Scientific Journal of Impact Factor (SJIF): 4.72
e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 5, Issue 01, January -2018

A Study on Clustering Classification Technique based on Machine Learning to Detect Android Malware Variants

Woong Go[1], Jun-hyung Park[1]

[1]*Korea Internet & Security Agency*

Abstract — *Mobile malware found these days are distributed for financial gain. Most of those malware are created and used as a malware variant that re-uses existing malicious behavior, because the financial objective can be achieved efficiently at low cost, compared with creating new malware. Another reason is that mobile malware with a short life cycle can be created massively to spread infection. However, anti-malware solutions available these days detect malware using the known signature of malware. Therefore, those solutions have a limit in detecting a malware variant that modifies existing malware partially. If many malware variants can be detected quickly, infection spread can be blocked in early stages and damages can be reduced. This paper proposes a clustering classification technique based on the unsupervised machine learning algorithm, which is designed to detect malware variants quickly that seek financial gain.*

Keywords-*Clustering; Machine Learning; Malware Variants; Detection, Classification*

I. INTRODUCTION

The Android-based smart phones is truly a representative device that occupies more than a 90% market share in the world. As the number of Android-based smart phone users increases, various contents and services are provided [1], which means that attackers can achieve their financial purposes easier. As a result, various Android-based malware is diffused. Kaspersky Lab announced that they detected more than 8.5 million malware in 2016 alone, which is an increase of 3 times over the last year. Some malware is newly created and diffused among those every-increasing malware. However, many malware reuses the existing code. Those malware variants modify existing malware partially and redistribute it after repackaging. According to TrendMicro, about 80% of top 50 applications registered in Google Play with various categories are actually variants [3][4].

Many malware variants are created and diffused to resolve the problem of a short life cycle and infect many smart phones. The life cycle of the malware is significantly shorter than that of the PC due to the App market policy and various anti-malware solutions. As a result, efficiency deteriorates even though a high-level attack techniques is applied. Therefore, attackers modify the code partially and reuse it to produce in large quantities and avoid detection by the signature-based anti-malware solution. Many similar malware is created and distributed using this method.

However, we have no choice but to analyze all Apps statically and dynamically, in order to respond to those malware. It causes unnecessary costs because only some of malicious behavior codes are modified in the malware variant. We can reduce costs and respond to malware faster as well, if we can determine malware quickly.

This paper accordingly proposes a clustering classification technique based on machine learning to detect malware variants. The proposing technique enables us to determine and respond to malware variants quickly.

II. RELATED WORKS

In this section, we describe the existing research related to proposed method simply.

2.1. Machine Learning

The concept of "machine learning" appeared first in the paper written by the professor Arthur Samuel, Sandford University in 1959. Machine learning is defined as "enabling the computer to learn by itself without programming" [5].
There was no significant progress in machine learning for a while due to a limit in computing capabilities, and research started in earnest since the 1980's. Machine learning has been used in various industries and various learning and application models are created these days, based on the machine learning technology and big data.
Depending on the type of feature information, machine learning can be divided into supervised, unsupervised, and reinforced type [6].

2.2. K-means Algorithm

The concept of the K-Mean algorithm was first introduced by Hugo Steinhaus in 1957, and the standard algorithm used now was devised by Stuart Lloyd in 1957, It was not until 1982 that the algorithm was released first in the science magazine.
K-Means clustering is one of the unsupervised learning algorithm, which classifies data into k groups based on the level of similarity. This algorithm is useful when classifying among the data set[7].

III. PROPOSED METHOD

The variant malware detection technique proposed in this paper detects malware by grouping them into 11 types using the K-Means algorithm. The following figure shows the basic flow of this technique.

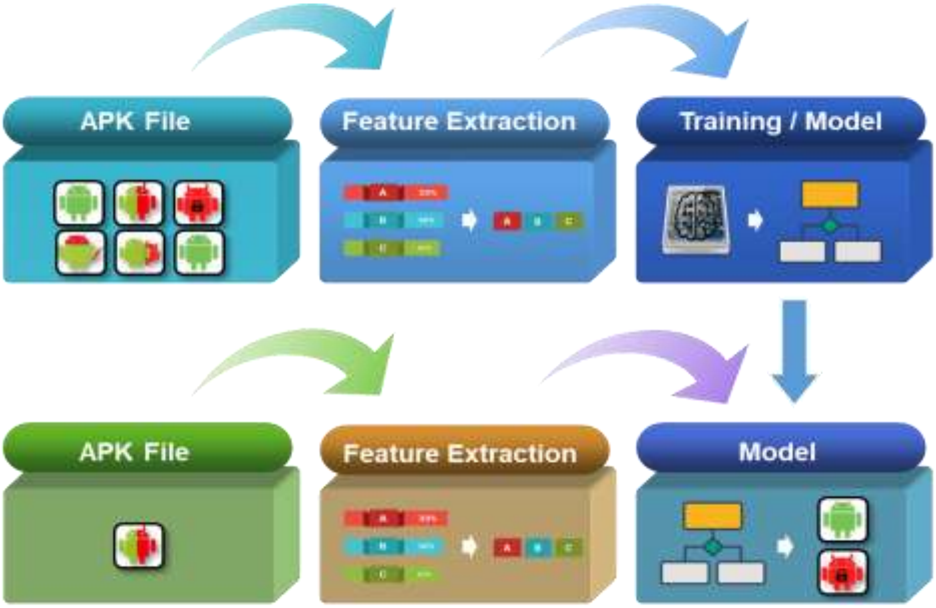

Figure 1. Basic Flow of Proposed Method

3.1. Types of mobile malware

11 malware types were classified by analyzing the attack pattern of malware designed for financial gain and the following table shows the classification items.

Table 1. Types of Mobile Malware for Clustering

Category	Definition
bankun1	Stealing financial information by making the same screen with the general bank App.
bankun2	
bankun3	
hello_jni	Stealing financial information when the user runs the bank App, by hiding the leak site in the SO library.
hello_jni_b	
pdex	Conducting malicious behavior by executing the p.dex file under the assets folder.
shella	Conducting malicious behavior by referring to the libshella.so, libshellx.so file under the lib folder. Running the executable OAT file that can be executed in the ART machine in the so file
soapi	Including /soapi/ in the leak site address and conducting malicious behavior after receiving a remote control command from the SMS message.
xxxxx	Conducting malicious behavior by installing the xxxxx.apk file under the assets folder.
view	Pretending to be a courier App that contains xxxxx.view in the class name. The leak site address is changed if the SMS message beginning with a string like "sorry!!" is received. Preventing App uninstallation by popping up a warning message, if the device manager right is revoked.
kbs	If the string /kbs/ is included in the leak site address and the user executes the bank App, an update message appears that prompts the user to uninstall the existing bank App, and download, install, and execute the false bank App.

3.2. Second-order headings

The feature information should be extracted and used selectively for unsupervised machine learning. Accuracy is not increased even though all feature information is used. It is important to select major characteristics that can classify the type well. This paper selects 181 APIs and string that are actually used frequently, using malware samples that can be classified into 11 types. The following steps are taken for classification.

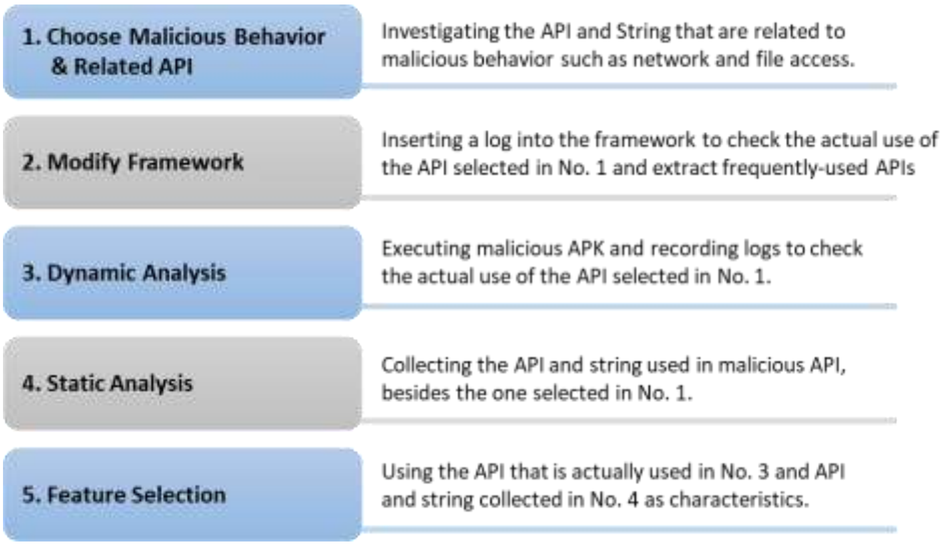

Figure 2. Processes of Feature Selection

A total of 141 APIs and 40 strings were extracted after the above five steps, and the following figure shows some of them.

API				String
delete	getSocketFactory	getNetworkType	putExtra	application/vnd.android.package-archive
exists	setEnableSessionCreation	getSimCountryIso	setAction	NPKI
mkdir	createSocket	getSimOperator	setData	android.app.action.ADD_DEVICE_ADMIN
query	checkClientTrusted	getSimOperatorName	setPackage	ContactsContract$Contacts;
insert	checkServerTrusted	getSimSerialNumber	abortBroadcast	ContactsContract$CommonDataKinds$Phone.CONTENT_URI
update	getConnectionInfo	getSubscriberId	onReceive	chmod
get	addPermission	getAccounts	createFromPdu	mkdir
put	addPermissionAsync	getAuthToken	getMessageBody	shutdown
getColumnIndex	getActivityInfo	getPassword	getOriginatingAddress	content://sms/
getDouble	getInstalledApplications	getUserData	describeContents	android.intent.action.NEW_OUTGOING_CALL
getFloat	getInstalledPackages	peekAuthToken	getDetailedState	incoming_number

Figure 3. Sample of Features

3.3. K-means based unsupervised learning

K-Means learning was conducted using 181 APIs and strings that had been extracted from 11 malware types and features, as analyzed before. As K-Means requires the cluster k value to group the entered data, 11 malware types analyzed before were set as the k value. Then, classification was conducted using 181 feature information. Classification passes through four steps as described below.

1) First, change the feature information of each App to the data with a coordinate value and display the feature information of all Apps on one coordinate system. Then, select a central point randomly as many as the number of clusters (k) and display it on the coordinate system. In this time, the displayed central point of the cluster is not accurate as it is displayed randomly, and the central point moves as operation continues.
2) Create a cluster by grouping data near the central point of each cluster.
3) Calculate the mean value of the data belonging to each cluster and move the central point of each cluster to the location of the calculated mean value.
4) Create a cluster by grouping the data again based on the moved central point. Repeat the above steps 2 ~ 3 times. Clustering is completed if the central point of each cluster doesn't move.

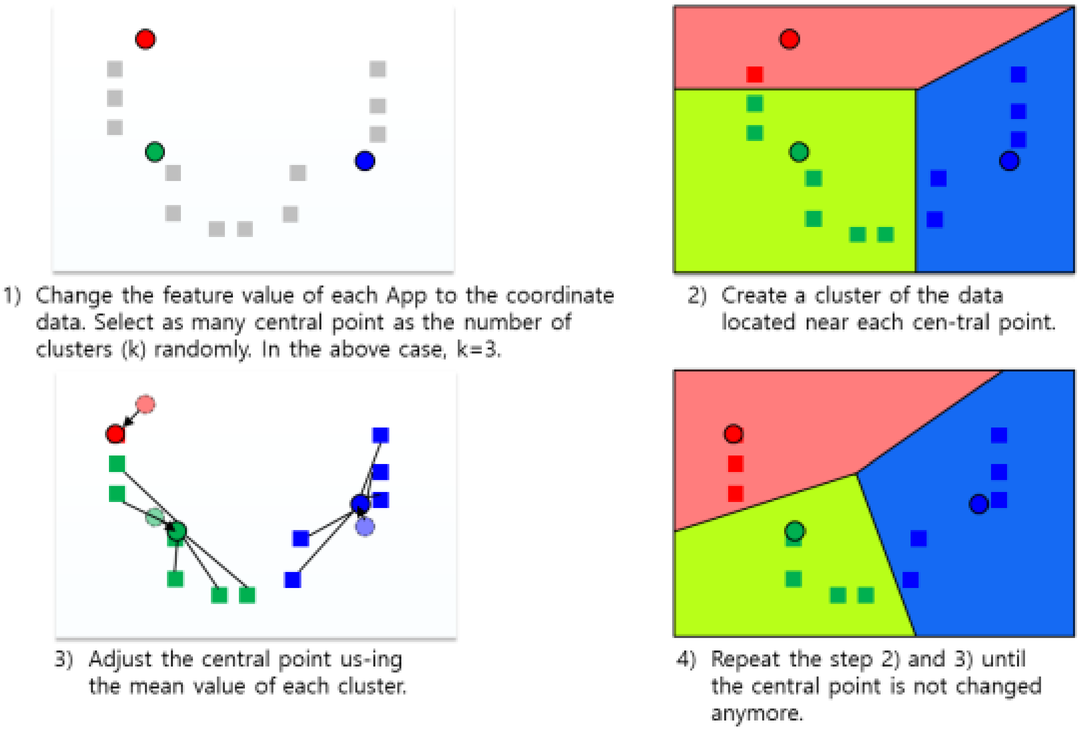

Figure 4. Processes of K-means Clustering

When the clustering process described above is completed, a total of 11 clusters will be created. If a new App is added after the clustering process is completed, 181 features included in the App will be extracted and displayed on the coordinate system. As a result, it can be checked to which cluster the new App belongs Afterwards, move the central point by repeating the clustering process and readjust by creating a new cluster.

3.4. Detection of mobile malware variants

When the clustering process is finished, the entered App comes to have a distance value from the central point in the cluster. The pertinent distance value is calculated according to the location displayed on the coordinate system, based 181 features. If the similar type API and string are used, the similar distance value will be obtained. In the end, there is a high possibility that Apps located close to a specific App is the variant of that App.
Therefore, calculate a distance value among each App and select N App that are close using the distance value, and classify it as a variant, in order to detect a malware variant finally. In this time, N indicates the threshold value.

IV. IMPLEMENTS

To verify the classification accuracy of the proposing technique, a technique is implemented that uses Python language on the Ubuntu 14.04 LTS OS. A total of 925 malware with 11 types was used for the test and the following table shows the number of malware by type.

Table 2. Amounts of Mobile Malware

Category	Amount	Category	Amount
bankun1	65	pdex	28
bankun2	136	shella	15
bankun3	34	soapi	62
hello_jni	226	xxxxx	31
hello_jni_b	13	kbs	55
view	260		

Malware used for the test has been already distributed and determined as malware, and the number of each malware is different because the distribution and collection case varies depending on the malware type.

For the test, a total of 925 malware was analyzed statically/dynamically to extract the API and string, and 118 APIs and strings were arranged that are used as a feature. If there is a feature used in malware, 1 will be saved.

Otherwise, 0 will be saved. A comma (',') is used as a separator. The following figure shows how the feature information is saved.

af_md5	af_features
00b09835092be02ad0fdcca84d6976ef	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.0.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
00DB2C5FA8059CADA9A20957764FFF5F	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.0.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
01BF0EDEA7F6D5575DB225732FCB0E77	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.1.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
0b728a78708e222176a7f7dc5f02a4a2	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.0.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
1B0BE139FEFDAFB24CC6996CC13A8A3E	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.1.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
1c340debcddd0ed0c761065561832335	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.0.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
1CBB4510391941606157BA31BD4D15DF	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.0.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
1D156A0999DB55F5A67D54036C614DD4	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.0.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
1DDF1E0FFAE94902E4A2A83CF4EF5FFF	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.1.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
1F659DCC859E23CFB8DEC41700550706	1.1.0.1.0.1.1.1.1.0.1.1.1.0.1.0.0.0.0.0.0.0.1.1.0.1.1.1.1.0.0.1.1.0.0.0.0.0.0.1.1.0.1.0.0.1.0.0...
NULL	NULL

Figure 5. Sample of Feature Information

The result of clustering that is classified by entering the arranged data into the implemented system has shown a high level of classification accuracy mostly. The following table shows the result of malware classification in each cluster.

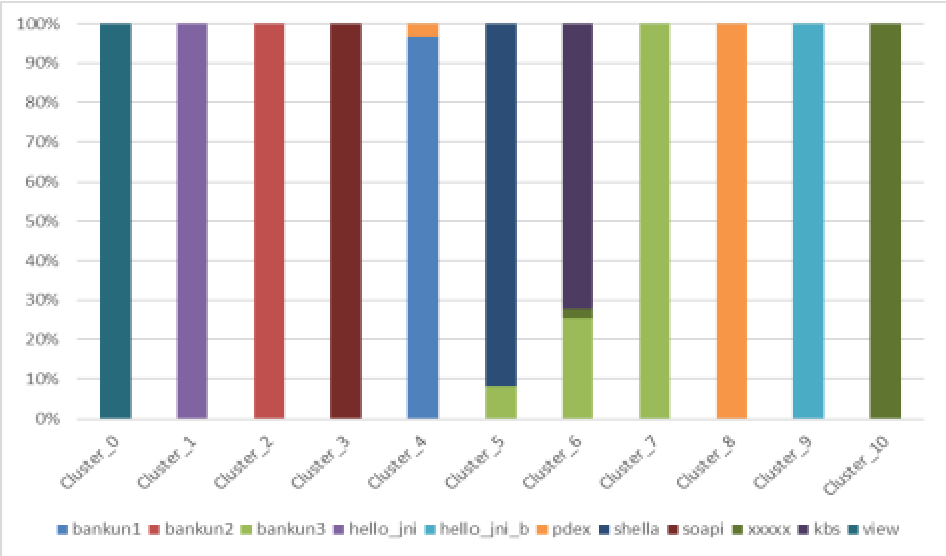

Figure 6. Result of Malware Classification in each Cluster

Table 3. Result of Cluster 0~10(C0~C11)

Category	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
bankun1	-	-	-	-	65	-	-	-	-	-	-
bankun2	-	-	136	-	-	-	-	-	-	-	-
bankun3	-	-	-	-	-	3	12	19	-	-	-
hello_jni	-	226	-	-	-	-	-	-	-	-	-
hello_jni_b	-	-	-	-	-	-	-	-	-	13	-
view	260	-	-	-	-	-	-	-	-	-	-
pdex	-	-	-	-	1	-	-	-	27	-	-
shella	-	-	-	-	-	15	-	-	-	-	-
soapi	-	-	-	62	-	-	-	-	-	-	-
xxxxx	-	-	-	-	-	-	1	-	-	-	30
kbs	-	-	-	-	-	-	55	-	-	-	-

As clustering is simple grouping, there is no information about the type of each cluster. However, we can classify the type of the pertinent cluster using the type of the App included in each cluster. When we review the malware type of each cluster, we can see that the same type malware actually belongs to the same cluster.

However, bankun3 type malware is grouped up to 56% and grouped in three clusters. The reason is that the type of the API available in the Android environment is limited, and the API/string used for malicious behavior can be used by many malware at the same time. Even so, we can see that most of the same type is grouped in cluster7.

Some malware grouped in the same cluster is as follows.

Cluster_7	0.69425820833016216	5E27E64AC122072A********************	bankun3
Cluster_7	0.69425820833016216	F99A306FB37500697********************	bankun3
Cluster_7	0.69425820833016216	D85F1DD92C22A4A********************	bankun3
Cluster_7	0.69425820833016216	054F85E16C11675D********************	bankun3
Cluster_7	0.86322207720298849	A21440AA18A51119********************	bankun3
Cluster_7	0.86322207720298849	CB60254CC0088C05********************	bankun3
Cluster_7	0.86322207720298849	870916D6143E005FE********************	bankun3
Cluster_7	0.86322207720298849	6216CBDCB61D6B9********************	bankun3
Cluster_7	0.92217976147027203	F2DDF03172BE6743********************	bankun3
Cluster_7	0.92217976147027203	77B08936C9B9F0E9********************	bankun3
Cluster_7	1.0552598766191281	A63CCB1D4DEE244********************	bankun3
Cluster_7	1.0552598766191281	32f178b3473f6cccfb********************	bankun3
Cluster_7	1.0799097120359586	4B520CDB4EC700B7********************	bankun3
Cluster_7	1.0799097120359586	FC84C5F4B2CB5CA9********************	bankun3
Cluster_7	1.1955596517474245	140b8366416267ac5********************	bankun3
Cluster_7	1.8375884473785382	6FC9090AE128F3186********************	bankun3
Cluster_7	1.8375884473785382	09A0A1D6E7B3E137********************	bankun3
Cluster_7	1.9352396116684509	33C42DB059E28DA4********************	bankun3
Cluster_7	1.9352396116684509	7AC30F22F5F85E170********************	bankun3

Figure 7. Sample of Malwares in a Cluster

The first column is the cluster type and the second column is the cluster distance value. The third column is the hash value of malware and the fourth column is the type of malware.

Classify the App by calculating the difference in the distance value to detect a malware variant.

V. CONCLUSION

As Android-based smart phone users increase sharply, Android malware targeting those smart phones also increases significantly. Many solutions were released to respond to such an increase in malware, most of them compare the hash data of malware only. However, more than 80% of malware all over the world is a variant that cannot be detected by simple hash data comparison. Therefore, this paper proposed a technique to detect and classify malware variants quickly using K-Means clustering.

Variants can be quickly detected that cannot be detected using the hash data, by classifying and detecting malware variants using the proposed technique. Eventually, the technique will reduce damages caused by malware drastically.

ACKNOWLEDGMENTS

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2015-0-00003, Development of Profiling-based Techniques for Detecting and Preventing Mobile Billing Fraud Attacks)

REFERENCES

[1] Masao Kakihara, "Mobile Apps in APAC: 2016 Re-port", Think with Google. Dec. 2016. https://apac.thinkwithgoogle.com/intl/en/articles/mobile-apps-in-apac-2016-report.html

[2] Kaspersky Lab, “Mobile Malware Evolution 2016,” Feb 2017. https://securelist.com/mobile-malware-evolution-2016/77681/

[3] Yajin Zhou, Xuian Jiang, “Dissecting Android Malware: Characterization and Evolution,” In securi-ty and Privacy(SP), 2012 IEEE Symposium on, pp.95-10, May 2012.

[4] Trend Micro, “A Look at Repackaged Apps and their Effect on the Mobile Threat Landscape”, (http://blog.trendmicro.com/trendlabs-security-intelligence/a-look-into-repackaged-apps-and-its-role-in-the-mobile-threat-landscape/), April, 2014.

[5] A. L. Samuel, “Some Studies in Machine Learning Using the Game of Checkers,” IBM Journals & Mag-azines, Vol 3 Issue 3, pp.210-229, 1959.

[6] Aihua Shen, Rencheng Tong, Yaochen Deng, “Application of Classification Models on Credit Card Fraud Detection,” Service Systems and Service Man-agement of the 2007 IEEE International Conference, pp.1-4, 2007.

[7] Wikipedia, K-means clustering, https://en.wikipedia.org/wiki/K-means_clustering, retrieved Jan 15, 2018.

