The Application of LightGBM in Microsoft Malware Detection

Qianguan Pan1, Weiliang Tang2, Siyue Yao3*
1School of Science, The Hebei University of Technology, Tianjin, China
2Faculty of Engineering, The Chinese University of Hong Kong, Hong Kong, China
3School of Computer Science, The China University of Geoscience, Wuhan, China
* Corresponding author: yaosiyue1@gmail.com

Abstract—The development of new technologies has caused computers one of the most popular electronic products. However, there is always a number of people who intend to take advantages of others through attacking others' computers. To avoid property damage as much as possible, a precise and efficient detection is essential. This work uses the dataset which was generated by combining heartbeat and threat reports collected by Microsoft’s endpoint protection solution to find out an effective solution. Since the dataset is large and has many categorical variables, reduction of memory and label encoding are used in data cleaning. Further, to handle the dimension problem and improve training efficiency, Chi-square testing is applied, and the top 42 fields are selected. Then, three algorithms (Logistic Regression, KNN and LightGBM) are chosen to build models and results are got respectively. The results show that LightGBM model achieves the best accuracy that AUC reaches 0.720687, and it is the most time-saving way. To the end, according to the feature importance from LightGBM algorithm, this work pick top-three important variables to analyze the underlying causes in the malware attack. One of the results reveals that the computer which has anti-virus software with bugs or pitfalls will suffer more attacks.

1. INTRODUCTION
Nowadays, electronic products have become necessities for most people. It also attracts hackers’ attention, which resulted in a considerable increase in the number of the malware. According to a report from Kaspersky Lab in 2017, there was an increasing trend in the types of the malware, since at least 360,000 new malicious files were detected every day in 2017 [1]. Thus, finding out which factors are related to the infection of computers is significant. This work figured out the relationship between different kinds of machines and the probabilities of being attacked, which provides necessary instructions to protect computers.

Generally, the approaches of malware prediction mainly include traditional static analysis method and some other methods, one of which use machine learning and data mining to predicting [2]. As Matthew g. Schultz et al. Studied in 2001 [3], static analysis of binaries is used to extract attributes, then training the new subset of the data.

Similarly, in this paper, the dataset will be cleaned, reduced dimension, and then model trained. It is worth mentioning that during the data cleaning process, this work figured out a method to deal with the version number that can better represent the consequential relationships among them. Besides, this
work used three different models to check whether computers will be attacked by malware, and two of these methods are used as the standard to evaluate the results.

2. Experimental Section

2.1 Dataset
The dataset comes from the Microsoft Malware Prediction dataset on Kaggle, and the purpose is to predict the possibility of a machine being infected by various malware according to the different attributes of the machine. Each machine in this dataset occupies a row, uniquely identified by a “Machine Identifier”. “Has- Detections” is an attribute to reflect whether Malware was detected on the machine. Besides, the dataset has nearly 9 million rows and 83 attributes, including computer hardware information, computer settings information, anti-virus software information, etc. In order to select a better model, the dataset is divided into training set and test set according to the ratio of 2:1.

2.2 Method

2.2.1 Data Cleaning. Since the dataset is vast, and it takes up too much memory and time to load, the work firstly reduced its memory storage by casting the data into a smaller data type. For example, from float 64 to float32. After this, the memory usage was reduced by 4 percent.

Next, this work did a statistic on the Nan value for every field. The field whose Nan value percentage was larger than 95 percent were removed, and replaced the Nan value in the remain fields by an unoccupied integer value.

There are six ordinal fields in the dataset, each of which is version identifier composed of different numbers and decimal dot. Reasonably, the closer the two versions are, the more similar vulnerability they should share, so this work need to let the model be aware of the ‘distance’ between two version numbers. Apparently, naive label encoding fails to achieve so because it only represents every version by a randomly assigned integer value. However, if the one-hot encoding is applied, the remaining fields will be encoded in the same way, and this will lead to dimension explosion. Thus, this work decided to label it in the following way. The work firstly split the version by its decimal dot four into several number segments, and treated each of the new segments as different fields respectively. As shown in figure 1, if two version numbers are closed to each other, more fields would share the same number. By doing so, the model can make use of the distance information of the version.

![Figure 1. Ways to encode labels](image)

Then, this work removed the field within which a single value dominant the whole field. After the above steps, the work reduced the quantity of the fields to 64.

2.2.2 Dimension Reduction. In order to further handle the dimension problem and improve training efficiency, the dimension of the dataset was also reduced. Firstly, PCA [4] and other linear reduction method did not apply here, for the reason that most of the fields are nominal and label values have no numerical meaning. Secondly, the work tried several non-linear methods like T-SNE [5] and LLE [6], but they are too complex and time-consuming. Finally, this work decided to use Chi-square testing to test the dependency of the two categorical fields and eliminate the fields that have little importance to
the label. Although this method only considers whether there is field in the sample and it does not consider the frequency of occurrences [7], this problem could be ignored in this dataset. As the result, the top 42 fields were selected, such as System Volume Total Capacity, AV Products Installed, OEM Model Identifier, etc.

2.3 Training
This work chose 3 models (Logistic Regression, KNN and LightGBM) and got the results respectively.

Logistic regression is a linear model and it performs well to reveal linear relationship within the data. KNN is a kind of clustering model, which uses data directly to classify instead of setting a model first [8,9]. It makes use of the distance among the samples in input space to assign the class with a shorter distance to the input. In order to get both efficient and accurate models, this work selected data in the top 20 important fields to train logistic regression and KNN model.

LightGBM is a gradient boosting framework that uses the tree-based learning algorithm, and it shows the best performance among the tree models [10]. The parameters were set as follows: num_leaves = 1024, min_data_in_leaf = 200, objective = binary, lambda_1 = 0.15, lambda_2 = 0.15, random_state = 42, verbosity = -1. After training 1000 times, the model showed the best results.

3. RESULT AND DISCUSSION

3.1 Experimental result
The dataset discusses in section 2.1 was used to train three models. As for the logistic regression model, figure 3 shows its ROC curve. As for the KNN model, the ROC is shown in figure 4. Although in Fairuz Amalina Narudin's research [11], the KNN model had the highest accuracy in predicting mobile malware, the best accuracy of KNN was 0.530, which was lower than the logistic regression model (0.569). Obviously, the results are unexpected because there are many nominal fields in the dataset, and the labels do not have numerical meaning. However, linear regression focuses on linear properties among different fields while KNN focuses on the distance for different samples in the sample space, so they are not able to fit our dataset.

Figure 2. ROC for logistic regression
Besides, the ROC curve of the LightGBM model is shown in figure 5. It achieves the best accuracy and it is very fast. This work inputted all data and got 0.720687 of AUC. Comparing our result 0.720687 with the top 3 solutions on the public leaderboard in Kaggle (see table 1), which indicates our trained model has a better performance. However, the official testing data is not accessible by us and the work used the testing data from the train/test split form the same file.

### TABLE I. COMPARISON OF RESULTS WITH THE TOP 3 SOLUTIONS IN KAGGLE

| Rank | Team Name     | AUC Score |
|------|---------------|-----------|
| 1    | sashimi power | 0.71444   |
| 2    | APTX4869 Power| 0.71167   |
| 3    | tofu power    | 0.71136   |
| ours | 0.72069       |           |

3.2 Analysis of relevant features

After the experiment, this work selected three features with a high correlation to evaluate the accuracy of the result. Firstly, “AvSigVersion2” was noticed as the most important field that contributes to the label, a part of the segments of “AvSigVersion” (possibly stands for the signature version for anti-virus version). This work plotted the value of label “HasDetections” against “AvSigVersion”, and found out that the value of “HasDetections” varied a lot among different values of “AvSigVersion”. This indeed shows that the two variables are related to each other. If different “AvSigVersion” stands for the different version of anti-virus software, this may reveal that some anti-virus software has bugs or...
pitfalls and will suffer more attacks. So, this plays a significant role to determine whether a machine will more likely to be hit or not.

Secondly, the work further discovered the variable “Census Processor Model Identifier” and “Census System Volume Total Capacity”, which ranks the 4th and the 2nd respectively according to the importance given by the model. However, the work surprisingly discovered that if these features were looked alone, nearly no relationships to the label was shown.
Overall, although LightGBM model could effectively predict malware attacks, there are still some problems. The first problem is that for some fields, even though they rank high in the importance list given by the LightGBM model, they show little relation to the label when looking isolated. The second problem is that some fields take too many different values that Chi-Square is not effective for these fields. For example, “CityIdentifier” takes up to 140 thousand values. Even though when analyzing individually, no relationship between it and the label is shown, it ranks the 5th in Chi-Square testing result.

4. CONCLUSION
In this report, our method to process “Microsoft Malware Detection Dataset” was introduced with different models tested and focused on some important fields. It showed that when training over dataset that had a variety of features, LightBGM had a higher accuracy. However, the AUC score still lower than 80% due to a number of null data and unimportant features. In the future, the official testing dataset might be used and different methods to compress the possible values for some fields should be conducted to make Chi-square more applicable to these fields.

REFERENCES
[1] Tran, K., Sato, H., Kubo, M. (2020). MANNWARE: A Malware Classification Approach with a Few Samples Using a Memory Augmented Neural Network. Information, vol.11, no.1, pp.51.
[2] Sahs, J., Khan, L. (2012). A machine learning approach to android malware detection. In 2012 European Intelligence and Security Informatics Conference, pp. 141-147.
[3] Schultz, M. G., Eskin, E., Zadok, F., & Stolfo, S. J. (2002). Data Mining Methods for Detection of New Malicious Executables. IEEE Symposium on Security & Privacy. IEEE.
[4] Pearson, K. (1901). LIII. On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, vol.2, no.11, pp.559-572.
[5] Maaten, L. V. D., Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, vol.9, no.Nov, pp.2579-2605.
[6] Roweis, S. T., Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. science, vol.290, no.5500, pp.2323-2326.
[7] ZHANG Rui, YANG Jiyun.(2014). Android malware detection based on permission correlation. Computer applications (05), 1322-1325
[8] Dasarathy, B V. (1990). Nearest neighbor (nn) norms: nn pattern classification techniques. Los Alamitos IEEE Computer Society Press, 13(100), 21 - 27.
[9] Dreiseitl, S., Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: a methodology review. Journal of biomedical informatics, vol.35, no.5-6, pp.352-359.
[10] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In Advances in neural information processing systems, pp. 3146-3154.
[11] Narudin, F. A., Feizollah, A., Anuar, N. B., Gani, A. (2016). Evaluation of machine learning classifiers for mobile malware detection. Soft Computing, vol.20, no.1, pp.343-357.