A STUDY ON ANDRIOD MALWARE USING VARIOUS DATA MINING CLASSIFICATION TECHNIQUES

[1]Subhash Prabhakar Kamath, [2]Vikas Boddu, [3]Gottapu Taraka Rama Rao
[1]UG Student, [2,3]Assistant Professor
[1,2,3]Computer Science and Engineering, GITAM Institute of Technology
[1,2,3]GITAM, Visakhapatnam, India

Abstract : There is a serious growth in the number of malicious apps in the Internet. Over the past few years malware has become a serious threat to many smart phone users. To defend this high rate of growth we have to create a system to detect malware apps. Data mining is a concept which we had used to categorize data, there are various algorithms and concepts which we can use to split the apps in two categories- malware or benign. In this comparative study we had analyzed various algorithms in data mining to classify android apps. The dataset used in this paper contains various permissions requested by apps from various sources in which it contains 58,519 apps out of which 53,422 were benign apps and the remaining 5,097 were malware apps. In order to classify the android apps we had performed preprocessing to the data and applied classification techniques on the data using the permissions requested by them**.**

***IndexTerms* - Andriod, Malware, Data Mining, Classification.**

I. INTRODUCTION

Smart phones are the devices which are most frequently affected by malware other than PC's.85% of the present market share is consists of android OS based devices [1], which makes it the most comfortable platform for transmission of malware.In the year 2016, spyware apps, under the category of malware, have increased by 23% in the Google play store compared to the previous year [2].

The most commonly found malware found in smart phones are -Root kits, Spyware, Adware, Trojan. There is an 54% increase in the new types of malware variants, out of which 63% steal the contacts from the users mobile and also a new malware app is introduced every 10.29 seconds in Quarter 1 of 2017 [3].
To defend Users from the malicious apps there is a need for a system to identify malware apps from benign apps. There are mainly two types of malware detection techniques -Static, Dynamic.

In Static method there are:
- Signature Based Approach.
- Permission Based Analysis.
- Virtual Machine Analysis.

In Dynamic method there are:
- Anomaly based.
- Taint Analysis.
- Emulation Based.

We have used Permission based technique to classify as it has faster Detection rates. Data mining techniques have been used for classification. Data mining is a concept used for finding patterns in the data and classifies the data according to their nature.

The detection of malware can also be done by using API calls made by the apps, STOWAWAY is a tool which uses this procedure to classify apps as malicious [5]. The detection done using PUMA detects malware apps by using permissions from mnifest.xml file by machine learning techniques [6]. SCANDAL another malware detection tool detects apps as malware by leakage of private information from source to remote server [7]. Another detection tool, DroidDetecter is malware classification engine which uses deep learning methods [8]

Rest of the paper is arranged as follows, Section II contains the related work of malware detection using classification, Section III explains the various procedures done to detect malware apps from dataset, Section IV discusses the performance results calculated by evaluating the classification algorithms and Section V concludes the overall research work done in the paper with future scope.

II. RELATED WORK

Many Efforts were put to detect malware apps by researchers. Few are shown below:

Jin Li Lichao Sun, Qiben Yan, Zhiqiang Li, Witawas Srisa-an and Heng Yen [9] performed multi-level data pruning to reduce the number of permissions and classified the android apps by using the Support Vector Machine classifier with decision tree.They had classified with an accuracy of 93.64%.

Yuxia Sun, Yunlong Xie, Zhi Qiu, Yuchang Pan, Jian Weng [10] used a tool called WaffleDectector to detect malware apps that uses data from API calls and permissions requested by apps which uses extreme learning techniques and they performed classification with an accuracy of 95.05%

Jungsoo Park, Hojin Chun, Souhwan Jung [11] classified the apps using API and Permissions and constructed a YARA rule,and matched each methods used in the application with their YARA rule by doing this they were able to detect new malware apps from the code in the apk package.

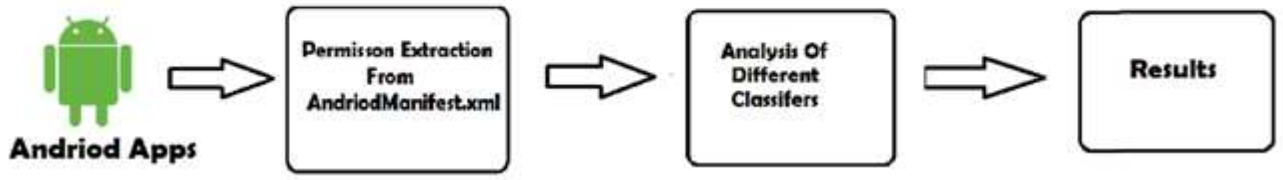
I. Burguera, U.Z., Nadijm-Tehrani,[11] created a system called CowDroid .CowDroid is a machine learning based framework that detects malware apps based on analysis of systems calls made by the apps while execution of action that requires human interaction.

III. RESEARCH METHODOLOGY

In this Study we had implemented few classification algorithms to test which algorithms produce a high accuracy rate in detecting malicious apps .To detect malware apps using data mining we must follow the steps shown below:

3.1 Data Collection
3.2 Data Preprocessing
3.3 Evaluation Of Classifiers

Above procedure have been followed to classify whether different applications are malicious or benign and the entire process is shown graphically in Fig.1.


Andriod Apps
Permisson Extraction From AndriodManifest.xml
Analysis Of Different Classifers
Results

.

Figure 1 Process of Evaluation of Classifiers

3.1 Data Collection

The data selected for this study was collected from an open source software development platform- Github [11]

The data we have collected consists of 58519 android applications that are of different target levels. Out of which 53422 are benign and 5097 are malware. Each tuple contains -34 different permissions. every permission attribute is binary valued and the class label is a nominal attribute and additional attributes that are Numeric valued .

The permissions are extracted from the andriodmanifest.xml file, present in application package, axmlprinter is used to convert the xml file to text file. After converting, each permission is noted in an csv file for that respective application from which have collected the manifest file

The attributes in android apps dataset are shown in the table below:

Table 1 Attributes List in the Andriod Dataset

ATTRIBUTES	TYPE	ATTRIBUTES	TYPE
@ATTRIBUTE SEND_SMS	Binary	@ATTRIBUTE RECORD_AUDIO	Binary
@ATTRIBUTE DELETE_SMS	Binary	@ATTRIBUTERECEIVE_BOOT_COMPLETED	Binary
@ATTRIBUTE INTERRUPT_SMS	Binary	@ATTRIBUTE RECEIVE_MMS	Binary
@ATTRIBUTE HTTP_POST	Binary	@ATTRIBUTE RECEIVE_SMS	Binary
@ATTRIBUTE DEVICE_ID	Binary	@ATTRIBUTE RECEIVE_WAP_PUSH	Binary
@ATTRIBUTE SIM_COUNTRY	Binary	@ATTRIBUTE SEND_SMS	Binary
@ATTRIBUTE INSTALLED_PKG	Binary	@ATTRIBUTE CALL_PHONE	Binary
@ATTRIBUTE LOAD_OTHER_CODE	Binary	@ATTRIBUTE CALL_PRIVILEGED	Binary
@ATTRIBUTE SUB_PROCESS	Binary	@ATTRIBUTE PROCESS_OUTGOING_CALLS	Binary
@ATTRIBUTE EXECUTE_OTHER_CODE	Binary	@ATTRIBUTE READ_CALL_LOG	Binary

@ATTRIBUTE JNI	Binary	@ATTRIBUTE READ_EXTERNAL_STORAGE	Binary
@ATTRIBUTE UNIX	Binary	@ATTRIBUTE READ_LOGS	Binary
@ATTRIBUTE INTERNET	Binary	@ATTRIBUTE ACCESS_COARSE_LOCATION	Binary
@ATTRIBUTE SET_DEBUG_APP	Binary	@ATTRIBUTE ACCESS_FINE_LOCATION	Binary
@ATTRIBUTE MODIFY_PHONE_STATE	Binary	@ATTRIBUTE BLUETOOTH	Binary
@ATTRIBUTE CAMERA	Binary	@ATTRIBUTE READ_CONTACTS	Binary
@ATTRIBUTE INSTALL_PACKAGES	Binary	@LABEL MALWARE	Nominal
@ATTRIBUTE NFC	Binary		

3.2 Data Preprocessing

The next step is to preprocess the data. For preprocessing of data we have to follow these steps below:

3.2.1 Data Cleaning

We have manually filled the missing values of the data with 0's so that there will be no problem during classifying of the data. The cleaning is performed to maintain the quality of data because without proper quality the execution time varies drastically which is not considered that good in the classification process.

3.2.2 Data Reduction

In this step we have removed unnecessary attributes from the data like application size, target level and other attributes which are in the Binary valued format and also which doesn't play any role in the classification process
Preprocessing is done to reduce most unnecessary parts from the dataset.

3.3 Data Classification

Classification is can be done by these two steps. The First Step is to learn from classification algorithms and build a classifier and the next step is to use that classifier to predict the class label and calculate the precision and accuracy

All the classifications performed in this paper are done in RapidMiner. RapidMiner is a data mining tool it was founded by Dr. Ingo Mierswa on 2007. It is mainly based on java programming language and it consists of large variety of data mining and machine learning tools for various tasks like modeling, prediction, clustering etc. [12]

The Below Shown are some algorithms which we had used

3.3.1 Naïve Bayes: It based mainly on Baye's algorithm. It is most commonly used algorithms that classifies data with a high accuracy. It uses the class conditional independence concept due to which it exhibits high learning capabilities

3.3.2 CHAID: It is a decision tree technique that uses an attribute selection method based on the statistical chi-square test. Independence of each attribute is attained using CHAID.

3.3.3 Gradient Boosted Tree: It is a technique used for regression and classification. In this a prediction model is created by ensemble of weaker models which is basically a decision tree

3.3.4 SVM: This is a linear classification technique divides the data into two distinct categories. This is mainly used to perform classification and to detect outliers. It is formally defined by a separating hyper plane. The classification is performed by the separating hyper plane that distinguishes the two categorical classes.

3.3.5 Decision Tree: It is tree -graph like structure, by which the classification process is done where each node is a test node and branch is the outcome for that test, this method works for both categorical and continuous input and output variables.

3.3.6 Linear Regression: In linear regression we fit a line to fit the attributes such that each attribute is used to predict the values of other attributes in the data set.

3.3.7 ID3: It is used to generate decision trees using top-down and greedy search approaches .Each attribute is tested at nodes and the final tree is used to classify the data

3.3.8 Deep learning: In Deep learning computer Models learn to classify data from examples given the dataset. it uses deep neural networks and deep belief networks to understand and analyze the data to further classify them.

3.3.9 K-NN: It is the one of the simplest algorithm used in machine learning for classification as well as regression. An object is classified by majority of the votes given by it's neighbors and also by the distance between the objects. The distance can either be Euclidean or Manhattan distance or whichever is specified by the user.

IV. PERFORMANCE MEASURES

The Performance of each classifier is measured using the following evaluation measures:
Precision: It is the ratio of all positive tuples that are correctly classified to the total number of positive tuples present

$$Precision = \frac{TP}{TP + FP} \qquad (4.1)$$

Sensitivity: It is the fraction of positive tuples that are correctly classified

$$Sensitivity = \frac{TP}{TP + FN} \qquad (4.2)$$

Specificity: It is the fraction of negative tuples that are correctly classified

$$Speciicity = \frac{TN}{TN + FP} \qquad (4.3)$$

Accuracy: it is the ratio of tuples which are correctly classified by the classifier to the total number of tuples present.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + TP} \qquad (4.4)$$

V. RESULTS AND DISCUSSION

We had analyzed various classification algorithms on the android applications dataset which we had collected and the performance of each classification algorithm is presented in the Table 2.

Table 2 Performance Evaluation of Various Classifiers

Classifiers	Accuracy	Precision	Sensitivity	Specificity
Naïve -Bayes	92.96	56.25	86.2	93.6
K-NN	96.29	100	57.38	100
Decision Tree	96.85	92.04	69.86	99.42
Random Forest	94.14	92.91	35.49	99.74
Gradient Boosted Tree	97.39	88.27	80.79	98.97
CHAID	98.36	91.52	89.46	99.21
ID3	98.64	93.52	90.66	99.4
RandomTree	91.29	76.2	14.89	99.55
Deep Learning	93.57	90.11	91.93	99.03
Neural Network	98.42	92.2	88.89	99.28
Auto MLP	98.38	83.75	74.02	98.62
Linear Regression	95.58	81.92	63.15	98.67
Logistic Regression	96.48	83.7	74.06	98.62
SVM	96.4	82.99	73.82	98.55

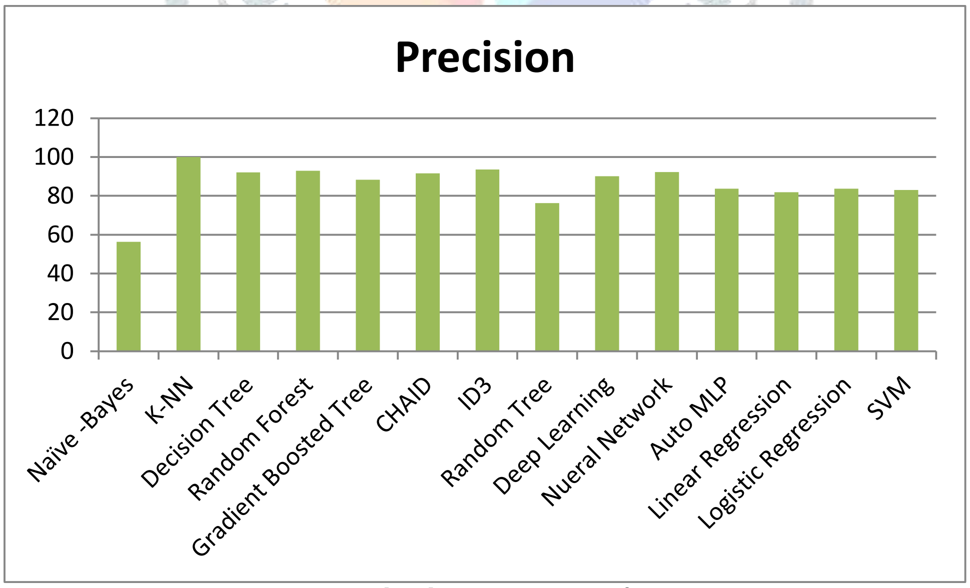


Figure 2 Graphical Representation of Precision

The Fig.2 shows the results in the measure of precision obtained by applying various classification techniques on the dataset. By observing the above graph, we can see how various classification techniques performed. The K- nearest neighbour classification technique proves to be the most efficient classifier that predicts whether an application is malware or benign with 100% precision.

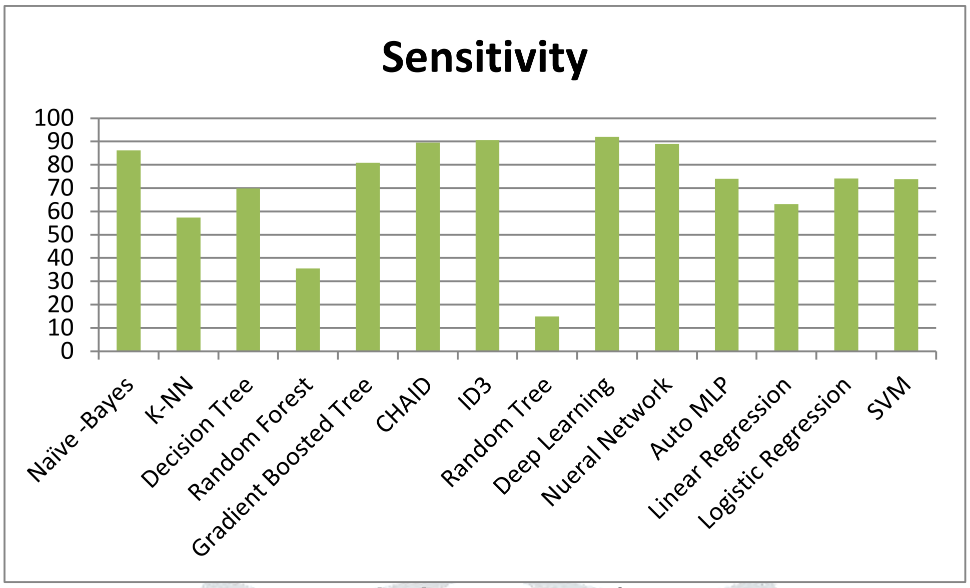


Figure 3 Graphical Representation of Sensitivity

The Fig.3 displays the sensitivity results obtained by applying various classification techniques on the dataset. By observing the graph, we can observe how various classification techniques predict with different values of sensitivity. The Deep Learning classification technique proves to be the most efficient classifier that predicts whether an application is malware or benign with 91.93% precision.

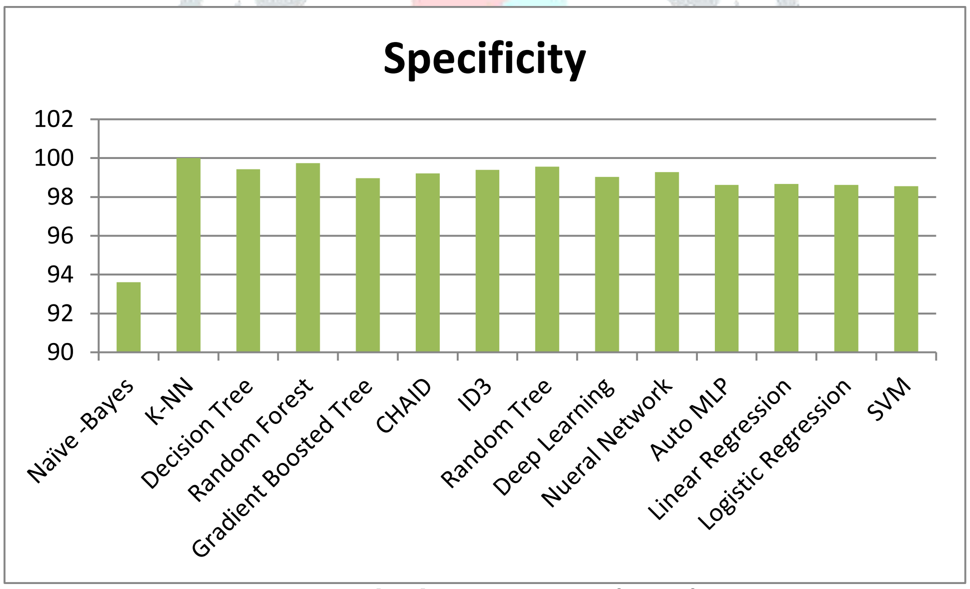


Figure 4 Graphical Representation of Specificity

The Fig.4 reflects the specificity results obtained by applying various classification techniques on the dataset. By observing the above graph, we can see how various classifiers predict with different specificity. The K-NN classification technique proves to be the most efficient classifier that predicts whether an application is malware or benign with 100% precision.

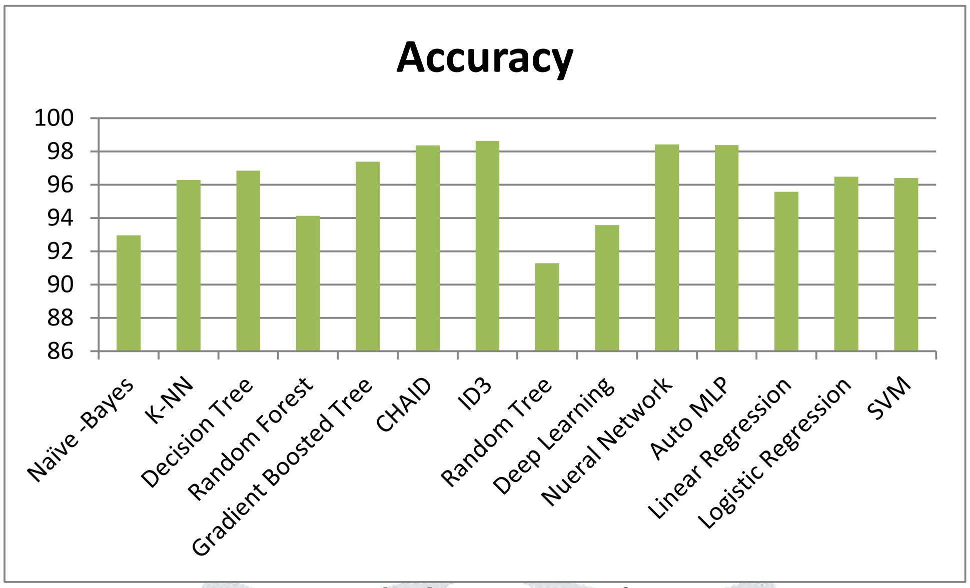


Figure 5 Graphical Representation of Accuracy

The Fig.5 visualizes the accuracy results obtained by applying various classification techniques on the dataset. By observing the above bar graph, we can see how various classification techniques predict with different accuracies. The ID3 classification technique proves to be the most efficient classifier that predicts whether an application is malware or benign with the most accuracy.

VI. CONCLUSION

With the increasing rate of android malware applications, there is an immense need for predicting the malware apps before installing them onto the phones. To achieve this we have designed a system by applying classification techniques on the dataset. In this comparative study, we have considered the most commonly requested permissions required by any application. Based on our dataset, that contains over 58519 apps, we considered only 34 out of 135 permissions to maintain the run-time performance. By observing the results, it is found that the ID3 classifier predicts malware with a high accuracy of 98.64% and the KNN classifier predicts malware with the highest precision and specificity of 100%. We were able to achieve these results due to the large amounts of data to train the system.

Further research can be done on this study by applying the ensemble methods to achieve even higher accuracies.

VII. ACKNOWLEDGMENT

We would like to convey our humble gradititude towards Prof K.Thammi Reddy,Head of the Department, Computer Science and Engineering, for the encouragement and resources provided by him to complete this work.

REFERENCES

[1] Internet Security Threat Report Volume 23, 2018 URL https://www.symantec.com/security-center/threat-report
[2] McAfee Mobile Threat Report Q1, 2018, URL https://www.mcafee.com/enterprise/en-us/assets/reports/rp-mobile-threat-report-2018.pdf
[3] New Malware Found Every 10 Seconds!, 2018, URL https://www.gdatasoftware.com/blog/2018/05/30735-new-malware-every-10-seconds
[4] Belal Amro, , 2017 Malware Detection Techniques For Mobile Devices, International Communications & Telematics Vol 7, No.4/5/6.Journal of Mobile Network
[5] Adrienne Porter Felt, Steve Hanna, Erika Chin, Dawn Song, and David Wagner, 2011. Android permissions demystified. Proceedings of the 18th ACM conference on Computer and communications security, pages 627–638.
[6] Borja Sanz, Carlos Laorden, Igor Santos, Xabier Ugarte-Pedrero, Gonzalo A´ lvarez, and Pablo Garcia Bringas, 2016.Puma: Permission usage to detect malware in android. International Joint Conference CISIS12-ICEUTE´ 12-SOCO´ 12 Special Sessions, pages 289–298
[7] Jinyung Kim, Kwangkeun Yi, Yongho Yoon, Junbum Shin, and SWRD Center. 2012. Scandal: Static analyzer for detecting privacy leaks in android applications. MoST, 12,.
[8] Zhenlong Yuan, Yibo Xue, and Yongqiang Lu. 2016.Droiddetector: android malware characterization and detection using deep learning. Tsinghua Science and Technology, 21(1):114–123

[9] Jin Li Lichao Sun,Qiben Yan, Zhiqiang Li,Witawas Srisa-an and Heng Yen.SiGPid. Identification for Machine Learning Based Android Malware Detection . 11th International Unwanted Software .ISBN 978-1-5090-4542-62017. Significant Permission Conference on Malicious and

[10] Yuxia Sun, Yunlong Xie, Zhi Qiu, Yuchang Pan, Jian Weng.Detecting. 2018. Android Malware Based on Extreme Learning Machine. 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech). ISBN 978-1-5386-1956-8

[11] Jiong Wang, Boquan Li, Yuwei Zeng. 2017. XGBoost-Based Android Malware Detection. 2017 13th International Conference on Computational Intelligence and Security (CIS) .ISBN 978-1-5386-4822-3

[12] I. Burguera, U.Z., Nadijm-Tehrani, S. ,2011. Crowdroid: Behavior- Based Malware Detection System for Android. In:SPSM’11, ACM

[13] Andriodall.csv, https://github.com/Choosue/apk-static-features-extraction/tree/master/weka_arff/

[14] RapidMiner, https://rapidminer.com/