CSC 7210 – Anomaly and Intrusion Detection Systems Fall 2020

Project Assignment: Intrusion Detection

Overview

As computers and the Internet become ubiquitous in our everyday lives, malicious activities in the cyberspace have increased significantly. Intrusion detection is an area of computer security that focuses on detecting these attacks reliably. Intrusion detection systems (IDS) usually have a knowledge base containing rules that characterize attacks. Building such a knowledge base manually can be time consuming. However, machine learning can help build such a knowledge base in a more efficient manner. In order to detect attacks, we need to differentiate between instances of normal and attack behavior. Based on previous instances of normal and attack behavior, a machine learning algorithm can gain the knowledge on how to differentiate between the two types of behavior and represent the knowledge in a form than can be used to predict if current instances are malicious or not.

Objectives

This project aims to apply machine learning techniques for detecting attacks/intrusions. More specifically, the objectives are:

- machine learning can be achieved from historical data (experience)
- machine learning algorithms can be applied to computer security
- understanding the learning task of trying to detect attacks
- understanding a decision-tree learning algorithm
- a better understanding of search and knowledge representation
- evaluation of machine learning algorithms

Project Description

Over the last decade, malicious activities in the cyberspace have increased significantly. Intrusion detection is an area of computer security that focuses on detecting these attacks reliably. Intrusion detection systems (IDS) usually have a knowledge base containing rules that characterize attacks. Building such a knowledge base manually can be time consuming. Machine learning can help build such a knowledge base in a more efficient manner.

In order to detect attacks, we need to differentiate between instances of normal and attack behavior. Based on previous instances of normal and attack behavior, a machine learning algorithm can gain the knowledge on how to differentiate between the two types of behavior and represent the knowledge in a form than can be used to predict if current instances are malicious or not.

For this project, **you will need to implement**[1] the following decision-tree learning algorithm (also found in Russell and Norvig's book "Artificial Intelligence, A Modern Approach"):

[1] You must implement the algorithm yourself – you are not allowed to use software/code from another source – that would be plagiarism!

```
function DECISION-TREE-LEARNING(examples, attributes, default)
returns a decision tree
   inputs: examples, set of examples
           attributes, set of attributes
           default, default value for the goal predicate

   if examples is empty then return default
   else if all examples have the same classification
        then return the classification
   else if attributes is empty
        then return MAJORITY-VALUE(examples)
   else
        best ← CHOOSE-ATTRIBUTE(attributes, examples)
        tree ← a new decision tree with root test best
        for each value v_i of best do
             examples_i ← {elements of examples with best = v_i}
             subtree ← DECISION-TREE-LEARNING(examples_i,
                       attributes − best, MAJORITY-VALUE(examples))
             add a branch to tree with label v_i and subtree subtree
        end
        return tree
```

You will then evaluate the accuracy of the algorithm on the provided training and test sets (described below).

1. Input to your program:
   - file name of the attribute description,
   - file name of the training set, and
   - file name of the test set.
2. Output from your program:
   - the tree using pre-order traversal with more indentation for nodes at deeper levels,
   - accuracy of the tree on the training set, and
   - accuracy of the tree on the (unseen) test set.

IDS Data Set

The IDS data set contains records of network activities that are normal or part of a denial of service (DOS) attack(s) called Neptune (aka SYN-flood). Neptune tries to make many "half" connections to a server. Due to limited resources, a server usually has a maximum number of connections that it can handle. Many malicious "half" connections can prevent legitimate connections to be made. That is, the server might be filled with useless "half" connections, and cannot accept legitimate connections and provide the intended service (hence "denial of service"). The provided data set is adapted from the much larger KDD Cup Data set (http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html). All values in the data set have been converted into discrete values.

Files for the data set (provided to you as a single zip file):

- Attribute description: ids-attr.txt
- Training set: ids-train.txt (800 records)
- Test set: ids-test.txt (200 records)

Submission

For this assignment, you must submit the following:

1. (30 points) Source code of your program
2. (10 points) Executable of your program (runnable in either Windows or Unix)
3. (20 points) Output from running your program with the provided data set.
4. (40 points) Report (2-3 pages) that includes a discussion of your experiences creating decision-tree learning software, and in general, with the decision-tree learner in terms of the inputs, outputs, and performance.

Submit the above **as a single zip file**, by **CLASS TIME ON THURSDAY OCTOBER 15**.