Computer Security: A Machine Learning Approach

We analyze two learning algorithms, NBTree and VFI, for the task of detecting intrusions.

SANDEEP V. SABNANI AND ANDREAS FUCHSBERGER

Royal Holloway
University of London

ABSTRACT

Information Security is one of the key areas today as securing computers especially against novel attacks becomes a daunting task. Intrusion detection is a method by which unauthorised access to one's assets is detected. In this paper, we present an application of the field of machine learning to computer security, particularly to intrusion detection. We analyse two learning algorithms (NBTree and VFI) for the task of detecting intrusions and compare their relative performances. We then comment on the suitability of NBTree algorithm over VFI for the intrusion detection task based on its high accuracy and high recall. We finally state the usefulness of machine learning to the field of computer security.

Computer Security: A Machine Learning Approach

Sandeep V. Sabnani

Information Security Group
Royal Holloway, University of London
Egham, Surrey, TW20 0EX,
United Kingdom
s.sabnani@rhul.ac.uk

Andreas Fuchsberger

Information Security Group
Royal Holloway, University of London
Egham, Surrey, TW20 0EX,
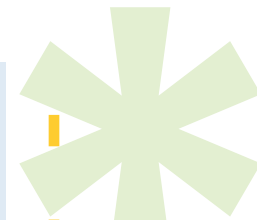United Kingdom
a.fuchsberger@rhul.ac.uk

1. INTRODUCTION

Computer security has become a challenging task these days with the rapid growth of the internet and the increasing complexity of communication protocols. New and complicated attack methods are being constantly developed by attackers thereby compromising the confidentiality, integrity and/or availability of one's data. CERT has reported 8064 new vulnerabilities in the year 2006 and the number has been increasing significantly over the past few years [1].

There have been quite a few approaches which prevent and/or detect *known* attacks. *Novel* or *unknown* attacks on the other hand are more difficult to detect and have received considerable attention in the recent past.

Another important problem in the field of computer security has been that of *insider* threats. Many of the insider threats may be unintentional, nevertheless it has become essential to ensure that insider behaviour is in sync with the security policy of the organisation.

These issues can prove to be quite expen-

sive for organisations to handle. The task of detecting intrusions involves the formulation of efficient rules which require a high level of domain expertise and analysis of large amounts of data, which might make the process slow and unreliable with human experts. For checking compliance with a security policy, an administrator may have to be extremely cautious as normal user behaviour may change over time.

Machine learning is a field related to artificial intelligence which deals with constructing computer programs that automatically improve with experience [12]. The *learning experience* is provided in the form of data and actual learning is achieved with the help of algorithms. The two main tasks that are addressed by machine learning are the ability to *learn* more about the given data and to *make predictions* about new data based on learning outcomes from the learning experience [9]; both of which are difficult and time-consuming for human analysts. Machine learning is thus, well-suited to problems that depend on rare, expensive and unreliable *human* experts.

This paper presents the intrusion detection problem and a machine learning based solution to it.

2. MACHINE LEARNING
2.1 Basic Concepts

Learning can be described in many ways including acquisition of new knowledge, enhancement of existing knowledge, representation of knowledge, organisation of knowledge and discovery of facts through experiments [11]. When such learning is performed with the help of computer programs, it is referred to as *machine learning*.

Every computer action can be modeled as a function with sets of inputs and outputs. A learning task may be considered as the estimation of this function by observing the sets of inputs and outputs. (We use the term estimation as the exact function may not be determinate.) The function estimating process usually consists of a *search in the hypothesis space* (i.e. the space of all such possible functions that might represent the input and output sets under consideration).

The authors in [14] formally describe the function approximation process. Consider a set of input instances $X = (x_1, x_2, x_3 \cdots x_n)$. Let f be a function which is to be guessed by the learner. Let h be the learner's hypothesis about f.

Also, we assume a priori that both f and h belong to a class of functions H. The function

> Every computer action can be modeled as a function with sets of inputs and outputs.

f maps the input instances in *X* as,

$$X \overset{h \in H}{\to} h(X)$$

A machine learning task may thus be defined as a search in this space *H*. This search results in *approximating* the relevant *h*, based on the training instances (i.e. the set *X*).

The approximation is then checked against a set of test instances which are then used to indicate the correctness of *h*. The search requires algorithms which are efficient and which best-fit the training data [12].

2.2 Inputs and Outputs

The inputs and outputs to a machine learning task may be of different kinds. Generally, they are in the form of numeric or nominal attributes. For instance, an attribute like *temperature* if used as a numeric attribute, may have values like 25o C, 28o C, etc. On the other hand, if it is used as a nominal attribute, it may take values from a fixed set (like *high, medium, low*). In many cases, the output may also be a boolean value (like *yes and no*).

2.3 Production of Knowledge

The way in which knowledge is learned is an important issue for machine learning. The learning element may be trained in different ways [3]. For classification problems like intrusion detection, knowledge may be learned in a *supervised*, *unsupervised* or *semi-supervised* manner. In this paper, we use supervised learning in which the learner is provided with training examples with the associated classes or values for the attribute to be predicted.
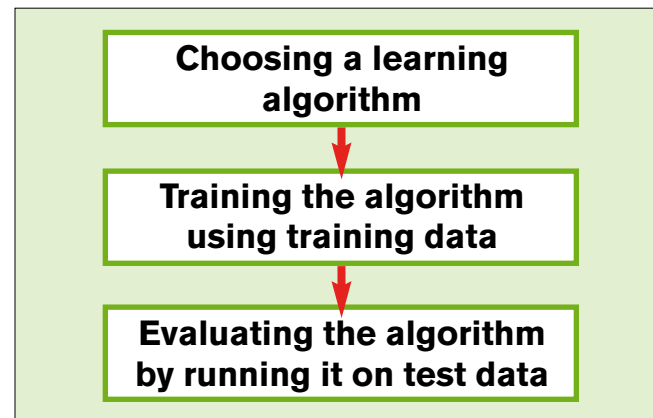

Choosing a learning algorithm

Training the algorithm using training data

Evaluating the algorithm by running it on test data

Figure 1. Life Cycle of a Machine Learning Task

2.4 Defining a Machine Learning Task

In general, a machine learning task can be defined formally in terms of three elements, viz. the learning experience E, the tasks T and the performance element P.

TM Mitchell in Machine Learning [12]

The inputs and outputs to a machine learning task may be of different kinds.

defines a learning task more precisely as follows:

"A computer program is said to learn from *experience E* with respect to some class of *tasks T* and *performance measure P*, if its performance at tasks in *T*, as measured by *P*, improves with experience *E*."

2.5 Life Cycle of a Machine Learning Task

Figure 1 shows the life cycle of a machine learning task. Depending on the nature of the knowledge to be learned, different types of algorithms may be chosen at different times. Also, the type of inputs and outputs are also instrumental in choosing an algorithm.

Once the algorithm is selected, the next step is to train the algorithm by providing it with a set of *training* instances. The training instances are used to build a model that represents the target concept to be learned (i.e. the hypothesis). This model is then evaluated using the set of test instances.

In the case where a large amount of data is available, the general approach is to construct two independent sets, one for training and the other for testing. On the other hand, if a limited amount of data is available, it

becomes difficult to create separate sets for training and testing. In such cases, some data might be held over for testing, and the remaining used for training. This is called the holdout procedure [20]. However, the data in this case might be distributed in an uneven way in the training and test sets and might not represent the output classes in the correct proportions. *Stratification* [4] and *cross-validation* [5] can be used to circumvent this problem.

2.6 Benefits of Machine Learning

The field of machine learning has been found to be extremely useful in the following areas relating to software engineering [12]:

1. Data mining problems where large databases may contain valuable implicit regularities that can be discovered automatically.

2. Difficult to understand domains where humans might not have the knowledge to develop effective algorithms.

3. Domains in which the program is required to adapt to dynamic conditions.

In the case of traditional intrusion detection systems, the alerts generated are analysed by human analysts who evaluate them and take suitable actions. However,

Once the algorithm is selected, the next step is to train the algorithm by providing it with a set of training instances.

this is an extremely onerous task as the number of alerts generated may be quite large and the environment may change continuously [16]. This makes machine learning well suited for intrusion detection.

3. MACHINE LEARNING APPLIED TO COMPUTER SECURITY

3.1 Intrusion Detection as a Machine Learning Task

A machine learning task can be formally defined as shown in section 2.4. We use this notation to formulate intrusion detection as a machine learning task.

Thus, for intrusion detection, we have,

1. **Task:** To detect intrusions in an accurate manner.

2. **Experience:** A dataset with instances representing normal as well as attack data.

3. **Performance Measure:** Accuracy in terms of correct classification of intrusion events and normal events and other statistical metrics including precision, recall, F-measure and kappa statistic which are described in section 3.5.

3.2 Data Set Description

The data set used for evaluation in this paper is a subset of the KDD Cup '99 data set for intrusion detection obtained from the UCI machine learning repository. The KDD Cup '99 data set is a version of a data set used at the DARPA Intrusion Detection Evaluation program (www.ll.mit.edu/IST/ideval/data/data index.html).

The data set consists of TCP dump data for a simulated Air Force LAN. In addition to normal LAN simulation, attacks were also simulated and the corresponding TCP data was captured. The attacks were launched on three UNIX machines, Windows NT hosts and a router along with background traffic. Every record in the data set represents a TCP connection. Each connection was labeled as normal or as a specific attack type [15]. The attacks fall into one of the following categories:

- DOS attacks (Denial of Service attacks)
- R2L attacks (unauthorised access from a remote machine)
- U2R attacks (unauthorised access to super user privileges)
- Probing attacks

A detailed description and format of the dataset can be found in [17].

In addition to normal LAN simulation, attacks were also simulated and the corresponding TCP data was captured.

3.3 Algorithms

The following algorithms were used in the experiments carried out for this paper.

3.3.1 NBTree

The NBTree algorithm is a hybrid between decision-tree classifiers and Naive Bayes classifiers. It represents the learned knowledge in the form of a tree which is constructed recursively. However, the leaf nodes are Naive Bayes categorizers rather than nodes predicting a single class[6]. For continuous attributes, a threshold is chosen so as to limit the entropy measure. The utility of a node is evaluated by discretizing the data and computing the fivefold cross-validation accuracy estimation using Naive Bayes at the node. The utility of the split is the weighted sum of utility of the nodes and this depends on the number of instances that go through that node. The NBTree algorithm tries to approximate whether the generalisation accuracy of Naive Bayes at each leaf is higher than a single Naive Bayes classifier at the current node. A split is said to be significant if the relative reduction in error is greater that 5% and there are at least 30 instances in the node[6].

3.3.2 VFI

The VFI4 algorithm is a classification algorithm based on the *voting frequency intervals*. In VFI, each training instance is represented as a vector of features along with a label that represents the class of the instance. Feature intervals are then constructed for each feature. An interval represents a set of values for a given feature where the same subset of class values are observed. Thus, two adjacent intervals represent different classes.

A detailed explanation of both the above algorithms can be found in[17].

3.4 Experimental Analysis

The experiments done in this paper consist of the evaluation of the performance of NBTree and VFI algorithms for the task of classifying novel intrusions. The dataset described in section 3.2 was used in the experiments.

Weka[20], a machine learning toolkit was used for the implementation of the algorithms described in sections 3.3.1 and 3.3.2. Due to the limitation in the available memory and processing power, it was not possible to use the full dataset described in section 3.2. Instead a reduced subset was

> The NBTree algorithm is a hybrid between decision-tree classifiers and Naive Bayes classifiers.

used and 10-fold cross-validation (explained in section 2.5) was used to overcome this limitation.

3.5 Evaluation Metrics
In order to analyse and compare the performance of the above mentioned algorithms, metrics like the classification accuracy, precision, recall, F-Measure and kappa statistic were used. These metrics are derived from a basic data structure known as the *confusion matrix*. A sample confusion matrix for a two-class problem is shown in Table 1.

	Predicted Class Positive	Predicted Class Negative
Actual Class Positive	a	b
Actual Class Negative	c	d

Table 1. Confusion Matrix for a two-class problem (Expected predictions)

In this confusion matrix, the value *a* is called a *true positive* and the value *d* is called a *true* negative. The value *b* is referred to as a *false negative* and *c* is known as *false positive*.
In the context of intrusion detection, a true positive is an instance which is normal and is also classified as normal by the intrusion detector. A true negative is an instance which is an attack and is classified as an attack.

3.5.1 Classification Accuracy
Classification accuracy is the most basic measure of the performance of a learning method. It determines the percentage of correctly classified instances. From the confusion matrix, we can say that:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d}$$

This metric gives the number of instances from the dataset which are classified correctly i.e. the ratio of true positives and true negatives to the total number of instances.

3.5.2 Precision, Recall and F-Measure
Precision gives the percentage of slots in the hypothesis that are correct, whereas recall gives the percentage of reference slots for which the hypothesis is correct.

In the context of intrusion detection, a true positive is an instance which is normal and is also classified as normal by the intrusion detector.

Referring from the confusion matrix, we can define precision and recall for our purposes as [19]:

$$Precision = \frac{a}{a+c}$$

$$Recall = \frac{a}{a+b}$$

The precision of an intrusion detection learner would thus indicate the proportion of correctly classified positive instances to the total number of predicted positive instances and recall would indicate the proportion of correctly classified positive instances to the total number of actual positive instances.

The F-measure is another metric defined as the weighted harmonic mean of precision and recall [8] to address a problem identified in [7], which may be present in any classification scenario.

$$F\text{-}measure = \frac{2*Precision*Recall}{Precision+Recall}$$

3.5.3 Kappa Statistic

The Kappa statistic is used to measure the agreement between predicted and observed categorizations of a dataset, while correcting for agreements that occur by chance. It takes into account the expected

figure and deducts it from the predictor's success and expresses the result as a proportion of the total for a perfect predictor [20].

In addition to the above statistical metrics, the time taken to build the model was also considered as a performance indicator.

3.6 Attribute Selection

Attribute Selection is the process of identifying and removing much of the redundant and irrelevant information possible. The experiments conducted in this paper use the *information gain* attribute selection method which is described in [17].

3.7 Summary of Experiments

Based on the above algorithms, attribute selection methods and the type of cross-validation, the following table 2 shows a summary of the experiments conducted.

Algorithm	Feature Reduction Method	Cross Validation
NBTree	–	10-fold
VFI	–	10-fold
NBTree	Information Gain	10-fold
VFI	Information Gain	10-fold

Table 2. Summary of Experiments

4. EVALUATION

The results of the experiments described in section 3.4 are discussed in this section. A comparison between NBTree and VFI methods is also made based on the values of the metrics defined in section 3.5.

For an IDS, the accuracy indicates how correct the algorithm is in identifying normal and adversary behaviour.

Metric	Value
Time taken to build the model	1115.05s
Accuracy	99.94 %
Average Precision	90.33 %
Average Recall	92.72 %
Average F-Measure	91.14 %
Kappa Statistic	99.99 %

Table 3. Results of NBTree with all attributes

Metric	Value
Time taken to build the model	38.97s
Accuracy	99.89 %
Average Precision	94.54 %
Average Recall	90.84 %
Average F-Measure	92.28 %
Kappa Statistic	99.82 %

Table 4. Results of NBTree with selected attributes using information gain measure

Metric	Value
Time taken to build the model	0.92s
Accuracy	86.58 %
Average Precision	41.27 %
Average Recall	80.54 %
Average F-Measure	44.05 %
Kappa Statistic	79.50 %

Table 5. Results of VFI with all attributes

Metric	Value
Time taken to build the model	0.2s
Accuracy	75.81 %
Average Precision	35.71 %
Average Recall	75.82 %
Average F-Measure	37.43 %
Kappa Statistic	66.21 %

Table 6. Results of VFI with selected attributes using information gain measure

Recall would indicate the proportion of correctly classified normal instances from the total number of actual normal instances whereas precision would indicate the number of correctly classified normal instances from the total number of instances identified as normal by the IDS. The Kappa statistic is a general statistical indicator and the

For an IDS, the accuracy indicates how correct the algorithm is in identifying normal and adversary behaviour.

F-Measure is related to the problem mentioned in section 3.5.2. In addition to these, the time taken by the learning algorithm for model construction is also important as it may have to handle extremely large amounts of data.
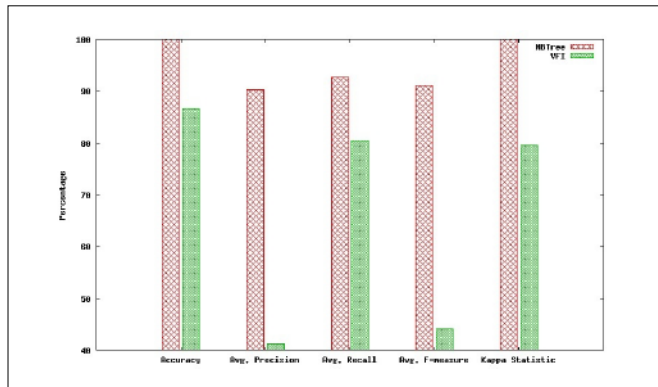

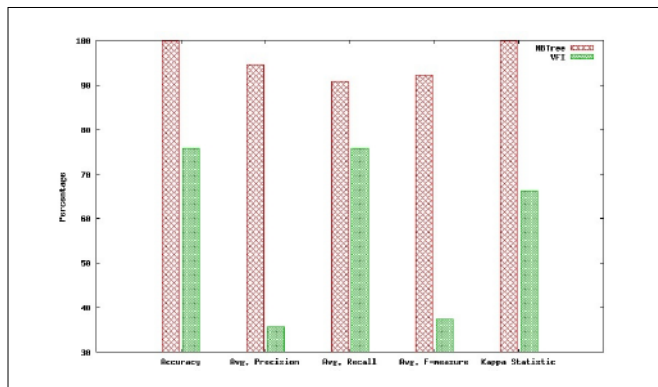**Figure 2. NBTree v/s VFI - All Attributes**

Figure 3. NBTree v/s VFI - Selected Attributes

The graphs in Figures 2 and 3 show a relative performance of NBTree and VFI for the intrusion detection task on the dataset under consideration. Figure 2 shows the comparison with all attributes under consideration and figure 3 depicts the comparison for attributes selected using the information gain measure.

As per the definitions in section 3.5.2, a good IDS should have a recall that is as high as possible. A high precision is also desired. From our results, we see that the classification accuracy of NBTree is better than that of VFI in both cases. There are tremendous differences in the precision and recall values of NBTree and VFI where the NBTree exhibits a relatively higher precision and higher recall. Also, when all attributes are used, NBTree has a lower precision value than the case when selected attributes are used. In both these cases, the recall is more or less the same. Also, the F- Measure value is high for NBTree in comparison to VFI. NBTree is seen to have a better performance as compared to the VFI in both the cases and it can thus be said that it is more suited to the intrusion detection task on the given data set.

There are tremendous differences in the precision and recall values of NBTree and VFI where the NBTree exhibits a relatively higher precision and higher recall.

5. CONCLUSIONS

Based on the experiments done in this paper and their corresponding results, we can state the following:

• Machine learning is an effective methodology which can be used in the field of computer security.

• The inherent nature of machine learning algorithms makes them more suited to the intrusion detection field of information security. However, it is not limited to intrusion detection. The authors in [10] have developed a tool using machine learning to infer access control policies where policy requests and responses are generated by using learning algorithms. These are effective with new policy specification languages like XACML [13]. Similarly, a classifier-based approach to assigning users to roles and vice-versa is described in [18]. Learning algorithms can also be used to develop applications which, for instance, can check whether people in an organisation are adhering to the defined security policy.

• It is possible to analyse huge quantities of audit data by using machine learning techniques, which is otherwise an extremely difficult task.

Finally it can be said that in order to realise the full potential of machine learning to the field of computer security, it is essential to experiment with various machine learning schemes towards addressing security-related problems and choose the one which is the most appropriate to the problem at hand. ✳

Ron Condon

UK bureau chief
searchsecurity.co.UK

Ron Condon has been writing about developments in the IT industry for more than 30 years. In that time, he has charted the evolution from big mainframes, to minicomputers and PCs in the 1980s, and the rise of the Internet over the last decade or so. In recent years he has specialized in information security. He has edited daily, weekly and monthly publications, and has written for national and regional newspapers, in Europe and the U.S.

REFERENCES

[1] CERT Vulnerability Statistics 1995 - 2006. http://www.cert.org/stats/vulnerability_

remediation.html, 2007.

[2] G. Demiroz and H. A. Guvenir. Classification by voting feature intervals. In European Conference on Machine Learning, pages 85-92, 1997.

[3] T. Dietterich and P. Langley. Machine learning for cognitive networks:technology assessments and research challenges, Draft of May 11, 2003. http://web.engr.oregonstate.edu/_tgd/kp/dl-report.pdf ,

2003.

[4] M. A. Hall. Correlation-based Feature Selection for Machine Learning. PhD thesis, University of Waikato, Department of Computer Science, 1999.

[5] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, pages 1137-1145, 1995.

[6] R. Kohavi. Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pages 202-207, 1996.

[7] M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: one-sided selection. In Proc. 14th International Conference on Machine Learning, pages 179-186. Morgan Kaufmann, 1997.

[8] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel.

Performance measures for information extraction. http://www.nist.gov/speech/publications/darpa99/html/dir10/dir10.htm, 1999.

[9] M. A. Maloof, editor. Machine Learning and Data Mining for Computer Security. Springer, 2006.

[10] E. Martin and T. Xie. Inferring access-control policy properties via machine learning. In POLICY '06: Proceedings of the Seventh IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'06), pages 235-238, Washington, DC, USA, 2006. IEEE Computer Society.

[11] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors. Machine Learning: An Artificial Intelligence Approach. Tioga

Publishing Company, 1983.

[12] T. M. Mitchell. Machine Learning. McGraw Hill, 1997.

[13] T. Mose. Oasis, extensible access control markup language, (xacml) version 2.0. http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf, 2005.

[14] N. J. Nilsson. Introduction to Machine Learning - an early draft of a proposed book. http://ai.stanford.edu/_nilsson/MLDraftBook/MLBOOK.pdf, 1996.

[15] U. Of California. The UCI KDD Archive, University of California. http://kdd.ics.uci.edu/databases/kddcup99/task.html, 1999.

[16] T. Pietraszek. Using adaptive alert classification to reduce false positives in intrusion detection. Recent Advances in Intrusion Detection, 3224:102-124, 2004.

[17] S. V. Sabnani. Computer security: A machine learning approach. Master's thesis, Royal Holloway, University of London, 2007.

[18] S. Sheng and S. L. Osborn. A classifier-based approach to user-role assignment for web applications. In Secure Data Management, pages 163-171, 2004.

[19] S. Tesink. Improving intrusion detection systems through machine learning. http://ilk.uvt.nl/downloads/pub/papers/the-sis-tesink.pdf, 2007.

[20] I. H. Witten and E. Frank. Data Mining - Practical MachineLearning Tools and Techniques, Second Edition. Elsevier, 2005.

[21] S. Wolthusen. Lecture 11 - Intrusion Detection and Prevention, Notes on Network Security, Royal Holloway University of London, 2006.