International Journal of Electrical and Electronics Engineers
Volume 13, Issue No. 02, July-Dec 2021
ISSN (O) 2321-2055
ISSN (P) 2321 -2045

Machine Learning Based Malware Classification & Detection

Abhishek Kukreja[1], Priyank Ahuja[2], Prashant Singh[3], Dhyanendra Jain[4]

[1,2,3,4] *Department of Information Technology*
[1,2,3,4] *Dr. Akhilesh Das Gupta Institute of Technology & Management, New Delhi*

abhishekkukreja863@gmail.com[1] , priyankahuja02@gmail.com[2], prashant.ert@gmail.com[3], dhyanendra.jain@gmail.com[4]

ABSTRACT

With the rise of malware in the current times it becomes easier for malware to disguise itself, which in turn makes it difficult for systems to recognize and eliminate them. Malware can be different types and can also use different techniques to target a system. Different malware types have different functions and can harm our systems in a lot of different ways which include encrypting our data files, stealing our private data or can be used to spy on us. In modern times Machine learning plays an important role in detecting and eliminating this malware from our system.

Keywords *– Malware, feature engineering, Machine Learning, classification*

Introduction

Malware can be really harmful and is spreading at an extremely fast rate. To fight this problem of malware detection we have used Machine Learning models on a huge dataset of 250 GB provided by Microsoft to classify and detect 9 classes of malware (Lollipop, Kelihos_ver3,Vundo,Simda,Tracur,Kelihos_ver1,Obfuscator.ACY, Gatak)

The dataset consists of two types of files, .asm files and .byte files. Our aim was to extract features from both .asm and .byte files using various feature engineering methods to create effective classifiers using basic multiple machine learning techniques such as K-Nearest Neighbors (KNN), Logistic Regression, Random Forest and XGBoost to find the best and most effective way of detecting malware with the best accuracy.

Techniques & Algorithms Used

We are dealing with a classification problem of the Malwares based on different attributes of .asm files and .byte file's dataset. We are using various Machine Learning algorithm for the process.

1. Random Model
2. Logistic Regression
3. KNN
4. Random Forest
5. XG Boost

International Journal of Electrical and Electronics Engineers
Volume 13, Issue No. 02, July-Dec 2021
ISSN (O) 2321-2055
ISSN (P) 2321 -2045

A. Random Model

Initially we will create a random model in which all the malware will be classified data points in any category. This is done in order to find the baseline. On this baseline other models will be compared to. Any model which performs worse than the random model will not be considered. Using any model less efficient than the random model is a waste of time as selecting data points randomly is a safer bet than using that model [6].

B. Logistic regression

The logistic regression model is used to find categorical predictors. The results of this model are predicted by the probability of occurrence of an event. Logistic regression model uses a logit function in order to categorically divide data points into the correct category.

C. KNN

KNN is also called the K Nearest Neighbor model. It is used as a classification predictor. K is the value of the nearest neighbors. In the KNN model the category of a data point is decided by the category of the nearest data point it resembles, according to the Euclidean distance. The data points are more likely to be like the data points which it is surrounded by as it shows a similar trend.

D. Random Forest

Random forest is an ensemble model. An ensemble model is where we run several models to get the result. In Random Forest we combine the result of all the decision trees and then decide the category of the data point. Category of the data point is chosen which has the most votes in all the decision trees.

E. XG Boost

XG Boost is a highly efficient library which is based on a distributed gradient boosting framework. The flexibility and efficiency make it useful. It can run multiple trees parallelly to boost the outcome and can provide extraordinary results in multiple problems.

Related Work

Many real-time applications employ the above-mentioned machine learning algorithms and models. The following are some of these applications:

A. K Nearest Neighbors

KNN may outperform more sophisticated classifiers despite its simplicity, and it is employed in a range of applications such as economic forecasting, data compression, genetics, and so on. For example, in 2006 research on functional genomics, KNN was used to assign genes based on their expression profiles.

B. Logistic Regression

A statistical strategy for predicting binary classes is logistic regression. It can calculate the likelihood of an event occurring and thus, has applications in Image Segmentation and Categorization, Cancer Detection, Geographic Image Processing, etc.

International Journal of Electrical and Electronics Engineers
Volume 13, Issue No. 02, July-Dec 2021
ISSN (O) 2321-2055
ISSN (P) 2321 -2045

C. Random Forest

It is a classification, regression, and other task-solving method based on the construction of a large number of decision trees. Random forest is a machine learning algorithm that can be used for medical diagnosis, stock market analysis, and e-commerce, among other things.

D. XGBoost

It's a gradient boosting-based ensemble learning system based on decision trees. It has had many real-world applications such as being used by CERN to classify signals from the LHC. It was also used in hourly prediction of PM 2.5 particles in China.

Methodology

We'd be working with a dataset that contains nine different types of malwares that we'd have to categorize for each data point. Since there are 9 classes, we could map this to a multi class classification problem in Machine Learning.

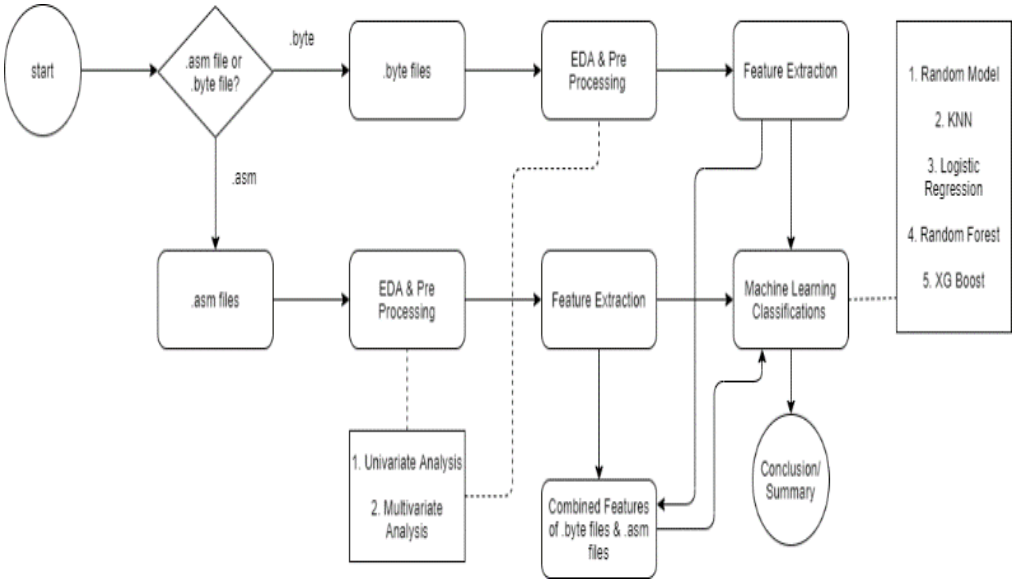

Fig 1. Workflow Diagram

In general, the first stage is data pre-processing which includes data loading, data analysis, exploratory data analysis and vectorization as shown in Fig 1.

In exploratory data analysis the first thing that we do is separate byte files and asm files into different folders to maintain hierarchy. To understand whether the problem belongs to balanced or imbalanced data we have plotted histograms, box plots, T-sne. Through these plots we can comprehend whether our data is an imbalanced data set. Following this we would do feature engineering to know if there's any useful information from the dataset.

International Journal of Electrical and Electronics Engineers
Volume 13, Issue No. 02, July-Dec 2021
ISSN (O) 2321-2055
ISSN (P) 2321 -2045

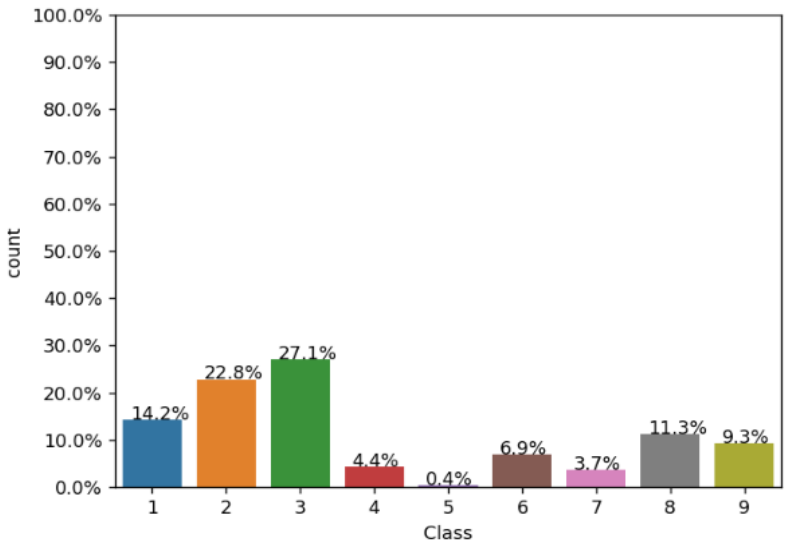

Fig 2. Distribution of Malware in the whole dataset

In Fig 2, feature engineering is done based on the size of files and to configure the size of each file, then using that file size we would conclude whether that would be useful or not.

Boxplots help in distinguishing some classes as well. Now we will convert all the hexadecimal files into text data by using Uni-Grams and we would be implementing the Uni Gram with our own bag of words. There is another approach we could convert into text data with "Count Vectorizer" but the problem is in this use-case the data is not available in main memory. This is because storing very huge (GB's) data in main memory is not a good idea here. After that we will do simple column normalization. Now for each file we have file size and bag of words as features.

After that we will perform feature extraction on byte files by converting them into text files by using a technique like Bag of words. Using T-SNE, we saw that our features are somewhat useful because they have nice grouping [2].

We divided our data into train (64%), test (20%) and cross-validation (16%) randomly. Let's check how the distribution classes are done in train, test and cross-validation. Modeling is done on both byte and .asm files, while modeling we use random models like random models, logistic regression, KNN, Random forests and XGBoost and then compare the log-loss of each and then decide whether it's a good model since log-loss is like threshold. In general, the multi log-loss has minimum value '0' and maximum value is 'Infinity'.

Performance Metrics

After doing the standard feature extraction, model selection, and implementation, and receiving a probability output, the next critical step is to determine the model's effectiveness using test datasets. Models can be evaluated using a variety of performance metrics.

International Journal of Electrical and Electronics Engineers
Volume 13, Issue No. 02, July-Dec 2021
ISSN (O) 2321-2055
ISSN (P) 2321 -2045

A. Log loss

The performance of a classification model with a prediction input of a probability between 0 and 1 is measured by logarithmic loss (log loss). As the anticipated probability diverges from the actual label, log loss grows. The purpose of models is to reduce this value as much as possible. As a result, a lower log loss is preferable, with a log loss of 0 indicating a flawless model.

B. Confusion Matrix

Multiclass classification is a method of examining the performance of a classification issue where the output can be of two or more forms of classes. A confusion matrix is a table having two dimensions: "Actual" and "Predicted," as well as "True Positives (TP)," "True Negatives (TN)," "False Positives (FP)," and "False Negatives (FN)" on both dimensions. We have used three types of performance metrics in order to evaluate our models that are classification accuracy, precision and recall as shown in Fig 3.

Confusion Matrix		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Fig 3. Confusion Matrix

The most frequent performance metric for multi-class classification algorithms is classification accuracy. It's the number of right guesses divided by the total number of predictions. To compute the accuracy of our classification model, we may use the accuracy score function in the sklearn.metrics library. Precision is employed in document retrieval and is defined as the number of right documents returned by our machine learning model, whereas recall is defined as the number of positives returned by our machine learning model.

Implementation

There are various types of malwares in the dataset. We must classify the dataset using the nine classes that we described earlier.

A. Feature Extraction

The given dataset consisted of 2 file types namely .asm and .byte. The .byte files were evaluated for their file size and we observed that class 2 and class 5 are well separated when compared to others. Also, the file sizes are different for each class as shown in the Fig 4 [4].

International Journal of Electrical and Electronics Engineers
Volume 13, Issue No. 02, July-Dec 2021
ISSN (O) 2321-2055
ISSN (P) 2321 -2045

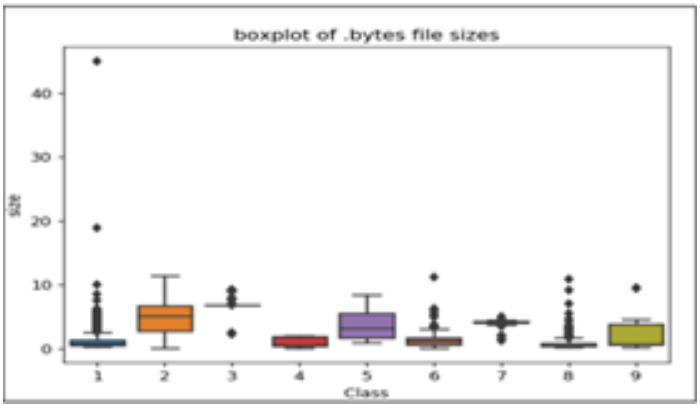

Fig 4. Boxplot of .byte file sizes

Now, all the hexadecimal values present in .byte files are converted into text data by using Unigrams. This converts the .byte files into a bag of words so that we can calculate frequency distribution of different hexadecimal values [7]. Thus, each .byte file has 257 features which include 256 hexadecimals and file size.

The evaluation on .asm files led us to select 52 features for their classification based on many references. These 52 features are different opcodes used in assembly language most frequently [3].

B. Random Model

Random Model is a model created by us which classifies the files randomly into any one of the 9 malware classes. This model was created to calculate the worst-case value of log loss that can be possible if we classify malware randomly. The log Loss value given by this model is taken as a threshold for other models and it makes it easier to compare the efficiency of other models [6].

C. K Nearest Neighbors

The k-nearest neighbors (KNN) technique is a straightforward supervised machine learning algorithm for solving classification issues. Here, we built a KNN model to classify .byte files as well as .asm files separately. The models were used to classify the given malware into their classes. The model made errors were not very accurate [5].

D. Logistic Regression

The logistic sigmoid function produces a probability value using the logistic regression algorithm, which is commonly used to handle categorical data. Our LR model classifies files into classes 1 to 9. This model compared to KNN performs worse.

E. Random Forest

Learning algorithms based on trees are considered to be one of the most popular and effective supervised learning methods. To produce a more reliable and accurate forecast, a random forest algorithm constructs numerous decision trees and blends them together. The random forest classifier here gives the best results compared to KNN and LR models

F. XGBoost

It is based on ensemble learning and stands for extreme gradient boosting. The XGBoost model also implements decision trees and works best to classify the malware files producing the lowest error from all used models..

International Journal of Electrical and Electronics Engineers
Volume 13, Issue No. 02, July-Dec 2021
ISSN (O) 2321-2055
ISSN (P) 2321 -2045

Result Analysis

These are the benchmarks which we got after applying various feature extraction techniques and using different machine learning models.

A. .byte Files

Models	LOG LOSS	Misclassification Error
Random Model	2.45	88%
K Nearest Neighbors	0.24	4.50%
Logistic Regression	0.528	12.32%
Random Forest	0.085	2.02%
XGBoost	0.070	1.24

Table I. Result of .byte files

B. .asm files

Models	LOG LOSS	Misclassification Error
K Nearest Neighbors	0.089	2.02%
Logistic Regression	0.415	9.16%
Random Forest	0.057	1.15%
XGBoost	0.048	0.87%

Table II. Result of .asm files

C. Merged Features of both .asm files & .byte files

Models	LOG LOSS	Misclassification Error
Random Forest	0.040	<0.87%
XGBoost	0.031	<0.87%

Table III. Result of Merged Features of both .asm files & .byte files

CONCLUSION

The classification of file types is a fundamental machine learning topic that has applications in a variety of products. We've split down the file type classification workflow into many parts in this report. We've recommended a tailored method for each stage based on the features of our specific dataset. In particular, using the feature extraction by using bags of words for .byte and .asm files, we suggest a model type by choosing them on the basis of their performance metrics scores [1].

First, we used all four models on .byte and . asm files and got maximum accuracy by using Random Forest (Log Loss = 0.057) and XG Boost (Log Loss = 0.048), so we only performed Random Forest and XGBoost on the merged features of both .byte and .asm files which further reduced the log loss values to for Random Forest (Log Loss = 0.040 & XG Boost (Log Los 0.031)

I.

REFERENCES

[1] Ahmadi, Mansour & Ulyanov, Dmitry & Semenov, Stanislav & Trofimov, Mikhail & Giacinto, Giorgio. Novel Feature Extraction, Selection and Fusion for Effective Malware Family Classification. (2016). 10.1145/2857705.2857713.

[2] Nataraj, Lakshmanan & Karthikeyan, Shanmugavadivel & Jacob, Grégoire & Manjunath, B. Malware Images: Visualization and Automatic Classification. (2011). 10.1145/2016904.2016908.

[3] Masud, Mehedy & Khan, Latifur & Thuraisingham, Bhavani. A scalable multi-level feature extraction technique to detect malicious executables. Information Systems Frontiers - ISF. (2008). 10. 33-45. 10.1007/s10796-007-9054-3.

[4] Bailey, Michael & Oberheide, Jon & Andersen, Jon & Mao, Zhuoqing & Jahanian, Farnam & Nazario, Jose. Automated Classification and Analysis of Internet Malware. 12th International Symposium on Recent Advances in Intrusion Detection. (2007). 4637. 178-197. 10.1007/978-3-540-74320-0_10.

[5] Bagga, Naman & Di Troia, Fabio & Stamp, Mark. On the Effectiveness of Generic Malware Models. (2018). 442-450. 10.5220/0006921504420450.

[6] Breiman, Leo. Machine Learning, Volume 45, Number 1 - SpringerLink. Machine Learning. (2001). 45. 5-32. 10.1023/A:1010933404324.

[7] Reddy, D & Pujari, Arun K. N-gram analysis for computer virus detection. Journal in Computer Virology. (2006). 2. 231-239. 10.1007/s11416-006-0027-8.