Acc): 0.901, (Prec) Label: 0.883 Norm: 0.903, (Rec) Label: 0.578 Norm: 0.981, (F1) Label: 0.699 Norm: 0.940
(Thr): 0.75, (Acc): 0.900, (Prec) Label: 0.883 Norm: 0.902, (Rec) Label: 0.573 Norm: 0.981, (F1) Label: 0.695 Norm: 0.940
(Thr): 0.80, (Acc): 0.899, (Prec) Label: 0.886 Norm: 0.901, (Rec) Label: 0.570 Norm: 0.982, (F1) Label: 0.693 Norm: 0.940
(Thr): 0.85, (Acc): 0.899, (Prec) Label: 0.889 Norm: 0.901, (Rec) Label: 0.566 Norm: 0.982, (F1) Label: 0.692 Norm: 0.940
(Thr): 0.90, (Acc): 0.896, (Prec) Label: 0.891 Norm: 0.897, (Rec) Label: 0.549 Norm: 0.983, (F1) Label: 0.679 Norm: 0.938
(Thr): 0.95, (Acc): 0.886, (Prec) Label: 0.893 Norm: 0.886, (Rec) Label: 0.489 Norm: 0.985, (F1) Label: 0.632 Norm: 0.933
(Thr): 1.00, (Acc): 0.800, (Prec) Label: 0.000 Norm: 0.800, (Rec) Label: 0.000 Norm: 1.000, (F1) Label: 0.000 Norm: 0.889
```

A threshold value of 0.2 was chosen given a relatively high recall and highest F1-score for probing attack. The accuracy on test data achieved using the threshold is 95.30%.

Confusion Matrix

	Predicted Normal	Predicted Attack	Total
Actual Normal	9383	328	9711
Actual DOS	242	2179	2421
Total	9625	2507	12132

Classification Report



```
Classification Report
              precision    recall  f1-score   support

      normal       0.97      0.97      0.97      9711
       probe       0.87      0.90      0.88      2421

    accuracy                           0.95     12132
   macro avg       0.92      0.93      0.93     12132
weighted avg       0.95      0.95      0.95     12132
```

The results obtained for this classification is quite desirable as there is a controlled balance between accuracy and low false positive rate.

6.0 Multi-Class Classification

On top of that, the team also explores neural network for multi-class classification: normal vs DOS attack vs U2R attack vs R2L attack vs Probing Attack.

Confusion Matrix:

	Predicted Normal	Predicted DOS	Predicted U2R	Predicted R2L	Predicted Probe Attack	Total
Actual Normal	9440	67	1	2	201	9711
Actual DOS	1002	6256	0	0	200	7458
Actual U2R	192	0	3	2	3	200
Actual R2L	2467	0	0	283	4	2754
Actual Probe Attack	631	166	0	53	1571	2421
Total	13732	6489	4	340	1979	22544

Classification Report

```
              precision    recall  f1-score   support

         dos       0.96      0.84      0.90      7458
      normal       0.69      0.97      0.81      9711
       probe       0.79      0.65      0.71      2421
         r2l       0.83      0.10      0.18      2754
         u2r       0.75      0.01      0.03       200

    accuracy                           0.78     22544
   macro avg       0.81      0.52      0.53     22544
weighted avg       0.81      0.78      0.74     22544
```

The classification method is more suitable for data classification and attack detection because the cumulative false positive rate is about 33% which is not considered low.

7.0 Limitations of datasets

- If the adversary is able to inject poisoned data into a training dataset, it can end up making a dataset less effective. It is important for organizations to not assume that the data that was used to train a data model is necessarily fully representative of all the data that is seen in the real world.
- The dataset may not be a perfect representative of existing real networks, because of the lack of public data sets for network-based IDS. The team believes it can still be applied as an effective benchmark data set to help researchers compare different intrusion detection methods.

8.0 Conclusion

After working on this assignment, the team had a deeper understanding of the process of data preparation and the strengths and limitations of the different algorithms. An example, in the case of the Isolation Forest algorithm, the model performed especially well when data was significantly skewed, but it did very poorly when the data used to train the model was not so skewed. For that, data which have a relatively even to high occurrence, classifiers such as Random Forest classifier and Voting Classifier worked better.

9.0 Reference

Asaithambi, S. (2018, Jan 31). Why, How and When to apply Feature Selection. Retrieved from Towards Data Science: https://towardsdatascience.com/why-how-and-when-to-apply-feature-selection-e9c69adfabf2

Badr, W. (2019, Feb 22). *Having an Imbalanced Dataset? Here Is How You Can Fix It.* Retrieved from Towards Data Science: https://towardsdatascience.com/having-an-imbalanced-dataset-here-is-how-you-can-solve-it-1640568947eb#targetText=It%20is%20also%20used%20to,of%20classes%20within%20a%20dataset.)

Gavin C. Cawley, N. L. (2010). On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation. *Journal of Machine Learning Research 11*, Gavin C. Cawley, Nicola L. C. Talbot; 11(Jul):2079−2107.

Haibo He, Y. B. (n.d.). ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced. *2008 International Joint Conference on Neural Networks (IJCNN 2008)*.

Jaitley, U. (2018, Oct 8). *models, Why Data Normalization is necessary for Machine Learning*. Retrieved from Medium: https://medium.com/@urvashilluniya/why-data-normalization-is-necessary-for-machine-learning-models-681b65a05029#targetText=Normalization%20is%20a%20technique%20often,dataset%20does%20not%20require%20normalization

L.Dhanabal, S. S. (2015). A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms. *International Journal of Advanced Research in Computer and Communication Engineering*.

Mahbod Tavallaee, E. B. (2009). A Detailed Analysis of the KDD CUP 99 Data Set. *CISDA 2009*.

Naahid, M. K. (2005). Analysis of KDD CUP 99 Dataset using Clustering based Data. *International Journal of Database Theory and Application*, 23-34.

S. Revathi, A. M. (2013). A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning. *International Journal of Engineering Research & Technology (IJERT)*.

Stewart, J. M. (2018). Chapter 1. In J. M. Stewart, *CompTIA Security+ Review Guide, 4th Edition.* Sybex.

UNB. (n.d.). *NSL-KDD dataset*. Retrieved from UNB Official Website: https://www.unb.ca/cic/datasets/nsl.html

Zhang, Z. (2019, Mar 27). *Understand Data Normalization in Machine Learning*. Retrieved from Towards Data Science: https://towardsdatascience.com/understand-data-normalization-in-machine-learning-8ff3062101f0

Appendix 1

Basic Features of Each Network Connection Vector (Part 1)

S/N	Attribute Name	Description
1	duration	Length of time duration of the connection
2	protocol_type	Protocol used in the connection
3	service	Destination network service used
4	flag	Status of the connection – Normal or Error
5	src_bytes	Number of data bytes transferred from source to destination in single connection
6	dst_bytes	Number of data bytes transferred from destination to source in single connection
7	land	if source and destination IP addresses and port numbers are equal then this variable takes value 1 else 0
8	wrong_fragment	Total number of wrong fragments in this connection
9	urgent	Number of urgent packets in this connection. Urgent packets are packets with the urgent bit activated
10	hot	Number of "hot" indicators in the content such as: entering a system directory, creating programs and executing program
11	num_failed_logins	Count of failed login attempts
12	logged_in	Login Status: 1 if successfully logged in; 0 otherwise
13	num_compromised	Number of ``compromised' ' conditions
14	root_shell	1 if root shell is obtained; 0 otherwise
15	su_attempted	1 if ``su root'' command attempted or used; 0 otherwise

Basic Features of Each Network Connection Vector (Part 2)

16	num_root	Number of ``root'' accesses or number of operations performed as a root in the connection
17	num_file_creations	Number of file creation operations in the connection
18	num_shells	Number of shell prompts
19	num_access_files	Number of operations on access control files
20	num_outbound_cmds	Number of outbound commands in an ftp session
21	is_host_login	1 if the login belongs to the ``hot'' list i.e., root or admin; else 0
22	is_guest_login	1 if the login is a ``guest'' login; 0 otherwise
23	count	Number of connections to the same destination host as the current connection in the past two seconds
24	srv_count	Number of connections to the same service (port number) as the current connection in the past two seconds
25	serror_rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in count (23)
26	srv_serror_rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in srv_count (24)
27	rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in count (23)
28	srv_rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in srv_count (24)
29	same_srv_rate	The percentage of connections that were to the same service, among the connections aggregated in count (23)
30	diff_srv_rate	The percentage of connections that were to different services, among the connections aggregated in count (23)

Basic Features of Each Network Connection Vector (Part 3)

31	srv_diff_host_rate	The percentage of connections that were to different destination machines among the connections aggregated in srv_count (24)
32	dst_host_count	Number of connections having the same destination host IP address
33	dst_host_srv_count	Number of connections having the same port number
34	dst_host_same_srv_rate	The percentage of connections that were to the same service, among the connections aggregated in dst_host_count (32)
35	dst_host_diff_srv_rate	The percentage of connections that were to different services, among the connections aggregated in dst_host_count (32)
36	dst_host_same_src_port_rat e	The percentage of connections that were to the same source port, among the connections aggregated in dst_host_srv_c ount (33)
37	dst_host_srv_diff_host_rate	The percentage of connections that were to different destination machines, among the connections aggregated in dst_host_srv_count (33)
38	dst_host_serror_rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_count (32)
39	dst_host_srv_serror_rate	The percent of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_srv_c ount (33)
40	dst_host_rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_count (32)
41	dst_host_srv_rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_srv_c ount (33)

Appendix 2

Training Dataset

Category	Attack Type	Count	Subtotal	Percentage	Summary
DOS	neptune	41214			
	back	956			
	smurf	2646			
	pod	201			
	land	18			
	teardrop	892	45927	36.46%	
U2R	buffer_overflow	30			
	loadmodule	9			
	rootkit	10			
	perl	3	52	0.04%	
R2L	warezclient	890			
	multihop	7			
	ftp_write	8			
	imap	11			
	guess_passwd	53			
	warezmaster	20			
	spy	2			
	phf	4	995	0.79%	
Probing Attack	portsweep	2931			
	satan	3633			
	nmap	1493			
	ipsweep	3599	11656	9.25%	46.54%
Normal	Normal	67343	67343	53.46%	53.46%
Total		125973	125973	100.00%	100.00%

Test Dataset

Category	Subtotal	Percentage	Summary
DOS	7458	33.08	56.92%
U2R	200	0.89%	
R2L	2754	12.22%	
Probing Attack	2421	10.74%	
Normal	9711	43.08%	43.08%
Total	22,544	100.00%	100.00%

Appendix 3

Shown below is a summary of the best models:

attack_ type	best_ model	best_ threshold	best_ features	oversampling_ amt	pca_ amt
DOS	Voting	0.1	src_bytes,dst_bytes,count,same_srv_rate,diff_srv_rate,dst_host_count,dst_host_srv_count,dst_host_serror_rate,dst_host_srv_serror_rate,service_ecr_i,service_http,flag_RSTR,flag_S0,wrong_fragment_0,logged_in_0	0	
Probe	RandomForest	0.2	src_bytes,dst_bytes,count,rerror_rate,dst_host_count,dst_host_same_srv_rate,dst_host_diff_srv_rate,dst_host_same_src_port_rate,dst_host_rerror_rate,service_finger,service_ftp_data,service_http,service_private,service_smtp,service_telnet	0	
R2L	IsoForest/ Voting	0.4/ 0.25	(all features)/ src_bytes,dst_bytes,hot,dst_host_count,dst_host_srv_count,dst_host_same_src_port_rate,dst_host_srv_diff_host_rate,service_ftp,service_ftp_data,service_http,service_imap4,flag_RSTO,is_guest_login_0,is_guest_login_1	0/ 50000	0/ 0
U2R	IsolationForest	0.5	(all features)	0	0