Pradeep Pundir, Dr.Virendra Gomanse ,Narahari Krishnamacharya / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622
www.ijera.com
Vol. 1, Issue 4, pp.1716-1722

Classification and Prediction techniques using Machine Learning for Anomaly Detection.

Pradeep Pundir, Dr.Virendra Gomanse ,Narahari Krishnamacharya.

***(** Department of Computer Engineering, Jagdishprasad Jhabarmal Tibrewala University, Jhunjhunu Rajasthan, 333001,

ABSTRACT

Network traffic data can be classified into binary class (i.e. anomaly-free and all others) or multi-level classes (e.g., anomaly-free, likely to be anomaly-free, anomaly-free, anomaly, likely to be anomaly, and unable to determined). In this article, the focus is on the common supervised learning algorithms and methods for binary classification. In the real world, it is possible that a data point belongs to more than one class or has similar attributes for multi-membership. In multi-level classification situation can be addressed by using multi-classification algorithms and then making decisions based on the membership functions acquired from the algorithms.

Keywords – **Machine Learning, Classification, Prediction, Supervised Learning, Unsupervised.**

I. INTRODUCTION

Classification and Prediction are two forms of anomaly packet detection that can be used to extract models describing important data classes or to predict future data trends. While **classification** predicts categorical labels (classes), **prediction** models continuous-valued functions. This paper is about recognizing, discovering and utilizing alternative techniques in network security data analytics. Attack detection systems trained on system usage metrics use inductive learning algorithms. To emulate a typical pattern recognition process using a computer models otherwise known as machine learning. Machine learning can be viewed as the attempt to build computer programs that improve performance of some task through learning and experience.

Preprocessing of the data in preparation for classification and prediction can involve data cleaning to reduce noise or handle missing values, relevance analysis to remove irrelevant or redundant attributes, and data transformation, such as generalizing the data to higher level concepts or normalizing data. Our goal of designing machine learning applications with regard

to network security is to reduce the tediousness and time consuming task of human audit analysis. The most commonly applied theory in many machine learning models is Pattern Classification. Predictive, accuracy, computational speed, robustness, scalability and interpretability are five criteria for the evaluation of classification and prediction methods. The Machine Learning field has been evolving from the broad field of Artificial Intelligence, which aims to copy intelligent abilities of humans by machines. In the field of Machine Learning one considers the important question of how to make machines able to "learn". The context of Learning is understood as inductive inference, where one observes examples that represent incomplete information about some "statistical phenomenon". In contrast with unsupervised learning or clustering one typically tries to uncover hidden regularities (e.g. distance base cluster analysis) or to detect anomalies in the data (for instance some unusual machine function or an intrusion). In supervised learning, there is a class label associated with each example. It is supposed to be the answer to a question about the example. If the label is discrete, then the task is called classification problem – otherwise, for real-valued labels we speak of a regression problem. Based on these examples (including the labels), one is particularly interested in predicting the answer for other cases before they are

explicitly observed. Hence, learning is not only a question of remembering but also of generalization to unknown cases.

1.1 Theory of Machine Learning

What exactly is machine learning? The Machine Learning field evolved from the broad field of Artificial Intelligence, which aims to mimic intelligent abilities of humans by machines. In the field of Machine Learning one considers the important question of how to make machines able to "learn". Learning in this context is understood as inductive inference; where one analyzes examples that represent incomplete information about some "statistical phenomenon".

In contrast with unsupervised learning or clustering one typically tries to uncover hidden regularities (e.g. distance base cluster analysis)or to detect anomalies in the data (for instance some unusual machine function or an intrusion).

In supervised learning, there is a class label associated with each example and is supposed to be the answer to a question about the example. If the label is discrete, then the task is called classification problem – otherwise, for real-valued labels we speak of a regression problem.

Based on the examples (including the labels), one is particularly interested in predicting the answer for other cases before they are explicitly observed. Hence, learning is not only a question of remembering but also of generalization to unknown cases. In machine learning, computer algorithms (learners) attempt to automatically distill knowledge from example data. This knowledge can be used to make predictions about novel data in the future and to provide insight into the nature of the target concepts. Applied to network security and intrusion detection, this means that a computer would learn to classify alerts into incidents and non-incidents. A possible performance measure for this task would be the accuracy with which the machine learning program classifies the instances correctly. The training experiences could be labeled instances. All of these will be elaborated on in subsequent sections.

1.1.1 Advantages of Machine Learning

First of all, for the classification of network security incidents, a vast amount of data has to be analyzed containing historical data. It is difficult for human beings to find a pattern in such an enormous amount of data. Machine Learning, however, seems well-suited to overcome this problem and can therefore be used to discover those patterns. Also an analyst's knowledge is often implicit, and the environments are dynamic. As a consequence, it is very hard to program IDS using ordinary programming languages that require the formalization of knowledge. The adaptive and dynamic nature of machine learning makes it a suitable solution for this situation. Third, the environment of an IDS and its classification task highly depends on personal preferences. The suspicious traffic on the network may seem to be an incident in one environment may be normal in other environments. This way, the ability of computers to learn enables them to know someone's "personal" (or organizational) preferences, and improve the performance of the IDS, for this particular environment

1.2 Machine Learning Categories

Machine learning can be divided in two categories; **supervised** and **unsupervised** machine learning algorithms.

In supervised learning, the input of the learning algorithm consists of examples (in the form of feature vectors) with a label assigned to them. The objective of supervised learning is to learn to assign correct labels to new unseen examples of the same task. As shown in Figure 1.1, a supervised machine learning algorithm consists of three parts: A learning module, a model and a classification module.
The learning module constructs a model based on a labeled training set. This model consists of a function that is built by the learning module, and contains a set of associative mappings (e.g. rules). These mappings, when applied to an unlabeled test instance, predict labels of the test set. The prediction of the labels of the test set is done by using the classification module.

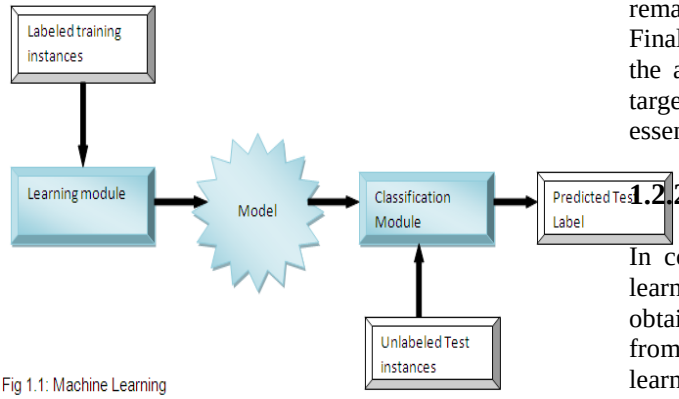


Fig 1.1: Machine Learning

1.2.1 Supervised Learning

Classification is an important task in Machine Learning, which is also referred to as pattern recognition, where algorithms are researched and build which are capable of automatically constructing methods for distinguishing between different exemplars, based on their differentiating patterns. A pattern could be given a name as it is an entity that could be vaguely defined and is also classified as "the opposite of chaos". There are different examples of patterns such as human faces, text documents, handwritten letters or digits, EEG signals, and the DNA sequences that may cause a certain disease. A pattern is defined by its features. These are the characteristics of the examples for a given problem. For instance, in a face recognition task some features could be defined as the color of the eyes or the distance between the eyes. Thus, the input to a pattern recognition task can be viewed as a two-dimensional matrix, whose axes are the examples and the features.

Pattern classification tasks are often divided into several sub-tasks such as:

- Data collection and representation.
- Feature selection and Reduction.
- Classification.

Mostly problem-specific statements are described under Data collection and representation. Therefore it is difficult to give general statements about this step of the process. Invariant features should be identified broadly and categorically which defines the differences. Feature selection and feature reduction attempt to reduce the dimensionality for the remaining number of feature steps of the task. Finally, the classification phase of the process finds the actual mapping between patterns and labels (or targets). In many applications the second step is not essential or is implicitly performed in the third step.

1.2.2 Unsupervised Learning

In contrast to supervised learning, in unsupervised learning the machine simply receives inputs, but obtains neither supervised target outputs, nor rewards from its environments. Unsupervised algorithms learn from unlabeled examples. Unsupervised learning can be thought of as finding patterns in the data and beyond what would be considered pure unstructured noise. The objective of unsupervised learning may be to cluster examples together on the basis of their similarity. Supervised learning methods will be used in this paper.

1.2.3 Eager Learning

Another distinction between types of machine learning is the one between eager and lazy learning. Eager learning is a form of supervised learning, which means that there is a learning module, a model and a classification module, as shown in Figure 1.1. Eager learning algorithms invest most of their effort in the learning phase. They construct a compact representation of the target function by generalizing from the training instances. Classification of new instances is usually a straightforward application of simple learned classification rules that employ the eager learner's model. A method is called eager when it generalizes beyond the training data before observing a new query, committing at training time to the network structure and weights that (i.e. the model) define its approximation to the target function.

1.2.3.1 Rule Induction

Rule induction is one of the types of eager learning. During the learning phase, rules are induced from the training sample, based on the features and class labels of the training samples. The goal of rule induction is generally to induce a set of rules from data that captures all generalizable knowledge within that data, and that is as small as possible at the same time. The rules that are extracted during the learning phase can

easily be applied during the classification phase when new unseen test data is classified.

Rule induction has lots of advantages. First of all, the rules that are extracted from the training sample are easy to understand for human beings. The rules are simple if-then rules and the rule learning systems outperform decision tree learners on many problems. A major disadvantage of rule induction, however, is that it scales relatively poorly with the sample size, particularly on noisy data.

1.2.4 Lazy Learning

Next to eager learning, there is also lazy learning as a form or variant of supervised learning. Lazy, instance based, exemplar-based, memory-based, case-based learning or reasoning are defined as memory-based learning algorithms. The reason for calling certain machine learning methods lazy is because they defer the decision of how to generalize beyond the training data until each new query instance is encountered. A key feature of lazy learning is that during the learning phase, all examples are stored in memory and no attempt is made to simplify the model by eliminating noise, low frequency events, or exceptions. The learning phase of a lazy learning algorithm consists simply of storing all encountered instances from a training set in memory. During the classification phase search for the optimal hypothesis takes place. A memory-based learning algorithm searches for the best matching instance or more generically a set of the k best matching instances in memory on being presented with a new instance during the classification phase. The algorithm takes the majority class and the instances in the set are then labeled as belonging to the class of the new instance after having found such a set of k best-matching instances. Pure memory-based learning algorithms implement the classic k -nearest neighbor algorithm.

A learning algorithm should not forget any information contained in the learning material in order to learn task successfully and it should not abstract from the individual instances. Computational optimizations of memory-based learning can be attained by replacing them by instance types and forgetting instance tokens as the memory that needs to be searched may become considerably smaller. A major disadvantage of lazy learning, however, is that noise in the training data can harm accurate generalization. Overall, lazy algorithms have lower computational costs than eager algorithms during training whilst they typically have greater storage requirements and often have higher computational costs when answering requests.

1.2.5 Hybrid Learners

Combination of k-NN classifier and rule induction generates hybrid learners. The reason for constructing hybrids is the difference in properties between memory-based learning and eager learning. Memory-based learners put time in the classification phase, whereas eager learners invest their time in the learning phase. Combining eager and lazy learners into hybrids; will produce machine learners that put effort in both the learning phase and the classification phase. This leads to the expectation that this double effort will be repaid with improved performance. The hybrid will use both the global hypothesis as induced by rule induction, as well as the local hypothesis created during memory-based learning.

The hypothesis then exists that combining the efforts in the learning task of eager learners with the efforts of the lazy learners' classification task will increase the accuracy with which incidents are predicted. Combining memory-based learning with eager learning into a hybrid may improve the generalization performance of the classifier. On the other hand, one of the draw-backs of eager learning is its insensitivity, by its generalization. By combining the learning module of an eager learner with the classification module, the results of the classification task of incidents should improve using hybrid machine learning.

1.3 Classification Algorithms

Although Machine Learning is a relatively young field of research, there are a myriad of learning algorithms that can be mentioned in this section but only six methods that are frequently used in solving data analysis tasks (usually classification) are discussed. The first four methods are traditional techniques that have been widely used in the past and work reasonably well when analyzing low dimensional data sets with not too few labeled training examples. Two methods Support Vector Machines & Boosting that have received a lot of attention in the Machine Learning community

recently will also be given a prominent mention. The advantages of Support Vector Machines & Boosting over traditional methods are that they are able to solve high-dimensional problems with very few examples quite accurately and also work efficiently when examples are huge (for instance several hundred thousands of examples).

1.3.1 k-Nearest Neighbor

One of the best known instance based learning algorithm is the k-Nearest Neighbor (k-NN). In pattern recognition, the k-nearest neighbor algorithm (k-NN) is a method based on learning by analogy, that is, by comparing given test sample with training samples that are similar to it. The training samples are described by n attributes. Each sample represents a point in n-dimensional pattern space. When given an unknown sample, K-NN searches the pattern space for the k-training samples that are closest training examples in the feature space. It is a type of lazy learning where the function is only approximated locally and all computation is deferred until classification. The key idea of k nearest neighbor algorithm is that the properties of any particular input point are likely to be similar to those of points in the neighborhood of input point (k is a positive integer, typically small). The object is simply assigned to the class of its nearest neighbor when k=1.

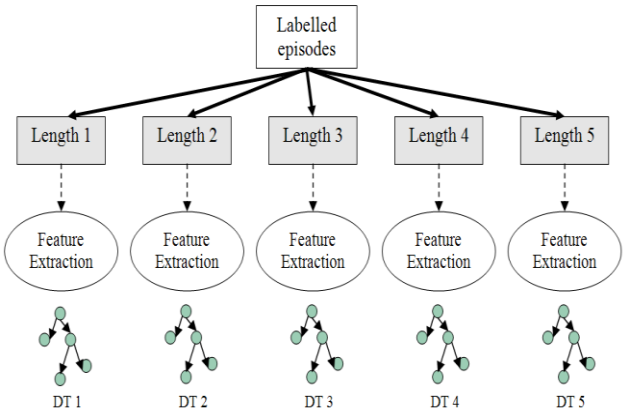
Here the k points of the training data closest to the test point are found, and a label is given to the test point by a majority vote between the k points. As was described earlier, the most important phase for a lazy learner is the classification phase. As a model of the target function k-NN algorithm uses all labeled training instances. During the classification phase, k NN uses a similarity-based search strategy to determine a locally optimal hypothesis function. Test instances are compared to the stored instances and are assigned the same class label as the k most similar stored instances. Since this method is highly intuitive, simple and low classification errors, but the disadvantage is; it is computationally expensive and requires a large memory to store the training data.

1.3.2 Linear Discriminant Analysis (LDA)

LDA computes a hyper plane in the input space that minimizes the within class variance and maximizes the between class distance. Even with large data sets it can be efficiently computed but often a linear separation is not sufficient. Nonlinear extensions using kernels exist, however it is difficult to apply to problems with large training sets.

1.3.3 Decision Trees

Decision Tree is another intuitive class of classification algorithms. Each algorithm uses an attribute selection measure to select the attribute tested for each non-leaf node in the tree. Classification problem are solved by these algorithms I which they repeatedly partitioning the input space, so as to build a tree whose nodes are as pure as possible (that is, they contain points of a single class). Classification of a new test point is achieved by moving from top to bottom along the branches of the tree, starting from the root node, until a terminal node is reached. Decision trees are simple yet effective classification schemes for small datasets. The computational complexity scales unfavorably with the number of dimensions of the data. Large datasets tend to result in complicated trees, which in turn require a large memory for storage. Pruning algorithms attempt to improve accuracy by removing tree branches reflecting noise in the data. Early decision tree algorithms typically assume that the data are memory resident. Several scalable algorithms such as SLIQ, SPRINT and Rain-Forest, have been proposed to address pattern classification.



1.3.4 Artificial Neural Networks

Neural networks are perhaps one of the most commonly used approaches in data classification. They are non-linear predictive models that learn

through training and look like biological neural networks in structure. Neural networks are a computational model inspired by the connectivity of neurons in animate nervous systems. A further boost to their popularity came with the proof that they can approximate any function mapping via the Universal Approximation Theorem. A simple scheme for a neural network is shown in Figure 1.2. Each circle denotes a computational element referred to as a neuron, which computes a weighted sum of its inputs, and possibly performs a nonlinear function on this sum. Any function can be identical if the function is computed by the network (specifically a mapping from the training patterns to the training targets), provided enough neurons exist in the network and proper training examples are provided.

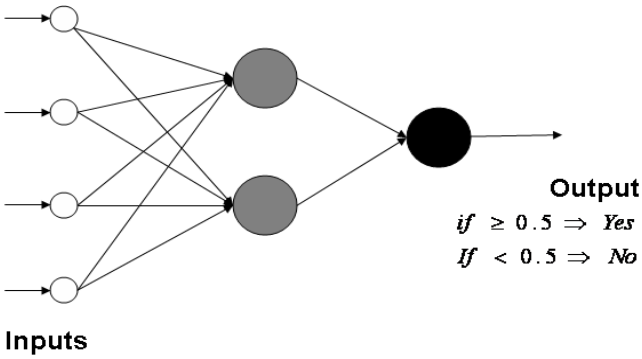


Figure 1.3: Many of these technologies have been in use for more than a decade in specialized analysis tools that work with relatively small volumes of data. These capabilities are now evolving to integrate directly with industry-standard data warehouse and OLAP platforms.

1.4 Maximum Margin Algorithms

Statistical Learning Theory is one of the theoretical foundation on which Machine Learning depends as it provides conditions and guarantees for good process of formulating general concepts by abstracting common properties of instances (known as generalization) of learning algorithms. A practical result of generalization theory is maximum margin classification techniques as they have emerged in last decades. The margin is the distance of the example to the separation boundary and a maximum margin classifier generates decision boundaries with large margins to almost all training examples is the role of maximum margin algorithm.. The two most widely studied classes of maximum margin classifiers are Support Vector Machines (SVMs) and Boosting.

1.4.1 Support Vector Machines

Single Layer networks have a simple and efficient learning algorithm, but have very limited expressive power as they can learn only linear decision boundaries in the input space. Multilayer networks, on the other hand are much more expressive as they can represent general non linear functions but are very hard to train because of the abundance of local minima and the high dimensionality of the weight space. Support Vector Machines (SVM) is a new method for the classification of both Linear and no linear data.

SVM is an algorithm that works as follows; It uses a nonlinear mapping to transform the original training data into a higher dimension. Given a training sample, the support vector machine constructs a hyperplane as the decision surface in such as way that the margins of separation between positive and negative examples is maximized. Within this new dimension it searches for the linear optimal separating hyperplane (that is, a “decision boundary” separating the samples of one class from another). Mapping to a sufficiently high dimension with an appropriate non linear examples, hyperplane is always used to separate data from two classes. SVM finds this hyperplane using support vectors (“essential” training samples) and margins (defined by the support vectors).

1.4.2 Boosting

In boosting, weights are assigned to each training samples. A series of k classifiers is iteratively learned. After a classifier is learned, the weights are updates to allow the subsequent classifier, to pay more attention to the training samples that were misclassified by first classifier. The final boosted classifier combines the votes of each individual classifier, where the weight of each classifier’s vote is a function of it accuracy. The boosting algorithm can be extended for the prediction of continuous values.

Adaboost is one of the popular algorithms in boosting. If accuracy of some learning method needs to be boosted, we are given data set of class labeled samples; Adaboost assigns each training sample an equal weight. It has been shown that Boosting has strong ties to support vector machines and maximum margin classification. Boosting techniques have been a used on very high dimensional data sets and can quite easily deal with more than a hundred thousand examples.

References:

1) www.idsia.ch/~ioannis/publications/pdf/besai2006.pdf

2) EXPLORATORY DATA ANALYSIS OF TRACE ELEMENTS IN CLINKER, János Abonyia , Ferenc D. Tamás, and Josef Tritthartc.

3) Abbott, D., Matkovsky, P. & Elder, J. (1998).An Evaluation of High-End Data Mining Tools for Fraud Detection. Proc. of IEEE SMC98.

4) Fan, W., Miller, M., Stolfo, S., Lee, W. & Chan, P. (2001). Using Artificial Anomalies to Detect Unknown and Known Network Intrusions. Proc. of ICDM01, 123-248.

5) Fanning, K., Cogger, K. & Srivastava, R. (1995). Detection of Management Fraud: A Neural Network Approach. Journal of Intelligent Systems in Accounting, Finance and Management 4:113-126.

6) Fawcett, T. (2003). "In Vivo" Spam Filtering: A Challenge Problem for KDD. SIGKDD Explorations 5(2): 140-148. Fawcett, T. (1997). AI Approaches to Fraud Detection and Risk Management: Papers from the 1997 AAAI Workshop. Technical Report WS-97-07. AAAI Press.

7) Fawcett, T. & Provost, F. (1999). Activity monitoring: Noticing Interesting Changes in Behavior. Proc. of SIGKDD99, 53-62. Fawcett, T. & Provost, F. (1997). Adaptive Fraud Detection.

8) Data Mining and Knowledge Discovery 1(3): 291-316. Foster, D. & Stine, R. (2004). Variable Selection in Data Mining: Building a Predictive Model for Bankruptcy. Journal of American Statistical Association 99: 303-313.

9) Ghosh, S. & Reilly, D. (1994). Credit Card Fraud Detection with a Neural Network. Proc. of 27th Hawaii International Conference on Systems Science 3: 621-630.

10) Goldberg, H., Kirkland, J., Lee, D., Shyr, P. & Thakker, D. (2003). The NASD Securities Observation, News Analysis & Regulation System (SONAR). Proc. of IAAI03.

11) Fraud through Neural Network Technology. Auditing 16(1): 14-28.

12) Hawkins, S., He, H., Williams, G. & Baxter, R. (2002). Outlier Detection Using Replicator Neural Networks. Proc. of DaWaK2002, 170-180.

13) Hodge, V. & Austin, J. (2004). A Survey of Outlier Detection Methodologies. Artificial Intelligence Review 22: 85-126.

14) Wong, W., Moore, A., Cooper, G. & Wagner, M. (2003). Bayesian Network Anomaly Pattern Detection for Detecting Disease Outbreaks. Proc. of ICML03, 217-223.

15) Yamanishi, K., Takeuchi, J., Williams, G. & Milne, P. (2004).

16) On-Line Unsupervised Outlier Detection Using Finite Mixtures with Discounting Learning Algorithms. Data Mining and Knowledge Discovery 8: 275-300.

17) Yeung, D. & Ding, Y. (2002). User Profiling for Intrusion Detection Using Dynamic and Static Behavioural Models. Proc. of PAKDD2002, 494-505.

18) Yoshida, K., Adachi, F., Washio, T., Motoda, H., Homma, T., Nakashima, A., Fujikawa, H. & Yamazaki, K. (2004). Density- Based Spam Detector. Proc. of SIGKDD04,486-493..