POLİTEKNİK DERGİSİ

JOURNAL of POLYTECHNIC

ISSN: 1302-0900 (PRINT), ISSN: 2147-9429 (ONLINE)

URL: http://dergipark.org.tr/politeknik

Hyperparameter tuning and feature selection methods for malware detection

Kötü amaçlı yazılım algılaması için hıperparametre ayarlama ve özellik seçım yöntemlerı

***Yazar(lar) (Author(s))**: Esra Kavalcı Yılmaz[1], Halit Bakır[2]*

ORCID[1]: 0000-0003-1314-4495

ORCID[2]: 0000-0003-3327-2822

To cite to this article: Yılmaz E. K. and Halit B., "Hyperparameter tunning and feature selection methods for malware detection", ***Journal of Polytechnic***, 27(1): 343-353, (2024).

Bu makaleye şu şekilde atıfta bulunabilirsiniz: Yılmaz E. K. ve Halit B., "Hyperparameter tunning and feature selection methods for malware detection", ***Politeknik Dergisi***, 27(1): 343-353, (2024).

Erişim linki (To link to this article): http://dergipark.org.tr/politeknik/archive

DOI: **10.2339/politeknik.1243881**

Hyperparameter Tuning and Feature Selection Methods for Malware Detection

Highlights

- *Comparison of 3 different methods for imbalanced data sampling*
- *Comparison using 4 different feature selection methods*
- *Hyperparameter tuning on different machine learning algorithms*
- *Comparison of hyperparameter tuning methods*

Graphical Abstract

In the study, firstly, three different methods were applied to solve the imbalanced data problem. Afterwards, by using feature selection methods, more important features were selected and classification processes were carried out using 5 different machine learning algorithms. Finally, hyperparameter tuning were conducted on the two most successful machine learning algorithms and the results were evaluated.

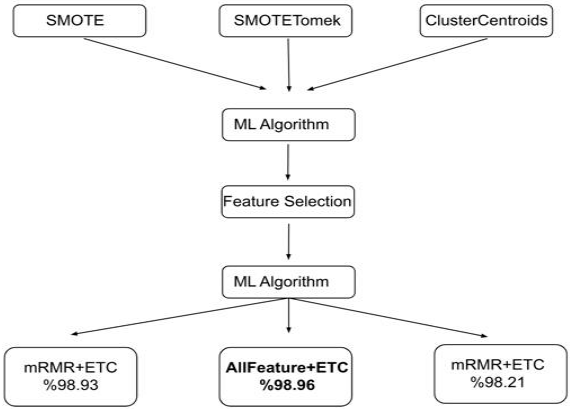

Figure. Working structure

Aim

The main purpose of this study is to examine the effects of imbalanced data sampling methods, hyperparameter tuning, and feature selection methods on machine learning algorithms used for malware detection on the android operating system.

Design & Methodology

Three different methods have been adopted, namely SMOTE, SMOTETomek, and Cluster Centroid for handling imbalanced data problem in our dataset. Afterward, feature selection process has been conducted using four different methods namely, mRMR, Mutual Information, Select From Model, and Select k Best. The classification process has been carried out using 5 different machine learning methods namely RF, SVM, LR, XGBoost, and ETC.

Originality

We believe that the difference of our study is that, the effects of many different methods (imbalanced data sampling, feature selection, machine learning, and hyperparameter tuning) on malware classification in android systems are investigated together.

Findings

The best accuracy in this study reached 98.96% obtained by using all features for training the ETC algorithm. Also, a very high classification accuracy reached 98.93% obtained using the mutual information feature selection algorithm with the ETC algorithm.

Conclusion

In the study, it was seen that the most successful results have been obtained using SMOTETomek with all features to train the ETC algorithm. It is seen that there is little difference between the accuracy obtained after applying mRMR and Mutual Information feature selection methods and the accuracy obtained using all the features in the dataset.

Declaration of Ethical Standards

The author(s) of this article declare that the materials and methods used in this study do not require ethical committee permission and/or legal-special permission.

Hyperparameter Tunning and Feature Selection Methods for Malware Detection

Araştırma Makalesi / Research Article

Esra Kavalcı Yılmaz, Halit Bakır*

Department of Computer Engineering, Sivas University of Science and Technology, Sivas, Turkey

(Geliş/Received : 30.01.2023 ; Kabul/Accepted : 13.03.2023 ; Erken Görünüm/Early View : 08.09.2023)

ABSTRACT

Smartphones have started to take an essential place in every aspect of our lives with the developing technology. All kinds of transactions, from daily routine work to business meetings, payments, and personal transactions, started to be done via smartphones. Therefore, there is a significant amount of very important user information stored in these devices which makes them a target for malware developers. For these reasons, machine learning (ML) methods have been used to detect malicious software on android devices quickly and reliably. In this study, a machine learning-based Android malware detection system has been developed, optimized, and tested. To this end, firstly, the data in the dataset has been balanced with 3 different methods namely SMOTE, SMOTETomek and ClusterCentroids. Afterward, the obtained results have been tried to be optimized by using different feature selection approaches including mRMR, Mutual Information, Select From Model, and Select k Best. Finally, the most two successful methods from the five tested ML algorithms (i.e. RF, SVM, LR, XGBoost, and ETC) have been tuned using GridSearch, Random Search, and Bayesian Optimization algorithms in order to investigate the effects of hyperparameter tuning on the performance of ML algorithms.

Keywords: Android malware detection, feature selection, imbalance data resampling, hyperparameter tuning.

Kötü Amaçlı Yazılım Algılaması için Hiperparametre Ayarlama ve Özellik Seçim Yöntemleri

ÖZ

Gelişen teknoloji ile birlikte akıllı telefonlar hayatımızın her alanında yer almaya başlamıştır. Günlük rutin işlerden önemli toplantılara, ödemelere ve kişisel işlemlere kadar her türlü işlem akıllı telefonlar üzerinden yapılmaya başlandı. Bu durumda, tüm kullanıcı bilgilerinin akıllı telefonlarda saklanması, akıllı telefonları kötü amaçlı yazılım geliştiricileri için bir hedef haline getirmektedir. Bu sebeplerden dolayı android cihazlardaki zararlı yazılımları hızlı ve güvenilir bir şekilde tespit etmek için Makine Öğrenmesi yöntemleri kullanılmaya başlanmıştır. Bu çalışmada öncelikle veri setindeki veriler SMOTE, SMOTETomek ve ClusterCentroids olmak üzere 3 farklı yöntemle dengelenmiştir. Daha sonra mRMR, Mutual Information, Select from Model ve Select k Best özellik seçim modelleri kullanılarak en yüksek doğruluk değeri elde edilmeye çalışılmıştır. Son olarak 5 farklı Makine Öğrenmesi algoritmasından (RF, SVM, LR, XGBoost, ETC) en başarılı 2 yöntem GridSearch, Random Search ve Bayesian Optimization yöntemleri kullanılarak ayarlanmıştır.

Anahtar Kelimeler: Android kötü amaçlı yazılım tespiti, öznitelik seçimi, dengesiz veri örneklemesi.

1.INTRODUCTION

Nowadays, with the growth and rapid development of technology, smartphones have begun to occupy a large area in our lives. Smartphones are used in many areas of our lives such as personal communication, entertainment, financial activities, health services, corporate meetings and correspondence. As a result, these type of devices stores a considerable amount of personal data such as contact information, location, financial information, multimedia files. This makes smartphones a popular target for malware developers, which leads to unauthorized actions or the use of users' personal data for other purposes. According to a study conducted in the first months of 2018, 88% of the phones sold worldwide between 2009 and 2018 were phones with Android operating system. Again, according to the same research, there are more than 2.6 million applications hosted on the Google Play Store [1]. Considering these results, it can be said that the Android operating system is the most frequently targeted platform by malware. The most dangerous side of this point is the end user's lack of cyber security awareness, unconscious, and carelessness. So, it is very important to take countermeasures to secure these devices and deal with such malicious software before it is presented to the end user. When evaluated in this context, the detection of the huge number of malware that targeted Android phones has led to the need to work in both academic and industrial fields in order to tackle this situation. For this reason, it has become widespread to detect malicious software in Android systems using artificial intelligence applications. This study aims to detect malware based on artificial intelligence methods and using DREBIN [2] well-known benchmark Android malware dataset.

* *Sorumlu yazar (Corresponding Author)*
e-mail: halit.bakir@sivas.edu.tr

2. RELATED WORK

The popularity of Android phones has indirectly led to an increase in malicious attacks on phones. Plenty researchers have carried out studies on the correct, reliable, and efficient detection of malicious applications to provide information security.

Wang et al. [3] proposed a hybrid method based on Deep Auto Coder (DAE) and Convolutional Neural Network (CNN) for malware detection. To develope detection accuracy, they first reconstructed high-dimensional features and used multiple CNN models to detect malware. In the training process, they aimed to prevent excessive learning by using a Serial Convolutional Neural Network (CNN-S). Thus, they created successful system in terms of feature extraction and malware detection with CNN-S. In the second stage, they used DAE as CNN's pre-training technique to decrease the training time. Then, with this proposed hybrid model, they conducted experiments on 10000 benign and 13000 malicious applications. As a result of the study, they found that compared to the SVM, the CNN-S increased success by 5%, while DAE-CNN reduced training time by 83%. Another study proposes a deep learning-based method to distinguish malicious apps from benignapps. For the classification process, first of all, system calls from malicious and benign applications have been used for training Long Short-Term Memory (LSTM) [4]. In another study, a Deep Residual LSTM-based model called MalResLSTM has been proposed for detecting malicious applications in Android environments. It has been stated that the proposed model is outperformed SVM, LR, RF and NN models [5]. Moreover. Unver and Bakour [6] proposed a malware visualization-based approach for detecting Android malware applications. In the study, firstly, feature extraction processes have been performed using multiple image-based local features and global features extraction algorithms. Afterwards, the extracted features have been used for training six different ML methods, namely Random Forest (RF), K-Nearest Neighbor (KNN), Decision Tree (DT), Bagging, AdaBoost and Gradient Boost. Baldini and Geneiatakis [7] examined the KNN ML method based on different distance measures. The experiments were carried out using the DREBIN data set and instead of the Euclidean distance, which is often used for KNN, Hamming, Correlation, Jaccard and CityBlock distances have been preferred to be used in the study, and it has been stated that the results of these two distance matrices can outperform those can be obtained using the Euclidean distance. Fiky et al. [8] proposed a model based on static malware analysis and ML methods for detecting malware before it's installation on android phones. In this context, they analyzed the proposed model by testing it on two different benchmark data sets, namely DREBIN and MALGENOME. It is stated that the suggested method has better performance than other similar models.

Gao et al. [9] proposed a Graph Convolutional Network (GCN)-based Android malware detection system for detecting malicious code in the Android platform, and it has been stated that obtained accuracy reached 97% on average.

Cao et al. [10] stated that when malware detection studies have been investigated, it has been concluded that the benign features have been generally used, and therefore most of the previous studies have focused on benign application detection studies. Thus, while achieving high accuracy, they emphasized that it is vulnerable to malicious applications that use features similar to benign ones. Therefore, in their study, they proposed an approach that focuses on its truly malicious features. Afterward, they evaluated the suggested approach in terms of both accuracy and durability against attacks. Last of all, they stated that the suggested model outperformed the basic techniques and was effective in obtaining accurate and reliable results.

Bakour and Unver [11] proposed the VisDroid model as a generic image-based method for classifying Android malware. In the study, Android malware sample sources have been reverse-engineered and used to create five different grayscale image datasets, each of which contains 4850 samples. The dataset has been used to train six different ML algorithms, including RF, KNN, DT, Bagging, AdaBoost and Gradient Boost. As a result, it has stated that the proposed model achieved 98.2% success and can be very effective in malware detection domain. In another study, Bakour and Unver [12] proposed DeepVisDroid model, in which image-based features are used in a hybrid manner with deep learning techniques for the detection of android malware. First of all, four gray image datasets have been constructed by converting the source code of Android applications into images. Afterward, the study was carried out by extracting some image-based local and global features from from the created gray scale image datasets. At the end of the study, it stated that the suggested model is a promising model with a success rate of over 98%.

3. MATERIALS AND METHODS

3.1. Dataset:

The dataset used in the study was prepared by Yerime et al. [13] based on applying static analysis on the Drebin well-known benchmark dataset. The dataset contains 5560 Malware samples taken from the publicly available DREBIN [2] Android malware dataset, and 9476 Benign samples taken from Google play app store. This dataset is called as DREBIN-215. So, the DREBIN-215 dataset consists of 15036 data samples belonging to two classes, and each data sample contains 215 features. The DREBIN-215 data set details are shared in Table.1.

The features in the dataset consist of 4 categories. 53% of the features are Manifest Permission, 33% are API call signatures, and 14% are Other (Commands signature, Intent.etc). Some examples of important features in these categories are shown in Table 2.

Table 1. DREBIN-215 dataset details

Samples	Malware	Benign	Features
15036	5560	9476	215

Table 2. Dataset feature categories.

Category	Feature
Manifest Permission	Read_External_Storage
	Receive_Boot_Completed
	Read_Call_Log
	Send_Sms
	Read_Social_Stream
API Call Signatures	Ljava.lang.Class.getResource
	TelephonyManager.getDeviceId
	Ljava.lang.Class.getMethods
	MessengerService
	getBinder
Commands Signatures	mount
	remount
	chmod
	chown
	/system/app
Intent	android.intent.action.BOOT_COMPLETED
	android.intent.action.PACKAGE_REPLACED
	android.intent.action.SEND_MULTIPLE
	android.intent.action.TIME_SET
	android.intent.action.PACKAGE_REMOVED

3.2. Data Re-Sampling Techniques:

If the class distribution of the dataset to be studied is imbalanced, the classification algorithm may tend to the majority class. In this case, the detection performance of the minority class may be very low. As a result, the success of the ML algorithms will negatively affected. For this reason, the dataset should be balanced in order to obtain healthier and more successful results. To this end, oversampling (adding data to the dataset) and undersampling (data is removed from the dataset) approaches are used [14].

- *SMOTE (Synthetic Minority Over-Sampling Technique):* It is a widespread oversampling technique that produces synthetic samples for the minority class using the k nearest neighbor algorithm. Synthetic samples are clustered around the real minority class data samples, in this way the classification accuracy of the minority class can be improved [15].

- *SMOTETomek:* It is a hybrid model in which both oversampling and undersampling methods are used to improve the classification performance. To make the data distribution balanced, first oversampling is conducted using the SMOTE technique. Afterwards, the ambiguous data samples are down-sampled using the Tomek links

algorithm for clarifying the difference between the classes in the dataset [16].

- *Cluster Centroids:* It is a down-sampling technique that decrease the number of the majority classe in order to make the distribution in the dataset balanced. This algorithm is based on clustering the data samples using the K-Means algorithm and eliminating those so far from the center of the clusters [17].

3.3. Feature Selection:

It is the process of extracting a subset of features from the original dataset using a specific algorithm. Feature selection methods are used to prevent overfitting caused by too many features in the dataset. These feature selection methods also make prediction results more predictable [18].

- *mRMR (Min Redundancy Max Relevance):* This method is based on calculating the maximum correlation between the features and the target variables in the dataset and redundant linkages between traits. Particularly, the mutual information is used for evaluating the features in the dataset and obtaining a subset of features from them [19].

- *Mutual Information:* It is a method based on statistical operations used to calculate the relationships between variables and their dependence on each other. Features are selected by calculating how much a change in any of the variables can affect the others [20].

- *Select From Model:* Feature selection is made based on a specified ML algorithm. Considering a treshold value, the feature selection is made according to the importance of the feature (feature_importances_) to the ML algorithm. Treshold value is taken as the average value by default [21].

- *Select k-Best:* This method is based on ranking the features according to the scores of the independent variables and the target variables. Then, statistical processes are applied to determine the relationship between them. Relationships are determined using some methods such as Chi-Squared or Classifier-F functions. Afterwards, k features will be selected from the original features to be used in ML algorithms' training process [22].

3.4. Tuning the Hyperparameters of the Algorithms:

Artificial intelligence is revolutionizing diverse domains such as healthcare, automating medical diagnosis, cypersecurity, data communication and treatment optimization, while also transforming transportation through self-driving vehicles and advanced traffic management systems [23-27]. It's enhancing customer experiences in retail by enabling personalized recommendations, and revolutionizing manufacturing with predictive maintenance and quality control using AI-driven analytics. Tuning or optimizing hyperparameters requires determining the best value for each hyperparameter in the ML algorithm [28][29], which can help to achieve the most successful results. It is always possible to detect the best values for the hyperparameters by conducting a wide range of experimental studies, but, these processes are very time-consuming [30]. So, it is better to adopt an optimization algorithm for selecting the best values for the hyperparameters of the used ML algorithm. The hyperparameter tuning algorithms used in this study are briefly explained below.

- *Grid Search:* Its basic principle is to try all possible parameter values. Suppose there are k parameters that need to be optimized in the relevant ML algorithm. In this case, a grid of size k×k will be defined and combinations of each parameter are processed separately. If the number of k parameters is large, the search will be more difficult, therefore, this algorithm is suitable to be used in smaller number of hyperparameter [31].

- *Random Search:* In the random search process, first of all, hyperparameters are determined by using preliminary information about the problem. Operations are performed with random hyperparameter value groups until the best result or expected value is obtained. It is faster compared to Grid Search as operations are performed on randomly selected subsets instead of all of the potential values for hyperparameters [32].

- *Bayesian Optimization:* It is a less costly method compared to other tuning methods. Bayesian optimization aims to reach the best results by creating a probabilistic model. It allows the best parameters to be found with the least iteration [33].

3.5. Classification Algorithms:

The classification algorithms used in the study are briefly explained below.

- *Random Forest Classifier (RF):* It is a machine learning algorithm in which a forest of decision trees represents an independent instance of each tree. It can be used for both classification and regression operations. During the training process of RF, sample and special selections are made randomly. In this way, overfitting is prevented [34].

- *Support Vector Classifier (SVC):* it is a supervised learning model used to classify binary or multiple datasets was first proposed in 1995. The working logic is to consider classification problems as a quadratic optimization problem. Thus, the result is achieved with less number of transactions and it provides a speed advantage compared to other algorithms. Thanks to this characteristic, it offers successful results both in problems with small-scale and large-scale datasets [35].

- *Logistic Regression:* It is a supervised learning model based on linear statistics. The working logic is to model the output probability according to the input. In this case, it does not perform a classification itself. The model chooses a cut-off value during operation. In binary classification, the inputs with probability less than the cut-off value can be classified into one class and the larger ones into the other class [36].

- Extreme Gradient Boosting (XGBoost) Classifier: Gradient Boosting is a Decision tree (DT) Based ensemble algorithm. It can be used for both regression and classification problems. The working logic of this algorithm is based on learning from mistakes by evaluating many weak classifiers. Individual classification models (trees) are created and each model is trained by the previous tree. Thanks to these features, xGboost offers better results compared to other boost algorithms [37].

-Extra Tree Classifier: Extra Tree Classifier is an ensemble learning method. The only difference from random forest classifier is the way the trees are created. In this method, each decision tree is created according to the dependency relationships of the data samples [38]. It uses the averaging method to increase the accuracy and prevent overfitting [39].

4. EXPERIMENTAL RESULTS AND DISCUSSION

When the class distribution of the dataset used in the study has been investigated, it has been noted that the data has an imbalanced distribution as shown in Table 3. So, in the study, firstly, the imbalance in the dataset distribution has been tried to be eliminated. For this purpose, three different data resampling methods have been adopted, namely 'SMOTE', 'SMOTETomek' and 'ClusterCentroids'. We have obtained three different balanced datasets after applying each of these data resampling methods. The dataset that has been constructed using SMOTE algorithm contains 9476 Malware and 9476 benign samples. The dataset that has been constructed using SMOTETomek algorithm contains 9472 Malware, and 9472 benign samples. Finally, the dataset that has been constructed using ClusterCentroids contains 5560 Malware and 5560 benign samples. Classification processes were carried out by applying ML algorithms to these datasets by separating them as 10% for testing and 90% for training. Then, the obtained balanced three datasets have been used for training and testing five different well-known machine learning classifiers namely RF, SVC, LR, XGBoost, and ETC.

The results of the classification processes performed after SMOTE, SMOTETomek and ClusterCentroid are presented in Table 4-5-6 respectively.

Table 3. Data distribution

Class	Samples
Malware	5560
Benign	9476

Table 4. ML algorithms' results after applying the SMOTE algorithm

Algorithm	Accuracy	Precision	Recall	F1 Score	CV Mean
RF	99.10	99.53	98.62	99.08	98.80
SVC	98.88	99.53	98.17	98.84	98.61
LR	98.04	98.27	97.71	97.99	97.73
XGBoost	96.93	97.67	96.00	96.82	96.78
ETC	99.21	99.65	98.74	99.19	**98.88**

Table 5. ML algorithms' results after applying the SMOTETomek algorithm

Algorithm	Accuracy	Precision	Recall	F1 Score	CV Mean
RF	98.46	99.45	97.41	98.42	98.80
SVC	98.20	98.90	97.41	98.15	98.36
LR	97.88	97.84	97.84	97.84	97.77
XGBoost	96.88	97.69	95.90	96.79	96.55
ETC	98.89	99.67	98.06	98.86	**98.96**

Table 6. ML algorithms' results after applying the ClusterCentroids algorithm

Algorithm	Accuracy	Precision	Recall	F1 Score	CV Mean
RF	98.30	98.71	97.71	98.21	97.98
SVC	98.30	98.96	97.46	98.20	98.01
LR	97.69	98.20	96.95	97.57	97.39
XGBoost	96.72	97.91	95.17	96.52	96.38
ETC	98.30	98.96	97.46	98.20	98.11

When the results of the classification processes have been investigated after the class distributions in the data set have been balanced, it has been seen that the best results have been obtained using the Extra Trees Classifier method for all three balanced datasets.

4.1. Feature selection experiments results

In the next stage, four different feature selection algorithms, namely mRMR, Mutual Information, Select From Model and Select k Best, have been applied for each dataset separately. The ML algorithm that obtained the best results namely ETC has been adopted to be used during applying the Select from Model feature selection process. Also, a range of values has been tested for each of the used threshold and the number of selected features as illustrated in Table 7. A group of features have been selected by testing all the possible values for the threshold and the number of selected features, and a classification process has been carried out in order to detect the best group of features.

Table 7. Defined parameter values for feature selection process.

Model	According by	Range
Select From Model	Treshhold	[0.001,0.002,0.003,0.004,0.005, 0.006,0.007,0.008,0.009,0.01, 0.02,0.03,0.04,0.05]
mRMR, Mutual Information, Select k Best	Number of Feature	[10,20,30,40,50,60,70,80,90, 100,110,120,130,140,140,160, 170,180,190,200,210]

When the feature selection methods were performed on the SMOTE-based dataset, the best selected number of features was as follows: 210 features have been selected using each of the mRMR and Mutual Information, 78 features have been selected using Select From Model with threshold = 0.001, and 190 features have been selected using Select k Best methods. Afterward, the classification process has been conducted again by training the machine learning classifiers using the selected features. The results obtained by training the machine learning classifiers using the the best subset of features selected from SMOTE-based dataset is illustrated in table 8.

When the results are examined, there is no increase in accuracy was observed after any feature selection method for the SVC method, and it can be said that the best results of the SVC algorithm is obtained by using all the features.

When evaluated in terms of feature selection methods, it can be seen that there is no increase in the results when the Select From Model method is used. Also, it can be noted that the mRMR was the method with the highest increase in the performance of the ML algorithms. It is seen that the highest success has been achieved by the ETC algorithm has been obtained after applying the mRMR and Mutual Information, where, its accuracy rate reached 98.93%.

Also, the previously mentioned feature selection methods have been applied to the constructed SMOTETomek-based balanced dataset. After selecting the best feature groups using each of the adopted feature selection approaches, the machine learning classifiers have been trained and tested using these feature groups. The results of this case study are illustrated in table 9

Table 8. Results obtained after applying SMOTE + Feature Selection. The value of cross-validation (cv) is 5.

ML Algorithm	All Feature	mRMR (210)	Mutual Info (210)	Select From Model (78)	Select k Best (190)
RF	98.80	98.84	98.74	98.74	98.72
SVC	98.61	98.56	98.59	98.36	98.52
LR	97.73	97.76	97.75	97.20	97.61
XGBoost	96.78	96.82	96.84	96.68	96.81
ETC	98.88	98.93	98.93	98.77	98.84

Table 9. Results obtained after applying SMOTETomek + Feature Selection. The value of cross-validation (cv) is 5.

ML Algorithm	All Feature	mRMR (210)	Mutual Info (210)	Select From Model (77)	Select k Best (190)
RF	98.80	98.82	98.86	98.75	98.73
SVC	98.36	98.28	98.35	98.16	98.25
LR	97.77	97.80	97.77	97.19	97.61
XGBoost	96.55	96.65	96.59	96.53	96.58
ETC	98.96	98.88	98.89	98.85	98.91

When the accuracy values obtained after SMOTETomek+Faeture Selection have been investigated, it has been noted that there was no increase after the Select From Model method has been recorded. methods, and the comparison of the accuracy values obtained when these features are used are presented in Table 10 (k=5 for CV in the table).

Table 10. Results obtained after applying ClusterCentroids + Feature Selection

ML Algorithm	All Feature	mRMR (210)	Mutual Info (210)	Select From Model (122)	Select k Best (200)
RF	97.98	97.90	97.98	97.85	98.02
SVC	98.01	98.01	97.98	97.97	97.94
LR	97.39	97.31	97.34	96.81	97.25
XGBoost	96.38	96.36	96.44	96.23	96.23
ETC	98.11	98.21	98.12	97.98	98.13

Also, there is no increase has been recorded for the SVC algorithm after applying the feature selection approaches. In accordance with Table 9, it can be observed that the highest accuracy value has been obtained when the classification process has been conducted using the RF algorithm after applying the Mutual Information feature selection approach (98.86%).

Finally, the feature selection methods have been applied to the ClusterCentroids-based balanced dataset. The number of features selected using the feature selection As in the other results, there was no increase when the Select From Model feature selection method has been used. It can be noted from the table that the best results obtained in this case study have been achieved using ClusterCentroids + ETC with an accuracy rate of 98.21%. When all the results of feature selection methods have been compared, we observed that the best accuracy value of 98.93% has been obtained by applying mRMR+ETC and Mutual Information+ETC methods over the SMOTE-based balanced dataset. On the other hand,

when all the results have been examined, we noted that when all features have been used, it was concluded that the best classification accuracy of 98.96% obtained using the ETC algorithm.

The flowchart of the experiments conducted in this work is illustrated in Figure 1.

4.2. Hyperparameter experiments results

In the second part of the study, hyperparameter tuning process have been carried out using three different methods, then the impact of these operation on the classification accuracy has been examined.

Three different hyperparameter tuning algorithms namely Halving Grid Search, Random Search, and Bayesian Optimization have been applied for tuning the AllFeature + ETC method and AllFeature + RF methods. The hyperparameter list used during the tuning processes is illustrated in Table 11. The best hyperparameters obtained for ETC and RF using Halving Grid Search and the results obtained when these parameters have been used are given in Table 12. The hyperparameters determined after applying Random Search and Bayesian Optimization algorithms and the results obtained using the hyperparameters tuned using these two algorithms are also illustrated in Table 13 and Table 14, respectively

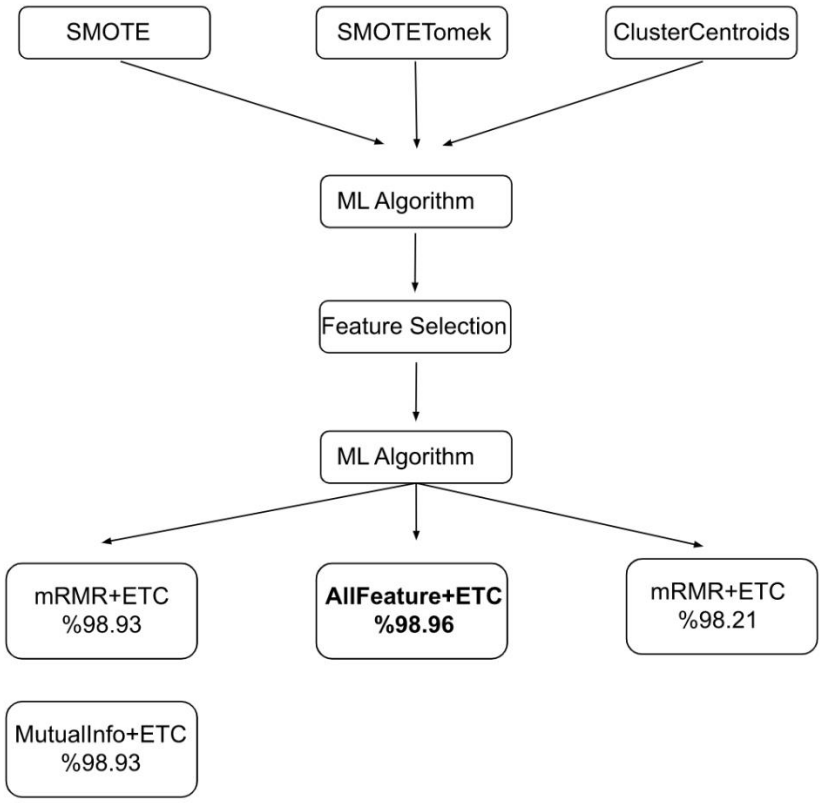

Figure 1. Conducted experimental studies structure

Table 11. Parameters used in the Tuning processes

Feature	Value Range
n_estimators	100, 200, 300,500,700
criterion	'entropy', 'gini'
max_depth	None, 1, 2, 3, 4
min_samples_leaf	2, 3, 4, 5

Table 12. Halving Grid Search results

Method	n_estimators	criterion	max_depth	min_samples_leaf	CV Mean	CV Mean (Before Tunning)
ETC	700	entropy	None	2	97.88%	98.80%
RF	300	entropy	None	2	98.93%	98.96%

Table 13. Random Search results

Method	n_estimators	criterion	max_depth	min_samples_leaf	CV Mean	CV Mean (Before Tunning)
ETC	100	gini	None	3	98.95%	98.80%
RF	500	entropy	None	4	98.89%	98.96%

Table 14. Bayesian Optimization results

Method	n_estimators	criterion	max_depth	min_samples_leaf	CV Mean	CV Mean (Before Tunning)
ETC	100	gini	None	2	**98.97%**	98.80%
RF	100	gini	None	2	98.94%	98.96%

An increase in RF results hase been observed in some cases when the RF, ETC algorithms have been tested after the hyperparameter tuning process. While the classification accuracy of RF algorithm was 98.80% before conducting the hyperparameter tuning process its accuracy reached 98.93%, 98.89% and 98.94% after applying Halving Grid Search, Random Search and Bayesian Optimization processes, respectively. We noted that some increase in the performance of ETC has been recorded after applying the hyperparameter process where its classification accuracy reached 98.97% after applying the Bayesian optimization algorithm.

5. COMPARISON STUDY

In this section, we have compared the results obtained in this work with some state-of-the-art previously conducted works. We have selected the compared works carefully such that all of them have been conducted using DREBIN well-known malware dataset. We noted that the data resampling algorithms have been used only in our work. Also, our work is the only work that adopted the hyperparameter tuning algorithms for adjusting the hyperparameters of the machine learning algorithms. The feature selection methods have been used in some of the compared works, but those that have been adopted in this

Table 15. The results of comparison study.

Authors	Year	Method	Re-Sampling Techniques	Feature Selection	Hyperparameters Tuning	Model performance (%)
Fiky et al. [8]	2021	RF	-	PCA+IG	-	98.4 (ACC)
GDroid [9]	2021	GCN	-	PCA	-	~ 97 (ACC)
VisDroid [11]	2021	Visualization-based	-	-	-	98.14 (ACC)
DeepVisDroid [12]	2021	Visualization-based	-	-	-	**98.96 (ACC)**
Yerime et al. [13]	2019	Ensemble model using K base machine learning classifier	-	IG	-	98.42 (F1-score)
D. J et al. [31]	2022	MLP-MFR Based	-	-	-	92.04 (ACC)
Odat and Yaseen [39]	2023	RF, DT, SVM	Random under-sampling	-	-	95 (ACC)
Proposed model	**2023**	**ETC**	**SmoteTomek**	**mRMR/MI**	**HGS/RS/BO**	**98.96** (ACC)

Multivariate Feature Ranking (MFR)
Graph Convolutional Network (GCN)
Halving Grid Search (HGS)
Random Search (RS)
Bayesian Optimization (BO)
Mutual Info (MI)

work have not been tested before in the compared previous works. When we investigated the compared works, we reviled that the best performance reached 98.98% has been achieved in our work. So, the conducted work outperformed the previous works in terms of detection accuracy and the used algorithms. The results of the conducted comparison study are illustrated in table 15.

6. CONCLUSION

In this work, the impacts of feature selection and hyperparameter tuning on detecting malware on android systems have been investigated. To this end, an experimental comparative study has been conducted using five different ML algorithms, four different feature selection approaches, and three different hyperparameter tuning methods. First of all, the classes in the dataset have been balanced by applying three different algorithms namely SMOTE, SMOTETomek and ClusterCentroids. As a result, three different balanced datasets have been obtained, these datasets have been used for training multiple machine learning algorithms. After that, four different feature selection approaches namely mRMR, Mutual Information, Select From Model, and Select k Best have been adopted in order to select the best feature groups from the dataset. The best accuracy in this study reached 98.96% obtained by using all features for training the ETC algorithm. Also, a very high classification accuracy reached 98.93% obtained using the mutual information feature selection algorithm with the ETC algorithm. Although the success rate obtained after reducing the number of features using the mutual information feature selection method was less than that obtained using all the features these feature selection methods can improve the training time of the Ml algorithms. When the Tunning methods have been evaluated, in general, it has been seen that the most successful results have been obtained using the Bayesian Optimization algorithm.

In future works, different feature selection methods will be tested and comparisons will be conducted. In addition, it is planned to increase the classification success by developing hybrid models instead of classical machine learning methods.

ACKNOWLEDGMENT

The authors would like to thank Arp et al. [2] for sharing their DREBIN android malware dataset, and Yerime et al. [13] for sharing their DREBIN dataset-based feature dataset.

DECLARATION OF ETHICAL STANDARDS

The author(s) of this article declare that the materials and methods used in this study do not require ethical committee permission and/or legal-special permission.

AUTHORS' CONTRIBUTIONS

Esra Kavalcı YILMAZ: Conducting the experiments, Visualization, and writing manuscript draft.

Halit BAKIR: Shared in conducting experiments, Visualization, Review the manuscript and Supervision.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

REFERENCES

[1] Dağlıoğlu A. and Doğru İ. A., "Android İşletim Sisteminde Kötücül Yazılım Tespit Sistemleri", ***Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi***, 11(2): 499-511, Haz. (2020).

[2] Arp D., Spreitzenbarth M., Hubner M., Gascon H. and Rieck K., "Drebin: Effective and explainable detection of android malware in your pocket", ***Network and Distributed System Security (NDSS) Symposium***14: 23-26, (2014).

[3] Wang W., Zhao M. & Wang J. "Effective android malware detection with a hybrid model based on deep autoencoder and convolutional neural network.", ***J Ambient Intell Human Comput*** 10:3035–3043 (2019).

[4] Xiao X., Zhang S., Mercaldo F., Hu G., Sangaiah A.K. "Android malware detection based on system call sequences and LSTM.", ***Multimed Tools Appl*** 78, 3979–3999 (2019).

[5] Alotaibi A., "Identifying Malicious Software Using Deep Residual Long-Short Term Memory", in ***IEEE Access***, 7, 163128-163137, (2019).

[6] Ünver H.M., Bakour K. "Android malware detection based on image-based features and machine learning techniques.", ***SN Appl. Sci***. 2, 1299, (2020).

[7] Baldini G. and Geneiatakis D., "A Performance Evaluation on Distance Measures in KNN for Mobile Malware Detection," ***2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)***, (2019).

[8] Fiky A. H. E., Elshenawy A. and Madkour M. A., "Detection of Android Malware using Machine Learning," ***2021 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)***, (2021).

[9] Gao H., Cheng S., Zhang W., "GDroid: Android malware detection and classification with graph convolutional network", ***Computers & Security***, 106, (2021).

[10] Cao M., Badihi S., Ahmed K., Xiong P. and Rubin J., "On Benign Features in Malware Detection," ***2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE)***, 1234-1238, (2020).

[11] Bakour K., Ünver H.M. "VisDroid: Android malware classification based on local and global image features, bag of visual words and machine learning techniques.", ***Neural Comput & Applic*** 33, 3133–3153, (2021).

[12] Bakour K., Ünver H.M. "DeepVisDroid: android malware detection by hybridizing image-based features with deep learning techniques.", ***Neural Comput & Applic*** 33, 11499–11516, (2021).

[13] Yerima S. Y. and Sezer S., "DroidFusion: A Novel Multilevel Classifier Fusion Approach for Android Malware Detection," in ***IEEE Transactions on Cybernetics***, 49(2):453-466, Feb. (2019).

[14] Srinilta C. and Kanharattanachai S., "Application of Natural Neighbor-based Algorithm on Oversampling SMOTE Algorithms," ***2021 7th International Conference on Engineering, Applied Sciences and Technology (ICEAST)***, 217-220, (2021).

[15] Tallo T. E. and Musdholifah A., "The Implementation of Genetic Algorithm in Smote (Synthetic Minority

Oversampling Technique) for Handling Imbalanced Dataset Problem," ***2018 4th International Conference on Science and Technology (ICST)***, 1-4, (2018).

[16] Sahid M. A., Hasan M., Akter N. and Tareq M. M. R., "Effect of Imbalance Data Handling Techniques to Improve the Accuracy of Heart Disease Prediction using Machine Learning and Deep Learning," ***2022 IEEE Region 10 Symposium (TENSYMP)***, pp. 1-6, (2022).

[17] Zheng H., Sherazi S. W. A. and Lee J. Y., "A Stacking Ensemble Prediction Model for the Occurrences of Major Adverse Cardiovascular Events in Patients With Acute Coronary Syndrome on Imbalanced Data," in ***IEEE Access***, 9, 113692-113704, (2021).

[18] Wang G., Lauri F. and Hassani A. H. E., "Feature Selection by mRMR Method for Heart Disease Diagnosis," in ***IEEE Access***, 10, 100786-100796, (2022).

[19] Zhou H. and Peng C., "Oil Spills Identification in SAR Image Using mRMR and SVM Model," ***2018 5th International Conference on Information Science and Control Engineering (ICISCE)***, (2018).

[20] Baruah H. S., Thakur J., Sarmah S. and Hoque N., "A Feature Selection Method using PSO-MI," ***2020 International Conference on Computational Performance Evaluation (ComPE)***, 280-284, (2020).

[21] Pirbazari A. M., Chakravorty A. and Rong C., "Evaluating Feature Selection Methods for Short-Term Load Forecasting," ***2019 IEEE International Conference on Big Data and Smart Computing (BigComp)***, 1-8, (2019).

[22] Agaal A. and Essgaer M., "Influence of Feature Selection Methods on Breast Cancer Early Prediction Phase using Classification and Regression Tree," ***2022 International Conference on Engineering & MIS (ICEMIS)***, 1-6, (2022).

[23] Bakır, Halit, and Rezan Bakır. "DroidEncoder: Malware detection using auto-encoder based feature extractor and machine learning algorithms." ***Computers and Electrical Engineering*** 110: 108804, (2023).

[24] Bakır, Halit, Ayşe Nur Çayır, and Tuğba Selcen Navruz. "A comprehensive experimental study for analyzing the effects of data augmentation techniques on voice classification." ***Multimedia Tools and Applications***: 1-28, (2023).

[25] Duran, Abdulmuttalip, and Halit BAKIR. "Hiperparametreleri Ayarlanmış Makine Öğrenimi Algoritmalarını Kullanarak Android Sistemlerde Kötü Amaçlı Yazılım Tespiti." ***International Journal of Sivas University of Science and Technology*** 2, no. 1: 1-19, (2023).

[26] Özcan, Büşra, and Halit Bakır. "Yapay Zeka Destekli Beyin Görüntüleri Üzerinde Tümör Tespiti." In ***International Conference on Pioneer and Innovative Studies***, 1,297-306. (2023).

[27] Doğan, E. R. O. L., and Halit BAKIR. "Hiperparemetreleri Ayarlanmış Makine Öğrenmesi Yöntemleri Kullanılarak Ağdaki Saldırıların Tespiti." In ***International Conference on Pioneer and Innovative Studies***, 1, 274-286. (2023).

[28] Demircioğlu, Ufuk, Asaf Sayil, and Halit Bakır. "Detecting Cutout Shape and Predicting Its Location in Sandwich Structures Using Free Vibration Analysis and Tuned Machine-Learning Algorithms." ***Arabian Journal for Science and Engineering***: 1-14, (2023).

[29] Bakır, Halit, and Kholoud Elmabruk. "Deep learning-based approach for detection of turbulence-induced distortions in free-space optical communication links." ***Physica Scripta*** 98, no. 6: 065521, (2023).

[30] Ragab M. G., Abdulkadir S. J. and Aziz N., "Random Search One Dimensional CNN for Human Activity Recognition," ***2020 International Conference on Computational Intelligence (ICCI)***, 86-91, (2020).

[31] Zhang J., Chen Y., Yang K., Zhao J. and Yan X., "Insider Threat Detection Based on Adaptive Optimization DBN by Grid Search," ***2019 IEEE International Conference on Intelligence and Security Informatics (ISI)***, 173-175, (2019).

[32] Tanyıldızı E. and Demirtaş F., "Hiper Parametre Optimizasyonu Hyper Parameter Optimization," ***2019 1st International Informatics and Software Engineering Conference (UBMYK)***, 1-5, (2019).

[33] Nguyen V., "Bayesian Optimization for Accelerating Hyper-Parameter Tuning," ***2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)***, pp. 302-305, (2019).

[34] Katırcı R., Yılmaz E. K., Kaynar O., Zontul M., "Automated evaluation of Cr-III coated parts using Mask RCNN and ML methods", ***Surface and Coatings Technology***, 422, 127571,ISSN 0257-8972, (2021).

[35] Adem K., "Diagnosis of breast cancer with Stacked autoencoder and Subspace kNN", ***Physica A: Statistical Mechanics and its Applications***,Volume 551,124591,ISSN 0378-4371, (2020).

[36] Zhao Y., "Credit Card Approval Predictions Using Logistic Regression, Linear SVM and Naïve Bayes Classifier," ***2022 International Conference on Machine Learning and Knowledge Engineering (MLKE)***, 207-211, (2022).

[37] Katirci R., Aktas H. & Zontul M., "The prediction of the ZnNi thickness and Ni % of ZnNi alloy electroplating using a machine learning method", ***Transactions of the IMF***, 99:3, 162-168, (2021).

[38] D. J, N. J and N. P, "Multimodal Feature Selection for Android Malware Detection Classifiers," ***2022 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)***, Chennai, India, 1-5, (2022).

[39] E. Odat and Q. M. Yaseen, "A Novel Machine Learning Approach for Android Malware Detection Based on the Co-Existence of Features," in ***IEEE Access***, 11. 15471-15484, (2023)