Araştırma Makalesi
Adıyaman Üniversitesi
Mühendislik Bilimleri Dergisi
23 (2024) 190-202

ANDROID MALWARE CLASSIFICATION USING BASIC MACHINE LEARNING METHODS

Tuğba PALABAŞ[1*]

[1] Zonguldak Bulent Ecevit University, Faculty of Engineering, Biomedical Engineering Department, Zonguldak, 67100, Türkiye

Geliş Tarihi/Received Date: 01.04.2024 Kabul Tarihi/Accepted Date: 17.07.2024 DOI: 10.54365/adyumbd.1462488

ABSTRACT

Both the complexity and volume of Android malware attacks increases day by day. Thus, Android users remain vulnerable to cyber-attacks. Researchers have developed many machine learning techniques to detect, block or mitigate these attacks. However, technological developments and the increase in Android mobile devices and the applications used on these devices have also increased problems in terms of user privacy due to malware. In this study, a comprehensive study is presented on the detection and classification of malicious applications using an up-to-date dataset containing 241 features. First, incorrect, and missing data are detected, and the relevant lines are removed, and then normalization-based scaling is performed. After this preprocessing step, the dataset is randomly split into 70% training and 30% testing data using cross-validation. Finally, the classification process is carried out using 6 different machine learning methods which are Bernoulli Naive Bayes (BNB), Multi-Layer Perceptron (MLP), Logistic Regression (LOGR), K-Nearest Neighbor (KNN), Decision Tree Classifier (DTC), Random Forest (RF). Comparison of modeling results show that the RF machine learning technique can achieve the best performance with a 97% accuracy level and various other metrics for malware detection on real-world Android application.

Keywords: *Android Malware, Goodware, Classification, Machine Learning, Random Forest Algorithm*

TEMEL MAKİNE ÖĞRENİMİ YÖNTEMLERİNİ KULLANARAK ANDROİD KÖTÜ AMAÇLI YAZILIM SINIFLANDIRMASI

ÖZET

Android kötü amaçlı yazılım saldırılarının hem karmaşıklığı hem de hacmi her geçen gün artmaktadır. Bu nedenle android kullanıcıları siber saldırılara karşı savunmasız kalmaktadırlar. Araştırmacılar bu saldırıları tespit etmek, engellemek veya azaltmak için birçok makine öğrenmesi tekniği geliştirmişlerdir. Ancak teknolojik gelişmeler, Android mobil cihazların ve bu cihazlarda kullanılan uygulamaların artması, kötü amaçlı yazılımlardan dolayı kullanıcı gizliliği açısından sorunları da artırmıştır. Bu çalışmada, 241 öznitelik içeren güncel bir veri seti kullanılarak kötü amaçlı uygulamaların tespiti ve sınıflandırılması konusunda kapsamlı bir çalışma sunulmaktadır. Öncelikle hatalı ve eksik veriler tespit edilerek ilgili satırlar kaldırılmıştır, ardından normalizasyon bazlı ölçeklendirme gerçekleştirilmiştir. Bu ön işleme adımından sonra veri seti, çapraz doğrulama kullanılarak rastgele %70 eğitim ve %30 test verisine bölünmüştür. Son olarak Bernolulli Naive Bayes (BNB), Çok Katmanlı Algılayıcı (MLP), Lojistik Regresyon (LOGR), K-En Yakın Komşu (KNN), Karar Ağacı Sınıflandırıcı (DTC), Rastgele Orman (RF) olmak üzere 6 farklı makine öğrenmesi yöntemi kullanılarak sınıflandırma işlemi gerçekleştirilmiştir. Modelleme sonuçlarının karşılaştırılması, RF makine öğrenimi tekniğinin, gerçek dünyadaki Android uygulamalarında kötü amaçlı yazılım tespiti için %97 doğruluk düzeyi ve diğer çeşitli ölçümlerle en iyi performansı elde edebileceğini göstermiştir.

Anahtar Kelimeler: *Android Kötü Amaçlı Yazılım, İyi yazılım, Sınıflandırma, Makine öğrenmesi, Rastgele Orman Algoritması*

[*] e-posta1 : tugba.palabas@gmail.com ORCID ID: https://orcid.org/ 0000-0002-6985-6494

1. Introduction

The most used smart mobile devices in the world are Android-based and generally occupy a large proportion of the market share [1]. In addition, in [2], "statista" data shows that Google Play users worldwide downloaded 1 million mobile applications in July 2013 and most recently placed at 2.59 million apps in 2023 [2]. The fact that Android-based systems allow the installation of scripts from unauthenticated sources such as application stores and file-sharing websites, and the high popularity of Android for many years has also increased the interest of cybercriminals who create unsafe or unwanted malicious applications that can steal personal information or damage your device [3]. For instance, in 2023, over 100 Android apps with more than 400 million downloads combined have been infected with a new malicious Android apps [4]. Cyber attackers use malicious applications created in many different forms, such as viruses, worms, Trojans, ransomware, spyware, adware, keylogger, botnets, rootkits, scareware, and many other types, to intentionally damage devices and networks [5]. Identifying and preventing of these malwares are very important for security.

Android malware detection solutions in the literature often capitalize on machine learning approach to detect pieces of malware. The proposed models may use a variety of features, such as security settings, permissions requested, managements of the documents, API calls made, network activity, etc. The model is trained using these features and relevant samples then new application may classify as malicious or non-malicious. For example, Sahs and Khan [6] performed the classification process with the Single Class Support Vector Machine (SVM) method after the feature extraction step, which allows taking advantage of the higher computing power of a server or server cluster for the detection of malware on Android devices. The authors have shown, based on various performance criteria, that the sensitivity of the algorithm decreases as the benign sample increases. Yerima et al. [7] propose a new Android malware detection approach consisting of function-based, tree-based, probabilistic, and two rule based algorithms using parallel machine learning classifiers. Thus, they introduced a scheme that takes advantage of the strengths of classifiers with different features. Wen and Yu [8] proposed a lightweight system that first feature extraction based on static analysis and dynamic analysis. The authors applied an approach using principal component analysis (PCA) and relief to reduce the dimensions of the features. Then, they performed a classification with the SVM algorithm. The results obtained showed that the system provides an effective method for Android malware detection. Similarly, Kakavand et al [9]. also performed Android malware detection using SVM and KNN algorithms. The results obtained showed that the KNN classifier showed better performance. Li et al. [10] developed a method includes three levels of pruning to identify the most important permissions, a malware detection system based on permission usage analysis which is Significant Permission IDentification (SigPID) and machine learning methods. Tahtacı and Canbay [11] presented machine learning models using n-gram features of smali files, which are Android packages, and evaluated their performance by combining them with different feature extraction/selection methods.

As stated before, Android applications on mobile systems are increasing rapidly, following this, malware also emerges. Therefore, many researchers have examined the Android malware detection problem and introduced different methods. These studies have suggested that machine learning methods are an effective method for detecting Android malware. In this context, in recent years, there have been many reviews aimed to examine the development of malware detection techniques based on machine learning algorithms focusing on the Android operating system in a wider range [12-15].

In this study, a comprehensive analysis is presented examining the performance of 6 different machine learning methods in detecting and classifying malicious applications using a current data set containing 241 features. In this context, the performance of the algorithms is evaluated comparatively according to various metrics.

2. Materials and Methods

2.1. Dataset

In this study, the "TUNADROMD dataset", open source accessible via the Kaggle platform, consisting of 241 attributes such as, updates, media storage, google photos, permissions, file systems access, location, message reading, time setting, read social stream and 4465 sample values marked as "0" or "1" has been examined. Here, 0 represents the malware label and 1 represents the goodware label. The out of 4465 samples, 3564 are labeled malware and 901 are labeled goodware. Additionally, features between 1-214 are Permission-based features and between 215-241 are API (Application Programming Interface)-based features.

2.2. Preprocessing and Classification

First, some rows containing incorrect or missing data are removed from the original dataset with 4465 rows and the dataset is scaled using Standard Scaler (Normalization) in preprocessing step. Then a data set consisting of 650 samples is obtained by removing the duplicate lines (with drop.duplicate). Secondly, classification is performed in two phases with preprocessed dataset: learning and prediction. In the learning phase, the model is fitted based on the training data. In the prediction phase, the model is used for predicting the test data. Thus, the preprocessed dataset with 4462 rows (where 3 rows with missing or incorrect data are deleted) and 241 columns is randomly separated as 70% training (455) and 30% test (195) using hold-out cross validation. Finally, in the classification step, the success of the most popular classification algorithms BNB, MLP, LOGR, KNN, DTC and RF methods are analyzed separately. All steps of proposed android malware detection model are shown in Fig.1.

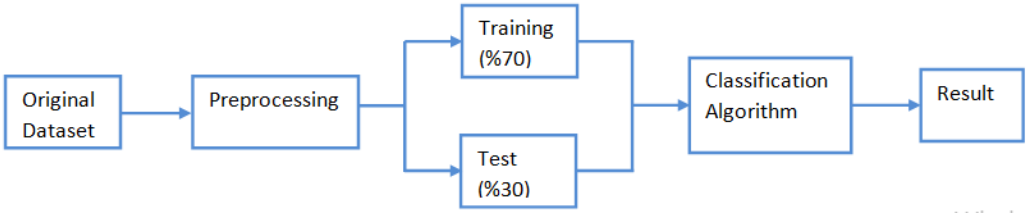

Figure 1. Proposed android malware detection steps.

2.2.1 Bernoulli Naive Bayes (BNB)

BNB is a subcategory of the Naive Bayes Algorithm that is a supervised machine learning algorithm based on Bayes Theorem which predicts the posterior probability of an event, given that another event has already occurred. Namely, if Y is an event has already occurred, the probability of occurrence of X is calculated as below [17]:

$$P(X|Y)=\frac{P(Y|X).P(X)}{P(Y)} \quad (1)$$

Here, P(X|Y) is the posterior probability which is probability of X being true given Y is true, P(Y|X) is the likelihood of the occurrence of the event namely Probability of Y being true given X is true, P(X) is the prior probability which is probability of A being true.

2.2.2.Multilayer Perceptron (MLP):

In MLP, which consists of input, hidden and output layers as shown in Fig.2, the input layer receives the incoming data and sends it to the hidden layer. The number of hidden layers may vary depending on the problem, at least one. The output of each layer becomes the input of the next layer. Thus, the output layer determines the output of the network by processing the data from the previous

layers, and the number of outputs of the system is equal to the number of elements in the output layer [18].

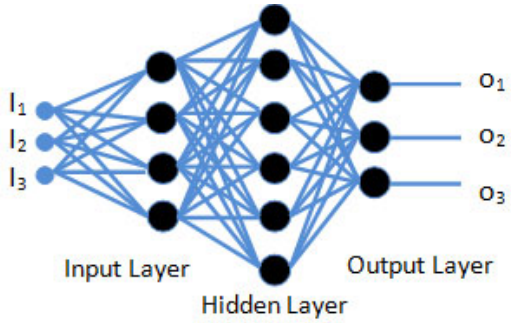

Figure 2. Multilayer Perceptron network

In this study the hyperparameters of MLP are set as default. Namely hidden layer sizes=100, activation function is "relu" which uses the rectified linear unit function, the solver for weight optimization is "adam" which is a stochastic gradient-based optimizer proposed, batch size set to "auto" that batch size=min(200, n_samples), learning rate is "constant" with learning_rate_init=0.001.

2.2.3. Logistic Regression (LOGR):

LOGR is a supervised machine learning classification algorithm used to estimate the relationship between a dependent categorical variable and independent variables. The class or category to be predicted contains data with binary values such as 0 or 1, yes or no, goodware or malware. The basis of this algorithm is the Sigmoid function. The sigmoid function, also called the logistic function, uses the result of the z linear regression equation to obtain a probability value (p) between 0 and 1 with the following equation, and class assignment is made according to this probability value [19].

$$p = \frac{1}{1+e^{-z}} \tag{2}$$

$$z = b_0 + b_1 x_1 + \cdots + b_n x_n \tag{3}$$

Here, x_i is independent variable for i=1, 2, ..., n; b_i is the slope coefficients of the logistic regression model for i = 0, 1, 2, ..., n that b_0 is the intercept of the model.

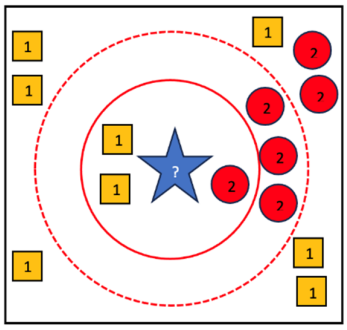

Figure 3. Determining the class (1 or 2) of a star sample using the KNN algorithm

2.2.4. K-Nearest Neighbor (KNN):

KNN algorithm predicts the class of the relevant variable based on the information in which class the nearest neighbors of the independent variables are concentrated. In this algorithm, determining the number of

neighborhoods (K) and calculating the distance of the point to be predicted from other points are two important issues [20]. For example, when K= 6, the class of the new sample (asterisk) is determined as class 2, since there are 2 sample from class 1 and 4 sample from class 2 samples in set of 6 closest sample as shown in Fig. 3.

Euclidean, Manhattan, and Chebyshev are typical measurements for distance calculation. In this study, the Euclidean distance method is used for the KNN classifier and K is determined as 5.

2.2.5. Decision Tree Classifier (DTC):

DT is a non-parametric supervised learning method used for classification and regression. It is a graphical representation for predicting the target variable by learning the decision rules inferred from features of the given dataset. It is called a decision tree because it starts with the root node, expands on further branches which represent the decision rules and includes internal node represents an attribute and leaf node represents the outcome as shown in Fig. 4.

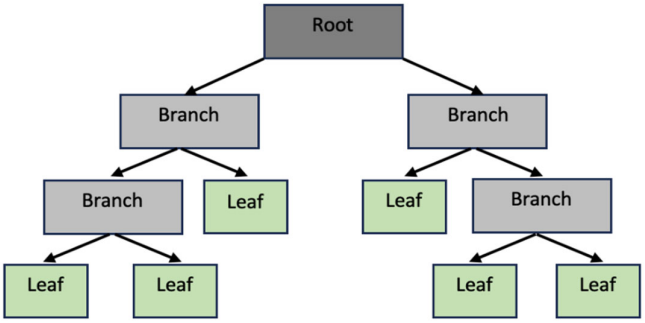

Figure 4. Decision Tree

The most important problem in decision trees is determining the root node and in what order the branching will proceed to the leaves. For this purpose, the entropy measurement shown in Equation (4), where pi is the probability value of a class, is frequently used [21].

$$E = -\sum_{i=1}^{n}(p_i)log_2\,(p_i) \tag{4}$$

Information gain is maximized by making choices that reduce the entropy value, which expresses the degree of randomness. For this, DT recalculates the error function in each question or node and selects the case with the lowest error.

Since the training time is longer when the entropy criterion is used, the Gini index as in Eq. 5 is also used as an alternative in the case of large datasets.

$$G = 1 - \sum_{i=1}^{n}(p_i)^2 \tag{5}$$

In this study, Gini index is taken into consideration as the splitting criterion in decision trees.

2.2.6. Random Forest (RF):

RF algorithm, which is an ensemble learning method, aims to increase classification success by producing multiple decision trees during the classification process. Individual decision trees, which are randomly selected subsets from the data set they are connected to, come together to form a decision forest [22]. In this study, the number of decision trees for the RF algorithm is determined as 100.

2.3. Performance Evaluation Metrics

There are many metrics used to evaluate classification success in the literature [23-25]. Precision, Specificity, Accuracy, Kappa and RMSE criteria, which are also used within the scope of this study, are shown in Equations 6-10 [26-29].

Precision (Pre) shows the extent to which malware class values are predicted to be malware and is calculated as follows:

$$Pre = \frac{TP}{TP+FP} \tag{6}$$

Specificity (Spec) indicates the extent to which non-malware class values are predicted to be non-malware and is calculated as follows:

$$Spec = \frac{TN}{TN+FP} \tag{7}$$

Accuracy (Acc-%) is the ratio of correctly predicted malware and goodware values to all values and is calculated as follows:

$$Acc = \frac{TP+TN}{\text{TP + TN + FP + FN}} \tag{8}$$

Kappa (**κ**) is the statistic that measures the agreement between two observers. This coefficient gives a strong result because it also considers whether the fit could be due to chance.

$$\kappa = \frac{p_0-p_e}{1-p_e} \tag{9}$$

Here, po the total proportion of cases correctly classified and pe is the probability of this case occurring by chance. **κ** =1 occurs when there is perfect agreement and κ =0 when the observed agreement due to chance.

RMS (Root Mean Square Error), which measures the magnitude of error, is the standard deviation of the difference between the actual values and the predicted values and is calculated as follows:

$$RMS = \sqrt{\sum_{i=1}^{n} \frac{1}{n}(y_i - \hat{y}_i)^2} \tag{10}$$

RMSE value of zero means that the model does not make any errors.

AUC (Area Under of Curve –ROC) expresses the success of the model in separating positive groups from negative groups, that is, its discrimination power, and is calculated using the values of Equation (6) and Equation (7). An AUC value greater than 0.5 means that the discriminatory power of the model is good.

3. Results

Various criteria have been used to evaluate the classification success of machine learning methods on the data set.

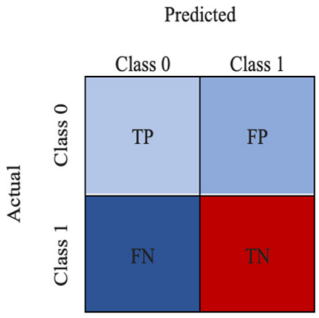

Figure 5. Confusion matrix

Firstly, the confusion matrix expressing the prediction accuracy success as a 2x2 matrix is obtained, as shown in Fig. 5. Thus, the comparison of predictions with actual values is clearly shown. Here, TP (True Positive) indicates the correctly predicted malware class value. FP (False Positive) is the incorrectly predicted malware class value. FN (False Negative) is the incorrectly predicted non-malware (goodware) class value. TN (True Negative) is the correctly predicted non-malware (goodware) class value.

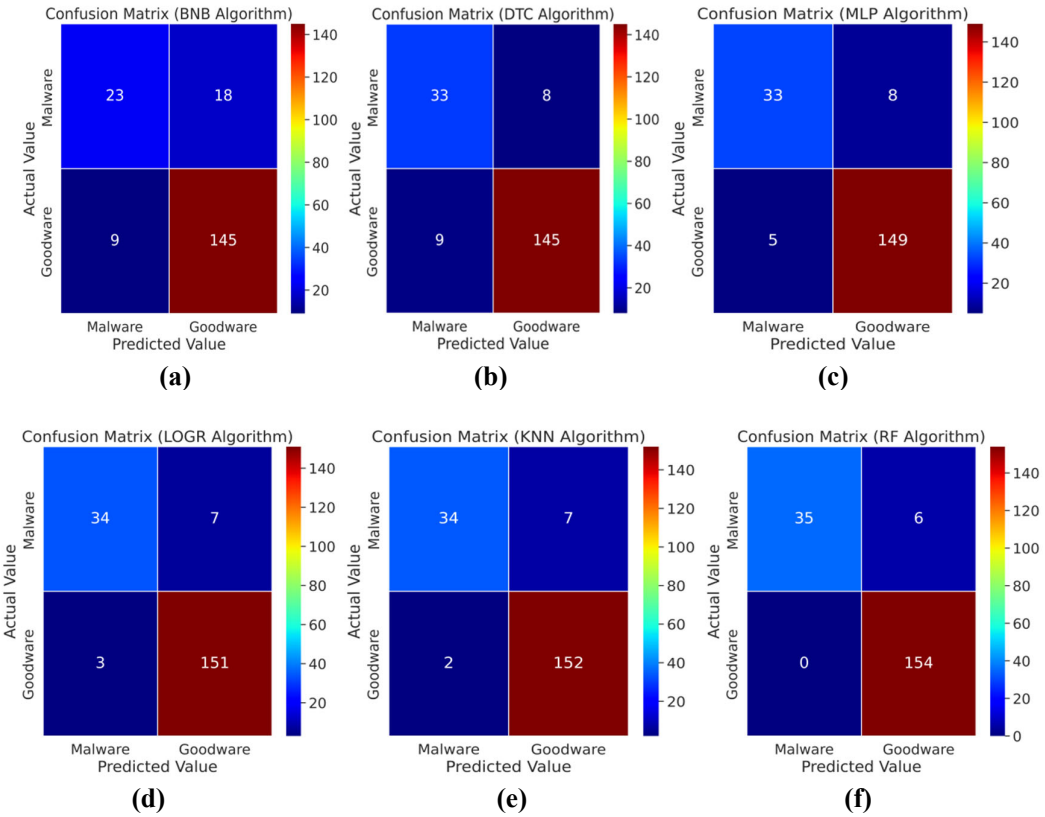

Figure 6. Confusion matrix (a) BNB Algorithm, (b) DTC algorithm, (c) MLP Algorithm, (d) LOGR Algorithm, (e) KNN Algorithm, (f) RF Algorithm.

Confusion matrices for six classifiers are shown in Fig. 6 to compare these values which are TP, FP, FN, TN which are important things to be considered while building a classification model. These matrices help to debug the model and to understand how the model behaves.

If expressed in more detail, the highest (9) FN value, which expresses the false detection of goodware samples, is obtained with the BNB and DTC algorithms and the lowest (0) with the RF

algorithm by using randomly selected 450 samples for training and 195 samples for testing from the preprossed dataset with 650. Conversely, the TP value, which is the correct detection of malware, is achieved to be lowest (145) in the BNB and DTC classifiers and highest (154) in the RF classifier. Thus, the success of the RF algorithm in android malware detection can be clearly seen from this figure.

In addition to the confusion matrix, performance criteria are calculated for all classification algorithms and are shown in Table 1, considering the order of success. As seen in the table, the best classification performance according to all metrics is achieved with the RF algorithm and android malware is classified with a success rate of 97%. BNB method obtains the lowest success. However, the execution time (s) of the algorithm is faster than other methods. RF, which is performed in the optimal training-detection and analyzing time among the 6 methods, also has the minimum error rate, RMS.

Table 1. The values of precision, specificity, accuracy, training- detection and analyzing time, kappa and RMSE metrics for BNB, DTC, MLP, LOGR, KNN and RF classifiers.

	Pre	Spec	Acc	Time	κ	RMS
BNB	0.56	0.88	86	1.14	0.54	0.37
DTC	0.80	0.88	91	1.17	0.73	0.29
MLP	0.80	0.91	93	1.73	0.79	0.25
LOGR	0.82	0.92	95	1.73	0.83	0.22
KNN	0.82	0.93	95	1.89	0.85	0.21
RF	0.85	0.94	97	1.36	0.90	0.17

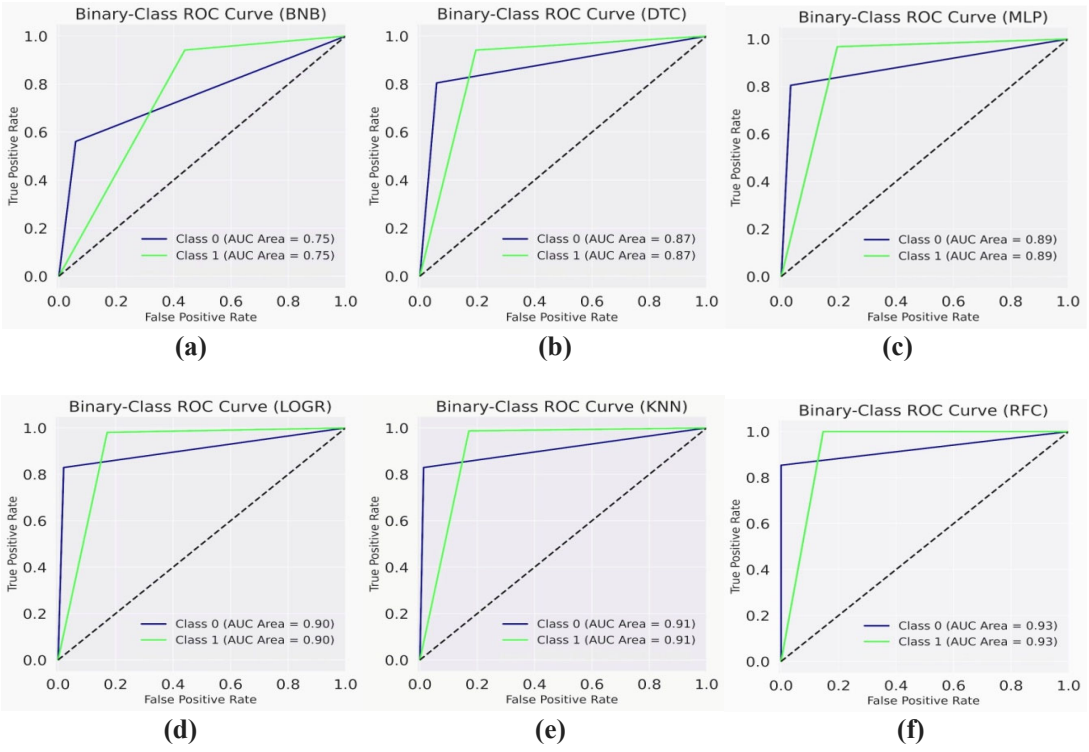

Figure 7. ROC curves (a) BNB Algorithm, (b) DTC algorithm, (c) MLP Algorithm, (d) LOGR Algorithm, (e) KNN Algorithm, (f) RF Algorithm.

The other important metric which expresses the successful of classification algorithm is ROC (receiver operating characteristic) curve [30]. As seen in the Fig. 7, the performance of classifiers based on RF algorithm is resulted in maximum Area under ROC curve (AUC) values "0.93".

Besides, ROC curves very close to the upper left corner are obtained with a perfectly discriminating distribution as seen in the six panels presented. On the other hand, the minimum AUC value is obtained with BNB, and in this panel the ROC curve is furthest from the upper left corner. Note that the ROC curves approach the True Positive Rate axis from (a) to (f).

One of the critical parameters affecting the success of the RF algorithm that exhibits the best classification performance is maximum depth information. As the maximum depth of randomly generated decision trees increases, the accuracy of the model increases, but after a certain level it decreases due to overfitting of the model. Therefore, it is important to determine it at the optimal level. In this study, the default value is set to "None". In other words, the nodes within the tree continue to grow until all the leaves become pure. These trees can grow very large and be difficult to visualize.

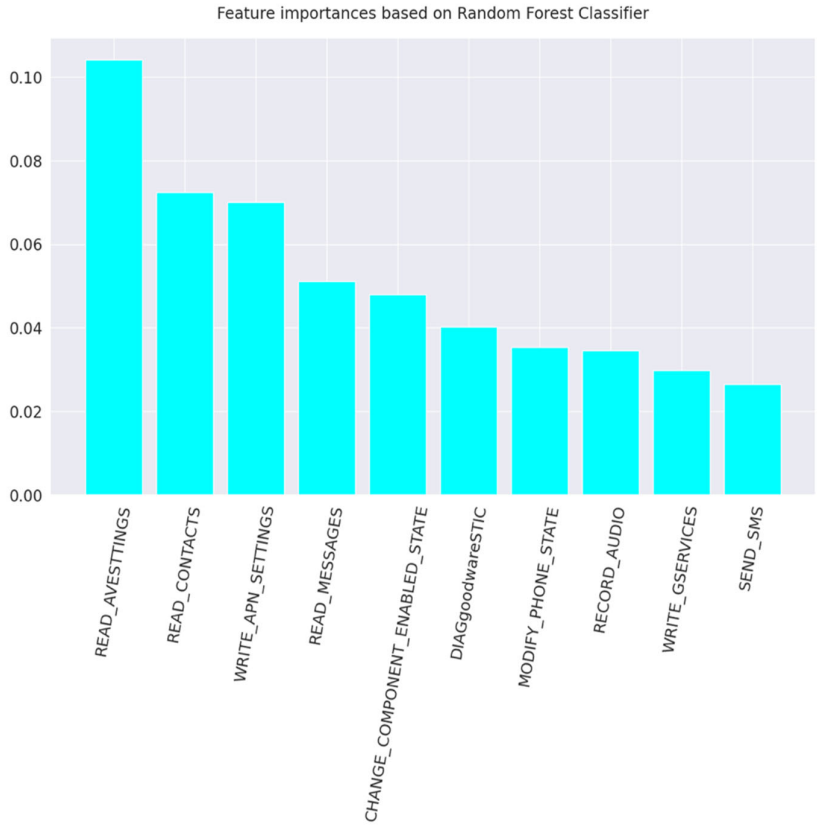

Figure 9. Feature importance based on RF classifier.

The RF algorithm determines the importance levels of the attributes in the data set in classification. Fig. 9 shows 100 different decision trees in the RF classification and the level of importance for the first 10 of 241 features in the dataset when the maximum depth is "None". As seen in the figure, the leftmost bar chart belongs to the attribute with maximum importance.

There are many suspicious permissions used for Android malware detection, it can read contact data, send or read messages or SMS. Here 10 permissions have been identified as important attributes for the RF classifier in Table 2.

Table 2. The important attributes and their definition for RF classifiers.

Attribute	**Definition**
read avesttings	App configuration file can read
read contacts	Suspicious permissions can read contact data
write apn settings	The Access Point Name setting on mobile phone
change component enabled state	There may be various components that constitute malware, this feature shows the active state of the components
DIAG goodware STIC	They are diagnostic tools to run a PC health check and fix various issues
Modify phone state	It is a system-only permission that allows situations such as changing the phone call status
record audio	suspicious permissions can record audio
write Gservices	A mysterious and undocumented Android Permission
Send SMS	SMS send information

Ambekar et al. named the NATICUSdroid dataset and TUNADROMD dataset, which contain these and many attributes, as dataset 1 and dataset 2 and compared the performances of various studies with the TabLSTMNet method they proposed and presented them in a table like Table 3. In this study, the achievements obtained by using the data2 dataset, which is up-to-date and contains several features and samples that can guarantee analysis accuracy, are taken into account.

Table 3. Comparative study of existing models with proposed method [33]

Model	**Accuracy**
Li et al. (2023) [34]	.87
Shu and Yan (2023) [35]	.89
Liu et al. (2022) [36]	.86
Islam et al. (2023) [37]	.91
Ullah et al. (2023) [38]	.92
Ambekar et al. [33] (10 fold cross validation)	.98
FR (hold out cross validation) (proposed)	.97

Using the TabLSTMNet method, the authors achieved 97.10% Android malware classification success on the NATICUSdroid dataset and 98% on the TUNADROMD dataset. The authors achieved a 98% correct classification result using the k-fold cross validation method by setting k = 10 in the preprocessing step. When they compare their accuracy performance with 5 different studies, as shown in Table 2, they state that they achieve the best performance. A close accuracy is also achieved in this study and it is added to the bottom line of the table. In the proposed study, the RF method, which has the best performance among BNB, MLP, LOGR, KNN, DTC and RF methods, provided correct

classification of %30 test data with 97% success after training step with 70% of the data using hold-out cross validation. On the other hand, the runtime value of the algorithm is not shared in the paper. However, since the dataset in question consists of 4465 instances and 241 attributes, this method, which is obtained by taking the average of 10 classification processes, will naturally require a long time.

4. Conclusions

In this paper, a machine learning-based Android malware detection method is proposed. As mentioned in the results section, Ambekar et al. [33] achieves a best performance with 98% using 10-fold cross-validation and TabLSTMNet classifier. In this study titled "TabLSTMNet: enhancing android malware classification through integrated attention and explainable AI", the authors present a table likes to Table 3 and compare the performance of 5 different studies with their own studies and state that they obtain the best success. In the proposed study, a similar success is achieved by using fewer samples for a rapid training step.

In another study, Borah et al. [32] evaluate the performance of 5 different classifiers comparatively by applying pre-processing, feature extraction and classification steps. When the authors analyze the methods', performance only based on the accuracy criterion, they state that the Extra Tree classifier show the best performance with 98.8%. However, it is not enough to evaluate classifier performance based on a single criterion. In this study, firstly, confusion matrices are obtained and Precision (0.85), Specificity (0.94), Accuracy (97%) values are calculated for 6 classifiers. Then, kappa coefficient (0.90) and RMSE (0.17) metrics are obtained. ROC analyzes based on the AUC Area (0.93) present to provide a more detailed representation. In addition, the method, which consists of only pre-processing and classification steps, with the less attribute obtains all analysis results very quickly.

As a result, it is important to detect and take precautions against changing threats due to constantly developing technology quickly and accurately. In this context, it is once again revealed that the random forest algorithm, which stands out with its high accuracy in classification, is also successful in detecting Android malware. However, it has been stated that very high performance can be achieved by using the hold out cross validation method in large data sets instead of the k-fold algorithm, which comes to the fore to guarantee success and accuracy in small data sets. In this context, it is thought that the success of the proposed method can be expressed as a generalized method in this regard by testing it with different data sets.

Conflict Of Interest

The authors declare that they have no conflict of interest

References

[1] "Smartphone OS market share 2023", 2024, https://www.idc.com/promo/smartphone-market-share.

[2] "Statista, Number of available applications in the Google Play Store from December 2009 to June 2023", 2024, https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store.

[3] Li, J., Sun, L., Yan, Q., Li, Z., Srisa-An, W., & Ye, H. (2018). Significant permission identification for machine-learning-based android malware detection. IEEE Transactions on Industrial Informatics, 14(7), 3216-3225.

[4] "Tom'sguide", 2024, https://www.tomsguide.com/news/over-400-million-infected-with-android-spyware-delete-these-apps-right-now.

[5] Ngo, F. T., Agarwal, A., Govindu, R., & MacDonald, C. (2020). Malicious software threats. The Palgrave Handbook of International Cybercrime and Cyberdeviance, 793-813.

[6] Sahs, J., & Khan, L. (2012, August). A machine learning approach to android malware detection. In 2012 European intelligence and security informatics conference (pp. 141-147). IEEE.

[7] Yerima, S. Y., Sezer, S., & Muttik, I. (2014, September). Android malware detection using parallel machine learning classifiers. In 2014 Eighth international conference on next generation mobile apps, services and technologies (pp. 37-42). IEEE.
[8] Wen, L., & Yu, H. (2017, August). An Android malware detection system based on machine learning. In AIP conference proceedings (Vol. 1864, No. 1). AIP Publishing.
[9] Kakavand, M., Dabbagh, M., & Dehghantanha, A. (2018, November). Application of machine learning algorithms for android malware detection. In Proceedings of the 2018 International Conference on Computational Intelligence and Intelligent Systems (pp. 32-36).
[10] Li, J., Sun, L., Yan, Q., Li, Z., Srisa-An, W., & Ye, H. (2018). Significant permission identification for machine-learning-based android malware detection. IEEE Transactions on Industrial Informatics, 14(7), 3216-3225.
[11] TAHTACI, B., & CANBAY, B. (2020, October). Android malware detection using machine learning. In 2020 Innovations in Intelligent Systems and Applications Conference (ASYU) (pp. 1-6). IEEE.
[12] Liu, K., Xu, S., Xu, G., Zhang, M., Sun, D., & Liu, H. (2020). A review of android malware detection approaches based on machine learning. IEEE Access, 8, 124579-124607.
[13] Kouliaridis, V., & Kambourakis, G. (2021). A comprehensive survey on machine learning techniques for android malware detection. Information, 12(5), 185.
[14] Odusami, M., Abayomi-Alli, O., Misra, S., Shobayo, O., Damasevicius, R., & Maskeliunas, R. (2018). Android malware detection: A survey. In Applied Informatics: First International Conference, ICAI 2018, Bogotá, Colombia, November 1-3, 2018, Proceedings 1 (pp. 255-266). Springer International Publishing.
[15] Pan, Y., Ge, X., Fang, C., & Fan, Y. (2020). A systematic literature review of android malware detection using static analysis. IEEE Access, 8, 116363-116379.]
[16] "Kaggle, Android Malware Detection", 2024, https://www.kaggle.com/datasets/joebeachcapital/tuandromd.
[17] Chiong, R., & Theng, L. B. (2008, June). A hybrid naive bayes approach for information filtering. In 2008 3rd IEEE Conference on Industrial Electronics and Applications (pp. 1003-1007). IEEE.
[18] Zhao, C., Gao, Y., He, J., & Lian, J. (2012). Recognition of driving postures by multiwavelet transform and multilayer perceptron classifier. Engineering Applications of Artificial Intelligence, 25(8), 1677-1686.
[19] Tsangaratos, P., & Ilia, I. (2016). Comparison of a logistic regression and Naïve Bayes classifier in landslide susceptibility assessments: The influence of models complexity and training dataset size. Catena, 145, 164-179.
[20] Viswanath, P., & Sarma, T. H. (2011, September). An improvement to k-nearest neighbor classifier. In 2011 IEEE Recent Advances in Intelligent Computational Systems (pp. 227-231). IEEE.
[21] Du, W., & Zhan, Z. (2002). Building decision tree classifier on private data.
[22] 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.
[23] Eroğlu, K., & Palabaş, T. (2016, December). The impact on the classification performance of the combined use of different classification methods and different ensemble algorithms in chronic kidney disease detection. In 2016 National Conference on Electrical, Electronics and Biomedical Engineering (ELECO) (pp. 512-516). IEEE.
[24] Palabas, T., & Eroğlu, K. (2018, May). Occupancy detection from temperature, humidity, light, CO2 and humidity ratio measurements using machine learning techniques. In 2018 26th Signal Processing and Communications Applications Conference (SIU) (pp. 1-4). IEEE.
[25] Erkaymaz, O., & Palabaş, T. (2018, May). Classification of cervical cancer data and the effect of random subspace algorithms on classification performance. In 2018 26th Signal Processing and Communications Applications Conference (SIU) (pp. 1-4). IEEE.
[26] Bowers, A. J., Sprott, R., & Taff, S. A. (2012). Do we know who will drop out? A review of the predictors of dropping out of high school: Precision, sensitivity, and specificity. The High School Journal, 77-100.
[27] Chu, K. (1999). An introduction to sensitivity, specificity, predictive values and likelihood ratios. Emergency Medicine, 11(3), 175-181.

[28] Vieira, S. M., Kaymak, U., & Sousa, J. M. (2010, July). Cohen's kappa coefficient as a performance measure for feature selection. In International conference on fuzzy systems (pp. 1-8). IEEE.
[29] Chicco, D., Warrens, M. J., & Jurman, G. (2021). The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. PeerJ Computer Science, 7, e623.
[30] Linden, A. (2006). Measuring diagnostic and predictive accuracy in disease management: an introduction to receiver operating characteristic (ROC) analysis. Journal of evaluation in clinical practice, 12(2), 132-139.
[31] "BusinessofApps Android Statistics (2023)", 2024, https://www.businessofapps.com/data/android-statistics.
[32] Borah, P., Bhattacharyya, D. K., & Kalita, J. K. (2020, December). Malware dataset generation and evaluation. In 2020 IEEE 4th Conference on Information & Communication Technology (CICT) (pp. 1-6). IEEE.
[33] Ambekar, N. G., Devi, N. N., Thokchom, S., & Yogita. (2024). TabLSTMNet: enhancing android malware classification through integrated attention and explainable AI. Microsystem Technologies, 1-19.
[34] Li J, He J, Li W, Fang W, Yang G, Li T (2023) Syndroid: an adaptive enhanced android malware classification method based on CTGAN-SVM. Comput Secur 137:103604
[35] Shu Z, Yan G (2023) Eagle: evasion attacks guided by local explanations against android malware classification. IEEE Trans Depend Secure Comput. https://doi.org/10.1109/TDSC.2023.3324265
[36] Li J, He J, Li W, Fang W, Yang G, Li T (2023) Syndroid: an adaptive enhanced android malware classification method based on CTGAN-SVM. Comput Secur 137:103604
[37] Islam R, Sayed MI, Saha S, Hossain MJ, Masud MA (2023) Android malware classification using optimum feature selection and ensemble machine learning. Internet Things Cyber-Phys Syst 3:100–111
[38] Ullah F, Cheng X, Mostarda L, Jabbar S (2023) Android-iot malware classification and detection approach using deep url features analysis. J Database Manag 34(2):1–26