International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

ENSIC: Feature Selection on Android Malware Detection Attributes Using an Enhanced Non-Linear SVM Integrated with Cross Validator

Ravi Eslavath[1], Upendra Kumar Mummadi[2]

Submitted: 18/09/2023 **Revised**: 17/11/2023 **Accepted**: 29/11/2023

Abstract: The digital telecom world made the humans to have android based mobiles. Hackers are realising more number of threatening applications daily to attack the android devices either to steal the personal information or to commit fraud money transaction. These cyber crimes, which attacks the devices functionality or security is known as "Malware Attacks". The proposed research aims to identify the impact of the android device factors like call signals, intents, and commands to access an application in classifying the malware attacks. Existing dataset contains 215 sub attributes for the given factors, which is treated as a high dimensionality according the principles of the machine learning. So, the previous researchers implemented traditional feature selection techniques like correlation, which consumes high computational time because bi-variate or multi variate with chi-square analysis needs to do lot of computations in between the different pairs of attributes. Using the recursive feature elimination has produced more number of attributes, which is more than half of the attributes and has got average accuracy with high mis-classification rate. The proposed research enhances the embedded approach by integrating the non-linear customized SVM (Support Vector Machine) with cross validated pipelining feature. This increases the accuracy of the model and reduces the number of essential attributes to 74 with 95.58% accuracy.

Keywords: *Correlation, Embedded Approach, Feature Selection, Intents, Recursive Feature Elimination*

1. Introduction

1.1. Introduction to Malware

Malware attacks include any harmful software created with the purpose of secretly damaging or destroying infrastructure, servers, clients, computer networks, and/or computers. Malicious software is where the word malware comes from. Without the victim's knowledge, hackers create malicious software to corrupt and access the victim's computer. Malware can penetrate a computer system in a variety of ways, including as spyware, ransomware, adware, Trojan horses, viruses, and worms. A programme is typically labelled malware based on the developer's intent rather than the features itself. The malware was initially created solely as a joke for the end user, but it subsequently developed with the introduction of more sophisticated technology to target victim PCs and make money. Any hacker who infects a computer with malware wants to steal sensitive data or encrypts and then demand payment to decrypt them. Malware infestations are launched on any victim machine by hackers for a variety of reasons. There are other contributing variables, including some OS flaws that grant access to too many resources.

1.2. Types of Malware Attacks

Figure 1 categorizes malware types that exists in the different network attacks

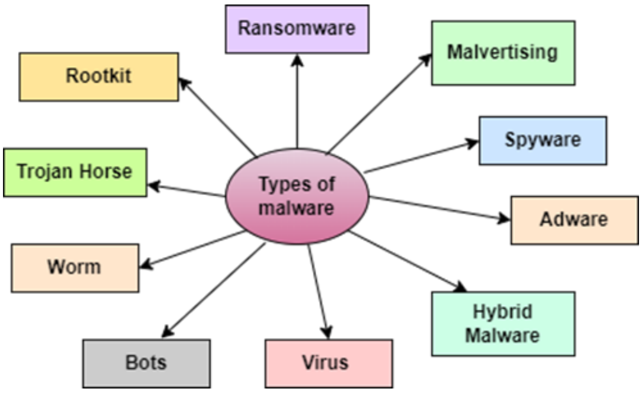


Fig. 1. Types of Malware Attacks

a. Virus: When malware is activated, it has the ability to alter other programmers and insert its hazardous code into them in order to replicate. The only type of virus that may "infect" other files is this one, and it is also among the most difficult to remove.

b. Worm: A worm can quickly travel from one computer to another, infecting entire networks, and appears to be able to replicate itself without the end user's help.

c. Bots: A bot is a type of malware that is created to do a particular set of functions. It was initially created with good intentions, but it has since turned evil. In addition to being used in botnets to launch DDoS attacks as web frameworks that can gather server data, they are designed to transmit

[1] Research Scholar, Department of CSE, University College of Engineering, Osmania University, Hyderabad, India.
Author Email: eslavathravi@gmail.com
ORCID ID: 0000-0002-7561-4344
[2] Professor, Department of CSE, Muffakham Jah College of Engineering and Technology, Hyderabad, India.
Author Email: upendra.kumar@mjcollege.ac.in
ORCID ID: 0000-0002-6269-6854

malware that is disguising itself as well-known search phrases on download websites.

d. Trojan: Trojan software, which seems to be a reliable programme, is one of the most difficult types of malware to detect. Once the victim executes this virus, the malicious software and instructions it contains can continue to run undetected. Other malware is frequently introduced into the system via it.

e. Hybrid malware: A "hybrid" or mashup of different harmful software types makes up a lot of modern malware. For instance, "bots" first resemble Trojan horses before acting as worms once they have been activated. They are frequently employed as part of a wider cyberattack against the entire network to target specific users.

f. Adware: Adware bombards users with intrusive and aggressive advertising, such as pop-ups.

g. Malvertising: Malvertising spreads malware to end-user computers using legal advertisements.

Spyware: Spyware monitors the unaware end user, gathering login information, passwords, browser history, and other information.

h. Rootkit: A rootkit is a particular kind of malicious software designed to gain access to a system without the users' consent and without being noticed by security tools. Once successfully deployed, rootkits are put on the target computer by malware developers. This allows hackers to remotely execute files and change configurations.

k. Ransomware: Computers are attacked by ransomware, which locks files in an encrypted state and refuses to release the key unless the client pays a fee. Attacks using ransomware that target businesses and the government are increasing, costing victims millions of dollars as some pay to recover crucial systems from the hackers. The Cryptolocker malware family, often known as Petya or Loky, is one of the most widespread and well-known ransomware families.

1.3. Feature Selection

While feature selection determines the best suitable set of properties for a specific target variable, structure learning demonstrates the relationships among all the parameters, generally by showing these linkages as a graph. It is basically expected when utilising a method of feature selection that the data may contain particular features that could be eliminated with minimal to no loss of data because they are redundant or unnecessary. The phrases "irrelevant" and "redundant" should not be used interchangeably because a relevant attribute may become redundant if another relevant feature also exists that is actually closely related. It is advisable to keep feature selection and feature extraction independent techniques. Through feature extraction, the ability of the distinctive features is exploited to create novel features, whereas feature selection just produces a subset of those functionalities. In areas where there are many characteristics but few samples or data points, feature selection methods are frequently used. The accuracy of learning is increased, computation time is decreased, and a better comprehension of the learning method or data is made possible by removing irrelevant data. Typically, not all characters in the input can be employed to actually construct a model using machine learning. Repetitive variables can diminish a classifier's accuracy and reduce a model's capacity for generalisation. A complicated model is also created when additional factors are added to an existing one.

1.4. Need of Feature Selection

Understanding the necessity for any approach, as well as the requirement for Feature Selection, is crucial before putting it into practice. It is common knowledge that a large current dataset is required for machine learning to provide improved outcomes. We collect a large amount of data to refine our model and support its learning. The dataset often consists of inaccurate data, irrelevant data, and a little quantity of useful data. The vast amount of data also slows down the training process for the model, and with disorder and inappropriate input, the model may not predict accurately or perform well. Therefore, it is essential to remove these disturbances from the dataset and include much less data. Attribute selection techniques are employed to accomplish this. The best features should be chosen to improve the model's performance.

1.5. Feature Selection Techniques

The two primary categories of feature selection strategies are presented in figure 2.

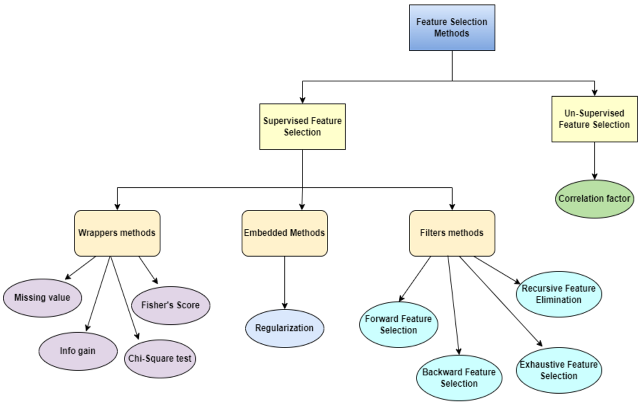

Feature Selection Methods
Supervised Feature Selection
Un-Supervised Feature Selection
Correlation factor
Wrappers methods
Embedded Methods
Filters methods
Missing value
Fisher's Score
Info gain
Chi-Square test
Regularization
Forward Feature Selection
Backward Feature Selection
Recursive Feature Elimination
Exhaustive Feature Selection

Fig. 2. Classification of Feature Selection Techniques

A Technique for Supervised Feature Selection

- The labeled dataset can be used with supervised feature selection methods that consider the target variable.
- The wrapper methodology approaches characteristic selection as a search problem, where several combinations are made, assessed, and contrasted with other options.

Utilizing the subset of attributes repeatedly trains the algorithm. Before a halting condition is met, variables are added to a starting (typically vacant) list of variables by forward selection.

• The repeating backward elimination procedure is the opposite of the forward selection process. Beginning with a consideration of all traits, the least significant one is first eliminated. The process of removing features is kept going until it has no further effect on the model's efficacy.

• Exhaustive feature selection is one of the best feature selection methods because it uses brute force technique to evaluate each feature set. It shows that every possible feature combination is created using this procedure, and the best set of features is then returned.

• Recurring greedy optimization techniques like recursive feature elimination choose features by repeatedly choosing a lower and smaller group of characteristics.

• Features in the filter method are chosen based on statistical metrics. In this approach, the features are chosen during a pre-processing stage that is separate from the learning algorithm.

• While transforming the dataset, information gain sets the amount of entropy that is reduced.

• A technique for determining the relationships between categorical variables is the chi-square test.

• One of the widely used supervised feature selection strategies is Fisher's score. It returns the variable's position on the fisher's criterion in descending order.

• The output of the missing value ratio can be used to compare the property set to the given input threshold value.

The advantages of both wrapper and filter approaches were merged into embedded methods by taking into consideration feature interaction and cheap processing costs. These efficient processing methods are superior than classification methods in that they are faster.

• Regularization refers to methods for calibrating models of machine learning to reduce the modified loss function and avoid overfitting and underfitting.

• A Method for Unsupervised Feature Selection

Unsupervised feature selection methods can be applied to unlabeled datasets and ignore the target variable.

• Correlation: Correlation can be utilised to choose characteristics since the target and ideal variables have a high correlation. Although they should be connected with the target, variables should not be correlated with one another.

2. Related Works

In [1], M. Abhijith et al. (2019) observed that malware dangers may be found in practically every program architecture, from supercomputers to device firmware. The first of its kind artificial neural network, a feedforward neural network on a Field-Programmable Gate Array, has been created by the authors and could be used to identify viruses on mobile devices. The computational strain on the mobile CPUs can be decreased by developing specialised malware detection accelerators. These days' top-tier smartphones have dedicated neural processors. Current Extremely large Scale Integrated (VLSI) technology advancements are to thank for the decrease in silicon and die area costs. The majority of malware infections on Android OS occur when malicious applications or Application Packages are installed on a device (APKs). It is suggested that an artificial neural network be employed as the classifier. A dataset with open sourcing is produced. There are 3417 malware samples and 8057 benign ware samples in this dataset, Considerable Permissions.

In [2], Zarni Aung et al, have suggested a framework for Android's mobile operating system that can identify malicious programs. The authors researched Android malware detection using permissions. Researchers describe a virus detection tool for Android-based smartphones that uses machine learning. A machine learning-based system called Crowdroid can identify Trojan-like infections on Android mobile devices. The team gathers extracted features from Android applications to create a dataset that will be used to create an Android pathogen detection framework. The researchers implemented the characteristics for malware detection after identifying the actual permissions that the application needed. Malware sample databases are utilised to power some of the malicious software programs. Machine learning methods are employed for the classify both malicious software and legitimate apps. Because there were insufficient samples of malicious applications, the model was not trained using big datasets.

In [3], A.-D. Schmidt et al. has been reported that existing countermeasures to smartphone malware have flaws because they rely heavily on signatures. Researchers claim their new method is significantly more effective than existing ones for spotting malware on Android cellphones. Ad-Hoc networks are the technology that makes it possible for Android devices to implement collaborative intrusion detection. Since malware has long been a menace to computer systems, it was only a matter of time until the first harmful software creators developed an interest in rising-in-popularity mobile platforms like Symbian OS. Simple classifiers proved to be an effective method for detecting malware on Android when utilising static ELF analysis. Malware executables and function call sequences are compared in order to categorise them using the Prism, PART, and Nearest Neighbor algorithms.

In [4], Justin Khan et al extract features from bundled Android applications using the open source Androguard project. There is a growing risk connected with virus targeted at portable devices since mobile platforms have recently become capable of running increasingly complicated software. Utilizing the Scikit-learn framework, the group trains a One-Class Svm Classifier using these extracted characteristics. The kernel, which is specified across strings of any length, is utilised for the string features. There are two basic difficulties in utilising a classifier based on machine learning to detect malware. The first step is to extract some kind of feature representation for an application. Second, a classification method must be chosen in order to train the system using classes because the training dataset is virtually entirely benign.

In [5], J. Todd McDonald et al, have created a machine learning system that is capable of finding malware on Android smartphones. In the market for mobile operating systems, the Android OS has constantly been a strong competitor. Apposcopy showed an accuracy of 90% using 1027 harmful samples against fifteen different malware families, with 103 negative result and only two false positive results. The purpose of this study is to evaluate the effectiveness of solitary machine learning methods on manifest permissions using a data set from an Android app that is of a sizeable enough. The results demonstrated that all of the ML methods that employed just the file alone produced considerable gains when contrasted to the diagnostic accuracy of readily accessible anti-virus engines. In order to categories program as dangerous or benign, this study tests the efficacy of four different algorithms for machine learning in conjunction with features taken from the Android manifest file permissions. Results from case studies show recall, accuracy, and precision rates of more than 80%. According to science, ongoing research will concentrate on developing a revolutionary algorithm that is specifically tailored to the identification of malware. For static identification using manifest permission features, ensemble methods and the inclusion of additional ML algorithms are planned.

In [6], İsmail Atacak et al. claimed that a hybrid machine learning strategy based on CNN's features extraction layers with ANFIS is provided for the identification of mobile malware applications. The authors suggested utilising a convolutional neural network to identify Android malware from the Google Play Store's application authorization data. In the first dataset, the suggested model had an accuracy of 92%, and in the second, it had an accuracy of 94.6%. System analysis and lively analysis are the two main methods used to find malware. With the aid of convolutional and pooling layers and with the aid of all the applications' permission data, feature extraction is carried out. The ANFIS model makes predictions based on the acquired features. Ten mobile users participated in the analysis. The investigators note that deep learning frameworks produce positive malware detection outcomes but are constrained by their high variable count and memory requirements. The outcomes demonstrated that identical values may be discovered using deep learning techniques. The team suggests that combining the data gleaned from the static and dynamic analysis of malicious programmes with the data gleaned from the dynamic analysis can lead to more effective results.

In [7], Takamasa Isohara et al. examined the study of kernel-based behaviour for detecting malware on Android. Numerous Android trojans, like Geinimi and DroidDream, are made available on the app store. We suggest a system that can identify harmful activities on the market places since malware infection through Android application markets is the biggest threat to Android users. Without any outside evaluation, any Software program can be released on the market. For Android malware assessment, the team has proposed a kernel-based behaviour analysis. All activity of programmes running on the Android system will be included in the log data. To detect the leaking of personal information, some data that is kept on the device is collected automatically and transformed into a signature.

In [8], Anshul Arora et al. reported on malware in Android-based mobile devices utilising network traffic analysis. Overtaking rivals like Symbian, Google's Android has emerged as the most popular operating system for smartphones. A group of researchers has created a method for identifying and analysing network activity from a mobile phone's SIM card in order to find Android malware. Based on the characteristics of the network traffic, a traffic analysis is carried out to find malware on Android smart phones. The initial work for malware detection on Android has been completed. The vast majority of methods recommended for malware detection on Android include dynamic analysis, static analysis, and cloud-based solutions. The main objective of the given assignment is to locate Android malware that may be controlled remotely by a website and receive instructions from it. A total of 43 malware and viruses and 5 instances of traditional mobile traffic are used as test data to assess the classifier's accuracy.

In [9], Mariam Al Ali et al. report that malware poses serious risks to the nation's infrastructure, the service industry, and society at large's online security. With a focus on mobile computing platforms, they aim to create an useful and efficient unusual case malware detection system in their research project. Mobile devices are now targets of malware assaults. To achieve the goal, the authors generate system measures and use a number of effective machine learning approaches. The algorithms under consideration that produce the greatest outcomes are Random Forest and Support Vector Machines. Some of the studies purport to refute past discoveries in this field: The framework put forward is comparable to that described by Shabtai et al, who collected system measurements and then used classification methods for detection to examine them. The main distinction is that classifiers used in the device itself, which turned out to have drawbacks.

In [10], Min Zhao et al. reported on antiMalDroid. In this paper, the AntiMalDroid active learning framework for supporting vector machine-based malware detection is designed. A framework called Anti-MalDroid can find harmful applications on Android smartphones. Test results demonstrate the suggested approach's strong applicability and scalability on a range of common malware. Smartphones are becoming more and more widespread as a result of technological advancements in embedded systems and high-speed wireless communication networks. The authors contend that antiMalDroid is capable of identifying and preventing some known as well as some undiscovered malwares from operating on Android platforms. Future research will keep looking for more effective algorithms to cut down on time use. [16], [17] they investigated the concept of security using machine learning and deep learning methods for malware detection, as well as android malware detection with classification based on hybrid analysis and N-gram feature extraction.

The research gaps identifies from the studied existing approaches are tabulated in the table 1.

3. Proposed Methodology

The proposed methodology aims to selects the best attributes to classify the malware attacks in the android-based system. It has collected the dataset from the public repository known as “Kaggle”, which contains 215 attributes and description of the attributes are presented in

Table 2. Analysis on Malware Attributes

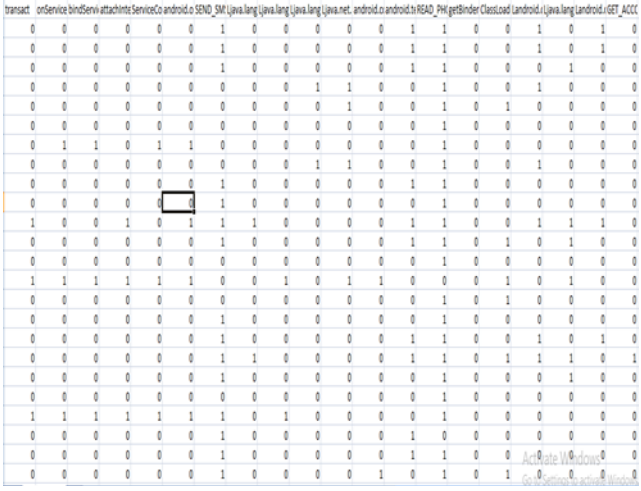
S. No	*Attribute Type*	*Count of Sub Types*
1	API Call Signature	73
2	Commands Signature	06
3	Intent	23
4	Manifest Permission	113
	Total	215

Table 1. Gaps Identified Using the Existing Approaches

S. No	*Author*	*Method*	*Merits*	*Demerits*	*Accuracy*
1.	M. Abhijith	FNN	Inference model is used, also uses sigmoid activation.	Complex and slow execution.	89.59%
2.	Zarni Aung	Crowdroid	Chose k-means clustering, create a dataset from extracted features of Android applications.	Smaller dataset is used,more features are need to be extracted.	93.45%
3.	A.-D. Schmidt	Artificial neural net- Work field-programmable gate array	low computational cost because of pre trained model.	Some more semantical features can be added so that for complex approaches.	93%
4.	Justin Khan	Androguard and one single SVM	Can work in complex software, use of Mercer kernel.	Didn’t perform classification	84.6%
5.	J. Todd Mcdonald	Machine learning-rf and SVM	4 machine learning algorithms in conjunction are used	General-purpose algorithms are not recommended.	81%
6.	İsmail Atacak	CNN	Low computational cost, uses adaptive neuro-fuzzy inference system	Static analysis and dynamic analysis should be performed.	92%
7.	Takamasa Isohara	Kernel-base Behavior analysis	Uses signature matching approach, can also detect the 37 different leaks.	Takes more time, expensive	92.05%
8.	Anshul Arora	Machine learning	3 level classification	Same approach can’t be used for different malwares.	93.75%.
9.	Mariam Al	Random forest	System metrics are generated first,	No security measures are suggested,	95%

the table 2.

A sample dataset which represents the storage of the values is projected in the figure 3 for the simple understanding



transact	onService	bindServi	attachInte	ServiceCo	android.o	SEND_SM	Ljava.lang	Ljava.lang	Ljava.lang	Ljava.net.	android.o	android.t	READ_PH	getBinder	ClassLoad	Landroid.r	Ljava.lang	Landroid.r	GET_ACCC
0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	1	1	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	0	1	0	1	1	1	0	0	0	0	1	1	0	0	1	1	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	1	1	1	1	1	0	0	1	0	1	1	0	0	0	1	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	1	0
0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	1	1	1	0	1
0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	1	1	1	1	1	1	0	1	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0

Fig. 3. A Sample description of the Dataset Storage Values

The proposed model initially pre-processes the class label using the label encoder technique to convert it into numerical value which makes the algorithm to process the elements at faster rate. Out of the two encoding techniques, i.e., label and one hot encoding, the proposed model has chosen label encoding because one hot encoding increases the dimensionality of the dataset. Already, the existing dataset is high in dimension. So to reduce the complexity of the model marks the initially attacks of malware as 0 and high impact malware as 1.

In order to select the features, the proposed model initially implements the fisher's score to select the K best features in traditional wrappers method. This method computes the score of each attribute using (1).

$$Fish\_Attr[i] = \frac{\sum_{i=0}^{n} C_n * \sum_{j=0}^{k} (g\_mean_{ij} - mean_j)^2}{\sum_{i=0}^{n} C_n * \sum_{j=0}^{k} \sigma_{ij}^2} \qquad (1)$$

By default, K is assumed as 10 in the fishers score i.e, this model selects top 10 best attributes and this static selection doesn't help for the efficient model designing because it is not true that for any attribute only top 10 attributes are sufficient. One can pass a dynamic value but choosing that random value is a brute force technique. The accuracy obtained using this approach is "79.03%." So, the proposed model tries to implement recursive feature elimination of filter technique as another comparison model.

3.1. Recursive Feature Elimination

RFE is a wrapper-style feature selection algorithm. With order to help with the selection of features, a unique machine learning strategy is presented, used as the product's core, wrapped in RFE. Contrarily, feature selections made with filters give each feature a score before selecting the those with the best (or lowest) score. RFE attempts to identify a set of traits by efficiently removing variables one at a time until the appropriate number of variables is left, beginning with all of the characteristics in the training sample. To do this, the core model's machine learning algorithm is first optimised, the features are sorted according to significance, the least significant features are eliminated, and the model is then re-optimized. This method is done until a predetermined number of qualities remain. A decision tree or other method of calculating significant scores must be provided by the algorithm. Different algorithms may be employed in RFE; the approach that best fits the chosen features is not required.

3.1.1. Procedure for Recursive Feature Elimination

- Feature elimination from training data for feature selection using RFE is an effective method.

- To achieve classification accuracy, RFE employs a tree structure and automatically chooses a number of features. The picked features are then fitted with a decision tree.

- The library's most recent iterations include RFE.

- RFE operates by commencing with all attributes in the training sample and exploring for a selection of those features.

- In addition to using internal filter-based feature selection, RFE is a feature selection technique in the wrapper style. RFE is simple to set up and use.

The proposed model selects 160 attributes as best using this recursive feature elimination in the forward direction. The accuracy obtained using this approach is "85.18%". This computation is not suitable for the proposed model because the malware detection process doesn't involve any computation of feature importance in the classification process. Utilization of RFE model sometimes makes the model to suffer from over-fitting because selection of unwanted attributes makes the likehood of these elements more on class labels. In general process of feature selection, RFE eliminates few features and reduces few features but that sub set generation is difficult to realize during the training process. So, the proposed model implements an embedded approach that combines the RFE with Non-linear Support Vector Machine by defining the best estimators using the cross-validation process. The model chooses non-linear SVM because the data distribution in the android packages is unpredictable due to different malware attacks. The right chose of kernel helps the model to convert the complex structure to linear easily.

3.2. Non-linear SVM algorithm

Non-Linear SVM is applied to non-linearly segregated data, which indicates that a dataset is non-linear data if it cannot be classified using a linear graph, and a Non-Linear Classification technique is used to do so. The information or vectors that are closest to the hyperplane and have the biggest impact on where it is placed are called support vectors. As a result of their support for the hyperplane, these vectors are known as support vectors. In order to find the line out of each class that is closest to the nearest point, the SVM approach is used. Support vectors are the names given to these points. What separates the vectors from the hyperplane is called the margin. This margin is to be raised by SVM. The best hyperplane is the one with the largest margin. When dealing with linearly organized data, the model can divide it using a straight line, but it cannot divide non-linearly organized data. As a result, the model has to include a new dimension to help it distinguish between these data values. For linear data, the system has employed the two dimensions x and y; therefore, for non-linear data, the system would add the three - dimensional object z. It can be calculated as z=x2+y2

Any SVM method defines a hyperplane that clearly shows the distinction between negative and positive classes. It is very hard to work with the actual data, so the SVM algorithm computes the dot product two attributes using the kernel function. There are three types of kernels as shown in figure 4.

Similarly, there is another parameter known as “C” that defines the strength of regularization plays crucial role in fitting the model to the data. Too much high or too much low value will adversely affects the model. Another parameter known as γ, represents the influence of single training record on the entire dataset. Lower the value, farther is the record from the hyper-plane and higher the value, closest is the record from the hyper-plane. To find the optimal values for these elements, the proposed model utilizes Grid Search optimization by checking all the possible combinations.

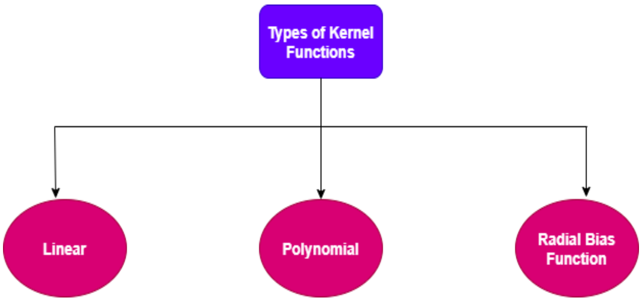


Fig. 4. Classification of Kernel Functions.

After completion of the process, it has identified that RBF is best kernel, γ is 0.001 and C is 100 for reducing the dimensionality. The entire process is clearly represented in figure 5.

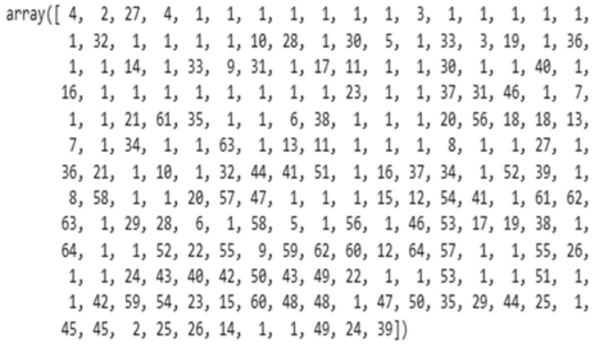
Fig. 5. Working of Proposed Model For Feature Selection

4. Results and Discussion

Table 3. Features Selected Using Fisher’s score

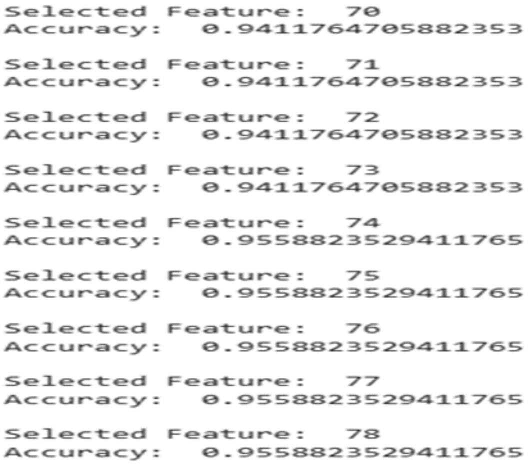
Attribute Number	*Attribute Name*
1	Transact
2	onServiceConnected
3	bindService
4	attachInterface
5	ServiceConnection
6	android.os.Binder
7	SEND_SMS
9	Ljava.lang.Class.getMethods
10	Ljava.lang.Class.cast
11	Ljava.lang.Class.getField

Figure 6 presents the ranking of each feature assigned by SVM algorithm in which presence of 1 represents that attribute is essential and any other value represents that it not essential. The model found that 89 attributes are important using this traditional SVM. The accuracy obtained in this approach is “92.64%”.



```
array([ 4,  2, 27,  4,  1,  1,  1,  1,  1,  1,  1,  3,  1,  1,  1,  1,  1,
        1, 32,  1,  1,  1,  1, 10, 28,  1, 30,  5,  1, 33,  3, 19,  1, 36,
        1,  1, 14,  1, 33,  9, 31,  1, 17, 11,  1,  1, 30,  1,  1, 40,  1,
       16,  1,  1,  1,  1,  1,  1,  1,  1, 23,  1,  1, 37, 31, 46,  1,  7,
        1,  1, 21, 61, 35,  1,  1,  6, 38,  1,  1,  1, 20, 56, 18, 18, 13,
        7,  1, 34,  1,  1, 63,  1, 13, 11,  1,  1,  1,  8,  1,  1, 27,  1,
       36, 21,  1, 10,  1, 32, 44, 41, 51,  1, 16, 37, 34,  1, 52, 39,  1,
        8, 58,  1,  1, 20, 57, 47,  1,  1,  1, 15, 12, 54, 41,  1, 61, 62,
       63,  1, 29, 28,  6,  1, 58,  5,  1, 56,  1, 46, 53, 17, 19, 38,  1,
       64,  1,  1, 52, 22, 55,  9, 59, 62, 60, 12, 64, 57,  1,  1, 55, 26,
        1,  1, 24, 43, 40, 42, 50, 43, 49, 22,  1,  1, 53,  1,  1, 51,  1,
        1, 42, 59, 54, 23, 15, 60, 48, 48,  1, 47, 50, 35, 29, 44, 25,  1,
       45, 45,  2, 25, 26, 14,  1,  1, 49, 24, 39])
```

Fig. 6. Ranks Assigned by Traditional SVM Approach



```
Selected Feature:  70
Accuracy:  0.9411764705882353

Selected Feature:  71
Accuracy:  0.9411764705882353

Selected Feature:  72
Accuracy:  0.9411764705882353

Selected Feature:  73
Accuracy:  0.9411764705882353

Selected Feature:  74
Accuracy:  0.9558823529411765

Selected Feature:  75
Accuracy:  0.9558823529411765

Selected Feature:  76
Accuracy:  0.9558823529411765

Selected Feature:  77
Accuracy:  0.9558823529411765

Selected Feature:  78
Accuracy:  0.9558823529411765
```

Fig. 7. Selection of Attributes using Embedded SVM Algorithm

Figure 7 presents the highest accuracy obtained by the embedded SVM model by choosing the minimum number of attributes. Using this approach with 74 attributes the model has got 95.58% accuracy, which is much better than traditional approaches.

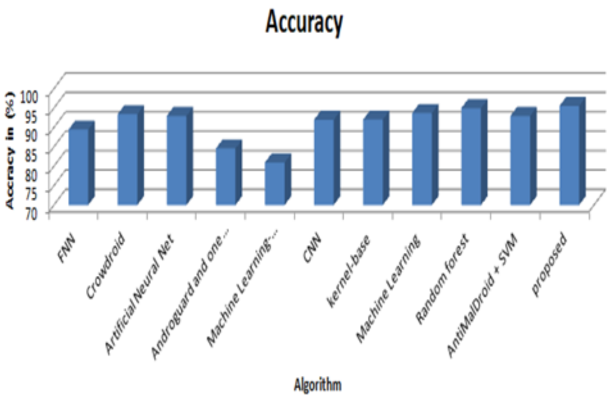


Fig. 8. Accuracy Analysis between Existing & Proposed Approaches

Figure 8 evaluates the effectiveness of the suggested model by comparing it to the accuracy levels attained by the current methods. It displays the X-axis for algorithm names and the Y-axis for accuracy metrics. Out of the existing approaches, the model has got more accuracy than deep learning as well as traditional bagging algorithms.

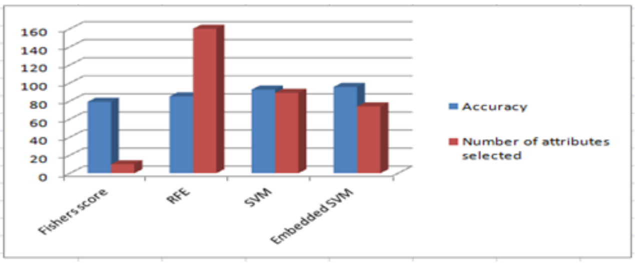


Fig. 9. Analysis on Accuracy & Feature Selection

Figure 9 presents the comparative analysis considered by proposed model by picking the different sub approaches of the traditional supervised feature selection techniques. Out of the approaches, even though fishers have 10 attributes by default, the proposed model has got correct attributes using the marginal analysis of the SVM.

5. Conclusion

Feature selection is a crucial processing step for the efficient classification of any machine learning approach. Feature selection for the supervised models can be done in three ways i.e., simplest and earlier approach is wrapper methods which is based on computation of score for each attribute. The computation makes the system as expensive because it involves local and global mean and standard deviations. Second easiest approach is RFE, which has the capability to generate subset in both forward and backward directions based on the entropy calculation. Generating more number of trees requires additional resources with high dimensionality datasets. To limit the decisions in tree generations, the proposed model has embedded SVM algorithm to RFE based on dataset utilized for malware detection. Since the distribution of data is unpredictable the tuning model projected that non-linear Radial Bias Function kernel with gamma value 100 is suitable for the integration process. This has reduced the attributes from 215 to 74 with high accuracy. Based on the separation in between feature and class label, the radial bias function takes the influence of the attribute into account. The significance of the feature is determined by the angular distance.

Author contributions

Ravi Eslavath: study conception and design, data collection, analysis and interpretation of results, and manuscript preparation. **Dr. Upendra Kumar Mummadi:** reviewed the results and approved the final version of the manuscript.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] Iyer, B., Deshpande, P. S., Sharma, S. C., &Shiurkar, U. (Eds.). (2020). Computing in Engineering and Technology. Advances in Intelligent Systems and Computing. doi:10.1007/978-981-32-9515-5

[2] Aung, Z., Zaw, W.: Permission-based android malware detection. Int. J. Sci. Technol. Res. 2(3), 228–234 (2018)

[3] Schmidt, A.D., Bye, R., Schmidt, H.G., Clausen, J.H., Kiraz, O., Yuksel, K.A., Camtepe, S.A., Albayrak, S.: Static Analysis of Executables for Collaborative Malware Detection on Android. In: Proceedings of IEEE International Conference on Communications, ICC 2019, Dresden, Germany, 14-18 June 2019, IEEE (2019) 1-5

[4] Sahs, J., & Khan, L. (2020). A Machine Learning Approach to Android Malware Detection. 2020 European Intelligence and Security Informatics Conference. doi:10.1109/eisic.2012.34

[5] McDonald, J. T., Herron, N., Glisson, W. B., Benton, R. K.,(2021). Machine Learning-Based Android Malware Detection Using Manifest Permission. Proceedings of the 54th Hawaii International Conference on System Sciences. https://doi.org/10.24251/HICSS.2021.839

[6] Atacak İ, Kılıç K, Doğru İA. 2022. Android malware detection using hybrid ANFIS architecture with low computational cost convolutional layers. PeerJ

Computer Science 8:e1092 https://doi.org/10.7717/peerj-cs.1092

[7] Isohara, T., Takemori, K., & Kubota, A. (2021). Kernel-based Behavior Analysis for Android Malware Detection. 2011 Seventh International Conference on Computational Intelligence and security. doi:10.1109/cis.2011.226

[8] Arora, A., Garg, S., &Peddoju, S. K. (2014). Malware Detection Using Network Traffic Analysis in Android Based Mobile Devices. 2014 Eighth International Conference on Next Generation Mobile Apps, Services and Technologies. doi:10.1109/ngmast.2014.57

[9] Ali, M. A., Svetinovic, D., Aung, Z., & Lukman, S. (2017). Malware detection in android mobile platform using machine learning algorithms. 2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS). doi:10.1109/ictus.2017.8286109

[10] Zhao, M., Ge, F., Zhang, T., & Yuan, Z. (2021). AntiMalDroid: An Efficient SVM-Based Malware Detection Framework for Android. Information Computing and Applications, 158–166. doi:10.1007/978-3-642-27503-6_22

[11] Ali Muzaffar, Hani Ragab Hassen, Michael A. Lones, Hind Zantout, An in-depth review of machine learning based Android malware detection, Computers & Security, Volume 121, 2022, 102833, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2022.102833.

[12] Sihag, V., Vardhan, M., Singh, P., Choudhary, G., & Son, S. (2021). De-LADY: Deep learning based Android malware detection using Dynamic features. Journal of Internet Services and Information Security, 11(2), 34–45. https://doi.org/10.22667/JISIS.2021.05.31.034

[13] Karbab, E.B., Debbabi, M. (2021). PetaDroid: Adaptive Android Malware Detection Using Deep Learning. In: Bilge, L., Cavallaro, L., Pellegrino, G., Neves, N. (eds) Detection of Intrusions and Malware, and Vulnerability Assessment. DIMVA 2021. Lecture Notes in Computer Science(), vol 12756. Springer, Cham. https://doi.org/10.1007/978-3-030-80825-9_16

[14] Mahindru, A., Sangal, A.L. MLDroid—framework for Android malware detection using machine learning techniques. Neural Comput & Applic 33, 5183–5240 (2021). https://doi.org/10.1007/s00521-020-05309-4

[15] Daoudi, N., Samhi, J., Kabore, A.K., Allix, K., Bissyandé, T.F., Klein, J. (2021). DEXRAY: A Simple, yet Effective Deep Learning Approach to Android Malware Detection Based on Image Representation of Bytecode. In: Wang, G., Ciptadi, A., Ahmadzadeh, A. (eds) Deployable Machine Learning for Security Defense. MLHat 2021. Communications in Computer and Information Science, vol 1482. Springer, Cham. https://doi.org/10.1007/978-3-030-87839-9_4

[16] Ravi, Eslavath, and Mummadi Upendra Kumar. "A Comparative Study on Machine Learning and Deep Learning Methods for Malware Detection." Journal of Theoretical and Applied Information Technology 100.20 (2022).

[17] Ravi, Eslavath, and Mummadi Upendra Kumar. "Android malware detection with classification based on hybrid analysis and N-gram feature extraction." International Conference on Advancements in Smart Computing and Information Security. Cham: Springer Nature Switzerland, 2022.

[18] Suresh Mamidisetti and A. Mallikarjuna Reddy, "A Stacking-based Ensemble Framework for Automatic Depression Detection using Audio Signals" International Journal of Advanced Computer Science and Applications(IJACSA), 14(7), 2023. http://dx.doi.org/10.14569/IJACSA.2023.0140767

[19] Cheruku, R., Hussain, K., Kavati, I. et al. Sentiment classification with modified RoBERTa and recurrent neural networks. Multimed Tools Appl (2023). https://doi.org/10.1007/s11042-023-16833-5.

[20] Naik, S., Kamidi, D., Govathoti, S. et al. Efficient diabetic retinopathy detection using convolutional neural network and data augmentation. Soft Comput (2023). https://doi.org/10.1007/s00500-023-08537-7.

[21] Mallikarjuna A. Reddy, Sudheer K. Reddy, Santhosh C.N. Kumar, Srinivasa K. Reddy, "Leveraging bio-maximum inverse rank method for iris and palm recognition", International Journal of Biometrics, 2022 Vol.14 No.3/4, pp.421 - 438, DOI: 10.1504/IJBM.2022.10048978.

[22] Sudeepthi Govathoti, A Mallikarjuna Reddy, Deepthi Kamidi, G BalaKrishna, Sri Silpa Padmanabhuni and Pradeepini Gera, "Data Augmentation Techniques on Chilly Plants to Classify Healthy and Bacterial Blight Disease Leaves" International Journal of Advanced Computer Science and Applications(IJACSA), 13(6), 2022. http://dx.doi.org/10.14569/IJACSA.2022.0130618

[23] A. Mallikarjuna Reddy, K. S. Reddy, M. Jayaram, N. Venkata Maha Lakshmi, Rajanikanth Aluvalu, T. R. Mahesh, V. Vinoth Kumar, D. Stalin Alex, "An Efficient Multilevel Thresholding Scheme for Heart Image Segmentation Using a Hybrid Generalized

Adversarial Network", Journal of Sensors, vol. 2022, Article ID 4093658, 11 pages, 2022. https://doi.org/10.1155/2022/4093658.

[24] P. S. Silpa *et al.*, "Designing of Augmented Breast Cancer Data using Enhanced Firefly Algorithm," *2022 3rd International Conference on Smart Electronics and Communication (ICOSEC)*, 2022, pp. 759-767, doi: 10.1109/ICOSEC54921.2022.9951883.