Machine Learning Algorithms for Intrusion Detection in Legacy and Modern Systems

Van N. Nguyen, MSOL Graduate Student, University of California, Riverside

ENGR 296D Final Project – Summer 2020

# Abstract

The KDD Cup 1999 Data is the most widely used dataset in intrusion detection and also the de facto benchmark for evaluating the performance of an intrusion detection system. The dataset has been popular among the academic community because of the ease of use and prelabeled samples. However, the KDD 1999 Cup Data is over 20 years old and has been studied so extensively that the weaknesses in that dataset have come to be known in the field. This project will examine whether the KDD Cup 1999 Data is relevant today and whether a more modern dataset like the CSE-CIC-IDS2018 Data is fundamentally different from a machine learning standpoint.

# Introduction

## Project Overview

In this project, we will examine legacy and modern datasets for intrusion detection. For the legacy dataset we will examine the KDD Cup 1999 Data looking at the performance of machine learning classification algorithms against that data and then compare the same machine learning classification algorithms against a modern network traffic dataset that is similar in features and samples. We will examine whether the KDD Cup 1999 Data is relevant today and whether a more modern dataset is fundamentally different from a machine learning standpoint. For modern dataset we will be using is CSE-CIC-IDS2018 Data, a labeled dataset of network intrusion by the Communications Security Establishment (CSE) & the Canadian Institute for Cybersecurity (CIC). We will compare the performance of standard machine learning algorithms at classifying normal network traffic and network intrusion traffic against the legacy KDD Cup 1999 Data and the CSE-CIC-IDS2018 Data.

## Significance of the Project

Network Intrusion Detection is an important use-case for data science and machine learning. Cybersecurity firms are increasingly using machine learning to thwart cyber threats for their clients. Machine learning is well suited for network intrusion detection because of the “cat and mouse nature” of the problem. As firms adapt their security to defend against threats, the cyber threats evolve and exploit gaps in the defense. Machine learning offers the ability to continuously learn and adapt to new threats. The global cyber security market size was estimated at $156.45 billion in 2019[[1]](#endnote-1). The global cost of cybercrime is estimated in the trillions of US dollars[[2]](#endnote-2). The opportunities to leverage this technology are enormous.

# Literature Review

## Previous research

The KDD is the most widely used dataset in intrusion detection and also the de facto benchmark for evaluating the performance of an intrusion detection system[4]. However, the KDD 1999 Cup Data is over 20 years old and has been studied so extensively that the weaknesses in that dataset have come to be known in the field. The KDD 99 Cup data has high redundancy of data (78%) [3] and extensively pre-processing of the data as others researchers have done[4], show weaknesses and bias in the data.

We know that prior research has shown the success of clustering algorithms with the KDD 199 Cup Data [1][3]. In this project we will validate some of these findings and try additional Machine Learning algorithms. We will also compare how these Machine Learning algorithms perform against modern datasets with more features and more samples.

Finally, we know that prior research has shown that kNN algorithm is prone is having high false negative rate[3][5]. We will verify these claims and see if these problems persist in modern datasets.

## Findings and unanswered questions

Although we see the extensive use of KDD Cup 1999 Data in the literature, we need to examine whether the KDD Cup 1999 Data is relevant today. We need to first answer can replicate the results and success of prior research using KDD Cup 1999 Data for intrusion detection. And to follow on to that question, whether a more modern dataset is fundamentally different from a machine learning standpoint.

## Your preliminary work on the project topic

Prior to this research, my work as an engineer for the Naval Surface Warfare Center Corona Division has been related to this field of study. Our group conducts readiness assessment for critical infrastructure systems. In additional to material readiness and reliability, we also assess cyber readiness, or the impact of cyber on reliability and availability of systems.

In 2018, my colleagues at Naval Surface Warfare Center Corona Division did research on using Machine Learning for Intrusion Detection. Although I did not participate directly in the research activities, I followed their work to know that certain Machine Learning algorithms were successful at classifying attack versus normal network traffic. The research utilized R programming language and wrote most of the Machine Learning algorithms from scratch rather than use popular libraries that are available today.

From the prior research performed by my colleagues, I became aware of the KDD 1999 Cup dataset. My colleagues successfully used k-Nearest Neighbors and Support Vector Machines for classification. The follow-on research was to take the algorithms and apply it to real world network traffic data collected on Navy systems from Test and Evaluation events. The PI for the research has moved on to another organization and the research is suspended at this time.

Part of the follow-on research was to answer whether we could take Machine Learning algorithms that were successful with the KDD 199 Cup Data and apply it to other datasets. This project hopes to help answer that question. We will retest the Machine Learning algorithms with KDD 1999 Cup Data and then test those algorithms against something from the modern area, in this case the CSE-CIC-IDS2018 Data.

## Remaining questions

We see in the academic literature that Machine Learning algorithms have mixed success at intrusion detection. Usually one Machine Learning algorithm performs significantly better than another. We also see that network intrusion is not a monolithic attack vector—it is a diverse set of conditions for different attack scenarios for different systems. The research shows that Machine Learning algorithms trained to a specific set of attack overall perform much better. Additional research could be performed on whether a layered defense approach utilizing multiple Machine Learning algorithms could reduce false negative rates without increasing false positives.

# Methodology

## Approach

We will utilize the KDD Cup 1999 Data as a representative sample of legacy intrusion detection dataset. We will then use the CSE-CIC-IDS2018 Data as a representative sample of modern intrusion detection dataset. We will first verify that machine learning algorithms that have had prior success in the literature can correctly classify network intrusions with high accuracy in the legacy dataset. Then we will apply the same algorithms to the modern dataset. We will compare the performance of the machine learning algorithms and resulting models based on accuracy against the legacy and modern datasets. We will postulate why there are differences in accuracy and test those theories if we can. We will also note the runtime performance of the algorithms however this is not our main focus because the models developed here will be running on mid-range consumer level hardware.

## Data Collection

The KDD Cup 1999 Data is a data set used for The Third International Knowledge Discovery and Data Mining Tools Competition at KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining. The competition task was to build a network intrusion detector, a predictive model capable of distinguishing between ``bad'' connections, called intrusions or attacks, and ``good'' normal connections. The data includes a wide variety of intrusions simulated in a military network environment.

The advantage of the KDD Cup 1999 Data is that samples are labeled allowing for supervised machine learning techniques. The labels represent the type of network intrusion attack. Non-attack samples are also labeled as “normal”. The data contains samples of 21 different types of network attacks. There are 41 different features with symbolic values for categories and continuous values for the remaining features. The data is similar to what is commonly gathered by network packet capture and Intrusion Detection Systems (IDS). However, this data is pre-processed in tabular format which has made it extremely popular for academic study over the past 20 years in the area of intrusion detection. The entire dataset is only 725MB which is relatively small by modern standards but the dataset is over two decades old and was meant to be processed with consumer-level hardware in 1999. We will see that the data has adequate number of samples of certain types of attacks but also the data is also sparse for certain types of attacks.

These are the KDD Cup 1999 Data labels of the attacks with non-attacks labeled as “normal”:

back,buffer\_overflow,ftp\_write,guess\_passwd,imap,ipsweep,land,loadmodule,multihop,neptune,nmap,normal,perl,phf,pod,portsweep,rootkit,satan,smurf,spy,teardrop,warezclient,warezmaster.

These are the KDD Cup 1999 Data features and the type of value:

duration: continuous.

protocol\_type: symbolic.

service: symbolic.

flag: symbolic.

src\_bytes: continuous.

dst\_bytes: continuous.

land: symbolic.

wrong\_fragment: continuous.

urgent: continuous.

hot: continuous.

num\_failed\_logins: continuous.

logged\_in: symbolic.

num\_compromised: continuous.

root\_shell: continuous.

su\_attempted: continuous.

num\_root: continuous.

num\_file\_creations: continuous.

num\_shells: continuous.

num\_access\_files: continuous.

num\_outbound\_cmds: continuous.

is\_host\_login: symbolic.

is\_guest\_login: symbolic.

count: continuous.

srv\_count: continuous.

serror\_rate: continuous.

srv\_serror\_rate: continuous.

rerror\_rate: continuous.

srv\_rerror\_rate: continuous.

same\_srv\_rate: continuous.

diff\_srv\_rate: continuous.

srv\_diff\_host\_rate: continuous.

dst\_host\_count: continuous.

dst\_host\_srv\_count: continuous.

dst\_host\_same\_srv\_rate: continuous.

dst\_host\_diff\_srv\_rate: continuous.

dst\_host\_same\_src\_port\_rate: continuous.

dst\_host\_srv\_diff\_host\_rate: continuous.

dst\_host\_serror\_rate: continuous.

dst\_host\_srv\_serror\_rate: continuous.

dst\_host\_rerror\_rate: continuous.

dst\_host\_srv\_rerror\_rate: continuous.

The CSE-CIC-IDS2018 Data, a labeled dataset of network intrusion by the Communications Security Establishment (CSE) & the Canadian Institute for Cybersecurity (CIC). The dataset was preprocessed and extraction from network traffic capture and systems logs and presented in tabular format. The dataset includes 80 features and each sample is label with the name of the attack or “Benign” for non-attack samples. The dataset includes several different attack scenarios: Brute-force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and infiltration. However, unlike the KDD Cup 1999 Data, the data is broken up into chunks specific to only one attack scenario. As we shall see, this allows training the algorithm for each specific type of attack. The entire dataset is 6.41GB however each attack scenarios block ranges from 105MB to 3.96GB. As we will see, even smaller blocks have adequate number of samples.

These are the CSE-CIC-IDS2018 Data labels of the attacks with non-attacks labeled as “Benign”:

Brute-force, Heartbleed, Botnet, DoS, DDoS, Web attacks, Infilteration, Benign

These are the CSE-CIC-IDS2018 Data features and the type of value:

Dst Port int64

Protocol int64

Flow Duration int64

Tot Fwd Pkts int64

Tot Bwd Pkts int64

TotLen Fwd Pkts int64

TotLen Bwd Pkts int64

Fwd Pkt Len Max int64

Fwd Pkt Len Min int64

Fwd Pkt Len Mean float64

Fwd Pkt Len Std float64

Bwd Pkt Len Max int64

Bwd Pkt Len Min int64

Bwd Pkt Len Mean float64

Bwd Pkt Len Std float64

Flow Byts/s float64

Flow Pkts/s float64

Flow IAT Mean float64

Flow IAT Std float64

Flow IAT Max int64

Flow IAT Min int64

Fwd IAT Tot int64

Fwd IAT Mean float64

Fwd IAT Std float64

Fwd IAT Max int64

Fwd IAT Min int64

Bwd IAT Tot int64

Bwd IAT Mean float64

Bwd IAT Std float64

Bwd IAT Max int64

Bwd IAT Min int64

Fwd PSH Flags int64

Bwd PSH Flags int64

Fwd URG Flags int64

Bwd URG Flags int64

Fwd Header Len int64

Bwd Header Len int64

Fwd Pkts/s float64

Bwd Pkts/s float64

Pkt Len Min int64

Pkt Len Max int64

Pkt Len Mean float64

Pkt Len Std float64

Pkt Len Var float64

FIN Flag Cnt int64

SYN Flag Cnt int64

RST Flag Cnt int64

PSH Flag Cnt int64

ACK Flag Cnt int64

URG Flag Cnt int64

CWE Flag Count int64

ECE Flag Cnt int64

Down/Up Ratio int64

Pkt Size Avg float64

Fwd Seg Size Avg float64

Bwd Seg Size Avg float64

Fwd Byts/b Avg int64

Fwd Pkts/b Avg int64

Fwd Blk Rate Avg int64

Bwd Byts/b Avg int64

Bwd Pkts/b Avg int64

Bwd Blk Rate Avg int64

Subflow Fwd Pkts int64

Subflow Fwd Byts int64

Subflow Bwd Pkts int64

Subflow Bwd Byts int64

Init Fwd Win Byts int64

Init Bwd Win Byts int64

Fwd Act Data Pkts int64

Fwd Seg Size Min int64

Active Mean float64

Active Std float64

Active Max int64

Active Min int64

Idle Mean float64

Idle Std float64

Idle Max int64

Idle Min int64

Label object

Although the data provided was specifically developed for Machine Learning we will go through the standard steps in preprocessing and cleaning the data as if the data came from a raw source. Both datasets are in tabular format with labels for each sample. This saves effort in de-normalizing the data, in the relational database sense, or flattening multiple tables/files in a single tabular format. With the tabular format in hand we checked the data for missing values, categorical data, non-numerical data, and good variance in the data. Because this data is pre-processed, we had very few problems with the data.

The KDD Cup 1999 Data had 3 categorical data features. We used a LabelEncoder function to assign numerical values to these categories. Other than that issue, the KDD Cup 1999 Data was clean with no missing values, clean numerical data, and had good variance.

The CSE-CIC-IDS2018 Data contained a Timestamp feature of type datetime. We decided that the Timestamp feature is not relevant to our analysis should be removed. We know that the time an attack occurs in not relevant to predicting future attacks. If we include the Timestamp feature in the training data we know that past attacks may be time correlated and the model may overfit to the Timestamp feature thus reducing the accuracy of our model. Therefore, we removed the Timestamp feature prior to training the model. The remaining features are all numerical values with no categorical features. However, with some simple pre-processing we discover that the data includes some values that converts to “NaN” or Not a Number. Some numerical values are also specified as Infinity or minus Infinity. This would cause an error when the model would try to convert the values. Although the number of samples with invalid numerical values is small, we decided to drop the samples from the dataset rather than assigning an arbitrary value such as zero or a large integer value because these arbitrary values may distort our model. The number of samples removed was very small with only less than 0.6% of the samples removed. The remaining dataset has no missing values, clean numerical data, and had good variance.

Before applying our Machine Learning models, we will attempt to reduce the number of features using some techniques of dimensionality reduction. Due to the number of features in both datasets we will select the Random Forest Algorithm over Principal Component Analysis (PCA). Random Forest algorithm is one of the most popular methods of dimensionality reduction or feature reduction and is easier to use and performs well with a large number of features. The algorithm is pre-built in sci-kit learn and will assign a feature importance score to each feature. This allows us to remove certain features in our model to reduce the complexity of the model and improve runtime if needed.

### KDD Cup 1999 Data Feature Reduction Analysis

The KDD Cup 1999 Data contains 41 features. Although this is not an extremely large number of features, it is still enough features to overwhelm certain machine learning algorithms such as Logistic Regression or Support Vector Machine (SVM). In addition, a large number of features, particularly irrelevant ones, can cause the model to overfit and reduce the accuracy of the prediction. Processing the KDD Cup 1999 Data through a Random Forest Algorithm results in the following Feature Importance Scores and a ranking of the most important features. The accuracy of the prediction was greater than 99.9%.

Random Forest Classifier Results with n\_estimators = 1000

Accuracy: 0.9997436018541634

Feature Importance Score

count 0.179769

srv\_count 0.124680

same\_srvrate 0.080571

src\_bytes 0.074948

protocol\_type 0.073758

dst\_host\_same\_src\_port\_rate 0.071520

diff\_srv\_rate 0.061298

flag 0.056767

service 0.055318

dst\_host\_same\_srv\_rate 0.034246

dst\_host\_diff\_srv\_rate 0.033881

dst\_bytes 0.033329

dst\_host\_srv\_count 0.019166

dst\_host\_serror\_rate 0.019150

logged\_in 0.015758

serror\_rate 0.013443

dst\_host\_count 0.010805

dst\_host\_srv\_serror\_rate 0.009024

dst\_host\_srv\_diff\_host\_ate 0.006068

dst\_host\_rerror\_rate 0.003736

hot 0.003478

wrong\_fragment 0.003440

num\_compromised 0.003362

dst\_host\_srv\_rerror\_rate 0.003152

rerror\_rate 0.002592

srv\_serror\_rate 0.002250

srv\_diff\_host\_rate 0.002100

duration 0.001076

srv\_rerror\_rate 0.000713

is\_guest\_login 0.000359

num\_failed\_logins 0.000091

root\_shell 0.000039

num\_ile\_creations 0.000028

num\_root 0.000027

land 0.000023

num\_access\_files 0.000013

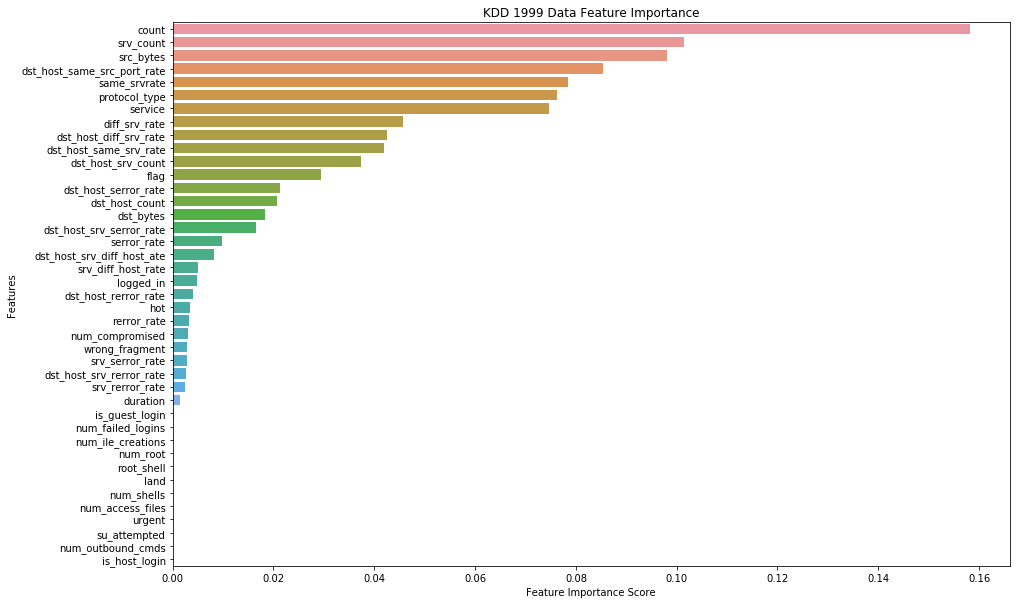
num\_shells 0.000011

urgent 0.000009

su\_attempted 0.000002

num\_outbound\_cmds 0.000000

is\_host\_login 0.000000



We see from the resulting Feature Importance Scores for KDD 1999 Cup Data that at least 12 features had very little relevance to the prediction. By eliminating these features we can improve the runtime of our model while maintaining 99.9% accuracy.

### CSE-CIC-IDS2018 Data Feature Reduction Analysis

The CSE-CIC-IDS2018 Data contains 80 features. This modern dataset has twice the number of features of the KDD 1999 Cup Data. With this dataset our models will have an expected longer runtime than with the KDD 1999 Cup Data. Also, with this many features we are vulnerable to long runtimes and overfitting the data.

We will remove the Timestamp feature because the time of the attack is not relevant to detecting future attacks. By leaving the Timestamp feature in the model is vulnerable to overfitting because within the training data the attacks may be time correlated. However, we know logically that the Timestamp feature is not relevant in detecting future attacks.

We perform a Random Forest classifier and obtain the following results:

Random Forest Classifier Results with n\_estimators = 1000

Accuracy: 0.7497841653547306

Feature Importance Score

Dst Port 0.100080

Fwd Pkts/s 0.056449

Flow IAT Max 0.056117

Flow Duration 0.053613

Flow IAT Min 0.053206

Flow IAT Mean 0.052637

Flow Pkts/s 0.051192

Bwd Pkts/s 0.051181

Flow Byts/s 0.041710

Init Fwd Win Byts 0.037802

Fwd IAT Max 0.024511

Fwd IAT Min 0.023884

Fwd Seg Size Min 0.023633

Fwd IAT Mean 0.023305

Flow IAT Std 0.021666

Fwd IAT Tot 0.020397

Fwd IAT Std 0.017259

Bwd IAT Min 0.013001

Bwd IAT Mean 0.012715

Bwd IAT Max 0.012361

Bwd IAT Std 0.012061

Pkt Len Mean 0.012033

Bwd IAT Tot 0.011217

Init Bwd Win Byts 0.010274

Pkt Size Avg 0.009991

Fwd Pkt Len Max 0.008927

Fwd Pkt Len Mean 0.008841

Fwd Header Len 0.008508

Pkt Len Max 0.008263

Pkt Len Var 0.007909

Fwd Seg Size Avg 0.007696

Subflow Fwd Byts 0.007648

Pkt Len Std 0.007605

TotLen Fwd Pkts 0.007345

PSH Flag Cnt 0.007330

Bwd Seg Size Avg 0.006940

Bwd Pkt Len Mean 0.006765

TotLen Bwd Pkts 0.006421

Subflow Bwd Byts 0.006414

Active Min 0.005612

Active Max 0.005263

Subflow Fwd Pkts 0.005230

Active Mean 0.005161

Tot Fwd Pkts 0.005098

Bwd Pkt Len Max 0.004822

Idle Min 0.004647

Idle Std 0.004320

Bwd Header Len 0.004129

Bwd Pkt Len Std 0.003915

Idle Mean 0.003903

Fwd Pkt Len Std 0.003697

Bwd Pkt Len Min 0.003425

Active Std 0.003294

ACK Flag Cnt 0.003017

Fwd Pkt Len Min 0.002982

Subflow Bwd Pkts 0.002981

Idle Max 0.002948

Tot Bwd Pkts 0.002925

Pkt Len Min 0.002917

URG Flag Cnt 0.001940

Down/Up Ratio 0.001858

Fwd Act Data Pkts 0.001264

RST Flag Cnt 0.001100

ECE Flag Cnt 0.001051

SYN Flag Cnt 0.000833

Fwd URG Flags 0.000653

Fwd PSH Flags 0.000633

CWE Flag Count 0.000632

Protocol 0.000541

FIN Flag Cnt 0.000307

Fwd Pkts/b Avg 0.000000

Fwd Blk Rate Avg 0.000000

Fwd Byts/b Avg 0.000000

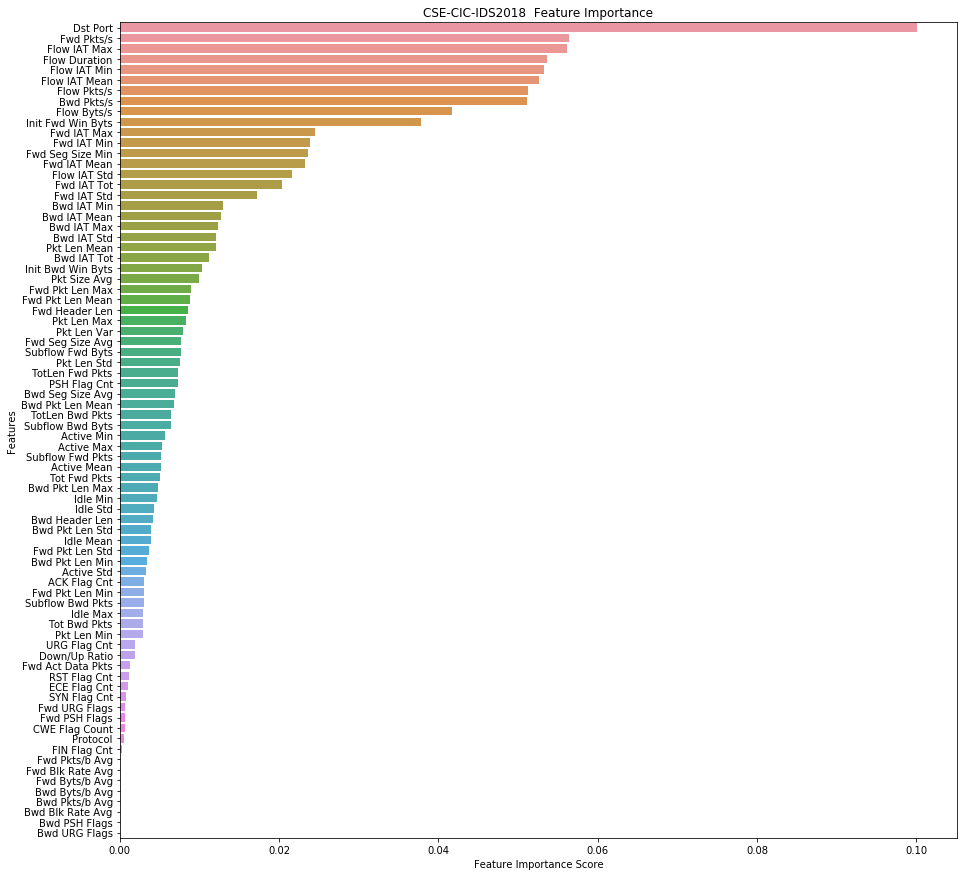
Bwd Byts/b Avg 0.000000

Bwd Pkts/b Avg 0.000000

Bwd Blk Rate Avg 0.000000

Bwd PSH Flags 0.000000

Bwd URG Flags 0.000000



We see that for this subset of the CSE-CIC-IDS2018 dataset that we obtain a lower accuracy on the classifier prediction than with the KDD 1999 Cup Data. Here, the accuracy is 75% versus 99.9% for KDD. The subset we are using is in the file “Thursday-01-03-2018\_TrafficForML\_CICFlowMeter.csv”. These are the Infiltration attacks. We will need to test the accuracy for other types of attacks.

We see that one feature is significantly more relevant to the model than the other features. However, there is almost a normal distribution among the feature importance scores. We can likely eliminate 10 or more features from the model and still obtain good results.

## Analytics Methods

Now that we have done the data preprocessing and dimensionality /feature reduction analysis, we want to select a model that will give a good prediction of attack versus normal data. We could test a large number of different machine learning models and test the accuracy of each model. By looking at our dataset, we can make an educated guess at which types of models will perform well in terms of accuracy and runtime. We know we have a large number of features or mostly numerical data and some categorical data. Linear and logistic regression models are not able to handle large number of categorical features/variables. They are also vulnerable to overfitting.

We know from our Random Forest Feature Reduction analysis that decision tree models work relatively well with this data. The academic literature also supports this hypothesis with numerous applications of decision trees are intrusion detection.

We also know from the literature that clustering algorithms work well with intrusion detection applications. The dataset we are testing also have the characteristics that are conducive to clustering. Clustering algorithms handle categorical data well and don’t have the typical overfitting problems with regression and SVM models. We will test the datasets with k-Nearest Neighbor algorithm, which is one of the most popular clustering algorithms. We expect good results from clustering in terms of predictive accuracy and runtime.

## Plans for interrupting results

# Results and Discussion

## Decision Tree Model

### KDD Cup 1999 Data

The following are the results of testing a Decision Tree Model for the KDD Cup 1999 Data with 80% training data and 20% test data.

precision recall f1-score support

back. 1.00 1.00 1.00 467

buffer\_overflow. 0.67 0.67 0.67 9

ftp\_write. 1.00 0.50 0.67 4

guess\_passwd. 1.00 0.83 0.91 12

imap. 0.75 0.75 0.75 4

ipsweep. 1.00 1.00 1.00 2443

land. 1.00 0.75 0.86 4

loadmodule. 0.33 0.50 0.40 2

multihop. 0.00 0.00 0.00 1

neptune. 1.00 1.00 1.00 214905

nmap. 0.99 0.99 0.99 452

normal. 1.00 1.00 1.00 194337

perl. 1.00 1.00 1.00 2

phf. 1.00 1.00 1.00 2

pod. 0.97 1.00 0.98 56

portsweep. 1.00 1.00 1.00 2015

rootkit. 0.00 0.00 0.00 0

satan. 1.00 1.00 1.00 3104

smurf. 1.00 1.00 1.00 561466

teardrop. 1.00 1.00 1.00 204

warezclient. 0.96 0.98 0.97 195

warezmaster. 0.75 1.00 0.86 3

accuracy 1.00 979687

macro avg 0.84 0.82 0.82 979687

weighted avg 1.00 1.00 1.00 979687

We see that for all the attack labels that had an adequate number of samples, the prediction accuracy was over 99.9% precision. The 3 worst performing predictions were for loadmodule, multihop, and rootkit which only had 2, 1, and 0 samples respectively. Even on low samples, the model was able to make a respectable prediction with buffer\_overflow at 67% precision with 9 samples and imap at 75% precision with 4 samples. There were several other examples of this accuracy at low sample number. With a high sample numbers, the model was extremely accurate with greater than 99.9% precision and recall.

### CSE-CIC-IDS2018 Data

The following are the results of testing a Decision Tree Model for the CSE-CIC-IDS2018 Data with 80% training data and 20% test data. The file used is “Thursday-01-03-2018\_TrafficForML\_CICFlowMeter.csv”.

precision recall f1-score support

Benign 0.79 0.81 0.80 47240

Infilteration 0.48 0.44 0.46 18397

accuracy 0.71 65637

macro avg 0.63 0.63 0.63 65637

weighted avg 0.70 0.71 0.70 65637

We see that in this data subset, the Decision Tree Model did reasonably well at classifying Benign with roughly an 80% accuracy. The question is whether this level of accuracy is adequate for intrusion detection. At this level of accuracy we have roughly a 1 in 5 false positive and 1 in 5 false negative. The number of sample is seems high enough such that additional training data likely would not help the accuracy of the model.

These results are testing the model against the file “Thursday-01-03-2018\_TrafficForML\_CICFlowMeter.csv”. Unlike the KDD dataset, this only contains one type of attack label. We saw that with KDD, the model accuracy varied with the type of attack. Although this is mainly due to the sample size of the different attacks. We will look at the Decision Tree model against a different type of attack in a different subset of data from the CSE-CIC-IDS2018 Data.

Using the file “Thursday-15-02-2018\_TrafficForML\_CICFlowMeter.csv “, the following are the results of testing a Decision Tree Model for the CSE-CIC-IDS2018 Data with 80% training data and 20% test data.

precision recall f1-score support

Benign 1.00 1.00 1.00 197791

DoS attacks-GoldenEye 1.00 1.00 1.00 8096

DoS attacks-Slowloris 1.00 1.00 1.00 2223

accuracy 1.00 208110

macro avg 1.00 1.00 1.00 208110

weighted avg 1.00 1.00 1.00 208110

We see in this file, which has attack labels for 2 different types of attacks: DoS attacks-GoldenEye and DoS attacks-Slowloris, the Decision Tree Model was greater than 99.9% accurate at classifying Benign and attack data.

We see a large disparity between the datasets for Infiltration type attacks versus Denial of Service (Dos) type attacks. By running the model on different subsets we are essentially training 2 separate models, one to detect Infiltration and one to detect DoS. With the KDD dataset the data was combined for all attack types therefore we trained only one model for all classification. Let us test what happens when we combine the different attack types into one dataset and train a single model for classification

These following results are testing the model against the file “Thursday-01-03-2018\_TrafficForML\_CICFlowMeter.csv” and file “Thursday-15-02-2018\_TrafficForML\_CICFlowMeter.csv “ combined. The following are the results of testing a Decision Tree Model for the combined files with 80% training data and 20% test data.

precision recall f1-score support

Benign 0.95 0.96 0.96 244569

DoS attacks-GoldenEye 1.00 1.00 1.00 8439

DoS attacks-Slowloris 1.00 1.00 1.00 2200

Infilteration 0.42 0.37 0.39 18538

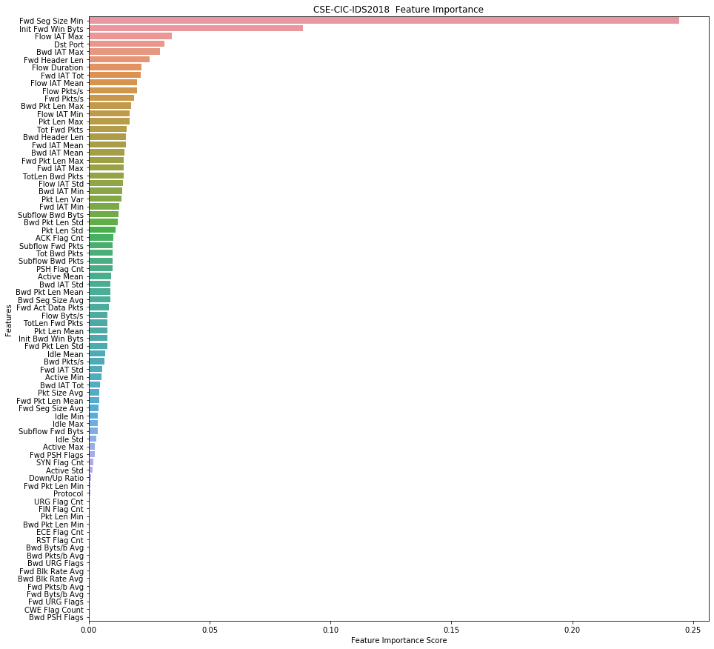
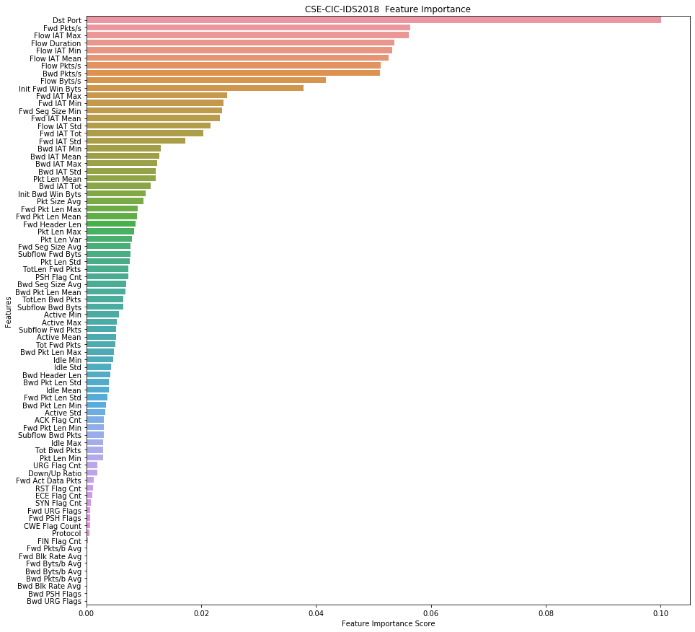
accuracy 0.92 273746

macro avg 0.84 0.83 0.84 273746

weighted avg 0.92 0.92 0.92 273746

We see that when we combine the data and train a single Decision Tree model, the accuracy of classifying DoS attacks is unchanged. However, the accuracy of classifying Infiltration attacks is substantially degraded from running it against the model train specifically for Infiltration attacks. This is likely due to overfitting of certain features that are not relevant to Infiltration attack detection.

Let us revisit the Random Forest Analysis of the CSE-CIC-IDS2018 Data. We will re-run the Random Forest against the DoS attack data subset. The following are the Feature Importance scores side by side:



We can clearly see that for Infiltration attacks (on left) and for DoS attacks (on right), the features that help classify the data are significantly different. Therefore, the classification model that has to predict both types of data will expectedly perform worse overall.

## K Nearest Neighbor Model

### KDD Cup 1999 Data

The following are the results of testing a k-Nearest Neighbor Model (kNN) for the KDD Cup 1999 Data with 80% training data and 20% test data.

precision recall f1-score support

back. 1.00 0.99 0.99 438

buffer\_overflow. 0.33 0.33 0.33 3

ftp\_write. 0.00 0.00 0.00 1

guess\_passwd. 1.00 1.00 1.00 8

imap. 0.00 0.00 0.00 1

ipsweep. 0.99 0.99 0.99 2534

land. 0.50 0.33 0.40 3

loadmodule. 0.00 0.00 0.00 3

neptune. 1.00 1.00 1.00 215040

nmap. 0.99 0.98 0.99 476

normal. 1.00 1.00 1.00 194639

perl. 0.50 1.00 0.67 1

pod. 1.00 1.00 1.00 56

portsweep. 1.00 1.00 1.00 2069

rootkit. 0.00 0.00 0.00 3

satan. 1.00 0.99 1.00 3188

smurf. 1.00 1.00 1.00 560820

teardrop. 1.00 1.00 1.00 198

warezclient. 0.92 0.94 0.93 203

warezmaster. 1.00 0.67 0.80 3

accuracy 1.00 979687

macro avg 0.71 0.71 0.70 979687

weighted avg 1.00 1.00 1.00 979687

We see that for all the attack labels that had an adequate number of samples, the prediction accuracy for the kNN Model was over 99.9% precision. The results were roughly inline with results from the Decision Tree Model.

### CSE-CIC-IDS2018 Data

The following are the results of testing a k-Nearest Neighbor Model (kNN) for the CSE-CIC-IDS2018 Data with 80% training data and 20% test data. The file used is “Thursday-01-03-2018\_TrafficForML\_CICFlowMeter.csv”. The n\_neighbors = 5.

precision recall f1-score support

Benign 0.79 0.90 0.84 47134

Infilteration 0.60 0.38 0.46 18503

accuracy 0.75 65637

macro avg 0.69 0.64 0.65 65637

weighted avg 0.73 0.75 0.73 65637

We see that the kNN Model for classifying Infiltration attacks performed significantly better than the Decision Tree Model for the same attack. The accuracy of classifying Benign data had similar precision with slightly better recall, 90% for kNN versus 81% for Decision Tree. This is likely due to fewer false negatives for kNN. For the attack data, kNN had a precision of 60% versus 48% for Decision Tree. This means that kNN is better at identifying true positives. For recall, kNN performed slightly worse than Decision Tree, 38% versus 44% respectively. This is likely due to more False Negatives for kNN.

The disparity in performance poses an interesting question, would it be better to allow false positives or false negatives? For false positives we have good traffic that is being blocked. For false negatives we have bad traffic that is not being blocked. For the data scientist it is academic but for a network security engineer it becomes a practical consideration.

The following are the results of testing a k-Nearest Neighbor Model for the CSE-CIC-IDS2018 Data with 80% training data and 20% test data. The file used is “Thursday-15-02-2018\_TrafficForML\_CICFlowMeter.csv “. The n\_neighbors = 5.

precision recall f1-score support

Benign 1.00 1.00 1.00 197559

DoS attacks-GoldenEye 1.00 1.00 1.00 8389

DoS attacks-Slowloris 1.00 1.00 1.00 2162

accuracy 1.00 208110

macro avg 1.00 1.00 1.00 208110

weighted avg 1.00 1.00 1.00 208110

Here we see the kNN performs equally well compared to Decision Tree. Both models are able to classify Benign versus 2 different types of DoS attack with almost 100% accuracy.

Let us now combine the datasets for the Infiltration and DoS attacks and test the kNN Model. The following results are testing the model against the file “Thursday-01-03-2018\_TrafficForML\_CICFlowMeter.csv” and file “Thursday-15-02-2018\_TrafficForML\_CICFlowMeter.csv “ combined. The following are the results of testing a k-Nearest Neighbor Model for the combined files with 80% training data and 20% test data. The n\_neighbors = 5.

precision recall f1-score support

Benign 0.95 0.99 0.97 244620

DoS attacks-GoldenEye 1.00 1.00 1.00 8302

DoS attacks-Slowloris 1.00 1.00 1.00 2189

Infilteration 0.68 0.32 0.44 18635

accuracy 0.94 273746

macro avg 0.91 0.83 0.85 273746

weighted avg 0.93 0.94 0.93 273746

We see that the kNN Model still was able to classify DoS with near 100% accuracy. However, unlike with Decision Tree where combining the attack types caused the model to perform worse at classifying Infiltration attacks, the kNN Model actually performed better with combined data at 68% precision for classifying Infiltration versus 60% precision when run against only Infiltration data. Recall was slightly worse, 32% versus 38%. This means with the combined data we are getting less false positives but more false negatives. In practical terms, we are letting more attacks slip through with kNN on combined attack data.

# Conclusion

In summary, this project tested 2 different machine learning algorithms for classifying data and 2 different data sets on network intrusion detection from different eras. We saw that with the legacy data set, KDD 1999 Cup Data, both our classification models, Decision Tree and k-Nearest Neighbor, performed equally well with almost 100% accuracy. With the modern dataset, CSE-CIC-IDS2018 Data, we saw a much more varied performance between different attack types that we were attempting to classify and varied performance between classification algorithms. The results would tend to validate anecdotal evidence that the KDD 199 Cup data is too simplistic to test real world performance of machine learning algorithms for intrusion detection. The results show that machine learning algorithms are still very valid for intrusion detection but it really depends on the attack type and the specific algorithm being used. The datasets are almost two decades apart. The Internet and computer networks have drastically changed over that time. It seems fitting that this represents the complexity and dynamic nature of modern computer networks.

# References

## Academic References

1. Porto-Diaz, Iago, Martiınez-Rego, David, Alonso-Betanzos, Amparo, & Fontenla-Romero, Oscar. Combining Feature Selection and Local Modelling in the KDD Cup 99 Dataset. Department of Computer Science, University of A Coruna, Spain
2. Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99. H. G. Kayacik, A. N. Zincir-Heywood, M. Heywood. Published 2005.
3. Devi, R. Rama, Abualkibash, Munther. Intrusion Detection System Classification Using Different Machine Learning Algorithms on KDD-99 and NSL-KDD Datasets - A Review Paper. Published 2019. Computer Science International Journal of Computer Science and Information Technology.
4. Wang, Yan\*, Yang, Kun, Jing, Xiang, Jin, Huang Long. Problems of KDD Cup 99 Dataset Existed and Data Preprocessing. Published 2014. Applied Mechanics and Materials (Volume 667)
5. Mao, J. K., & Zhan, F. (2015). Study on Intrusion Detection System Based on Data Mining. Applied Mechanics and Materials, 713–715, 2499–2502. https://doi.org/10.4028/www.scientific.net/amm.713-715.2499
6. Luo, Y. J., Meng, J., & Zhao, H. Y. (2014). Research on Intrusion Detection Technology Based on Ad Hoc Network. Applied Mechanics and Materials, 687–691, 2622–2625. https://doi.org/10.4028/www.scientific.net/amm.687-691.2622
7. Kumar, Satish, Sunanda, Arora, Sakshi. A Statistical Analysis on KDD Cup’99 Dataset for the Network Intrusion Detection System. Part of the Lecture Notes in Networks and Systems book series (LNNS, volume 125).
8. Cai, Ying and Wang, Xiaofei. The Analysis and Optimization of KNN Algorithm Space-Time Efficiency for Chinese Text Categorization. Dept. of Computer Science and Technology, Beijing Information Science & Technology University Beijing, 100101, P. R. China
9. Bubeck, Sebastien, von Luxburg, Ulrike. Nearest Neighbor Clustering: A Baseline Method for Consistent Clustering with Arbitrary Objective Functions. Journal of Machine Learning Research 10 (2009) 657-698.
10. P. H. Swain and H. Hauska, "The decision tree classifier: Design and potential," in IEEE Transactions on Geoscience Electronics, vol. 15, no. 3, pp. 142-147, July 1977, doi: 10.1109/TGE.1977.6498972.
11. Han, Jiawei, Kamber, Micheline, Pei, Jian. Data Mining: Concepts and Techniques (The Morgan Kaufmann Series in Data Management Systems) 3rd Edition.
12. Aggarwal, Charu C. Data Mining: The Textbook 2015th Edition.

## Non-Academic Articles

1. 3 New Techniques for Data-Dimensionality Reduction in Machine Learning. <https://thenewstack.io/3-new-techniques-for-data-dimensionality-reduction-in-machine-learning/>
2. Building an Intrusion Detection System using KDD Cup’99 Dataset. <https://medium.com/analytics-vidhya/building-an-intrusion-detection-model-using-kdd-cup99-dataset-fb4cba4189ed>
3. Decision Trees in Python with Scikit-Learn. <https://stackabuse.com/decision-trees-in-python-with-scikit-learn/>

## Data Sets Used

A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018) <https://registry.opendata.aws/cse-cic-ids2018/>

KDD Cup 1999 Data The UCI KDD Archive, Information and Computer Science, University of California, Irvine <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>

## Code Repository

Code developed for this project can be found at this repository:

<https://github.com/nnvan/CS296>

## Endnotes

1. <https://www.grandviewresearch.com/industry-analysis/cyber-security-market> [↑](#endnote-ref-1)
2. <https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016> [↑](#endnote-ref-2)