International Journal of Innovative Technologies
WWW.IJITECH.ORG
ISSN 2321-8665
Vol.03,Issue.01,
May-2015,
Pages:0030-0033

Navie Bayes Classification Dataset Feature Filtering For Malware Detection

K. JYOTHI[1], S. JESSICARITHA[2]

[1]PG Scholar, Dept of CSE, JNTU College of Engineering (Autonomous), Pulivendula, AP, India, E-mail: jkammari@gmail.com.
[2]Asst Prof, Dept of CSE, JNTU College of Engineering (Autonomous), Pulivendula, AP, India, Email: sarithajntuacep@gmail.com.

Abstract: An n-gram may be a sub-sequence of n things from a given sequence. N-grams square measure employed in varied areas of applied mathematics linguistic communication process and genetic sequence analysis. During which sequence analysis is that the method of comparison the sequence or series of attributes so as to seek out the similarity. Malware is nothing however malicious computer code that\'s designed by attackers for worrying computers. Malware variants can have distinct computer memory unit level representations whereas in principal belong to a similar family of malware. The computer memory unit level content is completely different as a result of tiny changes to the malware ASCII text file may end up in considerably different compiled code. During which programs square measure used as operational code (opcode) density histograms obtained through dynamic analysis. Dynamic analysis is that the method of testing and analysis of application or a program throughout period. A SVM is employed for classification or regression issues. It uses a way known as the kernel trick to remodel your information and so supported these transformations it finds associate degree optimum boundary between the doable outputs. We have a tendency to use static analysis to classify malwares. So in this paper we are using three algorithms for finding the malware those are naive baives, J48, random forest. It known a prefilter stage victimisation hex values of files, that may scale back the feature set and so scale back the coaching effort. The result shows that the relationships between options square measure complicated and straightforward statistics filtering approaches don't give a sensible approach. However, hex decimal primarily based produces an appropriate filter. The whole systems are going to be enforced in WEKA tool.

Keywords: Text File, Static Analysis, Testing.

I. INTRODUCTION

Recent years have seen huge growth in malware, with signature detection and observance suspected code for illustrious security vulnerabilities turning into ineffective and intractable. In response, researchers have to be compelled to adopt new detection approaches that trump the various attack vectors and obfuscation ways utilized by the malware writers. Detection approaches that use the host environment's native opcodes at run-time can circumvent several of the malware writers' attempts to evade detection. One such approach, as planned in this paper, is that the analysis of opcode density options victimization supervised learning machines performed on options obtained from run-time traces. In future analysis we tend to will expand the detection ways by work N-gram size, which will dramatically increase the quantity of options. With this anticipated explosion of options we've got chosen to analyze methods to prune digressive options. While Principle part Analysis (PCA) could be a standard method to cut back options in mathematical space, this paper aims to identify feature reduction within the original dataset area.

II. LITERATURE SURVEY

A. A Method for Detective Work Obfuscated Calls In Malicious Binaries

Information regarding calls to the OS (or kernel libraries) created by a binary workable is also accustomed verify whether or not the binary is malicious. Being tuned in to this approach, malicious programmers hide this data by creating such calls while not exploitation the decision instruction. as an example, the decision addr instruction is also replaced by 2 push directions and a sop instruction, the primary push pushes the address of instruction once the sop instruction, and also the second push pushes the address addr. The code is also additional obfuscated by spreading the 3 directions and by cacophonous every instruction into multiple directions. This work presents a way to statically observe obfuscated calls in code. The thought is to use abstract interpretation to observe wherever the conventional call-ret convention is desecrated. These violations may be detected by what's referred to as associate degree abstract stack graph. Associate degree abstract stack graph could be a sententious illustration of all potential abstract stacks at each purpose in an exceedingly program. Associate degree abstract stack is employed to associate every component within the stack to the instruction that pushes the component. Associate degree algorithmic program for constructing the abstract stack graph is additionally bestowed. Strategies for exploitation the abstract stack graph area unit shown to observe eight completely different obfuscations. The technique is incontestable by implementing a paradigm tool referred to as DOC (detector for obfuscated calls).

B. Automatic Analysis of Malware Behaviour using Machine Learning

Konrad Rieck1, Philipp Trinius2, In this framework for automatic analysis of malware behaviour using machine learning. The framework allows for automatically identifying novel classes of malware with similar behaviour (clustering) and assigning unknown malware to these discovered classes (classification). Based on both, clustering and classification, we propose an incremental approach for behavior-based analysis, capable of processing the behavior of thousands of malware binaries on a daily basis.

C. A New Classification Statistics Technique for Malware Detection and Risk Assessment in A Modern Computer And Network Systems

This paper, proposes the use of SVM as a means of identifying malware. It shows that malware, that is packed/encrypted, and it is identified a pre-filter stage using eigenvectors that can reduce the feature set and therefore reduce the training effort. The results presented in this paper exposed three key points. Firstly, the identification of a high population op-code: mov that is not only is a poor indicator of benign/malicious software, but inhibits the ability to correctly classify software when used with other op-codes such as ja, adc, sub, inc, add and rep. Secondly, a subset of op-codes can be used to detect malware. However, the SVM analysis demonstrates that ja, adc and sub are strong indicators of malware as they are four times more likely to be used in the correct classification of malware than the next most significant op-codes (inc). Several op-codes have been identified as potential indicators of malware, which provides the basis for an improvement in detection techniques beyond current state of the art. Finally, using the 'eigenvector' pre-filter, the dataset can safely remove irrelevant features.

D. N-Grams-Based File Signatures for Malware Detection

As the amount of malware and its variety is growing every year, malware detection rises to become a topic of research and concern. Further, classic signature methods provide detection when the threat is known in beforehand. It fails, however, when confronted to Unknown malware. Future work will focus on further research of the capability of n-gram analysis in malware detection, experiments with different and larger malware collections, and combination of this technique with behavioural analysis of malicious code.

E. A Fast Automaton-Based Method for Detecting Anomalous Program Behaviors

In this paper, we presented a new technique for intrusion detection based on learning program behaviors. Our method captures program behaviors in terms of sequences of system calls. These sequences are represented using a finite-state automaton. Unlike previous approaches, the FSA approach does not limit either the number or length of system call sequences. (Even without such limits, our representation ensures that the size of FSA itself is bounded – in the worst case, its size is linear in the size of the program.) Moreover, it captures the looping and branching structures of a program in a natural way, enabling it to recognize variations of behaviors learnt during training. The presence of program state information enables the FSA approach to perform more accurate detection of execution of unusual sections of code. Its ability to focus on program behaviors (while ignoring library behaviors) contributes to shorter training periods and smaller storage requirements.

F. Detecting Bots via Incremental LS-SVM Learning with Dynamic Feature Adaptation

Feilong ChenDept. of Computer Science, In this paper, we argue that, even in the face of encrypted traffic flows, botnets can still be detected by examining the set of server IP-addresses visited by a client machine in the past. However there are several challenges that must be addressed. First, the set of server IP-addresses visited by client machines may evolve dynamically. Second, the set of client machines used for training and their class labels may also change over time.

G. Malware Detection Using Machine Learning

Dragos¸ Gavrilut¸1,2, Mihai Cimpoes¸u1,2, In this paper we present the ideas behind our framework by working firstly with cascade one-sided perceptrons and secondly with cascade kernelized one-sided perceptrons. After having been successfully tested on medium-size datasets of malware and clean files, the ideas behind this framework were submitted to a scaling-up process that enable us to work with very large datasets of malware and clean files.

H. Using Multi-Feature and Classifier Ensembles to Improve Malware Detection

Yi-Bin Lu1, Shu-Chang Din2, In this paper, we improved the accuracy of machine learning from these two factors. On the one hand we combined features extracted from both content-based and behavior-based analyses to represent the instances; on the other hand, we used classifier ensembles to replace individual classifier. Based on our methodology, a hybrid-classifier was implemented to classify unknown executables as either malicious or benign.

I.Detecting Stealthy Malware Using Behavioural Features in Network Traffic

Ting-Fang Yen, In this thesis, we hypothesize that malware-infected hosts share characteristics in their network behaviors, which are distinct from those of benign hosts. Our approach works by aggregating "similar" network traffic involving multiple hosts. We identify key characteristics that capture basic properties of botnet operation, and that can be observed even within coarse network traffic summaries

III. EXISTING SYSTEM

The utilization of SVM as tools of distinctive malware. It shows that malware, that\'s packed or encrypted, is detected victimization SVMs and by victimization the opcodes chosen by the SVM as a benchmark, known a prefilter stage victimization eigenvectors which will cut back the feature set and thus cut back the coaching effort.

Limitations:

- A set of operation codes are often accustomed discover malware.
- Less accuracy.

IV. PROPOSED SYSTEM

In planned system we have a tendency to area unit victimisation navie Bayes formula for classification with accuracy as shown in Fig.1. A plus of naive Bayes is that it

solely needs a little quantity of coaching information to estimate the parameters (means and variances of the variables) necessary for classification. as a result of freelance variables area unit assumed, solely the variances of the variables for every category have to be compelled to be determined and not the complete variance matrix.

Advantages:

- Accuracy
- Each and every attribute classification.
- Effective

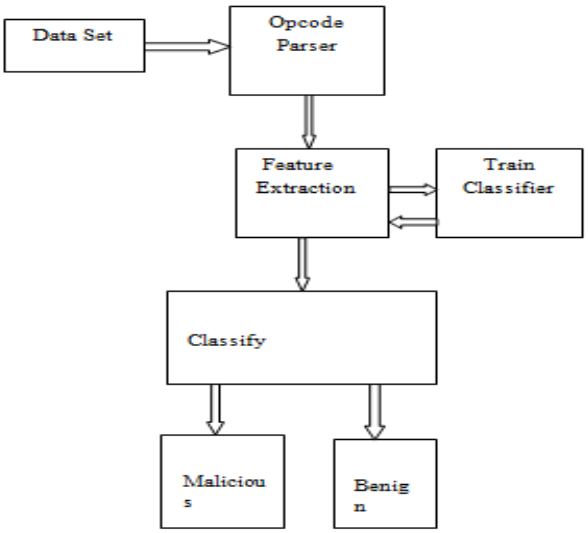

Fig.1. System Architechture.

V. MODULES

- Data Collection
- Dataset Creation
- Feature Extraction
- Classification0

A. Data Collection

In this stage, data set consists of 100 binaries out of which 90 are benign and 10 are spyware binaries. The benign files were collected from Download.com, which certifies the files to be free from spyware as shown in Fig.2. The spyware files were downloaded from the links provided by SpywareGuide.com. This hosts information about different types of spyware and other types of malicious software.

Fig.2. Data Collection.

B. Dataset Creation

In which byte sequences represent fragments of machine code from an executable file. We use xxd, which is a UNIX-based utility for generating hexadecimal dumps of the binary files as shown in Fig.3. From these hexadecimal dumps we may then extract byte sequences, in terms of n-grams of different sizes. ARFF databases based on frequency and common features were generated. All input attributes in the data set are represented by Booleans. These ranges are represented by either 1 or 0.

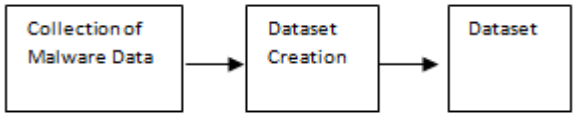

Fig.3. Dataset Creation.

C. Feature Extraction

In this stage output from the parsing is further subjected to feature extraction as shown in Fig.4. We extract the features by using following approaches, the Common Feature-based Extraction (CFBE) and Frequency-based Feature Extraction. The occurrence of a feature and the frequency of a feature. Both methods are used to obtain Reduced Feature Sets (RFSs) which are then used to generate the ARFF files.

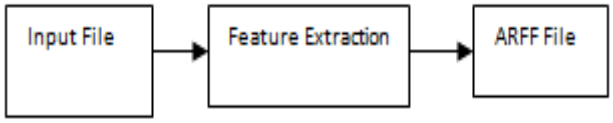

Fig.4. Feature Extraction.

D. Classification

Naive Bayes classifier is a probabilistic classifier based on Bayes theorem with independence assumptions, i.e., the different features in the data set are assumed not to be dependent of each other as shown in Fig.5. This of course, is seldom true for real-life applications. Nevertheless, the algorithm has shown good performance for a wide variety of complex problems. J48 is a decision tree-based learning algorithm. During classification, it adopts a top-down approach and traverses a tree for classification of any instance. Moreover, Random Forest is an ensemble learner. In this ensemble, a collection of decision trees are generated to obtain a model that may give better predictions than a single decision tree.

Fig.5. Classification.

VI. RESULTS

Results of this paper is shown in Figs.6 to 8.

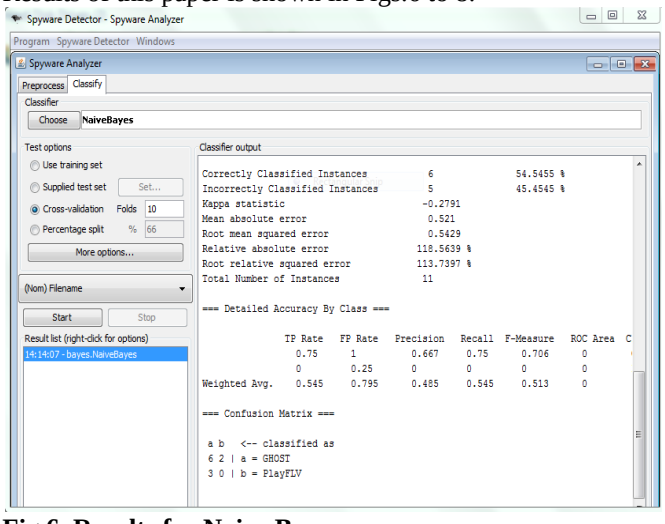

Fig.6. Results for Naive Bayes.

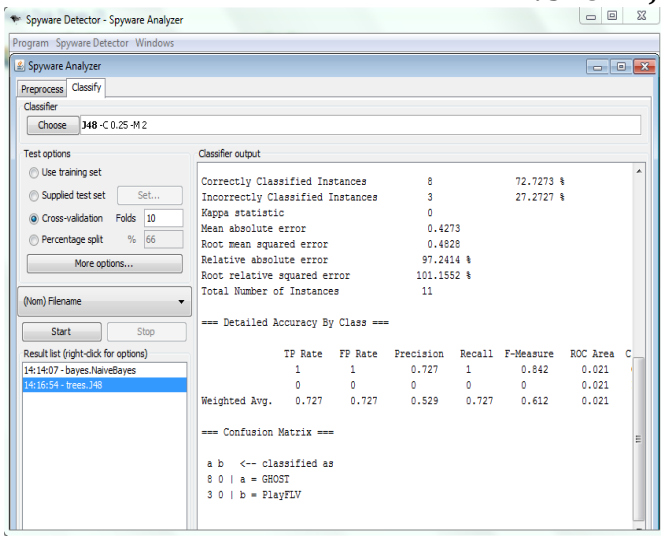

Fig.7. Results for J48 classifier.

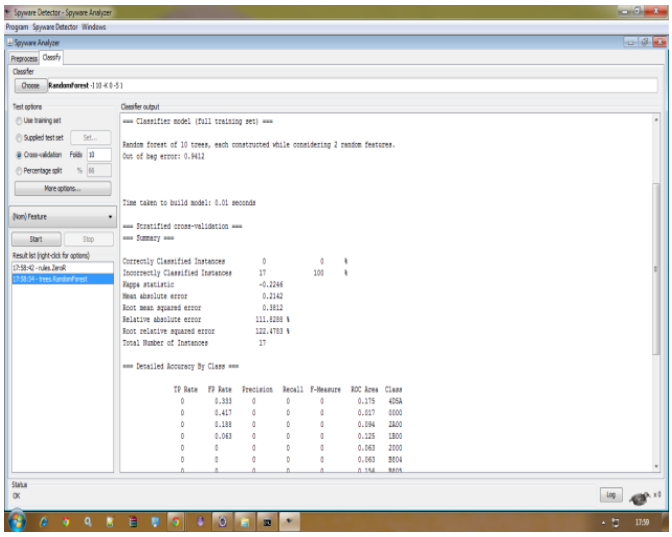

Fig.8. Results for Random Forest.

V. CONCLUSION

In planned system we have a tendency to area unit victimisation navie Bayes formula for classification with accuracy. a plus of naive Bayes is that it solely needs a little quantity of coaching information to estimate the parameters (means and variances of the variables) necessary for classification. as a result of freelance variables area unit assumed, solely the variances of the variables for every category have to be compelled to be determined and not the complete variance matrix.

Advantages:

- Accuracy.
- Each and every attribute classification.
- Effective.

VI. REFERENCES

[1]M. E. Karim, A. Walenstein, A. Lakhotia, and L. Parida, "Malware phylogeny generation using permutations of code," Journal in Computer Virology, vol. 1, pp. 13-23, 2005.

[2]Margaret H. Danham,S. Sridhar, ”Data mining, Introductory and Advanced Topics”, Person education , 1st ed., 2006.

[3]Wenke Lee, Salvatore J. Stolfo, Kui W. Mok, “A data processing Framework for Building Intrusion”

[4]Aman Kumar Sharma, Suruchi Sahni, “A Comparative Study of Classification Algorithms for Spam Email Data Analysis”, IJCSE, Vol. 3, No. 5, 2011, pp. 1890-1895

[5]W. L. K. Wang, S. Stolfo, and B. Herzog, “Fileprints: Identifying file types by n-gram analysis,” in Proc. 6th IEEE Inform. Assurance Workshop, Jun. 2005, pp. : 64–71.

[6]I. Santos, F. Brezo, J. Nieves, Y. K. Penya, B. Sanz, C. Laorden, and P. G. Bringas, “Opcode-sequence-based malware detection,” in Proc. 2nd Int. Symp. Eng. Secure Software and Syst. (ESSoS), Pisa, Italy, Feb. 3–4, 2010, vol. LNCS 5965, pp. 35–43.

[7]D. Bilar, “Opcodes as predictor for malware,” Int. J. Electro Security Digital Forensics, vol. 1, no. 2, pp. 156–168, 2007.

[8]Anshul Goyal, Rajni Mehta, “Performance Comparison of Naive Bayes and J48 Classification Algorithms”, IJAER, Vol. 7, No. 11, 2012, pp.

[9]http://stackoverflow.com/questions/10317885/deci sion-tree-vs-naive-bayes-classifier

[10]George Dimitoglou, James A. Adams, and Carol M. Jim,” Comparison of the C4.5 and a Naive Bayes Classifier for the Prediction of Lung Cancer Survivability”.