Research Article

Malware Classification Using Xgboost With Vote Based Backward Feature Elimination Technique

Munisamy Eswara Narayanan[1], Balasundaram Muthukumar[2]

[1]Research Scholar, Sathyabama Institute of Science and Technology, Chennai, India, Technology Lead, Infosys Ltd, Chennai, India.
[2]Professor, Department of Computer Science and Engineering, United Institute Of Technology, Coimbatore. India.

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 28 April 2021

Abstract

Malware is one of the most popular threats today, and it is rapidly becoming a significant threat to Internet security. Malware is computer code written by cyber criminals with the intent of causing extensive harm to data and infrastructure or gaining unauthorized access to a network. There are several methods are employed to detect the malware with signature based and behaviour based techniques. Several machine learning techniques are used for classification of malware files. The traditional techniques are not efficient to detect the malware. To efficiently classify the malware, we proposed the XGB with Vote based Backward Feature Elimination technique (XGB-VBFE) which selects the optimal features to build the model and classifies the files with higher accuracy. The performance of the proposed system is compared with other machine learning algorithms such as SVM and Random Forest and proved to be better in accuracy, precision and recall. The proposed XGB-VBFE classifies the files with the accuracy of 99.50%, precision 0.99 and recall 0.96.

1. Introduction

Malware detection and classification is a challenging task in OS security. OS are vulnerable to malware attacks. To secure the operating system, the malware should be detected and classified. The windows PE files are extracted to classify the malware and benign files. The Portable Executable (PE) file format is used in 32-bit and 64-bit Windows operating systems to store executables, object code, DLLs, and other files. The PE format is a data structure that contains all of the details that the Windows OS manager needs to handle the wrapped executable code.

In all over the world, a large number of people use operating system in their PC and laptop. It has a great, easy, simple, attractive GUI, easy functionality and many more similar characteristics. But it is not open source. As we know everything has its own two sides one is dark other is bright. Same happens in case of Windows, it has the user from all around the globe and this much user uses it as their operating system. So this creates great interest in the hackers to steal a thing from the user and tends them to penetrate into your privacy [1].

The ever-increasing number of malwares creates a variety of vulnerabilities to information and security. Malware is generally characterized as a program created by hackers to interrupt the computer system's operations and steal data's identity. Malware hijacks the infrastructure without permission, causing harm to individual users, businesses, and government agencies. When malware infects a user's computer, it searches the device for vulnerabilities and exploits them to cause harm. It also takes certain unintended acts that lower the system's efficiency [2].

To avoid the harm caused by malware intrusion, timely identification and eradication are critical. However, discovering previously unknown malware faces a significant risk. Despite the fact that antimalware tools utilize a range of monitoring methods to fight unknown malware, they are unable to identify it in a timely manner due to the malware's advanced characteristics. The static features of the file are removed without executing it in the signature-based identification technique. This method is ineffective in combating new sophisticated malware that employs evasion techniques.

A behavioural-based identification technique has emerged as an important supplement to solve this issue. This procedure runs the software in a different context in order to track and log the events that occur during implementation. It keeps track of the list of code calls or API calls made during the program's execution. Though this method is highly effective and robust compare to signature-based recognition, it does take time and resources. In order to build an accurate and efficient Malware Detection Approach, it is important to use heuristic-based techniques to examine and differentiate malware. It looks at things like API calls, application calls, structural specifics like information of header and opcode. The first step in reducing the dimensions of the features and enhancing the efficiency of the prediction of classifiers is to recognize significant features while eliminating noisy features. [3].

For major security tasks such as finding distribution of malware through email, the Web, and accurate file classification, a hybrid machine learning malware recognition model is used. Instead of a hybrid model with static and dynamic components, this model uses a variety of hybrid approaches to improve identification speeds, including hybrid classifiers, hybrid features. This hybrid approach improves accuracy of malware and benign file classification, making it an important tool for malware prevention, especially for automated analysis techniques. [4].

Microsoft Malware classification challenge dataset is used. A main stage in successfully examining and categorizing such a huge number of records is to organize and identify them into their corresponding families. In addition, new files encountered on computers can be subjected to certain grouping requirements in order to identify them as malicious and belonging to a certain family. It consists of a set of documented malware files from nine

separate families. Every malware file has a hash value of 20-characters which is an identifier that uniquely identifies the file, as well as an integer class code that describes one of the nine malware family names [5].
In order to enforce function collection, we reduce the data set's dimension. The model then employs a machine learning technique to detect device threats, as well as the ability to summarize data to aid intrusion detection. The proposed work necessitates feature extractions, and dimension reduction that allows for feature extraction, and feature collection. Feature extraction is the method of using all transition functionality, and then all the main features are combined [6].
Static or dynamic analysis is used to remove malware features. Static analysis analyses files that are either executable or disassembled without executing them. An APE (Portable Executable) file's headers and portions inform the dynamic linker how to map the file into memory. PE's IAT (Import Address Table) can be examined to allow use of DLL information. Byte sequence, byte entropy, n-gram, opcode (operational code), API sequence, and other features were selected [7].

In real-time malware detection, feature selection is critical in terms of detection accuracy, time, and training effort. 5 percent threshold is mostly selected by Backward Feature Elimination. The features are then learned using a machine learning algorithm, and the function with the highest P-value is discovered. The function is excluded from the initial feature list if the highest P-value obtained from training is higher than the maximum value. The alternative is preferred if the first choice is not available. The modified features are now trained again once, and the procedure is repeated up to the highest P-value feature is less than the threshold limit. [8].
Approaches that use machine learning and similitude mining to visualize static and dynamic malware detection. Several static malware detection approaches have distinguished themselves by examining various classification methods includes support vector machine (SVM), k-Nearest Neighbour (KNN), and Nave Bayes (NB), among others. According to the results, XGBoost classification can achieve a precision of more than 90%. [9].The contribution of the proposed work is as follows:

- The XGB-VBFE is proposed in this work using Microsoft Malware Classification challenge dataset. With hybrid features, the XGB-VBFE can accurately distinguish ransom ware and benign PE files.
- The features are selected by vote algorithm with data filtering method called Information Gain.
- The optimal features are selected by the wrapper method called Backward Feature Elimination.
- We implemented the proposed work and evaluated the performance with other state-of-the-art techniques. Our method achieves higher accuracy, training time and better classification.

The rest of the work is organized as follows. In Section2, reviewed the malware identification related works. Section 3 explains the outline of the proposed XGB-VBFE. In Section 4, the results are presented. In Section 5, the work is concluded.

2. Literature Survey

Dima Rabadi and Sin G Teo (2020) proposed and used a light-weight API-based dynamic function extraction technique to introduce a malware detection and category classification method. They use a fair dataset of 7774 benign and 7105 malicious samples from ten different malware forms to test their approach.

Catak et al. (2020) created a new dataset that represents malicious software activity from Windows operating system API calls. Adware, Downloader, Backdoor, Dropper, Trojan, Virus, spyware, Trojan, Virus, and Worm are among the malicious malware forms included in the dataset. The LSTM (Long Short-Term Memory) classification method was used in this research, and it is a commonly used method for classification in consecutive data.

Megira et al. (2018) examined malware using malware samples in order to better understand how it infects computers and devices, the extent of danger it presents, and how to protect devices from it. Malware can be dealt with by understanding how to operate when attacking a computer device.

P HarshaLatha and R Mohanasundaram (2020) described the overview of malware and types of malware, as well as define the new method of using machine learning techniques in anomaly based classification, and described the previous work related to malware analysis classification using machine learning techniques, as well as describe the important challenges that are faced.

Huang et al. (2020) suggested incorporating dynamic and static analysis to create a hybrid visualization of malware. They use the Cuckoo Sandbox to implement dynamic analysis on samples, transform the effects of the dynamic analysis into a visualization image using a programmed algorithm, and train the neural network on static and hybrid visualization images in hybrid visualization.

Yan et al. (2018) proposed a novel technique called MalNet, a malware recognition tool that spontaneously extracts features from raw data. To put it another way, by using the IDA decompilation tool to remove the malware's opcode sequences, create a grayscale version of the malware file. MalNet then employs

CNN and LSTM networks to benefit from grayscale pictures and opcode strings correspondingly, before executing malware classification using a stacking ensemble.

To find an effective solution, Pan et al (2020) uses a dataset that was created by integrating heartbeat and threat reports obtained by Microsoft's endpoint security solution. Then, to construct models, three algorithms (Logistic Regression, KNN, and LightGBM) are chosen, and results are obtained. The results illustrates that the LightGBM technique obtains the highest accuracy, with an AUC of 0.720687, and it saves the most time.

Shahini et al (2019) used several classification models to allocate a likelihood of computer which are affected by malware. The LightGBM method in this work is the best machine learning model because it is fast, more efficient, and uses less memory. The cross-validation ROC-AUC score for the LightGBM algorithm was 74 percent.

Zhang et al. (2020) introduced a new metric called a soft relevance value (s-value), for determining feature soft relevance that employs the mixed space criterion. They use the mixed distance criterion from pattern recognition to identify research models as a new family that hasn't been labelled in the training collection. Finally, they look at how malware databases is used to identify and classify new malware families using s-value.

To fix the limitations of conventional approaches, Fang et al (2019) implemented Deep Q-learning based Feature Selection Architecture (DQFSA). Without human interference, the proposed architecture selects a limited range of extremely distinguished features for the malware detection mission. By sequentially interacting with the features space, DQFSA trains an agent using Q-learning to optimize the estimated accuracy on a validation dataset of the classifiers.

Sergii Banin and Geir Olav Dyrkolbotn (2018) investigated whether low-level features could be used to classify multinomial malware. They differentiate between malwares of 10 families and 10 types of malware by looking at memory access patterns. They prove that their approach of classifying the families of malware works better than it does for classifying malware into forms, and they go into great detail about their accomplishments.

To distinguish malware families, Choi et al (2018) derived unique features from frequency distribution of malware. They break the malware down into sections and apply DCT/DFT to each one. The proposed approach achieves high accuracy and low operating cost, according to the results of the experiments.

Gholamreza Farahani (2020) introduced a new feature selection based on cross-correlation approach and related it to the feature selection based on mutual information and cuttlefish algorithm (CFA) using four different classifiers: naive Bayes (NB), K-nearest neighbour (KNN), support vector machine (SVM), and decision tree (DT). The proposed method outperforms the competition in terms of F1-score, precision, accuracy, and recall on the AWID, KDD Cup 99, CIC-IDS2017, and NSL-KDD datasets.

P. S. S. Siva Krishna and P. Venkateswara Rao (2019) presented a review of Liu et al.'s Automatic detection and classification of malware, Ashu Sharma et al.'s advanced malware classification technique. Bashari et al.'s ANN based detection and classification of malware. Mansour Ahmadi et al.'s effective Feature fusion for classification of malware family. Finally, they compared and contrasted all of the approaches listed above.

Rajesh Kumar and Geetha S (2020) proposed a malware classification scheme that uses computing resources with low-end and a wide balanced malware dataset to create a model. The model is optimized for efficiency by removing noisy features and reducing the dataset's feature sets using domain expertise in malware detection, as well as the XGboost feature value functionality and hyperparameter optimization.

Niranjan Agnihotri (2017) attempted to create a proactive classifier model based on Machine Learning that can detect ransomware based on the static attributes of PE files. Traditional signature-based detection is inefficient and unreliable, and it cannot detect zero-day attacks.

3. METHODOLOGY

In this section, the process flow of the proposed work is explained such as dataset, feature selection, machine learning techniques. The malware classification method consists of various steps such as feature extraction and training the machine learning classifier to categorize the malware and benign files. The features of PE files such as dynamic features and static features are extracted to analyse the malware. The extracted features are combined to get the hybrid features. The best features are selected from the hybrid features using Vote algorithm with the data filtering technique called Information Gain and then the optimized features are selected by using the wrapper method called backward feature elimination technique. Finally, the optimized feature set is provided to the XGB classifier to build the model. After, the XGB algorithm classifies the malware and benign files.

3.1 Dataset

In this article, the PE files are classified into benign and malicious types. The Program folders in the Windows 10 operating system contain benign PE files. Microsoft's malware classification challenge dataset includes PE files of malware. There are two sets of data in this dataset: a train set and a test set. The data is 200 GB in size (uncompressed). There are 10,868 test samples that can be used. For each sample, the corresponding.asm and .byte files are available. Using the interactive disassembler (IDA) method, each malicious file is disassembled into two files: assembly and binary file. The content of a binary file without a portable executable (PE) header is represented in hexadecimal numbers.

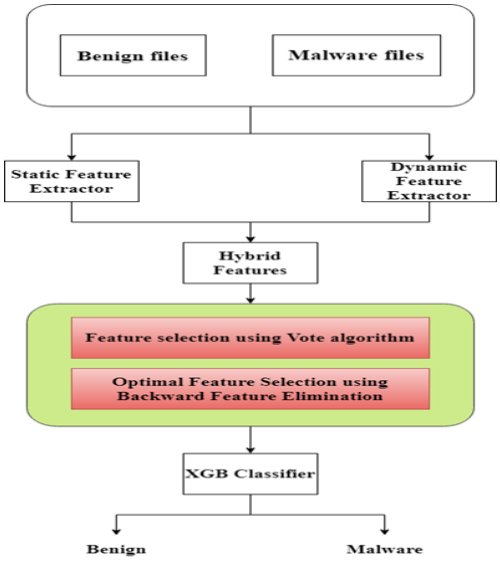

Figure 1 Architecture of Proposed System

There are nine families of malware in the dataset of Microsoft includes Lollipop, Simda, Kelihos_ver3, Ramint, Vundo, Obfuscator.ACY, Tracur, Kelihos_ver1, and Gatak. Vote algorithm and backward feature elimination feature selection techniques are used to select the most significant features in dataset. These two methods are very effective in reduction of data which filters the noise data that results in time complexity, less storage, and increase the accuracy of classifiers.

Table 1- Malware Family in the dataset

Malware Family	**Number of Samples**
Vundo	475
Lollipop	2478
Ramint	1541
Kelihos_ver3	2942
Tracur	751
Simda	42
Obfuscator.ACY	1228
Kelihos_ver1	398
Gatak	1013

3.2 Static Features

Static features are the features of windows PE files. Without involving the program's execution, static features are extracted from it. Static features in Windows PEfiles are extracted from one of two sources of information: the executable binary content or the programming languages source file obtained after decompiling and disassembling the binary executable. The detection of malware is focused on the study of static features. Opcodes, Dynamic Link Library related information, Printable String Information, N-grams (extracted input file's byte sequences), OH, FH, DOSH, or any combination from the PE files could be used.

Table 2- List of Features

Features	Number
DOS Header	12
File Header	10

Optional Header	25
Data Directory	23
Resource	29
DLL Imported	1
API called	1
The number of DLL imported	9
The number of API called	9
The header of .text sections	9
The header of .data sections	9
The header of .rsrc sections	9
The header of .rdata sections	9
The header of .reloc sections	9

3.3 Dynamic Features

Dynamic features are the features of the windows files that are gathered from malware execution during runtime. Monitoring malware as it runs (and detecting the actual sequence of instructions executed or the sequence of API functions triggered) or inspecting the machine after the malware has executed are both examples of dynamic analysis. It shows the existence of processes, the manipulation of files and registry entries, and the changes made to memory values, variables and registers. The retrieval of dynamic features are usually done in a sandbox tool that prevent the malware infection from spreading to the host machine.

3.4 Hybrid Features

Malware detection systems that rely solely on dynamic and static analysis are not capable of detecting advanced malware. As a result, hybrid features are created by combining static and dynamic features. To detect malware, hybrid features are used. As a result, based on hybrid features, malware detection approaches have appeared, integrating raw features (such as DOSH, OH, and others) with a range of derived features to implement more robust classification and detection.

3.5 Feature Selection

The vote algorithm, which uses the time complexity O(n) and constant space, is primarily used to find the majority of a sequence of elements. The simple feature filtering strategy, i.e. Information Gain, is the initial step for a vote-based feature selection strategy (IG). A voting scheme is used to pick the appropriate function subset. Each function in the subset is obtained using the feature filtering algorithms' plurality votes on the feature in this strategy.

The aim of feature selection approaches is (i) to improve the generalizability of the domain by reducing feature space, and (ii) improve the efficiency of machine learning algorithms.

3.5.1 Information Gain

Information Benefit (IG) is a common feature selection tool based on Shannon's entropy, which defines the degree of value between the given information X and a random variable Y. The knowledge gain of an attribute with respect to the class name is measured using IG in machine learning. With proper normalization, this approach can operate with both nominal and numerical function values. For the attribute A, IG score can be calculated as follows.

$$(A) = (S) - \sum Si\ S\ (Si) \quad (1)$$

where H(Si) is the entropy of the ith subset created by partitioning S based on function A, and H(S) is the total entropy of the dataset.

3.5.2 Backward Feature Elimination

The data is often noisy, and it may contain features (variables) that do not always match well with the output variable. The aim of 'feature filtering' is to investigate this relationship and only use variables that have a clear association. Backward elimination starts with all of the features and eliminates the least important function at each iteration, improving the model's performance. We repeat this process until no change is found as features are removed.

3.6 Machine Learning

Machine learning is a technique for analysing data and information in order to adapt it to new data. Machine learning uses a variety of algorithms to iteratively refine data, describe data, and forecast outcomes. For the detection and classification of malware, a variety of machine learning algorithms are used. The XGB classifier is a machine learning classifier that uses gradient boosting tree models to achieve high efficiency. The XGBoost outperforms the competition on a number of challenging machine learning tasks. The XGB classifier is used to perform malware detection in this article.

3.6.1 XGBoost Classifier

XGBoost is one of the machine learning algorithm that performs well and gave high accuracy. To become extreme gradient boosting, XGBoost utilize the construction of the simple gradient boosting tree model. When compared to other gradient boosting implementations, XGBoost is generally faster. XGBoost is designed for speed and performance. Its innovation aim is to bring the computing tools for boosted tree algorithms to their limits. The algorithm takes advantage of the utility of processing time and memory. It allows the most efficient use of energy in order to train the model. It takes care of the missing values in the dataset automatically. By adding new data to the original model, we will improve it even further. It is fast when compared to the random forest.

To improve the XGBoost performance, the hyperparameters are tuned. The following XGBoost model hyperparameters must be optimized: minimum child weight, a shrinkage parameter; alpha, the L1 regularization parameter; and lambda the L2 regularization parameter; maximum depth (max_ depth), which is a parameter of tree depth level, the subsample, the learning rate, a sample ratio of training data; colsample by a tree, the ratio of the sample column when constructing each tree; the learning rate, a shrinkage parameter; alpha, the L1 regularization parameter; and lambda, the L2 regularization parameter. In XGBoost, the n estimator hyper parameter describes the number of trees to match. The model's accuracy increases if the number of epochs the algorithm runs to connect a tree until the total number of trees approaches n estimator count. n_estimator has a default value of 100.

For hyperparameter tuning, the random search technique is used. We create a grid of possible hyperparameter values using the random search method. Each iteration attempts a different random combination of hyper parameters from this grid, reports the results, and then returns the best combination of hyperparameters. Randomsearch*cv* performs a search using a collection of random parameter combinations. At the end of the search, you can use the class's attributes to see all of the data. The best score observed and the hyperparameters that obtained the best score are perhaps the most critical qualities. If you've identified the right set of hyperparameters, you can create a new model, set the values for hyperparameter, and match the model to all available data. Scikit-optimize finds optimal solutions for hyperparameter search problems in less time by using a sequential model-based optimization algorithm.

After the XGB hyperparameters are tuned, the model is trained for the classification process then performs the classification process and classifies the malware and benign files with the accuracy of 99.5%.

4. Result and Discussion

In this work, the objective of the proposed XGB with Vote based Backward Feature Elimination was to accurately detect and classify the malicious executable using the hybrid features**.** The features are selected by vote algorithm with basic feature filtering technique called Information Gain and then the optimized features are selected by the backward feature elimination technique. The selected features are used to train the model. By using the features, the classifier classifies the malware files and benign files with the highest accuracy and less computation time. The proposed model is compared to evaluate the performance with the other machine learning algorithms such as Random Forest and SVM using the performance evaluation metrics such as precision, recall and accuracy.

4.1 Performance Evaluation metrics

To evaluate the performance of the model various evaluation metrics are used. The performance evaluation metrics are accuracy, precision, and recall.

Accuracy is the measure of the correct number of prediction to total number of predictions.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \quad (2)$$

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The formula for precision is

$$\text{Precision} = \frac{TP}{TP+FP} \quad (3)$$

Recall is the percentage of individual positives that a model classified correctly. The recall formula is given below:

$$\text{Recall} = \frac{TP}{TP+FN} \quad (4)$$

Table 3- Comparison of Accuracy of Random Forest, SVM, and XGBoost

No.of.Features	Precision		
	SVM	Random Forest	XGBoost
10	63.00%	96.33%	98.83%
20	95.96%	97.54%	99.12%
30	97.98%	98.76%	99.50%

Table 3- The accuracy of different algorithms such as SVM, Random Forest and XGB are compared with various numbers of features to show the performance of the proposed method.

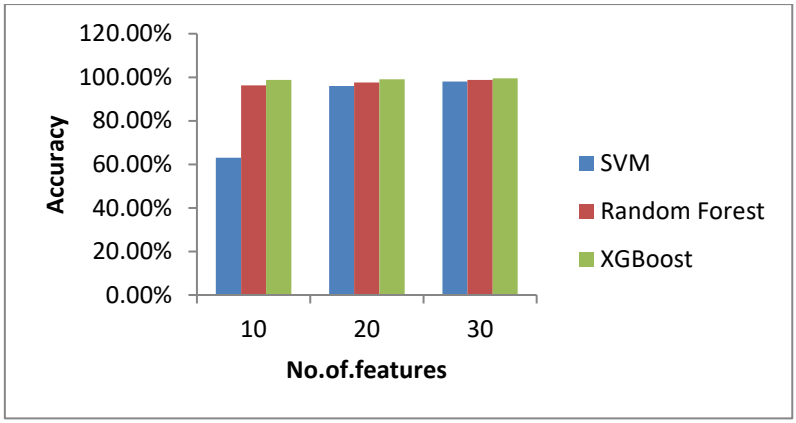

Figure 2- Comparison of Accuracy of Random Forest, SVM, and XGBoost

Figure 2- The accuracy of different algorithms such as SVM, Random Forest and XGB are compared with various numbers of features to show the performance of the proposed method.

Table 4- Comparison of Precision of Random Forest, SVM, and XGBoost

No.of.Features	Precision		
	SVM	Random Forest	XGBoost
10	0.65	0.76	0.91
20	0.72	0.82	0.95
30	0.79	0.89	0.99

Table 4- The precision of different algorithms such as SVM, Random Forest and XGB are compared with various numbers of features to show the performance of the proposed method.

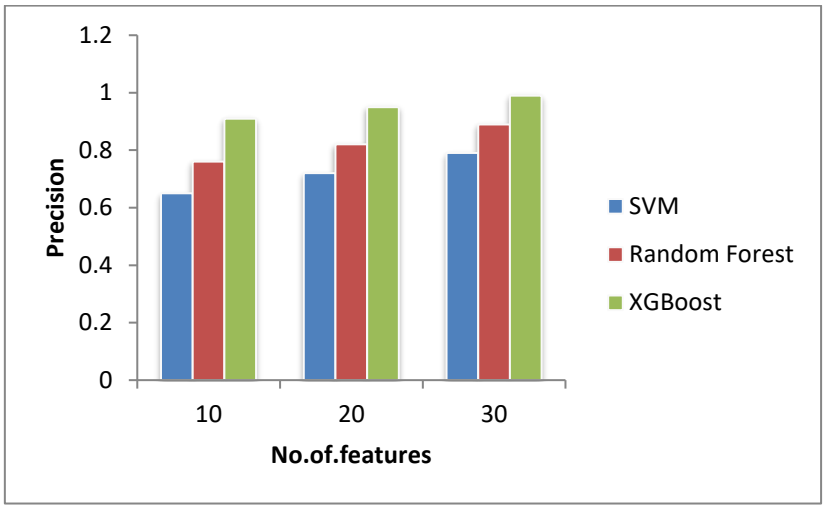

Figure 3- Comparison of Precision of Random Forest, SVM, and XGBoost

Figure 3- The precision of different algorithms such as SVM, Random Forest and XGB are compared with various numbers of features to show the performance of the proposed method.

Table 5 - Comparison of Recall of SVM, Random Forest and XGBoost

No.of.Features	Recall		
	SVM	Random Forest	XGBoost
10	0.54	0.66	0.89
20	0.65	0.79	0.93
30	0.68	0.82	0.96

Table 5- The recall of different algorithms such as SVM, Random Forest and XGB are compared with various numbers of features to show the performance of the proposed method.

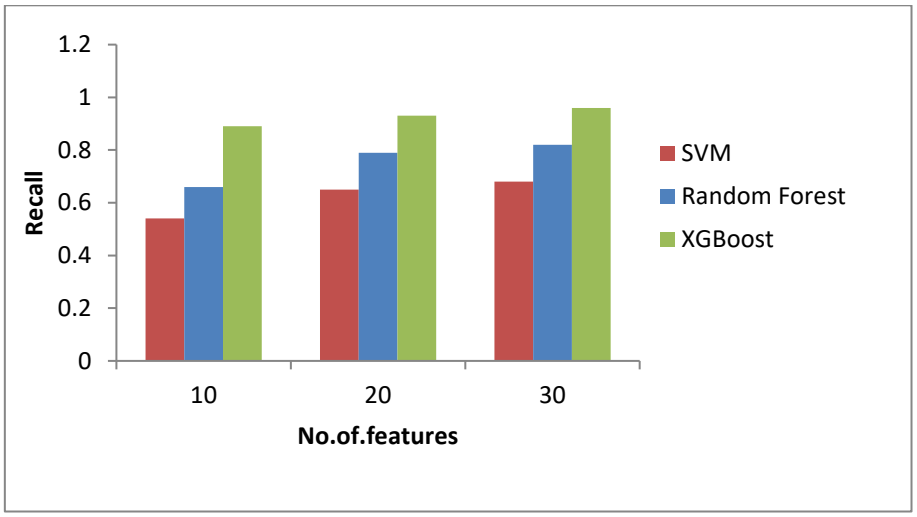

Figure 4- Comparison of Recall of Random Forest, SVM, and XGBoost

Figure 4- The recall of different algorithms such as SVM, Random Forest and XGB are compared with various numbers of features to show the performance of the proposed method.

Conclusion

This paper proposed the XGB with Vote based Backward Feature Elimination to classify the files of benign and malware. The optimal features are fed to the machine learning classifier. By using optimized features, the training time of the classifier is reduced and the accuracy is increased. The proposed system classifies the files with the highest accuracy and takes less computation time. The experimental model's efficiency is compared to that of other machine learning algorithms. The performance evaluation metrics are recall, precision and accuracy. The XGB classifier classifies the files with accuracy 99.50%, precision 0.99 and recall 0.96.

References

1. Mayank Kushwah, S. Bharathiraja, and S. Asha (2018), "Detection of malware in Windows operating system", International Journal of Pure and Applied Mathematics, Vol. 120, No. 6.
2. M. Asha Jerlin and K. Marimuthu (2018), "A New Malware Detection System Using Machine Learning Techniques for API Call Sequences", Journal of Applied Security Research, Vo.13, No.1.
3. S. L. Shiva Darshan and C. D. Jaidhar (2018), "Windows malware detection system based on LSVC recommended hybrid features", Journal of Computer Virology and Hacking Techniques.
4. SuyeonYoo, Sungjin Kim, Seungjae Kim and Brent Byunghoon Kang (2021)," AI-HydRa: Advanced hybrid approach using random forest and deep learning for malware classification", Information Sciences, Vol.546.
5. Royi Rone, Marian Radu, Corina Feuerstein, Elad Yom-Tov, and Mansour Ahmadi (2018)," Microsoft Malware Classification Challenge" DOI: 10.13140/RG.2.2.34695.91045
6. Sushant Kumar Pandey (2019)," Design and performance analysis of various feature selection methods for anomaly-based techniques in intrusion detection system", Security and Privacy, Vol.2, No.1.
7. Seoungyul Euh, Hyunjong Lee , Donghoon Kim and Doosung Hwang(2020)," Comparative Analysis of Low-Dimensional Features and Tree-Based Ensembles for Malware Detection Systems", IEEE Access, Vol.8.
8. S. Abijah Roseline and S. Geetha (2019)," An Efficient Malware Detection System using Hybrid Feature Selection Methods", International Journal of Engineering and Advanced Technology, Vol.9, No.1S3.
9. Sareen Fathima, Suzaifa, and Abdul Khader (2020)," Comparative Analysis of Malware Classification using Machine Learning Algorithms", International Journal of Engineering Research and Applications, Vol.10, No.12.
10. Dima Rabadi and Sin G Teo (2020)," Advanced Windows Methods on Malware Detection and Classification", ACSAC '20: Annual Computer Security Applications.
11. Catak FO, Yazı AF, Elezaj O, and Ahmed J. (2020)," Deep learning based Sequential model for malware analysis using Windows exe API Calls", PeerJ Computer Science.
12. S. Megira, A. R. Pangesti and F. W.Wibowo (2018)," Malware Analysis and Detection Using Reverse Engineering Technique", Journal of Physics.
13. P HarshaLatha, and R Mohanasundaram (2020)," Classification Of Malware Detection Using Machine Learning Algorithms: A Survey", International Journal Of Scientific & Technology Research, Vol. 9, No.02.
14. Xiang Huang, Li Ma, Wenyin Yang and Yong Zhong (2020)," A Method for Windows Malware Detection Based on Deep Learning", Journal of Signal Processing Systems.
15. Jinpei Yan , Yong Qi , and Qifan Rao (2018)," Detecting Malware with an Ensemble Method Based on Deep Neural Network", Security and Communication Networks.
16. Qiangjian Pan, Weiliang Tang, and Siyue Yao (2020)," The Application of LightGBM in Microsoft Malware Detection", Journal of Physics.
17. Maryam Shahini, Ramin Farhanian, and Marcus Ellis (2019)," Machine Learning to Predict the Likelihood of a Personal Computer to Be Infected with Malware", SMU Data Science Review,Vol. 2, No. 2.
18. Yongchao Zhang , Zhe Liu , and Yu Jiang (2020)," The Classification and Detection of Malware Using Soft Relevance Evaluation", IEEE Transactions On Reliability.
19. Zhiyang Fang , Junfeng Wang , Jiaxuan Geng , and Xuan Kan (2019)," Feature Selection for Malware Detection Based on Reinforcement Learning", IEEE Access, Vol.7.
20. Sergii Banin and Geir Olav Dyrkolbotn (2018)," Multinomial malware classification via low-level features", Digital Investigation, Vol.26.
21. Changhee Choi, Kyeongsik Lee, Hwaseong Lee, Ilhoon Jeong, and Hosang Yun (2018)," Malware Family Classification Based on Novel Features from Frequency Analysis", International Journal of Computer Theory and Engineering, Vol. 10, No. 4.
22. Gholamreza Farahani (2020)," Feature Selection Based on Cross-Correlation for the Intrusion Detection System", Security and Communication Networks.
23. P. S. S. Siva Krishna and P. Venkateswara Rao (2019)," An Analysis of Malware Classification Technique by using Machine Learning", International Journal of Computer Applications, Vol.181, No. 50.
24. Rajesh Kumar, and Geetha S (2020)," Malware classification using XGboost-Gradient Boosted Decision Tree", Advances in Science, Technology and Engineering Systems Journal, Vol. 5, No. 5.
25. Niranjan Agnihotri (2017)," Ransomware Classifier using Extreme Gradient Boosting", International Journal of Computer Science and Information Technologies, Vol. 9, No.2.