DDOS DETECTION USING MACHINE LEARNING

BY

Akshat Garg <ag2193@rit.edu>

Bharadwaj Sharma Kasturi < bk5953@rit.edu>

Megha Gupta <mg9428@rit.edu>

Shaista Syeda <ss7810@rit.edu>

Index

Contents

Introduction

Advances in technology have led millions of people to connect in some form of network and exchange critical data. Therefore, the need for security to protect the integrity and confidentiality of data is rapidly increasing. Efforts have been made to protect data transmissions, but attack technologies to infiltrate networks have continued to be developed simultaneously. Therefore, there is a need for a system that can adapt to these ever-changing attack techniques. In this paper, we have developed a system based on machine learning. Our goal is to find a suitable machine-learning algorithm to predict network attacks with the highest accuracy and establish a system to detect network intrusions using this algorithm. The algorithms compared are Naive Bayes, Decision Tables, K-nearest Neighbours, Random Forest, and AdaBoost. The dataset used to train the model is the KDD99 dataset. The reason I used machine learning is to give the system flexibility. For example, if a new type of attack is developed in the future, you can train your system to predict that attack. There are several types of intrusion detection systems, but our system is a knowledge-based intrusion detection system, also known as an anomaly-based system. Register anomalies and predict that such malicious networks will send alerts in the future. In this way, the network can be disconnected from such connections, and only secure connections are possible.

Problem Description

Information technology has advanced at a breakneck pace in the last two decades. Industry, business, and different aspects of human life all use computer networks. As a result, IT managers must focus on establishing reliable networks. On the other hand, the rapid advancement of information technology has created various problems in the laborious process of constructing trustworthy networks. Computer networks are vulnerable to various threats that jeopardize their availability, integrity, and confidentiality. One of the most widespread destructive attacks is the Denial-of-Service attack.

Data Research

This is the data set used for The Third International Knowledge Discovery and Data Mining Tools Competition. The task was to build a predictive model capable of distinguishing between ``bad'' connections, called intrusions or attacks, and “good” normal connections. The raw training data was about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. This was processed into about five million connection records. Similarly, the two weeks of test data yielded around two million connection records.

It is important to note that the test data is not from the same probability distribution as the training data, and it includes specific attack types not in the training data. This makes the task more realistic. Some intrusion experts believe that most novel attacks are variants of known attacks, and the “signature” of known attacks can be sufficient to catch novel variants. The datasets contain a total of 24 training attack types, with an additional 14 types in the test data only. There was a data quality issue with the labels of the test data, also there was high imbalance in the data. Finally, we learnt a classification model capable of distinguishing between legitimate and illegitimate connections in a computer network.

Literature Review

Being a classification problem, we came up with a supervised learning algorithm. Firstly, the class imbalance problem was overcome using a combination of oversampling as well as under sampling. When the skewness of the data was taken care of, various machine learning models were fed with the data to yield an efficient and usable yield. Algorithms like Regression, Naive Bayes, Decision Trees, Random Forest, Isolation Tree, XGBoost were utilized, and conclusions were drawn considering the test scores like recall, precision, and accuracy.

Analysis strategy

After studying the distribution of data, we identified that there were different subcategories of DDoS attack based on the layer of the network connection they attempt to attack. The result column had 60% of the normal data. and rest 40% attack types were unevenly distributed. So, clearly the major challenge was to handle the class imbalance problem. Our approach was to implement different sampling techniques to get the classes balanced. Since, for class imbalance problems, accuracy is not an appropriate metric for model evaluation because the accuracy score would be high and heavily biased towards the majority classes (normal class for KDDCup dataset). Hence our main goal was to identify the minority class (attack sub-classes). So, we focused on precision-recall and FPR (fallout rate) for the evaluation of the machine learning models. In other words, our aim was to minimize a bad connection that gets classified as normal.

Analysis code

1. Data Exploration

Since our objective was to cover the majority of the attack types, we combined the test and train data from the KDDCup dataset.

1. Identified the dataset for the null values. We found that there were no null values.
2. Checked for the duplicates in the data frame, around 70% of the data was duplicated so we dropped this.
3. Analysed the dataset's attributes and worked upon the numerical and categorical features individually.

2. Feature Selection

After plotting the correlation matrix, there was a total of nine pairs of highly correlated features; we selected one from each team. After which, there were a total of 32 numerical attributes.

3. Data Pre-processing

1. Numerical attributes: total count=32

   We standardized the numerical attributes, which had a range greater than 1.

2. Categorical attributes: total count=3 (service, flag, protocol type)

   For the columns, service and flag had a high number of subcategories. On converting numerical value using one-hot encoding result would have resulted in adding a column per subcategory. In this case it would result in adding 66 + 11 + 3 - 3 = 77 columns. This would have added to the complexity of the model. Hence, we used baseN encoding, which highly reduces the dimensionality as the value of N increases.

3. We used SMOTE (Synthetic Minority Oversampling Technique) for balancing the classes.

4. Model Selection

We used the following algorithms for training the model and hyper-tuned them.

1. Decision tree
2. Naïve Bayes
3. Random forest
4. Logistic regression

5. Model Comparison

Hyper tuned Random forest performed the best.

Work Planning and organization of each team member

We devised this project as an opportunity to apply the techniques we have learned in the class and to broaden our knowledge on each component. Every member of the group contributed individually to this project. According to them, whatever method was most effective was used either in Data pre-processing, Data Cleaning, or Feature Selection.

Everyone mastered the techniques they worked on and taught them to others as well.

We collectively chose the best option to improve our model once everyone had finished their parts.

Improving teamwork and collaboration

From the initial steps of the project, we as a team came up with our own inputs and had them discussed with the teammates in the weekly meetings. The most optimal methodology among the proposed ideas was considered and gave us exposure to how a specific task could be tackled in different ways. This collaborative approach has helped us to learn collectively and help us have end-to-end knowledge of the project.

Individual Contribution

Firstly, I extracted the dataset that contained the train and train dataset separately, added column names to it, and checked the size of the data. It had more than 4 million data points in the original train and test. So, we decided to use only 10% of the actual data and work on it.

I combined both the train and test dataset, having a total of more than 800000 data points, due to class imbalance, and ran exploratory data analysis and found there were many duplicate values, so I decided to remove all the duplicates, which reduced the data set by approximately 70%. Then I built a bar plot for the "result" column, which contains our result, before and after dropping duplicates.

Then after my teammates and I completed the exploratory analysis, I worked on One Hot encoder, but this resulted in the addition of too many columns (from 42, it went up to 77), hence increasing the complexity of the dataset. After converting all categorical columns to numeric, I split the dataset, firstly using "train_test_split," but this resulted in a few attack types not being present in the train set or test set, decreasing the accuracy. So, I used Stratified Shuffle split to ensure that we have every attack in both train and test sets.

The next thing I was working on was the class imbalance problem; since we had few attacks with more than minor 10 data points, I first tried to convert data into a binary classification problem by merging all types of attack as "0" and stable connection as "1". Then applied oversampling techniques which led to high accuracy, precision, recall, and f1-score, but that was not solving our problem as we were trying to find how are we attacked and by which one. So, we went back to the multi-classification problem and applied SMOTE technique to balance the class. This led to the balanced train set. I then standardized the train and test data.

I worked on the Logistic Regression model without any tuning and found the results out and decided whether it is a good model or not to predict the intrusion. Compared to other models, it was not a good fit because of its low precision and recall value.