International Interdisciplinary Virtual Conference on
'Recent Advancements in Computer Science, Management and Information Technology'
International Journal of Scientific Research in Computer Science,
Engineering and Information Technology | ISSN : 2456-3307 (www.ijsrcseit.com)

Android Malware Analysis using Classification Techniques

Dr. Deepti H. Pethkar
Shankarlal Khandelwal College, Akola, Maharashtra, India

ABSTRACT

Malware is currently one of the biggest threats aimed at mobile devices is growing as more sophisticated mobile platforms have just become available, and more sensitive applications like banking are increasingly employing mobile platforms to the Internet's security. Any malicious software intended to carry out harmful operations on a targeted framework is considered malware. The introduction of Android terminals into people's lives allowed Android malware to start having a real impact on people's lives. Attackers may easily obtain client private information due to Android's security flaws, and the information can then be exploited in APT assaults. This article discusses Android malware techniques, AI, and the use of deep learning to a malware detection system.

Keywords: Malware, Android Malware Detection, Signature-based Malware Detection Systems.

I. INTRODUCTION

Data Mining [2] is the process of extracting previously unknown information from a large dataset. Data mining is also known as knowledge mining from data, knowledgeextraction, data / pattern analysis, data archaeology, and data dredging. Remember that the mining of gold from rocks or sand is referred to as gold mining rather than rock or sand mining. [2] Thus, data mining should have been more appropriately named "knowledge mining from data," which is unfortunately somewhat long. Data mining is considered one of the most important frontiers in database systems and the KDD like Kaggle site for the gathering of data sets.

II. LITERATURE SURVEY

Sunil Kumar, et.al presents and compares the analysis of different Android malware detection frameworks dependent on various parameters, for example, detection system, examination technique and separated highlights [1]. We discover inquire about work in all the Android malware detection procedures that utilization AI, which additionally features the way that AI calculations are usually utilized around there for recognizing Android malware in nature.

Howard, M., et.al proposed a technique for expanding machine learning-based malware detection systems by anticipating characteristics of future varieties of malware and implanting them into the protected structure as a vaccination[3]. Our method uses significant learning to know the ways by development of malware.

Souri, A, et. al, presents a detailed and systematic review using malware detection method data mining techniques[4]. It also classifies malware sensing techniques into two primary classes, namely methods for signature and behaviour. The paper offers a detailed and confidential view of current solutions to the machine learning mechanisms; talks about the structure of realistic techniques used in detection approaches of malware and summarizes the problems of malware methods in data mining; and addresses significant data mining malware classification approaches.

Tieming Chen, et. Al proposes a novel lightweight static detection model, TinyDroid, utilizing guidance rearrangements and AI strategy[5,6]. Initial, an image-based improvement strategy is proposed to extract the opcode succession decompiled from Android Dalvik Executable records. At that point, N-gram is utilized to separate highlights from the streamlined opcode succession, and a classifier is prepared for the malware detection and order assignments. The test results show that TinyDroid can get a higher precision rate and lower bogus alert rate with fulfilled proficiency.

III. WEKA

Weka was developed at the University of Waikato in New Zealand; the name stands for Waikato Environment for Knowledge Analysis The system is written in Java and distributed under the terms of the GNU General Public License. It runs on almost any platform and has been tested under Linux, Windows, and Macintosh operating systems and even on a personal digital assistant. It provides a uniform interface to many different learning algorithms, along with methods for pre and post processing and for evaluating the result of learning schemes on any given dataset. Weka provides implementations of learning algorithms that can be easily apply to dataset. It also includes a variety of tools for transforming datasets, such as the algorithms.

The Weka workbench is a collection of state-of-the-art machine learning algorithms and data preprocessing tools. It is designed so that we can quickly try out existing methods on new datasets in flexible ways. It provides extensive support for the whole process of experimental data mining, including preparing the input data, evaluating learning schemes statistically, and visualizing the input data and the result of learning. As well as a variety of learning algorithms, it includes a wide range of preprocessing tools. This diverse and comprehensive toolkit is accessed through a common interface so that its users can compare different methods and identify those that are most appropriate for the problem at hand. All algorithms take their input in the form of a single relational table in the ARFF format. The easiest way to use Weka is through a graphical user interface called Explorer. This gives access to all of its facilities using menu selection and form filling.

The Weka contains a collection of visualization tools and algorithms for data analysis and predictive modelling, together with graphical user interfaces for easy access to this functionality. Advantages of Weka include:

- Free availability under the GNU General Public License
- Portability, since it is fully implemented in the Java programming language and thus runs on almost any modern computing platform.

- A comprehensive collection of data pre-processing and modelling techniques.
- Ease of use due to its graphical user interfaces.

Weka supports several standard data mining tasks, more specifically, data pre-processing, clustering, classification, regression, visualization, and feature selection. All of Weka's techniques are predicated on the assumption that the data is available as a single flat file or relation, where each data point is described by a fixed number of attributes (normally, numeric or nominal attributes, but some other attribute types are also supported). Weka provides access to SQL databases using Java Database Connectivity and can process the result returned by a database query.

Weka's main user interface is the Explorer, but essentially the same functionality can be accessed through the component-based Knowledge Flow interface and from the command line. There is also the Experimenter, which allows the systematic comparison of the predictive performance of Weka's machine learning algorithms on a collection of data sets from Kaggle.

IV. CLASSIFICATION

Decision Table Classifier: Decision Table is an accurate method for numeric prediction from decision trees and it is an ordered set of If-Then rules that have the potential to be more compact and therefore more understandable than the decision trees[7,8]. Selection to explore decision tables because it is a simpler, less compute intensive algorithm than the decision-tree-based approach.

The algorithm, decision table, is found in the Weka classifiers under Rules. The simplest way of representing the output from machine learning is to put it in the same form as the input. It summarizes the dataset with a "decision table‟ which contains the same number of attributes as the original dataset. The use of the classifier rules decision table is described as building and using a simple decision table majority classifier. The output will show a decision on a number of attributes for each instance. The number and specific types of attributes can vary to suit the needs of the task[9,10,11,12]. Decision Table classifier algorithm is used to summarize the dataset by using a decision table containing the same number of attributes as that of the original dataset. A new data item is allocated a category by searching the line in the decision table that is equivalent to the values contained in the non-class of the data item.

The entire problem of learning decision tables consists of selecting the right attributes to be included. Usually this is done by measuring the tables cross validation performance for different subsets of attributes and choosing the best performing subset. Fortunately, leave-one-out cross-validation is very cheap for this kind of classifier. Obtaining the cross-validation error from a decision table derived from the training data is just a matter of manipulating the class counts associated with each of the tables entries, because the table's structure doesn't change when instances are added or deleted. The attribute space is generally searched by best-first search because this strategy is less likely to get stuck in a local maximum than others, such as forward selection. Decision Table are one of the simplest hypothesis spaces possible and usually they are easy to understand. Decision Table builds a decision table majority classifier[13]. It evaluates feature subsets using best-first search and can use cross-validation for evaluation. An option uses the nearest-neighbour method to determine the class for each instance that is not covered by a decision table entry, instead of the table`s global majority, based on the same set of features.

ZreoR Classifier:

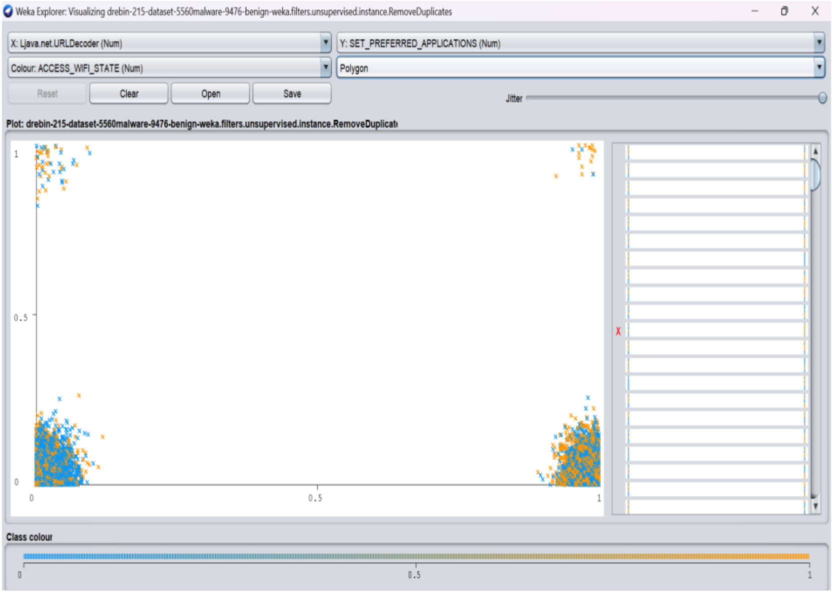
ZeroR is the simplest classification method which relies on the target and ignores all predictors. ZeroR classifier simply predicts the majority category (class). Although there is no predictability power in ZeroR, it is useful for determining a baseline performance as a benchmark for other classification methods.

M5Rule Classifier:

Generates a decision list for regression problems using separate-and-conquer. In each iteration it builds a model tree using M5 and makes the "best" leaf into a rule.

M5Rules generates a series of M5 trees, where only the "best" (highest coverage) leaf/rule is retained from each tree. At each stage, the instances covered by the best rule are removed from the training data before generating the next tree[14]. The algorithm is similar to the PART method for classification trees, except that always builds a full tree at each stage and does not employ the partial tree building speed-up of PART. M5P builds a single decision tree. It is certainly possible that an M5 rules classifier could outperform M5P on a given dataset.

Fig. 1: ZeroRClassifiers Wi-Fi state with Referred Application



Weka Explorer: Visualizing drebin-215-dataset-5560malware-9476-benign-weka.filters.unsupervised.instance.RemoveDuplicates

X: Ljava.net.URLDecoder (Num)

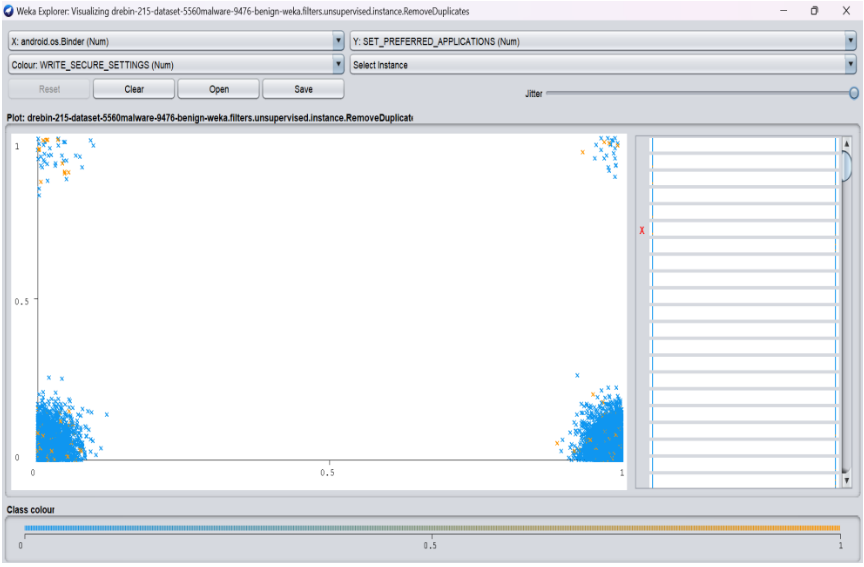
Y: SET_PREFERRED_APPLICATIONS (Num)

Colour: ACCESS_WIFI_STATE (Num)

Polygon

Plot: drebin-215-dataset-5560malware-9476-benign-weka.filters.unsupervised.instance.RemoveDuplicat

Fig. 2: M5Rules Classifiers Wi-Fi state with Referred Application



Weka Explorer: Visualizing drebin-215-dataset-5560malware-9476-benign-weka.filters.unsupervised.instance.RemoveDuplicates

X: android.os.Binder (Num)

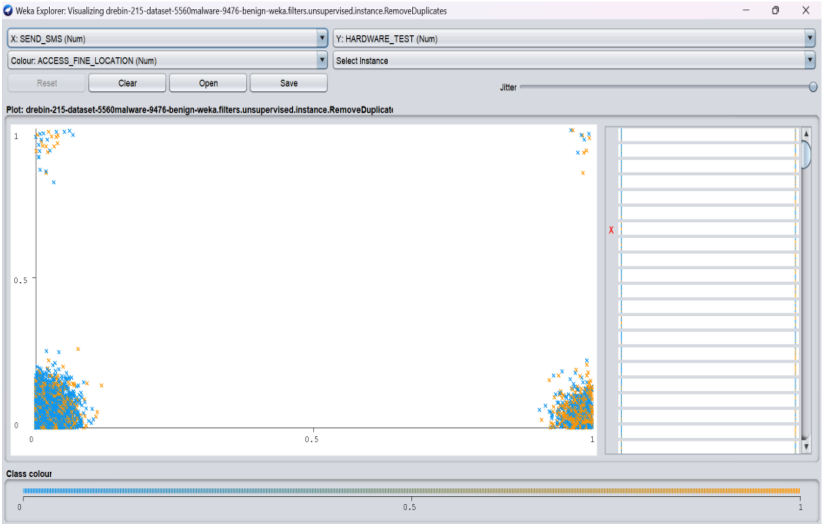
Y: SET_PREFERRED_APPLICATIONS (Num)

Colour: WRITE_SECURE_SETTINGS (Num)

Select Instance

Plot: drebin-215-dataset-5560malware-9476-benign-weka.filters.unsupervised.instance.RemoveDuplicat

Fig. 3: Decision TablesClassifiers Wi-Fi state with Referred Application



Weka Explorer: Visualizing drebin-215-dataset-5560malware-9476-benign-weka.filters.unsupervised.instance.RemoveDuplicates

X: SEND_SMS (Num)

Y: HARDWARE_TEST (Num)

Colour: ACCESS_FINE_LOCATION (Num)

Select Instance

Plot: drebin-215-dataset-5560malware-9476-benign-weka.filters.unsupervised.instance.RemoveDuplicat

TABLE1: Comparative Study

Name of Classifiers	M5Rules	Decision Tables	ZeroR
Time taken to build model:	1.04 Sec	0.07 sec	0.08 sec
Correlation coefficient:	0.964	0.9394	0.5454
Mean absolute error	13508.405	971379.82	1087916.76
Root mean squared error	40923.16	2045139	2294065.24
Relative absolute error	21.88%	53.17%	59.55%
Root relative squared error	26.62%	75.48%	84.67%
Total No. of Instances	7261	7261	7261

Table-1 represents the three classifiers M5Rules, Decision Tables and ZeroR with the WEKA tool by using these classifiers with the following contents. In these Time taken to build model, Correlation coefficient, Mean absolute error, Root mean squared error, Relative absolute error, Root relative squared error, Total no. of Instances with Ignored Class Unknown Instances.

According to Table 1 M5Rule is not superior than Decision Tables and ZeroR classifiers.

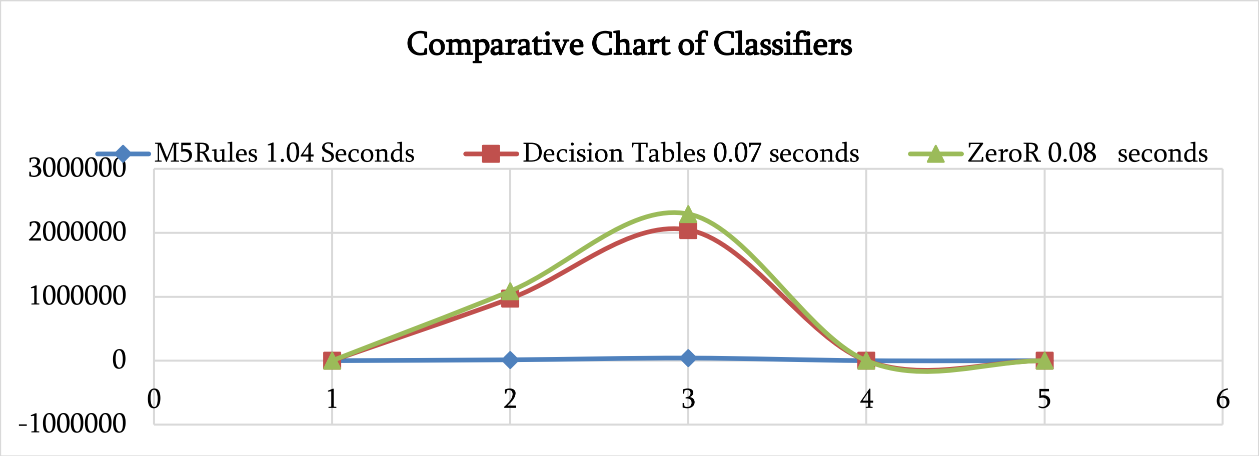


Fig. 5 - Comparative Analysis of Classifiers

V. CONCLUSION AND FUTURE SCOPE

This research proposes a malware detection module based on advanced data mining and machine learning. It can be implemented at enterprise gateway level to act as a central antivirus engine to supplement antiviruses present on end user computers. It can help protect invaluable enterprise data from security threat, and prevent immense financial damage. Data mining techniques and algorithms such as classification help to find the patterns to decide upon the future trends to expand the hospitality in this pandemic crisis.This paper focuses on the existing literature in the field of Classification Techniques (DTs, M5Rules, ZeroR) and research challenges in Data Mining. It found that there is no single technique that is consistent with all domains, and that Classification Techniques and algorithms perform better than the other existing methods. Each technique has its own strength and weakness, and can be selected based on the needed conditions. ZeroR classifier is one of the best classifiers.

VI. REFERENCES

[1]. Sunil Kumar Muttoo, Shikha Badhani, “Android malware detection: state of the art”, Springer(march 2017).

[2]. J. Han and M. Kamber, “Data Mining: Concepts and Techniques”, 2nd Ed., The Morgan Kaufmann Series in Data Management Systems, Jim Gray, Series Editor 2006.

[3]. Howard, M., Pfeffer, A., Dalai, M., &Reposa, M. (2017). Predicting signatures of future malware variants. 2017.

[4]. AbdelmonimNaway, Yuancheng LI, “A Review on The Use of Deep Learning in Android Malware Detection”(2018).

[5]. Tieming Chen, Qingyu Mao, Yimin Yang, MingqiLv, and Jianming Zhu “Tinydroid: A Lightweight and Efficient Model for Android Malware Detection and Classification”, Mobile information systems (2019).

[6]. N. Abirami, T. Kamalakannan, Dr. A. Muthukumaravel” A Study on Analysis of Various Data Mining Classification Techniques on Health Care Data” IJETAE 2013.

[7]. V. Vaithiyanathan, K. Rajeswari, Kapil Tajane, Rahul Pitale”, Comparison of Different Classification Techniques Using Different Datasets” IJETAE 2013www.sustaninlane.com

[8]. SyedaFarhaShazmeen,MirzaMustafaAliBaig"PerformanceEvaluation of Different data mining Classification Algorithm andPredictiveanalysis",IOSR-JCE2013

[9]. M.Soundarya,R.Balakrishnan"SurveyonClassificationTechniquesin Datamining”, IJARCCE201

[10]. Ms.AparnaRaj,Mrs.BincyG,Mrs.T.Mathu"SurveyonCommon Data Mining Classification Techniques”, InternationalJournalofWisdomBasedComputing2012

[11]. M.Soundarya1,R. Balakrishnan. "SurveyonClassificationTechniquesinDataMining”,International Journal ofWisdomBasedComputing 2012

[12]. PeimanMamaniBarnaghi,VahidAlizadehSahzabi,AzuralizaAbuBakar "AComparativeStudyforVariousMethodsofClassification”, ICICN2012

[13]. G. H. John and P. Langley, “Estimating Continuous Distributionsin Bayesian Classifier” Proceedings of the 11th Conference onUncertainty in Artificial Intelligence," San Francisco, 1995, pp.338-345.

[14]. Harshang G Patel, Prof. Ketan Sarvakar, "Research Challenges and Comparative Study of Various Classification Technique Using Data Mining", IJLTEMAS, ISSN 2278 – 2540, Volume III, Issue IX, September 2014, Pg. No. 170 – 176.