EPiC Series in Computing

Volume 91, 2023, Pages 19–26

Proceedings of 38th International Conference on Computers and Their Applications

Viability of Machine Learning in Android Scareware Detection

Aayush Gautam and Nick Rahimi
School of Computing Sciences & Computer Engineering
The University of Southern Mississippi, USM
aayush.gautam@usm.edu, nick.rahimi@usm.edu

Abstract

With the rapid increase in the use of mobile phones and other technologies, there has been a proportional growth in malware that tries to collect sensitive user data. Android is the most popular operating system for smartphones and has a great potential of becoming a target for malware threats. Scareware is a type of malware that tries to get users to provide valuable information or download malicious software through social engineering. This research aims to find out if machine learning is a viable option to prevent the consequences of scareware by accurately detecting them. In this investigation, four supervised machine learning (ML) algorithms were used in a dataset called CICAndMal2017 with 85 attributes for each of 11 Android scareware families and benign samples. We were able to achieve an accuracy of 96% which helped us to conclude that machine learning can and should be used to detect scareware. The machine learning models were then tested to calculate the accuracy for classifying each scareware family.

1 Introduction

After the first commercial release by IBM in 1994, smartphones have taken the world by storm. In 2021, there were 15 billion operating smartphones in the world, of which 2.5 billion were Android devices [1]. From bank transactions to calls and messages, people rely on their phones for everything. The convenience of using mobile apps for everything comes with the risk of a security breach by different malware. One of them is scareware. In simple terms, scareware is a form of malware that scares people into installing harmful software on their devices. They are socially engineered to cause the user to panic by showing "problems" on their phones or tablets which require to be immediately fixed [2]. Because of our reliance on technology, it is essential to research models that can properly detect malware. While there has been a few research into scareware detection before, the accuracies have been on the lower end which makes the result less effective.

Social engineering is at the heart of scareware. As shown in Fig.1, they usually show up as ads and popups saying something like: “Virus Found, Action Require” and try to either download some malicious files or make the victims voluntary give away their own personal data.

According to Miguel Morales, 70,000 people are exposed to scareware every day [3]. In March 2019, Office Depot and Support.com settled charges of $35 million for tricking their customer into believing that their computer has been affected by malicious software [4]. Even back in 2010, the FBI charged 3 people from Illinois in connection with a scheme that caused Internet users in more than 60 countries, including the United States, to buy more than $100 million worth of bogus scareware software [5]. Figure 1 shows a sample popup message.

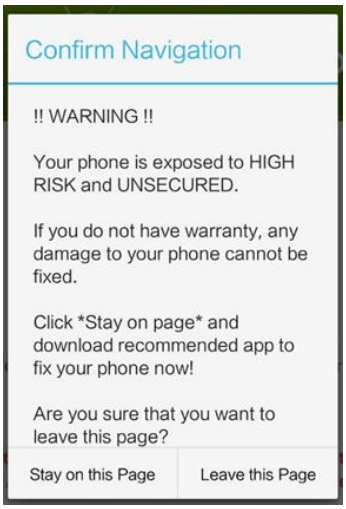


Figure 1: An Example of Scareware and Popup Messages

In this research, we are proposing a program that will be able to detect different scareware, thus preventing users from being targeted in different kinds of attacks. Four supervised machine learning models: Decision Tree (DT), Random Forest (RF), Naive Bayes (NB), and K-Nearest Neighbors (KNN) are used to get the most accurate result.

The paper contains six more sections where Section Two discusses about the dataset that was used for training the model, Section Three explains the data preprocessing that was carried out, Section Four describes the four kinds of machine learning algorithms that were used, Section Five analyses the results obtained, Section Six briefly describes each family and presents the results obtained by running the model on each scareware family, and finally, Section Seven provides a conclusion based on the evaluation results

2 Dataset

In this research, the CICAndMal2017[5] dataset has been used to build and test the models. This dataset was created by running both malware and benign applications on real smartphones so that runtime behavior modification of advanced malware samples that are able to detect the emulator environment can be detected. Data for 11 Scareware families were collected which included: AndroidDefender, AndroidSpy, AVpass, AVforAndroid, FakeApp, FakeApp.AL, FakeAV, FakeJobOffer, FakeTaoBao, Penetho, and VirusShield. This dataset contains multiple features which provide insight into different aspects of network traffic of Android scareware. Features like packet

length, time-based, content-based, etc., provide comprehensive information about the behavior of Android scareware family.

3 Data Preprocessing

Because of the discrepancy between the number of benign and scareware data, a python code was written to get random 37,400 data points for benign samples and 3,400 data points for each scareware family, for a total of 74,800 data points.

The combined dataset was then discretized, and attribute selection was done by ranking the data using Information Gain. The Information's Gain [6] calculates a certain attribute's usefulness for the data classifications which will be discussed in more detail later in the paper. Initially, the dataset has 83 attributes, but after applying information gain, it was reduced to 33. The data was then separated into two parts for training and testing based on percentages, where 80% was used for training and 20% was used for testing.

4 Algorithms

After the preprocessing of the data, it was run through four different machine learning algorithms. All the algorithms used, alongside their results, are discussed below. Since we are trying to identify whether machine learning can be a viable option to counter the rapid growth in Android scareware, we attempted to use the most widely available algorithms for the research. This selection of algorithms provides a foundation for more in-depth research, as well as the potential for widespread adoption and the development of new products to counter Android scareware threats. We tried to use both linear and nonlinear algorithms to compare the performance. All the algorithms used, alongside their results, are discussed below.

a. Decision Tree (DT)

A decision tree algorithm constructs a tree-like structure using different questions to classify the data. Information Gain and Gini Impurity are commonly used to create splitting criteria for the tree. For this research, Information Gain was used. Information Gain is the difference in entropy. Entropy measures the purity of a certain split, where the values range from 0 to 1, 0 being the purest. It is calculated using the formula (for a dataset with 2 classes):

$$E(S) = -P_{+} \times log2P_{+} - P_{-} \times log2P_{-}$$

where, S is the dataset used, and P_{+} and P_{-} are the probability of the classes in the dataset on that node.

$$IG(S, a) = Entropy(S) - \Sigma (|S_V| / |S|) Entropy (S_v)$$

where S is the dataset used, S_v is the dataset at the current node, and a is the current attribute [7].

b. Random Forest (RF)

A random forest algorithm creates a 'forest' of decision trees to classify a data point. It's an extension of the bagging method introduced by Leo Breiman in 1996. Decision trees consider all the possible feature splits but random forest only and consider single splits. This way, random forest reduces the risk of overfitting and variance, but it requires more computing power and time compared to Decision Tress [8].

c. K Nearest Neighbor (KNN)

KNN is termed a lazy algorithm because it does no computation during the training part. It classifies data as the class that has the greatest number of close data points to it. There are different methods to

calculate the distance like Manhattan Distance, Euclidean distance, Hamming Distance, etc [9]. For this project, Euclidean distance was used.

$$d = \sqrt{[(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2]}$$

where n is the number of attributes in the dataset. The testing phase in KNN is where everything happens, so it takes more memory and time. KNN works better when the number of attributes is less, as the dimension of the data increases, it becomes less accurate. Many studies have shown that in large dimensions Euclidean distance is not useful anymore [10]. Parameters must be set for the number of neighbors for data to belong to a class. For this research, different values were tested for the number of neighbors and the one with the best accuracy was used. Weighting can also be done for the distance so that the data points closer to the data would have more weight than farther ones. We tested both uniform and inverse weighting and used the best result.

d. Naive Bayes (NB)

Naive Bayes is a probability-based classification algorithm. The probability models in Naive Bayes contain strong independence assumptions, which is what makes them Naive. In the project, we used the Gaussian Naive Bayes algorithm [11].

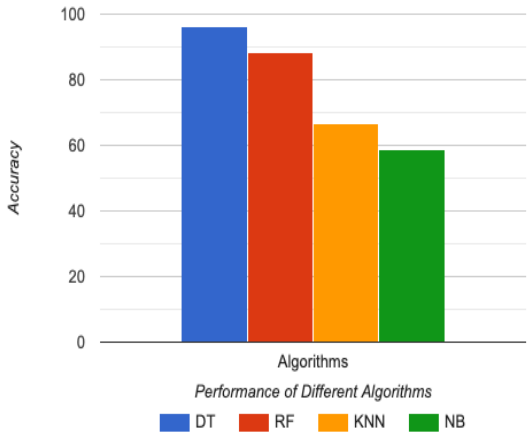


Figure 2: Performance of the Algorithms

5 Results

As shown in Figure 2, the decision tree provided the best result, with an accuracy of 96.12%. Random Forest had the second-best result with an accuracy of 88.26%. The other two algorithms showed fairly low performance with the accuracy of KNN being 66.81% and Naive Bayes being 58.60%. The reason for Decision Tree and Random Forest performing well is the presence of a few attributes that have a better correlation than others. KNN didn't perform well because the dimensions in the data were high. For Naive Bayes to perform well, it requires the predictors to be independent, but the dataset we used had dependent predictors thus it had a poor performance.

6 Analysis Of Each Scareware Family

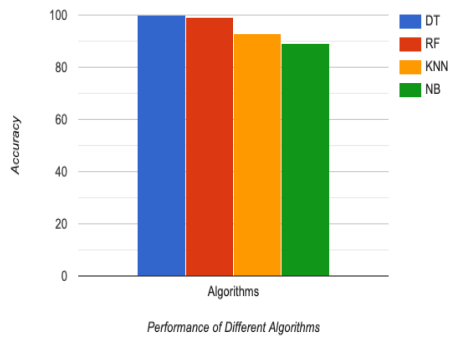
As mentioned above, 11 Scareware families were analyzed for this research: AndroidDefender, AndroidSpy, AVpass, AVforAndroid, FakeApp, FakeApp.AL, FakeAV, FakeJobOffer, FakeTaoBao, Penetho, and VirusShield. Now, we will test the models for each of these families. The results are presented in Figure 3.

Android Defender allows users to scan their phones for free, and as expected, shows that there is a presence of some kind of malware. It then prompts the user to pay for certain software that would protect their phones from the "viruses" that are present. This not only tricks people into paying but also continues providing fake updates leaving the device even more vulnerable [12][13].

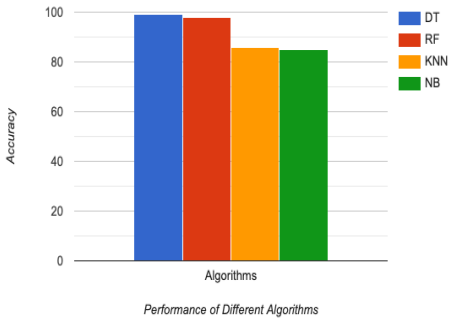
Android Spy, AVforAndroid, FakeApp, and other scareware do similar jobs where they try to mimic existing protection software and threaten you to provide your credentials.

FakeJobOffer does the job differently, where you might be prompted with a job offer but you have to either pay a certain fee, give your card details, or something similar.

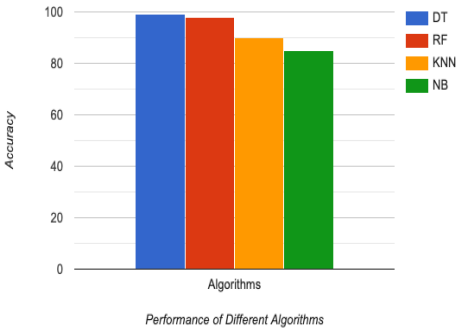
AV pass is the only Scareware family that interacts with the Anti-Virus detection system installed on the device. It leaks the detection model for the Android Malware detection system. [14]



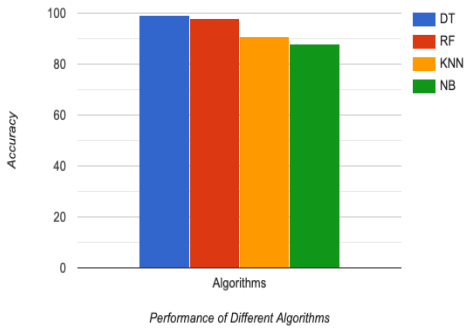
a. Accuracies of Algorithms on AndroidDefender



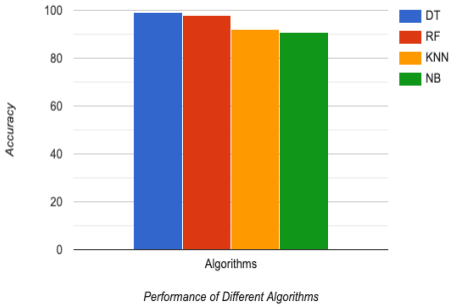
b. Accuracies of Algorithms on AndroidSpy



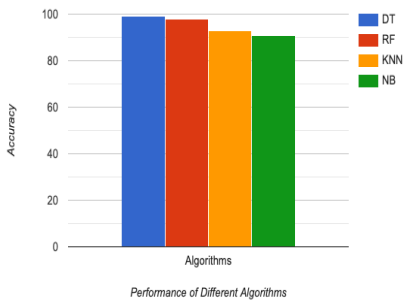
c. Accuracies of Algorithms on AV for android



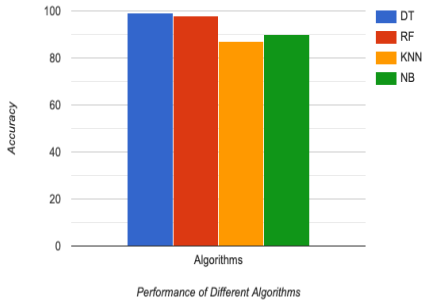
d. Accuracies of Algorithms on AV Pass



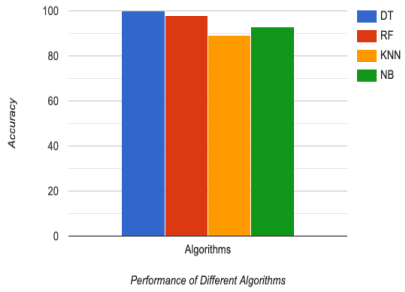
e. Accuracies of Algorithms on FakeApp



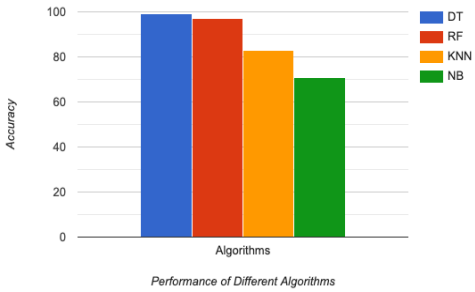
f. Accuracies of Algorithms on FakeApp.AL



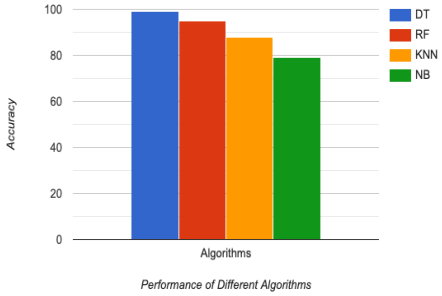
g. Accuracies of Algorithms on FakeAV



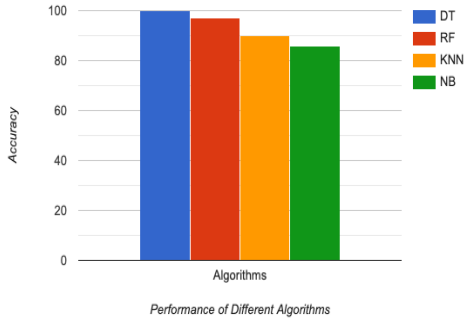
h. Accuracies of Algorithms on FakeJobOffer



i. Accuracies of Algorithms on FakeTaoBao



j. Accuracies of Algorithms on Penetho



k. Accuracies of Algorithms on VirusShield

Figure 3: Performance of the Algorithms in Classifying each scareware family.

7 Conclusion

Tree-based algorithms, namely Decision Tree and Random Forest, were found to be the most effective in detecting Android Scareware. With an accuracy of more than 96%, it can be concluded that machine learning indeed can be really effective in predicting scareware. Thus, security systems should be built using Machine Learning algorithms to protect users from the multi-million malware/scam industry.

Acknowledgments

This research was supported in part by the University of Southern Mississippi.

References

[1]. Rahimi, N., Nolen, J., & Gupta, B. (2019). Android security and its rooting—a possible improvement of its security architecture. Journal of Information Security, 10(2), 91-102.

[2]. Giles, J. (2010). Scareware: the inside story. New Scientist, 205(2753), 38-41.

[3]. Morales, Miguel. “Scareware in 2022: There's Scary Malware behind You.” My IT Guy, 1 Feb. 2022, https://www.gomyitguy.com/blog-news-updates/scareware.

[4]. Financial Fraud Enforcement Task Force Announces Results of Largest-Ever Nationwide Operation Targeting Investment Fraud. (n.d.). FBI. https://archives.fbi.gov/archives/news/pressrel/press-releases/brokentrust_120610

[5]. Arash Habibi Lashkari, Andi Fitriah A. Kadir, Laya Taheri, and Ali A. Ghorbani, “Toward Developing a Systematic Approach to Generate Benchmark Android Malware Datasets and Classification”, In the proceedings of the 52nd IEEE International Carnahan Conference on Security Technology (ICCST), Montreal, Quebec, Canada, 2018.

[6]. Simic, Milos. “Information Gain in Machine Learning.” Baeldung on Computer Science, 10 Mar. 2022, https://www.baeldung.com/cs/ml-information-gain.

[7]. Pal, M., & Mather, P. M. (2003). An assessment of the effectiveness of decision tree methods for land cover classification. Remote sensing of environment, 86(4), 554-565.

[8]. Belgiu, M., & Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. ISPRS journal of photogrammetry and remote sensing, 114, 24-31.

[9]. Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. In On the Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings (pp. 986-996). Springer Berlin Heidelberg.

[10]. Uddin, S., Haque, I., Lu, H., Moni, M. A., & Gide, E. (2022). Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction. Scientific Reports, 12(1), 1-11.

[11]. Jahromi, A. H., & Taheri, M. (2017, October). A non-parametric mixture of Gaussian naive Bayes classifiers based on local independent features. In 2017 Artificial intelligence and signal processing conference (AISP) (pp. 209-212). IEEE.

[12]. “Fake 'Android Defender' Promises Security, Delivers Malware.” NBCNews.com, NBCUniversal News Group, 31 May 2013, https://www.nbcnews.com/id/wbna52060727.

[13]. Rahimi, N., Maynor, J., & Gupta, B. (2020, March). Adversarial Machine Learning: Difficulties in Applying Machine Learning to Existing Cybersecurity Systems. In CATA (pp. 40-47).

[14]. Saad, M. H., Serageldin, A., & Salama, G. I. (2015, November). Android spyware disease and medication. In 2015 the second international conference on information security and cyber forensics (InfoSec) (pp. 118-125). IEEE.