Malware Detection Using Machine Learning

Ajay Kumar[1], Kumar Abhishek[1(✉)], Kunjal Shah[2], Divy Patel[2], Yash Jain[2], Harsh Chheda[2], and Pranav Nerurkar[2,3]

[1] NIT Patna, Patna, India
{ajayk.phd18.cs,kumar.abhishek.cse}@nitp.ac.in
[2] Veermata Jijabai Technological Institute, Matunga, Mumbai, India
{kshah_b18,panerurkar_p17}@ce.vjti.ac.in,
{dspatel_b17,ysjain_b17,hkchheda_b17}@it.vjti.ac.in
[3] NMIMS Mumbai, Mumbai, India
pranav.n@nmims.edu

Abstract. Decision making using Machine Learning can be efficiently applied to security. Malware has become a big risk in today's times. In order to provide protection for the same, we present a machine-learning based technique for predicting Windows PE files as benign or malignant based on fifty-seven of their attributes. We have used the Brazilian Malware dataset, which had around 1,00,000 samples and 57 labels. We have made seven models, and have achieved 99.7% accuracy for the Random Forest model, which is very high when compared to other existing systems. Thus using the Random Forest model one can make a decision on whether a particular file is malware or benign.

Keywords: Security · Malware · Machine learning

1 Introduction

Decision making is an important process today in almost all domains. Cyber security is a particular field wherein we need to make decision on files whether they are malware or benign. Malware is malicious software or a program or a script which can be harmful to any instance of computing. These malicious programs are capable of performing multiple tasks, including data theft, encoding, or straight-away deleting sensitive data, modifying or hijacking basic system functionalities, and keeping track of (spying on) the actions performed by humans/human driven software on the computers [7,26]. Malware is the short form for 'malicious software', a technical word for noting some particular computer program or code which undertakes illegal tasks without the owner's permission. In the last decade, the number of newly found malware has risen exponentially. As specified earlier, malware can have disastrous effects on the system if left uncontrolled. Therefore, we should always have a capable protection from such malignant programs. Currently we can group prevalent anti-malware mechanisms into three kinds: signature, behaviour and heuristics-based techniques. However, they cannot handle malware whose definitions are not known or the latest, frequently developed post discovery of a zero-day exploit. The number and types of malware are multiplying daily, and a dynamic system is essentially required for the classification of files as "malware" or "benign". As per the statistics published by the Computer Economics 1, the fiscal damage caused by the various malware attacks has risen from 3.3 billion dollar in 1997 to 13.3 billion dollar in 2006. It is a huge jump. The definition of Year of Mega Breach must be manipulated every few years to inculcate protection to the attacks performed in that particular year [2].

The static features of malware when it is not in the running state are: overall characteristics, PE structure characteristics, binary code characteristics, and assembly code characteristics [8,21].

We now converge our discussions to a specific type of file called the PE file, which is usually executed on the Microsoft Windows Operating System. Windows Portable Executable Files (PE files) are those who stand for the main file format for executables that are binary in nature and DLLs under Windows. A PE file comprises DOS Header, PE Header, Section Headers (Section table), and several sections [7]. PE files can be classified as benign or malicious, depending on their nature and attributes [1].

We divide our paper into the following sections: Sect. 2 has a comprehensive literature survey on the existing systems, Sect. 3 describes the dataset, Sect. 4 explains in detail the proposed methodology and Sect. 5 highlights the conclusion.

We have built seven different models and concluded with saying that Random Forest Model has provided the highest accuracy of all. Our model has improved accuracy over several existing systems.

2 Literature Survey

Classifying malware has been of great interest to researchers throughout the world, owing to the blast in demand for cybersecurity. Cybersecurity issues have become national issues [9], and not only machine learning but also blockchain [22], IoT [13], and cloud technologies involving heterogeneous client networks [5,14] have been used for fighting against them. Android [30] requires protection from Malicious PE files can cause data leakage [25] and other dangers to the security level.

Ren and Chen [19] have devised a new graphical analysis technique for researching malware resemblance. This technique converts malignant PE files into local entropy images for observing internal features of malware and then normalizes local entropy images into entropy pixel images for classifying malware. Zhang and Luo [30] have proposed a behaviour based analysis technique based on the method-level correlation relationship of application's abstracted API calls.

Mahmood Yousefi-Azar has shown work quite similar to what we have done. He has put forward a technique that he named ‘Malytics’ which consists of three parts: extracting features, measuring similarity, and classifying everything. The three parts are presented by a neural network [15] with two hidden layers and one single output layer. The author could achieve an accuracy of 99.45% [28]. Rushabh Vyas has worked on four different types of PE files and has extracted 28 features, packing, imported DLLs and functions from them. He could achieve 98.7% detection rates using machine learning [27]. Erdogan Dogdu has presented a paper wherein a shallow deep learning-based feature extraction method named as word2vec is used to show any given malware based on its opcodes. Classification is done using gradient boost. They have used k-fold cross-validation for validating the model performance without compromising with a validation split. He has successfully achieved an accuracy of ninety-six percent (96%) [3].

Muhammad Ijaz has used two techniques: static and dynamic to extract the features of files. Under static mechanism he could achieve an accuracy of 99.36% (PE files) and under dynamic mechanism he could achieve an accuracy of 94.64% [17]. In static analysis, the executable file is analyzed on structure bases without executing it in controlled environment. In dynamic analysis, malware behavior is analyzed in dynamic controlled environment.

Beyond all this, in the latest technology domain, data fusion models have also been prepared for malware detection [12].

3 Dataset

We have used the Brazilian Malware Dataset [4] for the project. It contained about 1,21,000 rows and 57 columns corresponding to 57 different attributes of PE files. The classes that it had were malicious and benign. Here is a description of the data items given in Table 1. We have shown the column name(which is a characteristic of the PE file), the type of data (numeric or categorical), the number of distinct values in each, and the correlation of each feature with the target.

Table 1. Exploration of data

Column name	Type	Distinct count	Correlation with target
AddressOfEntryPoint	Columnar	23,110	0.072
BaseOfCode	Columnar	385	0.04
BaseOfData	Columnar	1,106	0.101
Characteristics	Columnar	104	0.13
DIICharacteristics	Columnar	74	0.107
ExportNb	Numeric	670	0.4
FileAlignment	Numeric	9	0.103
ImportsNb	Numeric	954	0.27
ImportsNbDLL	Numeric	48	0.252

(*continued*)

Table 1. (*continued*)

Column name	Type	Distinct count	Correlation with target
ImportsNbOrdinal	Numeric	337	0.19
Legitimate Target	Columnar	2	—
LoadConfigurationSize	Numeric	39	0173
LoaderFlags	Columnar	15	0.001
Machine	Columnar	3	0.078
MajorlmageVersion	Columnar	38	0.079
MajorLinkerVersion	Columnar	41	0.099
MajorOperatingSystemVersion	Columnar	12	0.103
MajorSubsystemVersion	Columnar	6	0.089
MinorlmageVersion	Columnar	70	0.108
MinorLinkerVersion	Columnar	62	0.045
MinorOperatingSystemVersion	Columnar	12	0.043
MinorSubsystemVersion	Columnar	10	0.049
NumberOfRvaAndSizes	Numeric	23	0.004
ResourcesMaxEntropy	Numeric	23,004	0.329
ResourcesMaxSize	Numeric	50	0.313
ResourcesMeanEntropy	Numeric	42,745	0.17
ResourcesMeanSize	Numeric	16,013	0.253
ResourcesMinEntropy	Numeric	17,929	0.361
ResourcesMinSize	Numeric	1,011	0.422
ResourcesNb	Numeric	496	0.396
SectionAlignment	Numeric	12	0.112
SectionMaxRawsize	Numeric	4796	0.186
SectionMaxVirtualsize	Numeric	29,123	0.268
SectionsMaxEntropy	Numeric	49,062	0.35
SectionsMeanEntropy	Numeric	58,807	0.222
SectionsMeanRawsize	Numeric	9,233	0.168
SectionsMeanVirtualsize	Numeric	36,811	0.251
SectionsMinEntropy	Numeric	25,505	0.343
SectionsMinRawsize	Numeric	694	0.376
SectionsMinVirtualsize	Numeric	6,515	0.27
SectionsNb	Numeric	28	0.259
SizeOfCode	Numeric	3,809	0.334
SizeOfHeaders	Numeric	30	0 092
SizeOfHeapCommit	Numeric	21	0.1
SizeOfHeapReserve	Numeric	30	0.101
SizeOflmage	Numeric	2,312	0.289
SizeOflnitializedData	Numeric	3,217	0.288
SizeOfOptionalHeader	Columnar	5	0.078
SizeOfStackCommit	Numeric	40	0 099
SizeOfStackReserve	Numeric	40	0.46
Subsystem	Columnar	4	0.096
VersionlnformationSize	Numeric	20	0.473

4 Proposed Methodology

The dataset is first preprocessed, and then important features were selected, the model was trained on the selected features. The entire stages of the process can be shown in the block diagram Fig. 1. Each section can be elegantly described as:

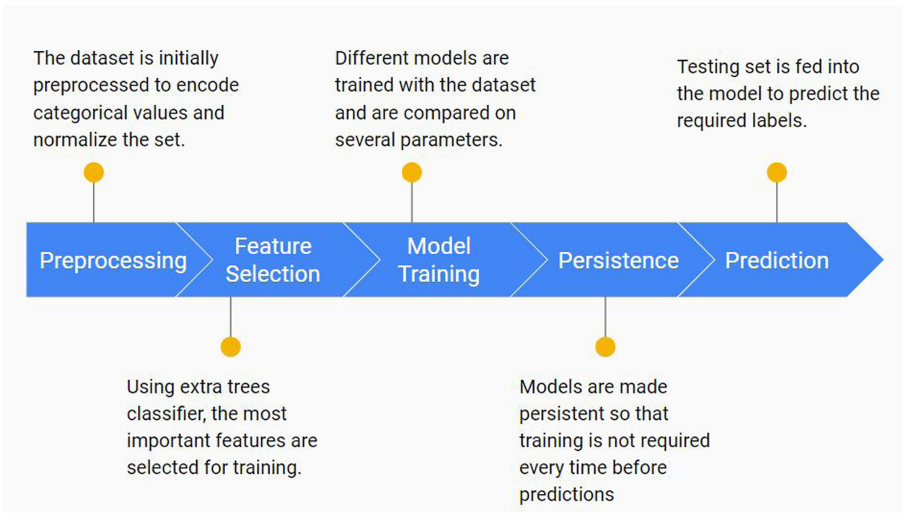

The dataset is initially preprocessed to encode categorical values and normalize the set.
Different models are trained with the dataset and are compared on several parameters.
Testing set is fed into the model to predict the required labels.
Preprocessing
Feature Selection
Model Training
Persistence
Prediction
Using extra trees classifier, the most important features are selected for training.
Models are made persistent so that training is not required every time before predictions

Fig. 1. Overall block diagram

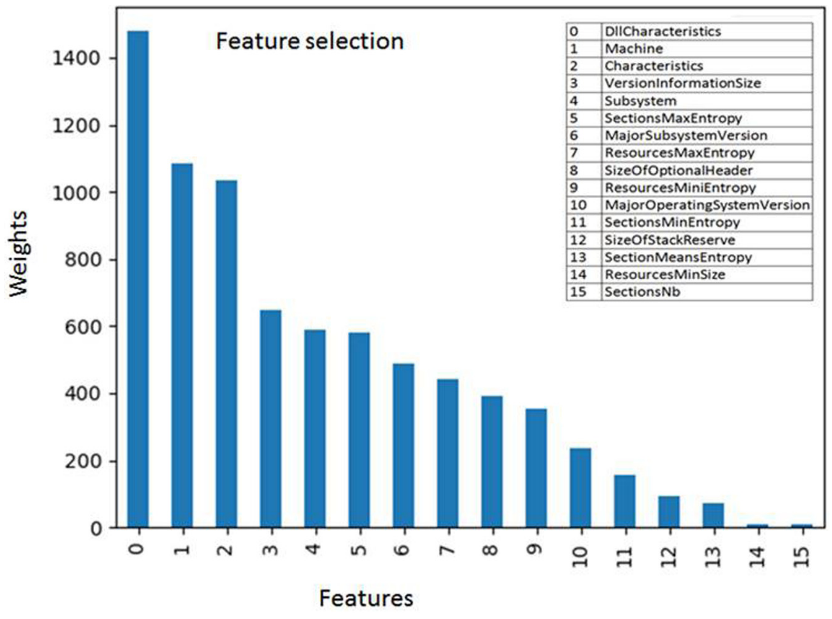
4.1 Preprocessing

From the given PE file, about 56 features were extracted, for example, md5, ImageSize, etc. Some features were categorical, so we applied label encoding [10] followed by one-hot encoding to process those features. Some features were human-readable features like Name, so we manually discarded such features. The two tasks done were:

1. Label Encoding:
   It implies converting the labels to number format for changing it over to the machine-readable form. Machine learning algorithms [6] and calculations would then be able to choose in a superior manner on how those labels must be worked. It is a significant pre-handling step for the organized dataset in supervised learning [11].
2. One-Hot Encoding: It refers to splitting the column containing categorical data to several columns depending on the number of types in that column. Each column has "0" if it is not placed and "1" if it is placed [20] (Fig. 2).

4.2 Feature Section and Model Training

On the pre-processed set of features we applied ExtraTreesClassifier [24] from scikit learn [16] for feature selection. So we selected around 15 features from the set of features and trained our model using those features [29]. We trained several models on the dataset set; we used models provided by scikit learn with slight tweaks. The following Table 2 depicts the accuracies that we could attain for various models:



0	DllCharacteristics
1	Machine
2	Characteristics
3	VersionInformationSize
4	Subsystem
5	SectionsMaxEntropy
6	MajorSubsystemVersion
7	ResourcesMaxEntropy
8	SizeOfOptionalHeader
9	ResourcesMiniEntropy
10	MajorOperatingSystemVersion
11	SectionsMinEntropy
12	SizeOfStackReserve
13	SectionMeansEntropy
14	ResourcesMinSize
15	SectionsNb

Fig. 2. Weights of top 15 features

Table 2. Accuracies

Key	Model	Accuracy
6	Decision Tree	99.7
1	Random Forest	99.7
4	Gradient Boost	98.48
2	SVM	96.9
0	Logistic Regression	96.8
5	XGBoost	96.7
3	AdaBoost	94.3

We also calculated precision, recall, training times and confusion matrices for the following figures. For the key please refer to Table 2 again (Figs. 3, 4, 5, 6).

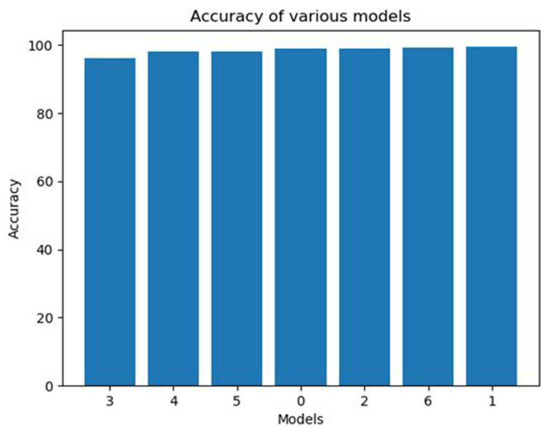
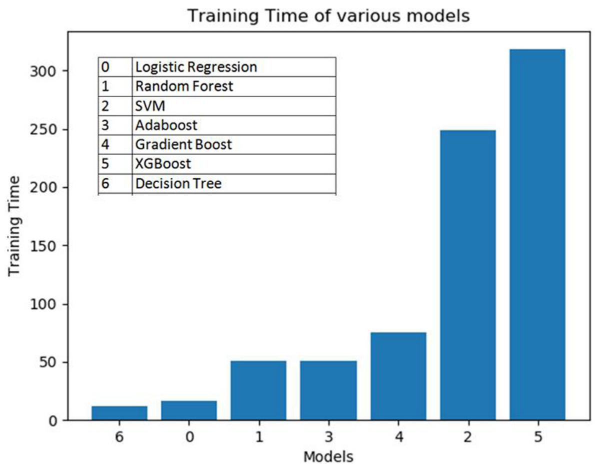


Fig. 3. Accuracies



0	Logistic Regression
1	Random Forest
2	SVM
3	Adaboost
4	Gradient Boost
5	XGBoost
6	Decision Tree

Fig. 4. Training times

In Fig. 7 we show various graphs for the model of Random Forest, for which we have received the highest accuracy. We show how the number of trees affects the training time as well as model accuracy.

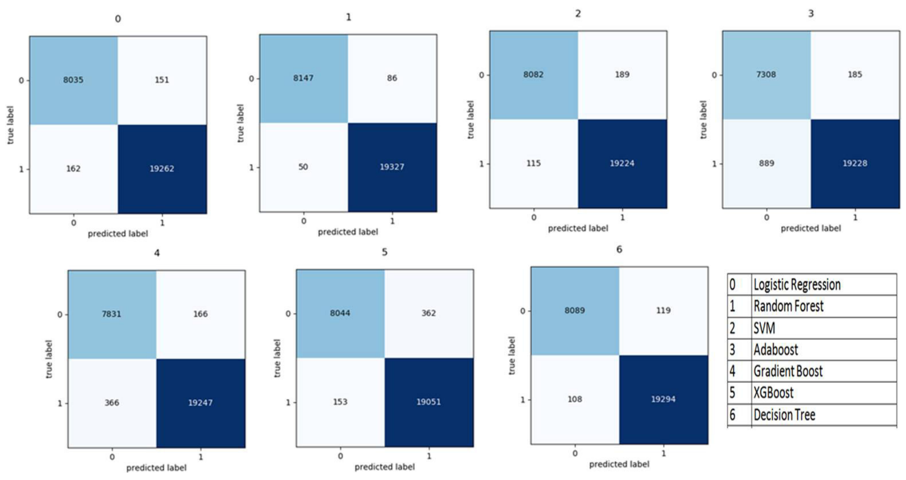


Fig. 5. Confusion matrices

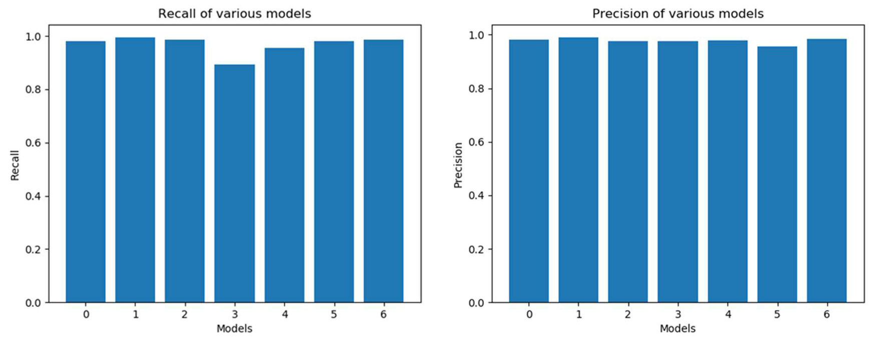


Fig. 6. Precision and Recall

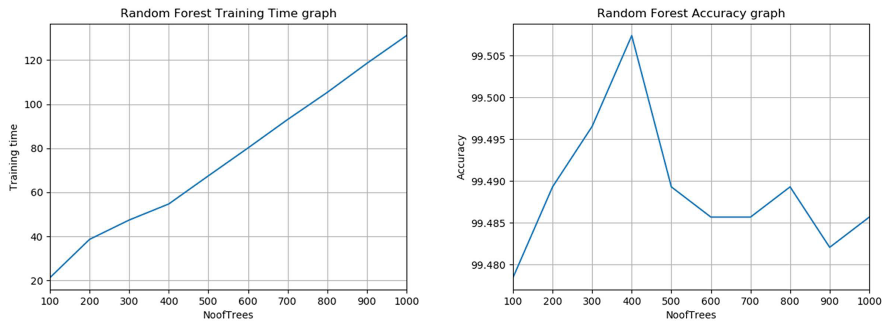


Fig. 7. Graphs for Random Forest

4.3 Persistence and Prediction

The models are made persistent so that training is not required every time before predictions. Persistence is very important for any machine learning model [18,23]. Finally, we feed in the testing set to predict the required labels. The results are compared with existing systems: 99.45% by Yousefi-Azar [28], 98.7% by Vyas [27] and 96% by Meng [3]. We show an accuracy of 99.7%.

5 Conclusion

From our analysis, we can conclude that the RandomForest model can give us the highest possible accuracy on the given dataset. Such machine learning methods can be used efficiently in cybersecurity in order to provide protection against malicious software around. In recent times viruses have become a very significant issue. Traditional protection methods (similar to those signature-based techniques) used by anti-virus will not be able to handle the latest malware issues. In our article, we have looked at malware analysis as an artificial intelligence-based problem. We have used state-of-the-art techniques in coding these models like cross-validation and feature selection. We have used feature selection methods. In the end, we could converge to the conclusion that the RandomForest model can give the highest possible accuracy. This model shows improvement over other systems.

References

1. Abdessadki, I., Lazaar, S.: New classification based model for malicious PE files detection. Int. J. Comput. Netw. Inform. Secur. **11**(6) (2019)
2. Belaoued, M., Mazouzi, S.: A real-time PE-malware detection system based on CHI-square test and PE-file features. In: Amine, A., Bellatreche, L., Elberrichi, Z., Neuhold, E.J., Wrembel, R. (eds.) CIIA 2015. IAICT, vol. 456, pp. 416–425. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19578-0_34
3. Cakir, B., Dogdu, E.: Malware classification using deep learning methods. In: Proceedings of the ACMSE 2018 Conference, pp. 1–5 (2018)
4. Ceschin, F., Pinage, F., Castilho, M., Menotti, D., Oliveira, L.S., Gregio, A.: The need for speed: an analysis of Brazilian malware classifers. IEEE Secur. Priv. **16**(6), 31–41 (2018)
5. Dey, S., Ye, Q., Sampalli, S.: A machine learning based intrusion detection scheme for data fusion in mobile clouds involving heterogeneous client networks. Inform. Fusion **49**, 205–215 (2019)
6. Gaurav, D., Tiwari, S.M., Goyal, A., Gandhi, N., Abraham, A.: Machine intelligence-based algorithms for spam filtering on document labeling. Soft Comput. **24**(13), 9625–9638 (2020)
7. Gheorghe, L., et al.: Smart malware detection on android. Secur. Commun. Netw. **8**(18), 4254–4272 (2015)
8. Han, W., Xue, J., Wang, Y., Liu, Z., Kong, Z.: MalInsight: a systematic profiling based malware detection framework. J. Netw. Comput. Appl. **125**, 236–250 (2019)
9. Kemmerer, R.A.: Cybersecurity. In: Proceedings of the 25th International Conference on Software Engineering, pp. 705–715. IEEE (2003)

10. Kumar, A., et al.: Multilabel classification of remote sensed satellite imagery. Trans. Emerg. Telecommun. Technol. e3988 (2020)
11. Lazzeri, F., Bruno, G., Nijhof, J., Giorgetti, A., Castoldi, P.: Efficient label encoding in segment-routing enabled optical networks. In: International Conference on Optical Network Design and Modeling (ONDM), pp. 34–38. IEEE (2015)
12. Meng, T., Jing, X., Yan, Z., Pedrycz, W.: A survey on machine learning for data fusion. Inform. Fusion **57**, 115–129 (2020)
13. Mishra, S., Sagban, R., Yakoob, A., Gandhi, N.: Swarm intelligence in anomaly detection systems: an overview. Int. J. Comput. Appl. 1–10 (2018)
14. Nerurkar, P., Chandane, M., Bhirud, S.: Survey of network embedding techniques for social networks. Turkish J. Electr. Eng. Comput. Sci. **27**(6), 4768–4782 (2019)
15. Nerurkar, P.A., Chandane, M., Bhirud, S.: Exploring convolutional auto-encoders for representation learning on networks. Comput. Sci. **20**(3) (2019)
16. Pedregosa, F., et al.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
17. Raghuraman, C., Suresh, S., Shivshankar, S., Chapaneri, R.: Static and dynamic malware analysis using machine learning. In: Luhach, A.K., Kosa, J.A., Poonia, R.C., Gao, X.-Z., Singh, D. (eds.) First International Conference on Sustainable Technologies for Computational Intelligence. AISC, vol. 1045, pp. 793–806. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-0029-9_62
18. Rahul, M., Kohli, N., Agarwal, R., Mishra, S.: Facial expression recognition using geometric features and modified hidden Markov model. Int. J. Grid Utility Comput. **10**(5), 488–496 (2019)
19. Ren, Z., Chen, G.: EntropyVis: malware classification. In: 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), pp. 1–6. IEEE (2017)
20. Rodríguez, P., Bautista, M.A., Gonzalez, J., Escalera, S.: Beyond one-hot encoding: lower dimensional target embedding. Image Vision Comput. **75**, 21–31 (2018)
21. Rudd, E.M., Rozsa, A., Günther, M., Boult, T.E.: A survey of stealth malware attacks, mitigation measures, and steps toward autonomous open world solutions. IEEE Commun. Surv. Tutor. **19**(2), 1145–1172 (2016)
22. Sengupta, J., Ruj, S., Bit, S.D.: A comprehensive survey on attacks, security issues and blockchain solutions for iot and iiot. J. Netw. Comput. Appl. **149**, 102481 (2020)
23. Shah, K., Bhandare, D., Bhirud, S.: Face recognition-based automated attendance system. In: Gupta, D., Khanna, A., Bhattacharyya, S., Hassanien, A.E., Anand, S., Jaiswal, A. (eds.) International Conference on Innovative Computing and Communications. AISC, vol. 1165, pp. 945–952. Springer, Singapore (2021). https://doi.org/10.1007/978-981-15-5113-0_79
24. Sharaff, A., Gupta, H.: Extra-tree classifier with metaheuristics approach for email classification. In: Bhatia, S.K., Tiwari, S., Mishra, K.K., Trivedi, M.C. (eds.) Advances in Computer Communication and Computational Sciences. AISC, vol. 924, pp. 189–197. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-6861-5_17
25. Shu, X., Yao, D., Bertino, E.: Privacy-preserving detection of sensitive data exposure. IEEE Trans. Inf. Forensics Secur. **10**(5), 1092–1103 (2015)
26. Udayakumar, N., Saglani, V.J., Cupta, A.V., Subbulakshmi, T.: Malware classification using machine learning algorithms. In: 2nd International Conference on Trends in Electronics and Informatics (ICOEI), pp. 1–9. IEEE (2018)

27. Vyas, R., Luo, X., McFarland, N., Justice, C.: Investigation of malicious portable executable file detection on the network using supervised learning techniques. In: IFIP/IEEE Symposium on Integrated Network and Service Management (IM), pp. 941–946. IEEE (2017)
28. Yousefi-Azar, M., Hamey, L.G., Varadharajan, V., Chen, S.: Malytics: a malware detection scheme. IEEE Access **6**, 49418–49431 (2018)
29. Zheng, W., Zhu, X., Wen, G., Zhu, Y., Yu, H., Gan, J.: Unsupervised feature selection by self-paced learning regularization. Pattern Recogn. Lett. **132**, 4–11 (2020)
30. Zhou, Y., Jiang, X.: Dissecting android malware: characterization and evolution. In: IEEE Symposium on Security and Privacy, pp. 95–109. IEEE (2012)