Malware detection on android using Adaboost algorithm

MSc Research Project

Msc Cyber Security

Liston Pallippattu Mathai

X20126433

School of Computing

National College of Ireland

Supervisor: Imran Khan

National College of Ireland
MSc Project Submission Sheet
School of Computing

Student Name: Liston Pallippattu Mathai

Student ID: X20126433

Programme: MSc Cyber Security **Year:** 2020 - 2021

Module: MSc Research Project

Supervisor: Imran Khan

Submission Due Date: 16-8-2021……

Project Title: Malware Detection on android using Adaboost Algorithm

Word Count: 5009…… **PageCount**……19……

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Liston Pallippattu Mathai

Date: 16-8-2021

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□

Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Malware detection on android using Adaboost algorithm

Liston Pallippattu Mathai

X20126433

MSc in Cybersecurity

National College of Ireland

Abstract

Advanced expansion of technologies and mobile devices has directed in to key cyber-attacks in the contemporary ages. Andriod is a famous operating platform, which made use in tablets and smartphones and also befit a fundamental target of untrustworthy obligations performed by various malwares. It is reported there is over 50 billion android OS download and more than 1.3 millions of android applications accessible in the official market of google and the popularity is still increasing. Broadening of the consumers in this particular operating system welfares the enemies to produce immense malwares which distress defectively with time. This study observes that, even though there are methods to detect malware in android system using machine learning techniques where the accuracy seems to be low. Machine learning Is an approach used in the past years to detect malwares which not produce a better result for the recent malwares developed (Yerima et al., 2015). Ensemble learning is an approach similar to machine learning which can give a healthier outcome, this encourage me to create a system to detect the malware in android using one of the ensemble learning alogorithm named “Adaboost” which is a boosting technique. Boosting is a method of merging distinctive low accuracy model which create high accuracy model. Mentioned Adaboost algorithm will be trained on Andriod data to understand how efficient is the algorithm to detect malware on android.

Keywords: Malware, Andriod, Machine Learning, Ensemble Learning, Boosting, Adaboost.

1 Introduction

In the topical era, the speedy progress of high-speed mobile communication system has caused

in transportable devices such as smartphones and tablets coming beyond common nowadays. Individuals make use of phones for calling, texting, browsing the internet etc., and doing several other deeds. Studies shows concerning mobile operating systems used in the world shows, android has most users or android is the popular operating system used world-wide with 71.81% (statcounter.com, n.d.). Android is a major popular operating system that helps users to download various applications. These acceptance causes high damages like hacking, staling of data etc. Even though there are various methods to detect malware on android, it always fails to detect newly created malwares which motivated to study on machine learning that can give a better result for this issue. Existing solutions for anti-virus detection are less capable of avoiding exponentially rising threats since they are more relied on signature based detection. Various android developers modified android system, and the attackers also modified the files where by a click of the security system it can inject malware.

Suleiman described malware as it is nothing but a malicious program that get inserted to a user device without their knowledge which cause high damage to the information's and data's. Numerous supervised algorithms random forest, SVM and mixture of clustering and classification has been made use to classify the malwares in mobile. Some of the works are grounded on one classification and some with multi-level classification, among this multi-level gives higher accuracy. Here in this context I propose a method which is Adaboost, a boosting technique which is efficient to detect malware on android (Rocca, 2019). A unique data set has been anticipated in the initial part and then it provides similar weight to individually of the inspection. If a prediction goes wrong while applying first learner, a high weightage is given to observation for the one which is predicted incorrectly. This method stays and keep counting the learners unless it touches the limit in accuracy.

1.1 Research aim

Since the development of computer The research aims to investigate and obtain understanding on the technique of malware detection on android and the workflow by their performance with the help of Adaboost algorithm. Reliable secondary means are responsible for attaining information on the system and its infrastructure for examining the processes in terms of avoiding mischievous attacks and keep the confidential data safe.

1.2 Research question

How efficiently Adaboost algorithm can detect malware on android?

In the recent years there is a vast growth in the use of smartphones and other devices which use android operating system which helps to raise the popularity of android OS. There are billions of users the specific operating system, this popularity results in huge number of cyberattacks. Rapid increase of this malware attack has led me to propose a new research topic. Even though there are numerous studies has been made in this area to prevent malware attacks all the traditional methods which implemented are not much capable of identifying new and unknown malwares. This paper discusses an ensemble learning method which uses Adaboost technique (Wang et al., 2018).

Previous research has been done by Andrew H sun (Rana & Sung, 2020) by using different approaches in ensemble learning including sacking, boosting etc., this study came to a conclusion that the boosting algorithm can efficiently decrease most of the main errors by differentiating strong learner from the weak learner. This paper motivated me to use boosting technique in my project. While researching on boosting technique I came to a conclusion that Adaboost can be more efficient in malware detection on android. Adaboost is an ensemble learning algorithm which boost the performance of any machine learning algorithm and has low generalization error, it can work with a broad range of classifiers since it is very easy to implement.

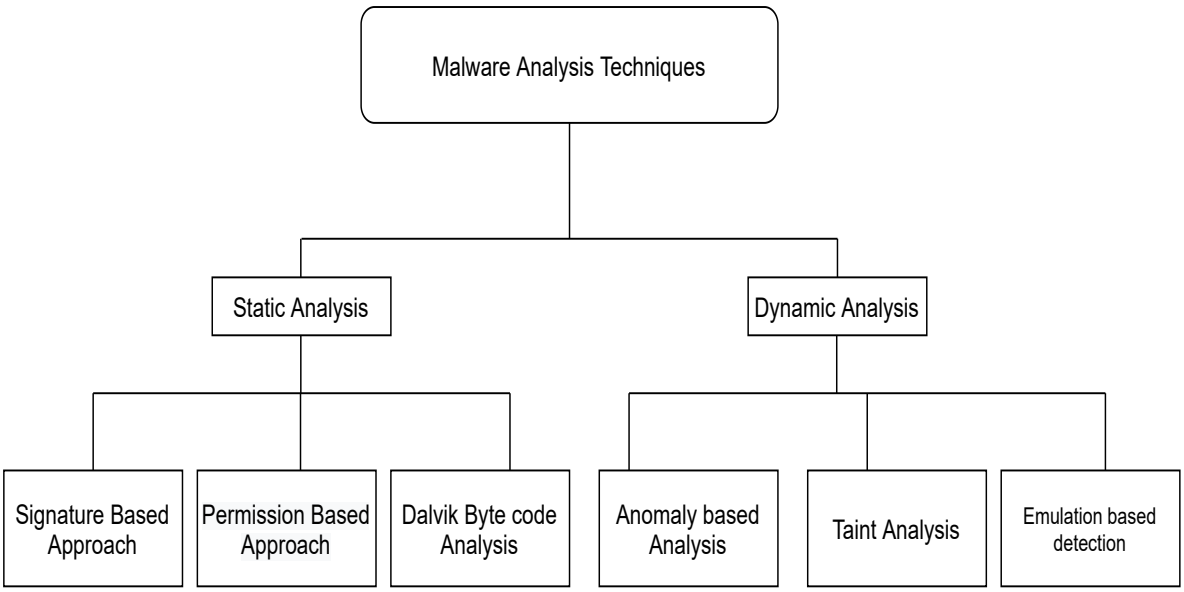
2 Related Work

This section deliberates the studies and researches conducted previously on the topic android malware detection. Society relays on technology nowadays for most activities. First android phone was invented in the year 2008 which has then blasted since then. People started using android for various uses which attracted cyber criminals to make attacks. As reviewed above development of technologies triggered a comprehensive usage of android phones and tablets that works on android operating system. Tablets and smart phones has become a daily escort to human beings in the present world, since they are much helpful for the communicate, purchase and to do events needed for a person. Society totally trust on smart phones for distinctive purposes which upsurge the risks of attacks like data stealing etc. Many of the malware programmed applications which is designed

by attacker's leads to kindof different attacks. In this regards of the attacks researchers compelled themselves to develop malware detection algorithms with the help of machine learning and through other methods.

Suleiman explained, a malware is a program which is injected in various processes by the attackers without the understanding of the owner, which cause damage to personal data's which is later used for immoral purposes.

In the year 2010 a malware was detected on android for the first time and then later the number increased steadily. (Zhou et al., 2012) conducted a study which discovered there are distinctive category of malicious applications has been exposed in the consequent year. Succeeding that a study has been made by (Di Cerbo et al., 2011) in the same year recognize the malwares in the mobile applications, where it trusts totally on model features of the security, later summarized six behavior of the malware that is interrupting SMS, data stealing, misuse telephony services etc, conducted by Parvez Faruki (Faruki et al., 2015) analyzed and précised security problems. Traditional methods which used last few years have become poor in this era since it cannot compromise to the newly build android applications. Detections methods are divided in to mainly two categories shown as follows,



A static analysis method was proposed by Dong-jie Wu (D. J. Wu et al., 2012) to find out malware in android with the help of API tracing of calls. A mechanism named feature-based configuration and tracing of API calls were used to detect malware by introducing a DroidMat system. System was much fruitful in detecting malware, but it has few of the drawbacks as well. One of the main drawback with the system was it can only identify the malware on single sample where also it has less capable of detecting the families like Droid Kungfu and Basebridge (Saracino et al., 2018).

Later another statistical model was suggested by Andria sarancino so-called a MADAM detector. MADAM is a Multi-level anomaly detector. Main objective of the system is to find the misconduct of applications and to take further actions which stop harm to the device. Experiment is conducted using different data sets and it reported 96 % of accuracy.

Jun song (Song et al., 2016) then presented a static recognition method for android malware detection, where an integrated detection has been introduced with a filtering method. Cut-out the workload which result in high efficiency was the main advantage of this work also the outcome created shows a rate of 98.80%. There are numerous researches piloted during the mid of the last period.

A method of identifying malware by captivating the raw opcode by LSTM related HDN, was prpopsed by Jipei Yan (Yan et al., 2018) which related to hierarchical denoising and to identify malware on android and they make use of a hierarchical structure because the sequence is too lengthy.

Many of the false positive has being treated by static detection methods and also it is not able to control the dynamic code loading process. Speed is considered to be another drawback and a sandbox method was proposed by the researchers which is able to run the application inside. Ambra deontis (Demontis et al., 2019) proved that by a research that it is much secure when we use machine learning technique. An adversary aware methodology has been followed since the crafted algorithm can be much resistant to avoidance. Training of linear classifier which has extra feature weight using theoretically learning algorithm which results in advanced system security, in the future different methods have been executed by using machine learning technique.

A category based malware detection method was proposed by Huda Ali Alatwi(Ali Alatwi et al., 2016) which, specifically emphases to refining the performance of the model under undisputable category. There was a high dependency for accuracy on the quality of the features. In this study to increase the performance clasiifier training has been done on every group separately. A system has been later proposed by Ali Feizolla (Feizollah et al., 2017) named andriodialysis that is proficient of inspecting two unrelated intent object entitled implicit and explicit where the study evaluates the adeptness of the intends as the feature for detecting malware and make use of a comprehensive dataset which involve 7406 applications. There was an efficiency of 91% reported when manipulating android intend and for the consent it was 83%. There was a major limitation for the paper which was that the indent feature is not considered to be the final last solution.

Later studies showed that dynamic analysis can give higher accuracy, a study by Paramvir and Aravind (Mahindru & Singh, 2017) gives a clear idea how dynamic analysis is accurate. They used several machine learning methodologies in many data sets collected and the evaluation of the same gives very high success rate and also the selection data sample with less number and the classifier caused a false positive. A related study has been conducted by Taniya Bhatia (Bhatia & Kaushal, 2017) which doubles how efficient is is dynamic analysis for detecting malware on the android. A monkey tool which produce gestures has been used in this method and also they recommended to use a virtual box for the execution, they collected many traces and the data collected are used for analysing various learning methodologies which provided an accuracy level more than 80% and the paper concluded that dynamic analysis is an efficient method to detect malware. W C Wu and S H Hung proposed a dynamic framework named DriodDolphin (W. C. Wu & Hung, 2014) to detect malware on android. It uses a method of running application in a virtual environment and hence extracting the information from API calls and 13 activities. The proposed work resulted in an accuracy of 86.1 percentage and F1 score of 87.5 percentage. Another study named EnDriod (Feng et al., 2018) was proposed by P.Feng, J.Ma, CSun, X.X and Y Ma which is an effective framework. Proposed paper aims the implementation of highly accurate detection of malware based on numerous types of dynamic performance features. One of the major advantage is that EnDriod accept the feature selection algorithm in order to avoid the noisy or inappropriate features and critical behavior feature has been extracted. OmniDriod (Martín et al., 2019) was an another study conducted by A Martin, R Lara Cabrera and D Camacho which is a big and widespread dataset which includes 22000 real set of malware and goodware samples, the main objective of the tool is to help the the

researchers and creators when creating anti-malware tools. Dataset was created with the help of AndroPyTool and a set of clasiifiers.

Hongliang Liang (Liang et al., 2017) proposed an end to end detection model which uses call structure by the way of text and process as extraction. To identify the malware, the call sequence has been observed which is the end to end technique model. One of the major difference with the similar studies are nothing but the abstraction of deep seated information from the long sequence and that not relate to any of the pattern and the study reported an accuracy level of 93%. Yi Zhang (Zhang et al., 2018) proposed a DeepclassifyDriod approach which is a three step method uses a conventional neural network. There are three main steps involved named extraction, detection and embedding. Proposed study is much faster when compared with SVM and KNN technique and also the method gives a high accuracy outcome more than 90. Following this a system named MalDozer has been proposed by ElMouatez Billah karbab (Karbab et al., 2018) which is an effective framework that grounded on excavating of sequence by the use of neural network. Here in this method this proceeds sequence API call method as the input which can be seen in DEX file. Throughout the training method proposed approach is able to identify many of the dangerous patterns which affect the system by the use of raw method call in the code, this method gives a high accuracy with an F1 score of 96 percentages.

Hybrid analysis is another field of study similar to static and dynamic, as we discussed above there are various studies conducted on static and dynamic method. Rehman and Zahoor Ur (Rehman et al., 2018) proposed a different area of approach that is the hybrid method, which is a machine learning assisted signature and heuristic detection method. A reverse engineering technology is used in the android applications which is proficient of extracting manifest files, binaries and make use of machine learning algorithms to detect the malware. Major objective of this study is that to detect the malwares before it is getting installed to the device. A comparison is made between both constant string of downloaded APK before the installation process. A dissimilarity or a mismatch in the string means it is malicious. Accuracy level achieved foe the system was around 80 to 85 percentage. Following that another hybrid paper was proposed by Yu Liu (Liu et al., 2016) which claims high accuracy rate and well-ordered complexity. Various features has been accomplished for the process and the training procedure is performed with different classifiers known as KNN and SVM. Even if the application is not

able to decompiled, the particular approach spontaneously executes the methods for detection. This paper concluded with an accuracy result of 93.33 percentage.

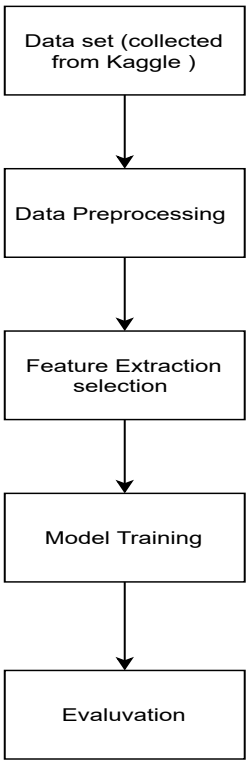
A researcher named Wei Wang (Wang et al., 2018) discusses a paper which express to mine 23744340 features and they use collaboration of various classifiers in higher accuracy. Large number of features has been extracted in this proposed work from the APK files which is able to use for the detection of the behavior. The features mentioned that is, 23744340 features have been extracted from a single application that drop to 11 types. SVM method has been employed which is used to rank the features which result in improving the efficiency and later several classifiers to such as Random forest, support vector machine, K-nearest neighbours, Naïve bayes and classification a regression tree. Investigations summarize that ensemble method can be more powerful than the base classifiers and it can give more accuracy than all others moreover it can detect newly created malwares as well. Andrew H sun (Rana & Sung, 2020) a researcher proposed a study on different strategies in machines learning by employing some of the methodologies in ensemble learning such as sacking, boosting etc., and the studies concluded that the boosting algorithm can play an important role and it can work efficiently because it reduce most of the major errors by distinguishing strong learner form the weak. The perception of this learning is nothing but it's a method of diversification centered on the evaluation of machine learning.

Plaindriod (Idrees et al., 2017) which is a novel malware detection on android system using ensemble learning was proposed by Fauzia idress, muttukrishnan Rajarajan, Muro Conti and Thomas M.chen which is considered to be the first solution that make use of a mixture of permissions and intentds supplemented with the method which can give accurate detection on malware. In this study they applied the suggested approach in to 1745 real world applications which resulted in a high accuracy and the study concluded that the proposed work is much effective in the detection of malware applications. This work is considered to be the first work which associate permissions and intents that helps for collaborative detection of malware.

3 Research Methodology

Above discussed are many of the traditional approaches in which they are not able to identify latest malwares. Machine learning and ensemble learning are two main methods used presently

which give high Accuracy, F1 score and recall metrics used at the time of training. Below figures shows the steps to be taken for the procedure.



A malware android data set should be selected to start the procedure, trailing that the data is pre-processed. Missing values are being checked in this step and also all the null values are unfurnished and it is executed towards data generalization process. model training would be the next part, where we will be using a part of the data set for training the model. Once after the training, we make use of rest of the data set for testing and the result would be evaluated using Accuracy.

Feature selection: - One of the major step to be processed in a machine learning model is Feature selection. In this step we will be deciding what features should be used for training which helps to reduce the time of training. Chi square method has been used for evaluating the significance of extricated feature.

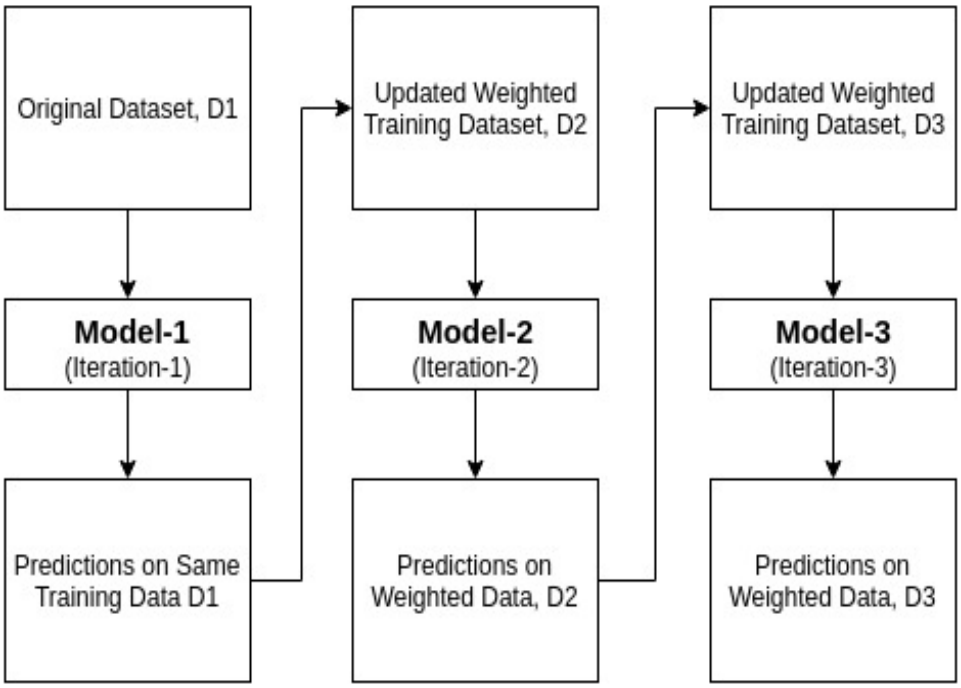
Model Training: - Adaboost module is used for for training the extracted features which will be then passed towards to it.

Evaluation: - Performance of the model is evaluated using the metrics like Accuracy.

4 Design Specification

Adaboost is the algorithm that we used in the implementation which use a boosting technique and is proposed in 1996 by Yoav Freund and Robert Schapire also it is an iterative ensemble method. Adaboost classifiers made a solid classifier which associate various weakly presenting classifier which results to get high accuracy classifier. The idea is to fix weights of classifiers and training the data sample in individual iteration such that it ensures the accurate prediction of unfamiliar observations. There are two conditions to be met for Adaboost algorithm, that is training should be done for classifiers interactively on different weighted training examples and in each training it tries to provide an excellent for the examples by minimizing training error. Working of Adaboost algorithm is as follows (*AdaBoost Classifier Algorithms Using Python Sklearn Tutorial - DataCamp*, n.d.),

1. A training subset is selected randomly.
2. Depending on the accurate prediction from the last training, a training set is selected which train the Adaboost machine learning model.
3. Assigns the higher weight to incorrect organised observations so that the observations will get the high probability for classification in the preceding iteration.
4. Also, It assigns the weight to the trained classifier in each iteration according to the accuracy of the classifier. The more accurate classifier will get high weight.
5. This process iterate until the complete training data fits without any error or until reached to the specified maximum number of estimators.
6. To classify, perform a "vote" across all of the learning algorithms you built.



4.1 Accuracy

Accuracy is considered to be a statistical measurement scale for a model. It is considered that the number of values which a model can calculate efficiently. It can be calculated as follows,

Accuracy = (True Positive + True negative/ (True positive + True negative + False positive + False negative)

Actually true values are represented by true positive and the model is also true. Actually negative values are represented by true negative values and model is also will be expected negative. False positive is the sum of actually negative values and model will be true. False negative is the sum of actually true values and model will be expected negative.

4.2 F1-Score

F1 score is nothing but it's the harmonic mean of the precision and recall values (Liu et al., 2016).

Precision = True positive / (True positive + False positive)

Recall = True positive / (True Positive + False negative)

True positive is nothing but they are actual values and the model also expected to be true. Real negative is actually negative values and the model is also expected to be negative. False positive is considered to be actual negative values and model is expected to be true. False negative is actually the number of actual costs and the model is expected to be negative.

5 Implementation

5.1 Environment Setup:

The projected model is coded in the python programming language. Anaconda software, and Jupyter Notebook (IDE) is used for writing and executing the code.

5.2 Dataset Description:

Data set is taken from Kaggle (*Kaggle: Your Machine Learning and Data Science Community*, n.d.) which is a publically available platform to download various datasets. It is downloaded in CSV format which consist of 214 mobile application and 15037 samples of application features, The dataset feature involves trsact, onService-Connected, ServiceConnection and send SMS etc.

5.3 Package Installation

Various packages and libraries are installed to perform the research they are as follows,

Pandas : Used to read the data set, that is the dataset.

Numpy : Used for array operations

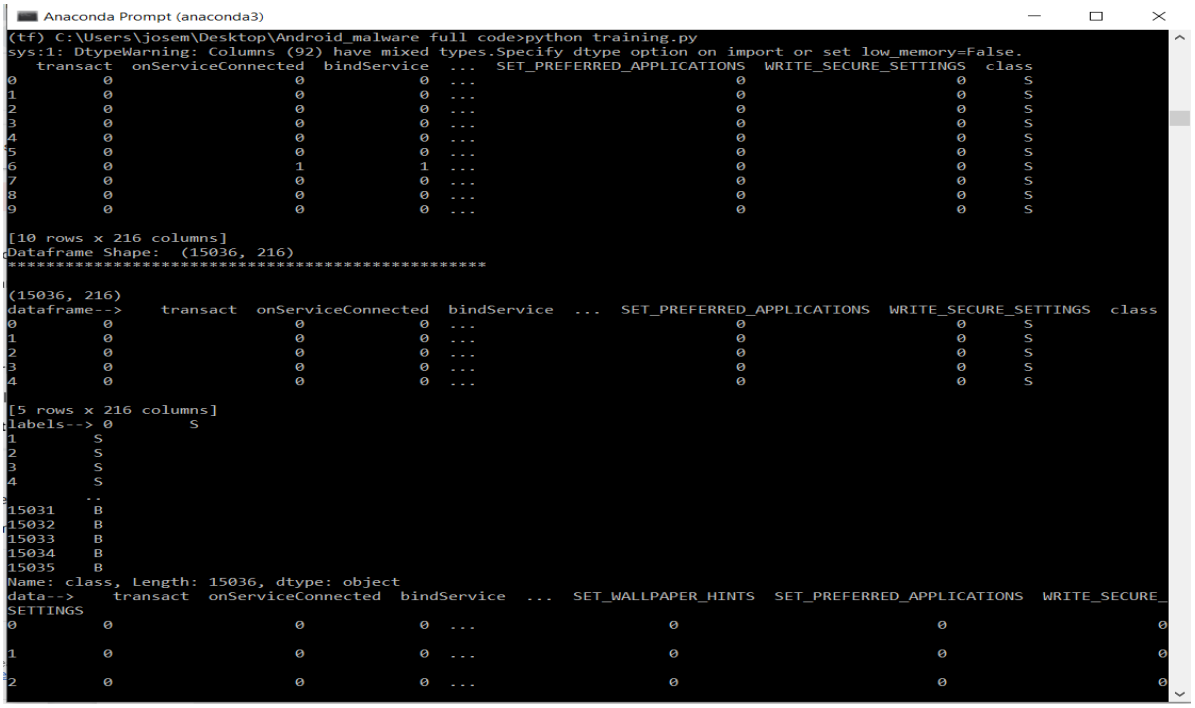
Pickle : Used to save the model

Metrics : calculate and print the accuracy

Selectkbest : Used for the selection of features

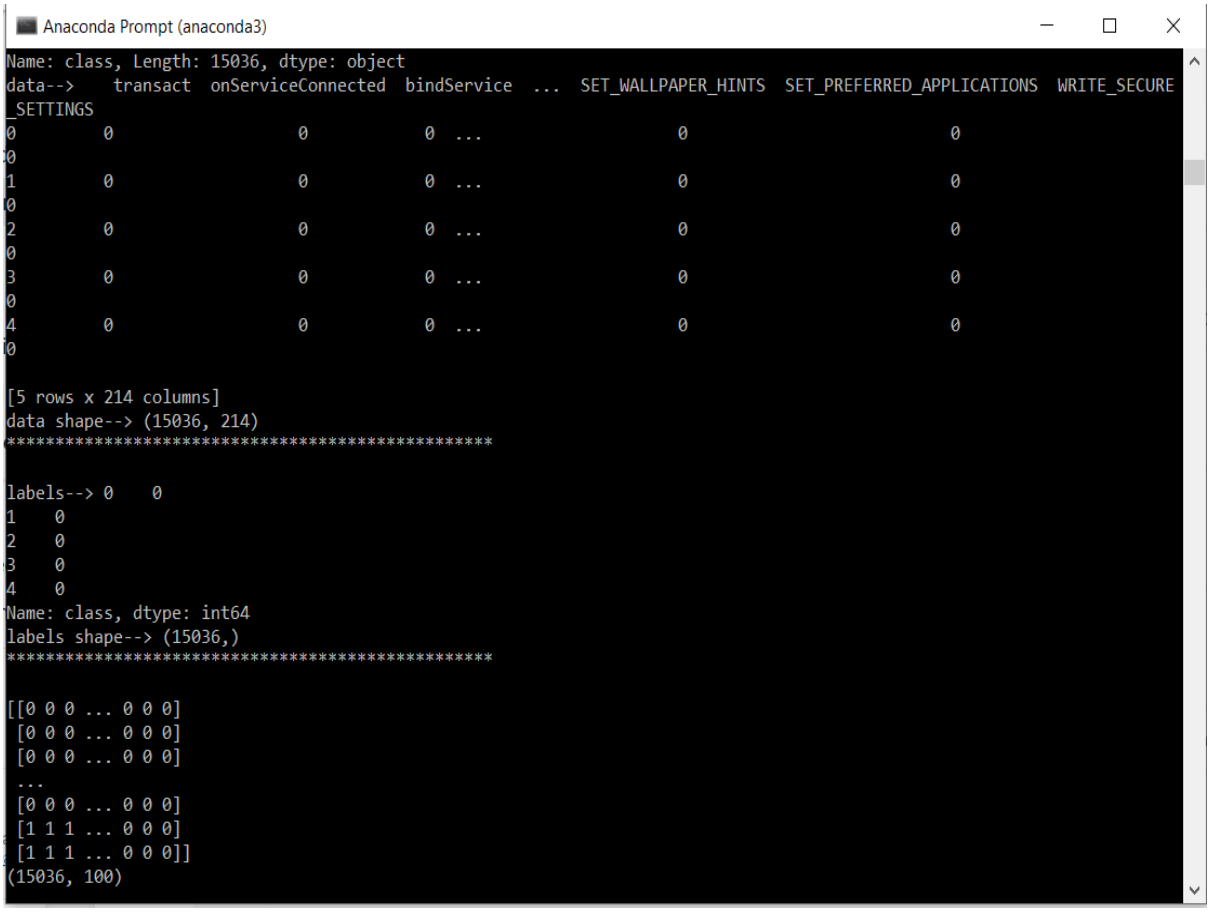
5.4 Pre-processing

Pre-processing is the first procedure to be carried out, where a basic statistical description of the feature is done, cleaning of the data set missing values are checked and made sure it doesn't contain any of such values. Following ae the main process performed in pre-processing.
In this process we drop all the null columns from the data set and fill the null values with zeros there after we separate data and labels from the data set. We fill null values in the labels with "B" and then we replace "S" and "B" that is class and the labels with "0" and "1" respectively where zero denotes the malware class and one denote the normal class. Object type column in data are removed since we are dealing with integer type data, object type contains both string and numbers and then convert this data and labels to numpy array.



5.5 Feature selection

Feature extraction is an important process used for the selection of correct number of features to train the model and remove unwanted features to increase the precision and to reduce training time. Data features and target sets are separated and the features are evaluated using the chi-square method. Using selectkbest chi2 method we select the best 100 feature from the data set.



5.6 Training and testing

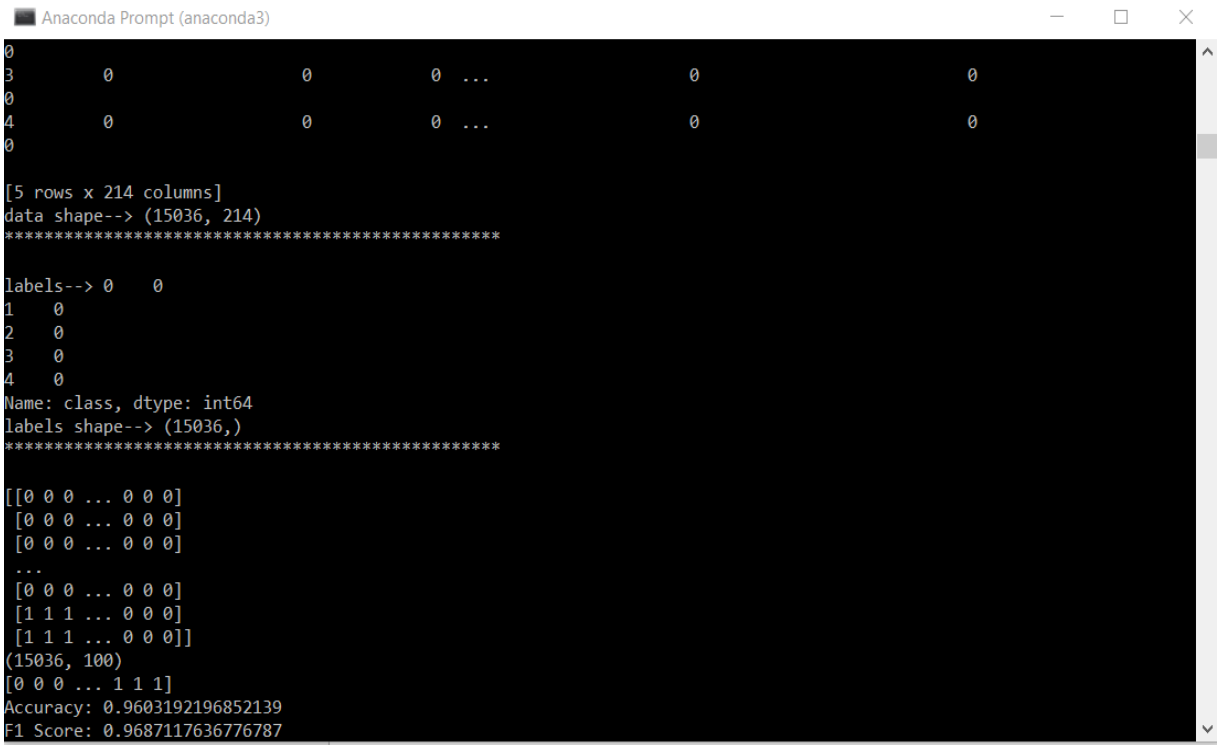
Data and labels has been split with the help of train test split method, an Adaboost model has been created and the splitted data and labels has been fitted to the created Adaboost model and we train the model using training data set data's and the labels. Trained model has been then saved. Finally, we calculate the accuracy and F1 score of the trained model using the testing set data's.

6 Evaluation

Adaboost model has been implemented and performance has evaluated. Below given table shows the evaluation result for different single machine learning algorithms (Milosevic et al., 2017).

Algorithm	Precision	Recall	F-Score
C4.5 decision trees	0.827	0.827	0.827
Random forest	0.871	0.866	0.865
Bayes Networks	0.747	0.747	0.747
SVM with SMO	0.879	0.879	0.879
JRip	0.821	0.819	0.819
Logistic regression	0.823	0.822	0.821

The accuracy and F1 score calculated when we use the particular data set is 96.03% and 96.87% respectively. Given below fissure shown the result for the same.



Malware detection on android is a topic which has been done using different methods. There are a few important papers which I have researched on this topic and most of the previous research gives less accuracy compared to my result. Comparison of those papers are shown below,

Research Paper	Summary	Comments	My methodology using Adaboost Algorithm
Malware Detection using MADAM detector	Multilevel anomaly detector.	Accuracy 96%	Accuracy 96.03%
Detection using Andriodialysis.	Inspection using two unrelated intent objects.	Accuracy 91%	
Study by Bhatia & Kaushal, 2017	Dynamic analysis for android malware detection	Accuracy more than 80%	
Research by W. C. Wu & Hung, 2014	Dynamic framework DriodDolphin	Accuracy 86.1%	
Study by Liang et al., 2017	End to end detection using call structure	Accuracy 93%	
Study by Liu et al., 2016)	Hybrid method for malware detection	Accuracy 93.33%	
Study by Song et al., 2016	Static method for malware detection	Accuracy 98.80%	

7 Conclusion and Future Work

The growth in the acceptance of smartphones remains to develop and this is what the reason Android malware is becoming a beneficial dealing for immoral thespians. As witnessed here in this research, the conventional security methods are not capable in justifying the threat of intrusion outstanding to the malware. As attackers use high-level methods and events, there is an awful necessity for an effective precaution to safeguard Android systems from criminal intrusions. Our research question was to find whether Adaboost model would be efficient on detecting malware on android and we were able to get an accuracy of 96.03%, which seems to be satisfactory and hence we can conclude that Adaboost is an efficient algorithm to detect

malware on android.

For the feature work, other ensemble learning boosting techniques like XG Boost can be implemented and can be compared with the existing model moreover the neural networks also can be performed to see the difference.

8 References

AdaBoost Classifier Algorithms using Python Sklearn Tutorial - DataCamp. (n.d.). Retrieved August 15, 2021, from https://www.datacamp.com/community/tutorials/adaboost-classifier-python?utm_source=adwords_ppc&utm_campaignid=898687156&utm_adgroupid=48947256715&utm_device=c&utm_keyword=&utm_matchtype=b&utm_network=s&utm_adpostion=&utm_creative=332602034352&utm_targetid=dsa-429603003980&utm_loc_interest_ms=&utm_loc_physical_ms=1007850&gclid=EAIaIQobChMItMz6r_6y8gIVVuDtCh3rZQKsEAAYASAAEgLI__D_BwE

Ali Alatwi, H., Oh, T., Fokoue, E., & Stackpole, B. (2016). Android Malware Detection Using Category-Based Machine Learning Classifiers. *Proceedings of the 17th Annual Conference on Information Technology Education*, 54–59. https://doi.org/10.1145/2978192.2978218

Bhatia, T., & Kaushal, R. (2017, October 18). Malware detection in android based on dynamic analysis. *2017 International Conference on Cyber Security And Protection Of Digital Services, Cyber Security 2017*. https://doi.org/10.1109/CyberSecPODS.2017.8074847

Demontis, A., Melis, M., Biggio, B., Maiorca, D., Arp, D., Rieck, K., Corona, I., Giacinto, G., & Roli, F. (2019). Yes, Machine Learning Can Be More Secure! A Case Study on Android Malware Detection. *IEEE Transactions on Dependable and Secure Computing*, *16*(4), 711–724. https://doi.org/10.1109/TDSC.2017.2700270

Di Cerbo, F., Girardello, A., Michahelles, F., & Voronkova, S. (2011). Detection of malicious applications on android OS. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6540 LNCS*, 138–149. https://doi.org/10.1007/978-3-642-19376-7_12

Faruki, P., Bharmal, A., Laxmi, V., Ganmoor, V., Gaur, M. S., Conti, M., & Rajarajan, M. (2015). Android security: A survey of issues, malware penetration, and defenses. *IEEE Communications Surveys and Tutorials*, *17*(2), 998–1022. https://doi.org/10.1109/COMST.2014.2386139

Feizollah, A., Anuar, N. B., Salleh, R., Suarez-Tangil, G., & Furnell, S. (2017). AndroDialysis: Analysis of Android Intent Effectiveness in Malware Detection. *Computers and Security*, *65*, 121–134. https://doi.org/10.1016/j.cose.2016.11.007

Feng, P., Ma, J., Sun, C., Xu, X., & Ma, Y. (2018). A novel dynamic android malware detection system with ensemble learning. *IEEE Access*, *6*, 30996–31011. https://doi.org/10.1109/ACCESS.2018.2844349

Idrees, F., Rajarajan, M., Conti, M., Chen, T. M., & Rahulamathavan, Y. (2017). PIndroid: A novel Android malware detection system using ensemble learning methods. *Computers and Security*, *68*, 36–46. https://doi.org/10.1016/j.cose.2017.03.011

Kaggle: Your Machine Learning and Data Science Community. (n.d.). Retrieved August 15, 2021, from https://www.kaggle.com/

Karbab, E. M. B., Debbabi, M., Derhab, A., & Mouheb, D. (2018). MalDozer: Automatic framework for android malware detection using deep learning. *DFRWS 2018 EU - Proceedings of the 5th Annual DFRWS Europe*, *24*, S48–S59. https://doi.org/10.1016/j.diin.2018.01.007

Liang, H., Song, Y., & Xiao, D. (2017). An end-To-end model for Android malware detection. *2017 IEEE International Conference on Intelligence and Security Informatics: Security and Big Data, ISI 2017*, 140–142. https://doi.org/10.1109/ISI.2017.8004891

Liu, Y., Zhang, Y., Li, H., & Chen, X. (2016). A hybrid malware detecting scheme for mobile Android applications. *2016 IEEE International Conference on Consumer Electronics, ICCE 2016*, 155–156. https://doi.org/10.1109/ICCE.2016.7430561

Mahindru, A., & Singh, P. (2017). Dynamic Permissions based Android Malware Detection using Machine Learning Techniques. *Proceedings of the 10th Innovations in Software Engineering Conference*, 202–210. https://doi.org/10.1145/3021460.3021485

Martín, A., Lara-Cabrera, R., & Camacho, D. (2019). Android malware detection through hybrid features fusion and ensemble classifiers: The AndroPyTool framework and the OmniDroid dataset. *Information Fusion*, *52*, 128–142. https://doi.org/10.1016/j.inffus.2018.12.006

Milosevic, N., Dehghantanha, A., & Choo, K. K. R. (2017). Machine learning aided Android malware classification. *Computers and Electrical Engineering*, *61*, 266–274. https://doi.org/10.1016/j.compeleceng.2017.02.013

Rana, M. S., & Sung, A. H. (2020). Evaluation of Advanced Ensemble Learning Techniques for Android Malware Detection. *Vietnam Journal of Computer Science*, *07*(02), 145–159. https://doi.org/10.1142/s2196888820500086

Rehman, Z. U., Khan, S. N., Muhammad, K., Lee, J. W., Lv, Z., Baik, S. W., Shah, P. A., Awan, K., & Mehmood, I. (2018). Machine learning-assisted signature and heuristic-based detection of malwares in Android devices. *Computers and Electrical Engineering*, *69*, 828–841. https://doi.org/10.1016/j.compeleceng.2017.11.028

Rocca, J. (2019). *Ensemble methods: bagging, boosting and stacking | by Joseph Rocca | Towards Data Science*. https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205

Saracino, A., Sgandurra, D., Dini, G., & Martinelli, F. (2018). MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention. *IEEE Transactions on Dependable and Secure Computing*, *15*(1), 83–97. https://doi.org/10.1109/TDSC.2016.2536605

Song, J., Han, C., Wang, K., Zhao, J., Ranjan, R., & Wang, L. (2016). An integrated static detection and analysis framework for android. *Pervasive and Mobile Computing*, *32*, 15–25. https://doi.org/10.1016/j.pmcj.2016.03.003

statcounter.com. (n.d.). *Mobile Operating System Market Share Worldwide | StatCounter Global Stats*. Retrieved April 19, 2021, from https://gs.statcounter.com/os-market-share/mobile/worldwide

Wang, W., Li, Y., Wang, X., Liu, J., & Zhang, X. (2018). Detecting Android malicious apps and categorizing benign apps with ensemble of classifiers. *Future Generation Computer*

Systems, *78*, 987–994. https://doi.org/10.1016/j.future.2017.01.019

Wu, D. J., Mao, C. H., Wei, T. E., Lee, H. M., & Wu, K. P. (2012). DroidMat: Android malware detection through manifest and API calls tracing. *Proceedings of the 2012 7th Asia Joint Conference on Information Security, AsiaJCIS 2012*, 62–69. https://doi.org/10.1109/AsiaJCIS.2012.18

Wu, W. C., & Hung, S. H. (2014). DroidDolphin: A dynamic android malware detection framework using big data and machine learning. *Proceedings of the 2014 Research in Adaptive and Convergent Systems, RACS 2014*, 247–252. https://doi.org/10.1145/2663761.2664223

Yan, J., Qi, Y., & Rao, Q. (2018). LSTM-Based Hierarchical Denoising Network for Android Malware Detection. *Security and Communication Networks*, *2018*. https://doi.org/10.1155/2018/5249190

Yerima, S. Y., Sezer, S., & Muttik, I. (2015). High accuracy android malware detection using ensemble learning. *IET Information Security*, *9*(6), 313–320. https://doi.org/10.1049/iet-ifs.2014.0099

Zhang, Y., Yang, Y., & Wang, X. (2018). A Novel Android Malware Detection Approach Based on Convolutional Neural Network. *Proceedings of the 2nd International Conference on Cryptography, Security and Privacy*, 144–149. https://doi.org/10.1145/3199478.3199492

Zhou, W., Zhou, Y., Jiang, X., & Ning, P. (2012). *Detecting repackaged smartphone applications in third-party android marketplaces*. 317. https://doi.org/10.1145/2133601.2133640