Unsupervised Clustering for Data Leakage Detection

Abstract
Data leakage is one of the key problems in security protection. One potential solution is to apply clickstream analysis. It involves unsupervised clustering, community detection and noise detection and is one of the cost effective ways to address such an issue. Thus, in this paper, unsupervised methods were applied for abnormal behavior detection on four kinds of data sets converted from original log data. They are clickstream distance matrix, transition matrix and matrixes consisting of selected session features. Additionally, dimension-reduction mechanism was used for noise reduction and visualization. Finally, abnormal data were selected from common subsets of different results.

1. Introduction

1.1 Data Leakage

Data leakage is one harmful phenomenon in which internal data or information is transmitted to outside company by employees through hard copy, web email or photos. Customer data leakage will incur loss on both reputation and money. This is especially true to banking industry. Moreover, this may also encourage fraudulent cases.

The damage rendered by data leakage is very huge. In article, "The Real Cost of Data Loss And How To Prevent It", (Sheldon, 2015), it says that, "A recent Verizon report suggests that 'small' data breaches which see less than 100 records lost come with an average cost of $18,120 to $35,730, but in a worst-case scenario could reach $555,660. Large data breaches (100 million records or more) cost an average of $5 million to $15.6 million and top out at $200 million."

Thus, internal data leakage detection is necessary to help protect the reputation for the company and discourage any potential fraudulent case.

1.2 Clickstream Analysis

Clickstreams are traces of click events conducted by users in an application on the company websites. And this kind of data is recorded in a log file. (Wikipedia, 2016b)

Clickstream analysis is one of the useful algorithms to find the patterns of users and to detect abnormal behavior patterns. In electronic business area, clickstream analysis is utilized to determine which kind of users will buy products or to predict what kind of products one user will be interested in through analyzing several click-through events. (Antonellis et al., 2009)

Because it's proven to be successful in electronic business and Sybil detection (Wang et al., 2013), thus, it's worth a try to use clickstream analysis on application log data to look for abnormal behaviors. (Hofgesang and Kowalczyk, 2005)

1.3 Unlabeled Data and Unsupervised Clustering

Traditionally, machine learning methods, such as Support Vector Machine (SVM), Logistic Regression, Random Forest and Neural Network are applied into labeled data and Back Propagation (BP) algorithm is used to tune models and try for the best set of parameters for specific model.
However, in real world, cost of obtaining labeled data is very huge because it requires experts to label each point of data. For large number of data points, it's impossible to investigate each point of data and label all the data correctly for various reasons, especially when there is a lack of knowledge about abnormal patterns. Therefore, how to use unlabeled dataset is important and attractive to reduce the cost and to utilize large data set.
The useful method to cope with unlabeled data problem is to employ unsupervised machine learning algorithms, such as K-means clustering (Kanungo et al., 2002), community detection (Fortunato, 2010) and noise detection. The logic of these algorithms is to group data points which represent similar behaviors, and to find groups consisting of abnormal behaviors or abnormal noise points based on assumption that abnormal data points are relatively rare compared to normal data points.
Transfer learning can be utilized for unsupervised clustering, so that hyper-parameters are better estimated based on other similar researches. (Liu et al., 2012) (Laskov et al., 2005) (Ma, 2003) (Raina et al., 2007)

1.4 Noise Detection

Noise is defined as certain data point that falls outside of normal distribution. Noise detection is very useful as an unsupervised learning method to detect abnormal behaviors.

2. Problem Statement

There are some problems caused by internal data leakage in companies, such as damage to companies' reputation and loss of potential profit. One of the leakages is leakage by application users. They may copy data to their local hard drives while accessing an online application. Data used in this paper contained no label saying whether the related behavior was normal or not. Therefore, in this paper, clickstream analysis and unsupervised learnings were adopted.

3. State of the Art

Nowadays, it becomes more and more attractive to use unlabeled data and unsupervised machine learning algorithms. The reason is that labeling dataset is costly as it needs a lot of time for experts to investigate and decide which label is to be assigned to which data point. Moreover, the accuracy of the model is another problem that even experts will make mistakes in labeling data points, which may render the entire effort useless. Thus, it is valuable to utilize unlabeled dataset with unsupervised learnings or semi-supervised learnings.
Amongst the unsupervised learning algorithms, clustering, community detection and noise detection are the popular choices. They are always used for unlabeled data to partition data points into several kinds of groups based on their attributes, such as consumer segments.
With the increasing popularity of online shopping, a useful data analysis method has been put forward, called clickstream analysis. This algorithm analyzes the click path of the users when accessing the website. It has already

been used in some sales websites and in some research papers. It is also mentioned how to detect abnormal behaviors of users using clickstream analysis in a research paper called, "Clickstream analysis for Sybil detection". (Wang et al., 2013) (Zhang and Guan, 2008) (Portnoy et al., 2001)

4. Solution: Description of Approaches

4.1 Methodology

4.1.1 Unsupervised Learning

In this paper, there are three kinds of unsupervised learning algorithms used. One is community detection, one is unsupervised clustering and another is noise detection.

In network construction, the data used is in the form of clickstream which was derived from log file. (Singh et al., 2010) And in clustering and noise detection, there are four kinds of data sets used. They are distance matrix of clickstream, data frame containing click attributes of users' daily behaviors, data frame containing session attributes of all the sessions and data containing vectors representing transition matrix for each session of clickstream.

The community detection methods utilized in this paper are naive statistical method, label propagation algorithm (Xie and Szymanski, 2013) and other community detection methods. And clustering partitioning mechanisms used in this paper are hierarchical clustering (Zhao and Karypis, 2002), K-means clustering and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996). Moreover, noise detection with auto-encoder is another method employed.

4.1.2 Visualization

After the model building stage, visualization is also important to understand data and the performance of models. The approach picked in this paper is t-distributed Stochastic Neighbor Embedding (t-SNE).

There are a lot of ways to reduce the dimensions for vectors in high-dimension space and to project data points into low-dimension space, such as three dimensions or two dimensions. The characteristic of these methods is to retain similarity or distance between each pair of points, which means similar points will be close in the low-dimension space.

t-SNE is a machine learning algorithm for dimensionality reduction. In t-SNE, similar nodes in high-dimension will have relatively higher probability to be picked by each other as neighbor nodes in low-dimension space. (Maaten and Hinton, 2008)

For this case, t-SNE was adopted for visualization after clustering and noise detection algorithms.

4.2 Data Exploration

4.2.1 Data Acquisition

The data used in this paper is from the transaction log file stored in the server database. Log file records the interaction between applications and users in form of click events conducted by users. The transaction type in log file includes login, inquiry and so on. (Wikipedia, 2016h)

Table 4.2.1 is structure of original data used in this paper.

Table 4.2.1 Log Data Format

Date	Time	User Id	Branch Id	Transaction Type
20160601	154722	1	0	CUSTOMER INQUIRY
20160601	154722	3	359	CUSTOMER INQUIRY
20160601	154723	6	157	TRANSACTION HISTORY

In Table 4.2.1, Branch Id is the branch office that users belong to (0 means missing value). Moreover, every record in dataset represents one transaction conducted by one user at a specific time. Every user has a large number of records, but time period of transaction log file is 24 hours.
Originally, there are 50 categories of transaction types. For simplicity and accuracy, transaction types are collapsed into fewer categories. Here, 10 categories were chosen based on meaning of each transaction type.

Table 4.2.2 Transaction Type

1	2	3	4	5
Database	Loans	Pdf database	Pdf loans	Pdf print
6	**7**	**8**	**9**	**10**
Pdf property	Pdf trans records	Property	System transaction	Trans and records

4.2.2 Data Distribution

Time spanning is in Figure 4.2.1, from 00:00:00 to 23:59:59.

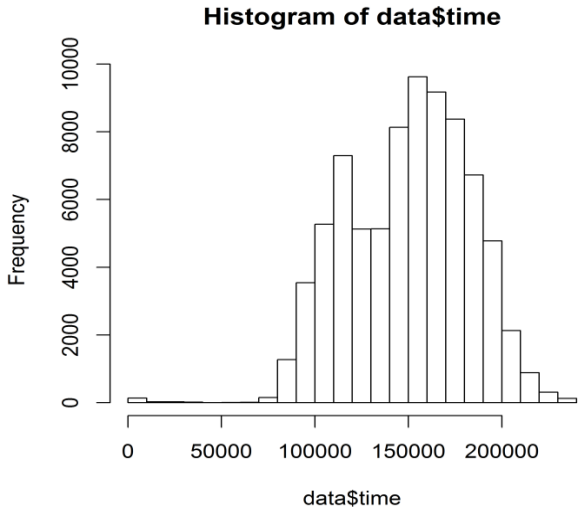


Figure 4.2.1 Time Distribution

The date acquired is on 01/06/2016.
The distribution of transaction types is shown in Figure 4.2.2, with 10 levels.

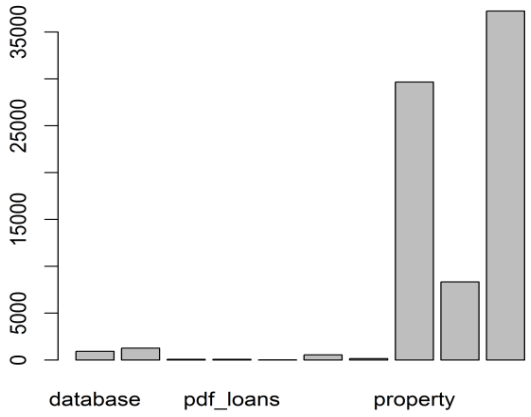


Figure 4.2.2 Transaction Type Distribution

The number of unique users is 1917.

The distribution of branches is in Figure 4.2.3, there are 164 branch IDs, including missing value 0.

4.3 Preliminary Analysis

4.3.1 Clickstream

4.3.1.1 data conversion

Original dataset should be combined and converted into target clickstream format. However, first step is to define time period of one session. After data exploration and consultation from experienced users, it is more suitable to define sessions through setting session's interval time without any transaction between two click events. After data exploration, appropriate interval time is roughly 30 minutes.

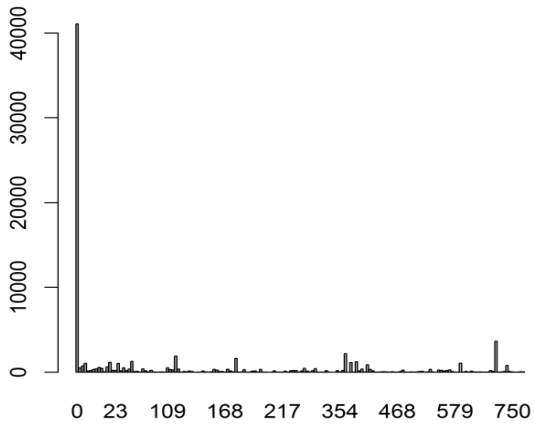


Figure 4.2.3 Branch Distribution

Another step should be taken into consideration is to separate all types of transaction into several predefined categories, which is proven to be useful in research paper (Wang et al., 2013). It's also instinctive because the more the

categories, the less possibility two clickstream data would be similar to each other. Thus, there will be few similar clickstreams, which will cause failure of the algorithm. Thus transaction types were split into 10 categories in Table 4.2.2.
The original dataset was then converted into clickstream format. Every record represents one session of one user. The row name is "date + time + user ID + branch ID", and each record includes a path of click events.
Data format is in Table 4.3.1. Every number, from 1 to 10, represents corresponding transaction type in Table 4.2.2.

Table 4.3.1 Clickstream Data Format

1 + 20150401 + 120000 + 0	3	9	4	8	8	2	3	7	2	4	3	7	8
1 + 20150401 + 121200 + 1	7	3	6	7	9	4	9	8	1	3	5		
1 + 20150401 + 122400 + 0	3	5	7	4	6	2	1	3	3	6	7		

The Alternative version of clickstream includes time interval between each pair of transactions. After discretization, time intervals were separated into several predefined categories, such as within 10 minutes, less than 120 minutes but more than 10 minutes, and more than 120 minutes. Aided by the suggestions from the results in paper (Wang et al., 2013), the version of clickstream dataset without time interval was chosen for continuing analysis.

4.3.1.2 data exploration

After clickstream was obtained, some basic data exploration and analysis was performed for better understanding of dataset, such as statistical analysis about frequency of each category (in Figure 4.2.2), the distribution of session length (in Figure 4.3.1) and Markov Chain (in Figure4.3.2) to show relationship between each pair of categories.

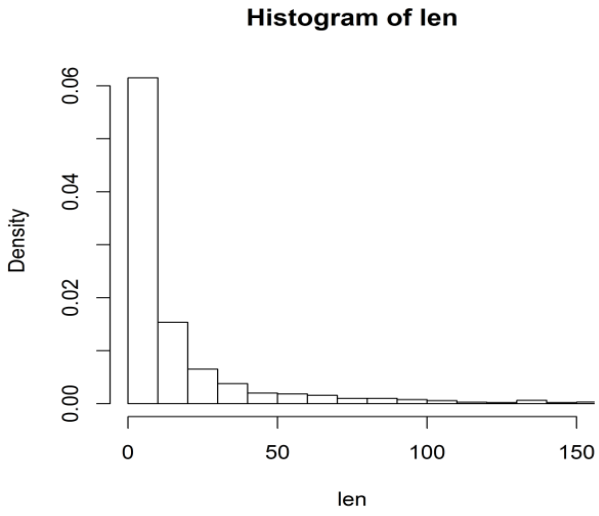


Figure 4.3.1 Distribution of session length

In Figure 4.3.2, Markov Chain is a random process, in which each position has different probabilities to transit to the others. And one important property of

Markov Chain is that it only considers the current state to render probabilities. (Hazewinkel et al., 2001)

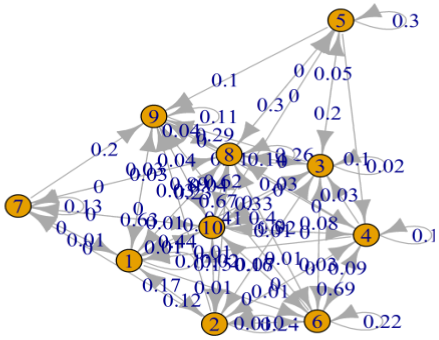


Figure 4.3.2 Markov Chain without Time

4.3.2 Transition Matrix

Transition matrix, is a matrix in which each cell value represents a transition probability from one state to another. It's a kind of representation of Markov Chain.

The basic assumption here is that every session of clickstream data can be represented efficiently using corresponding transition matrix. Similar transition matrixes mean similar sessions, which further mean similar behaviors.

And in this paper, transition matrix here is a 10*10 matrix. To represent clickstream behaviors in high-dimension space, the first step was to convert matrixes into one-dimensional vectors sitting in high-dimensional space. However, 100-dimensions vector is too large to compute. Thus, dimension reduction method was employed to reduce computational cost as well as eliminating noises.

Auto-encoder is one kind of artificial neural networks, and it is always used for dimension reduction, noise elimination, and weight pre-training. The aim of it is to reproduce the input after several hidden layers and output layer. (Baldi, 2012)

After encoding the high-dimensional vectors into lower-dimensional vectors through auto-encoder, every session of transactions was represented by a vector embedded in 20-dimension space.

4.3.3 Click and Session Attributes

User's sessions have statistical properties, such as average session length, average number of clicks per session and so on. The assumption here is that different kinds of sessions have variant statistical properties. Thus, attributes are extracted from clickstream to construct new dataset representing users' behaviors. In new dataset, every record represents daily behavior of one user, which captures patterns of users' daily behaviors, based on assumption that the smallest unit to differentiate different kinds of behaviors is one day. And

in second new dataset, every record represents session's behavior, based on assumption that the smallest unit is session.
The 13 attributes are extracted from clickstream dataset:

- Session Features: average clicks per session, average session length, and average interval time between clicks.
- Click features: as mentioned, there are 10 categories of clicks activities in log file. For each user, the percentage of clicks in each category is extracted as feature. (Wang et al., 2013)

Then there were new datasets containing click and session attributes, which could be employed for clustering algorithms.

4.4 Modeling and Clustering

4.4.1 Assumptions

To find abnormal behaviors, there are some assumptions. One is that abnormal behaviors are sparse and thus they are in relatively small group or could even be seen as noises. Another assumption is that abnormal behaviors are different from normal ones in terms of frequency of single click event or K-gram patterns of clickstream.

4.4.2 Clickstream Analysis and Network Partitioning

4.4.2.1 clickstream analysis

After clickstream was created from original dataset, the most important thing is to calculate dissimilarity or distance between each pair of instances. Therefore, sequence dissimilarity was introduced.
Sequence dissimilarity is a method to calculate distance between two sequences using frequencies of common sub-sequences. The calculation was conducted on frequencies of common sub-sequences and total sub-sequences. There are several calculation methods, such as Euclidean Distance and Polar Distance. According to suggestion of paper (Wang et al., 2013), sequence similarity considering frequencies of sub-sequences based on Polar Distance was adopted in this paper.
Specifically, a sequence $S = (s_1, s_2, s_3, s_4, s_5, s_6)$ was formalized from clickstream data. In sequence S, s_i is the i^{th} element in S. Then C_k was calculated from sequence S as the set of all possible K-grams, which are subsets consisting of K click events from sequence S. To compute the distance of two sequences, for instance, S_j and S_k, firstly the total unique set of K-grams was $T = S_j \cup S_k$. Then, the normalized frequency array c_i consisting of frequency of each K-gram was calculated from each sequence. Finally, formula, $Distance = \frac{1}{\pi}\cos^{-1}\frac{\sum_{j=1}^{n} c_{1j} \times c_{2j}}{\sqrt{\sum_{j=1}^{n}(c_{1j})^2} \times \sqrt{\sum_{j=1}^{n}(c_{2j})^2}}$, was utilized to calculate Polar Distance between two sequences. The value range was from 0 to 0.5.
The subsequent problem was to choose suitable parameters in this model for better performance. However, without any labeled data, it was impossible and costly to try all the hyper-parameters to select the optimal value. The possible and reasonable way was to select hyper-parameters based on other similar academic researches. Thus, according to the result of paper (Wang et al., 2013), in which there are eight categories of transactions in clickstream data and optimal K is five after experiment using labeled dataset, thus, 5-gram was

also adopted in this paper because there were similar number of categories of transactions in clickstream dataset. Additionally, K-gram with large K would capture long sequences that are unlikely to repeat as a pattern and the number of features, as well as computational costs also increases with K. That's why the number five was suitable for hyper-parameter K.
After applying sequence similarity formula into clickstream dataset, the distance matrix was created in which each grid contained dissimilarity value for corresponding pair of clickstream sequences. In this way, there was a relationship matrix to build the network.

4.4.2.2 network construction

The next step is to construct network using distance matrix of clickstream dataset. Each node in network represented one session behavior, and its identification number was session's date, time, branch ID and associated user ID. And each edge between each pair of nodes represented relationship between two nodes. The weight of each edge was the corresponding distance in distance matrix.
Currently, the network built was fully connected. Nevertheless, it was still difficult to detect communities, understand structure of the network and obtain meaningful results if it was complex network with densely connected nodes. For this reason, there should be several steps to cope with this problem.
Because of large amount of nodes existing in network, it's not easy to visualize the network or better understand it. Thus, the first step was to reduce the number of nodes in network. The method was to combine nodes with similar property into one node. Moreover, in this case, zero distance means corresponding pair of nodes are the same. After processing, the number of nodes was reduced from 3145 to 807.
Then, the second thing was to cut down edges using suitable cut-off values and to leave out some nodes with zero node degree.
According to the formula calculating the distance, the maximum value, which represents this pair of nodes has no same sub-sequence, was chosen as cut-off to cut down unrelated edges. Moreover, the next step was to leave out the nodes which have zero node degree.
The Figure 4.4.1 is the resulting network.

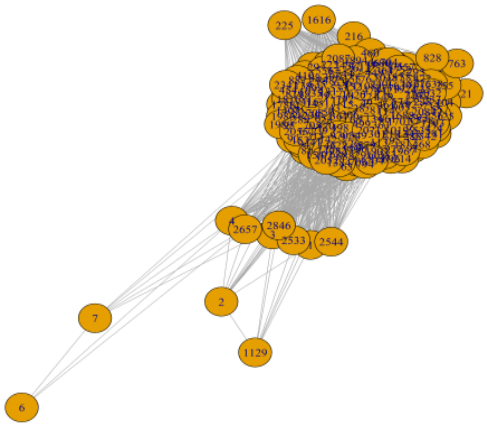


Figure 4.4.1 Densely Connected Network Visualization: after cutting unrelated edges and leaving out nodes with zero node degree, the network was drawn

4.4.2.3 network exploration and statistic method

After construction and pruning of network, it was useful to understand the network through some data explorations, such as distribution of node's degree. The Figure 4.4.2 is the distribution of node's degree.

One way to detect abnormal behaviors was to use suitable cut-off for edge distance, so that noise points not involving in large communities could be easily found. The intuition here is that if cut-off is too large, there might be no noise at all and if cut-off is too small, there might be too many isolated nodes. Several cut-offs were tested until one small separate group emerged.

After suitable pruning of network, there were 680 nodes left. Nonetheless, out of 680, there were 671 nodes which represent sessions with length of clickstream less than five. And in distance calculation mechanism, these nodes were not involved. Therefore, there were nine nodes left which should be closely analyzed.

These nine noise nodes represent nine abnormal groups with less than 20 nodes. The representation nodes' ids were 460, 1816 and 2071. Nodes in these abnormal groups were also considered as abnormal data points. In short, there are a total of 10 abnormal data points deserving closer analysis.

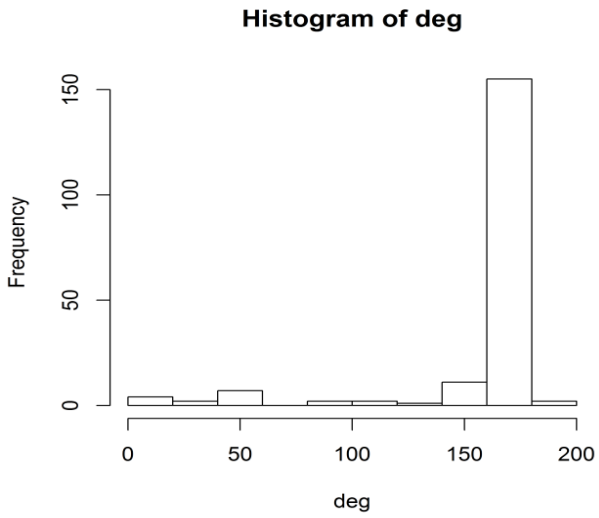


Figure 4.4.2 Distribution of Node Degree: in network without any isolated node

4.4.2.4 community detection

After leaving out the nodes with zero node degree, Minimum Spanning Tree (MST) was applied to densely connected network for pruning, better understanding structure of network and detecting reasonable communities (Pettie and Ramachandran, 2002).

The community detection method is to employ different algorithms to split the network into several parts, and each part includes similar nodes which have similar properties. (Fortunato, 2010)

Actually, there are various kinds of community detection methods to find similar groups. To determine which method is the best one, there are several kinds of approaches. One is to use accuracy of test dataset as target function, another one is to calculate modularity score for each method and select the specific method which maximizes the modularity score. (Newman, 2006)

In this paper, because there is no labeled data involved, thus, it's impossible to use accuracy as target function to choose the best community detection method. So, target function maximizing the modularity score was chosen for community detection method selection. The Table 4.4.1 are the results of modularity scores of all the community detection methods. And Fast Greedy was selected as best one.

Table 4.4.1 Community Detection Method Comparison

Community detection method	Fast greedy	Infomap	Label propagation	Leading eigen	Louvain method
Modularity score	0.8496	0.7925	0.7661	0.8258	0.8494

The Figure 4.4.3 are the results after applying Fast Greedy method into network.

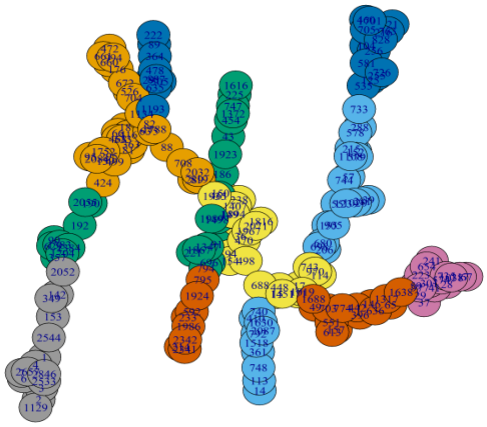


Figure 4.4.3 Community Detection: after using fast greedy algorithm, different colors mean different similar groups

And from partitioned communities, the smallest size group was selected as abnormal group based on the previous assumptions. Then, 43 nodes were selected from network as abnormal points.

Other methods used on distance matrix, such as hierarchical ranking, K-means and DBSCAN based on t-SNE and so on were listed in next section Unsupervised Clustering.

4.4.3 Data Processing

4.4.3.1 transition matrix

In the structure of Auto-encoder, there are three layers, one input layer, one hidden layer and one output layer. The input layer contains 10*10 units of neurons, in which 10 represents the number of categories. And the number of neurons in output layer is the same as number in input layer. There are 20 neurons in the hidden layer, which are designed to reduce the dimensions of the original vectors.

And hyper-parameters in training process are in Table 4.4.2.

Table 4.4.2 Auto-encoder Hyper Parameters

Hyper parameter	lambda	beta	rho	epsilon	Max.iterations
Chosen value	0.0002	6	0.01	0.001	2000

In Table 4.4.2, lambda is the of weight decay parameter, beta is the weight of sparsity penalty term, rho is the desired sparsity parameter, epsilon is a small parameter for initialization of weights and max.iterations is the number of iterations in optimizer.

After predicting reduced-dimension vectors using trained auto-encoder, the original transition matrix dataset was converted into new dataset which contains lower-dimension vectors representing all the sessions.

As for normalization, because values in transition matrix were already in the form of percentage, thus, there was no need to normalize the dataset before clustering.

4.4.3.2 click and session attributes extraction

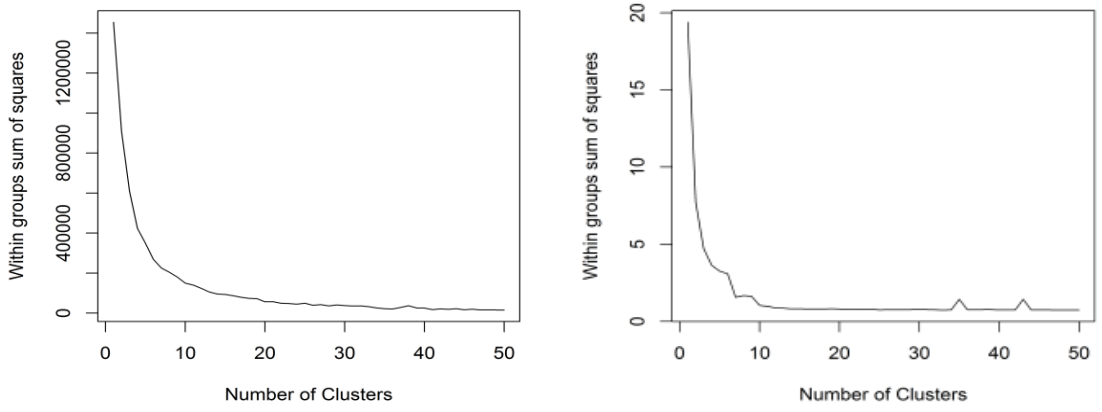
The dataset containing click and session attributes includes 13 attribute columns. However, there are some noises in dataset. Thus, auto-encoder was employed to eliminate noises to some extent. The number of neurons in hidden layer in this auto-encoder is 10. After processing, the dataset with 10 attributes columns was created. Because auto-encoder already scaled all the attribute columns, there was no need for any more normalization.

4.4.4 Unsupervised Clustering

4.4.4.1 hierarchical clustering

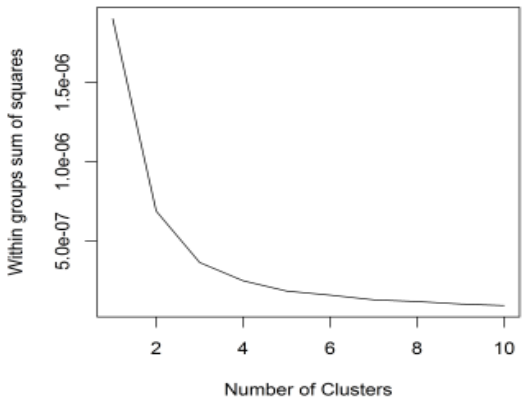
The first method is hierarchical clustering, whose metric is based on distance between each pair of nodes in 20-dimension space. The first problem was how to decide how many clusters should be obtained after clustering. The number of clusters was estimated from the plot in Figure 4.4.4, in which x-axis is number of clusters and y-axis is the total within-cluster sum of squares (variance within the clusters). As Figure 4.4.4 shows, the point where the drop begins to flatten is the suitable point to decide the correct number of clusters. The best numbers of clusters from Figure 4.4.4 are six, six and three, respectively.

Thus, the best number of clusters was employed to split data points in hierarchical tree. The results are shown in Figure 4.4.5.



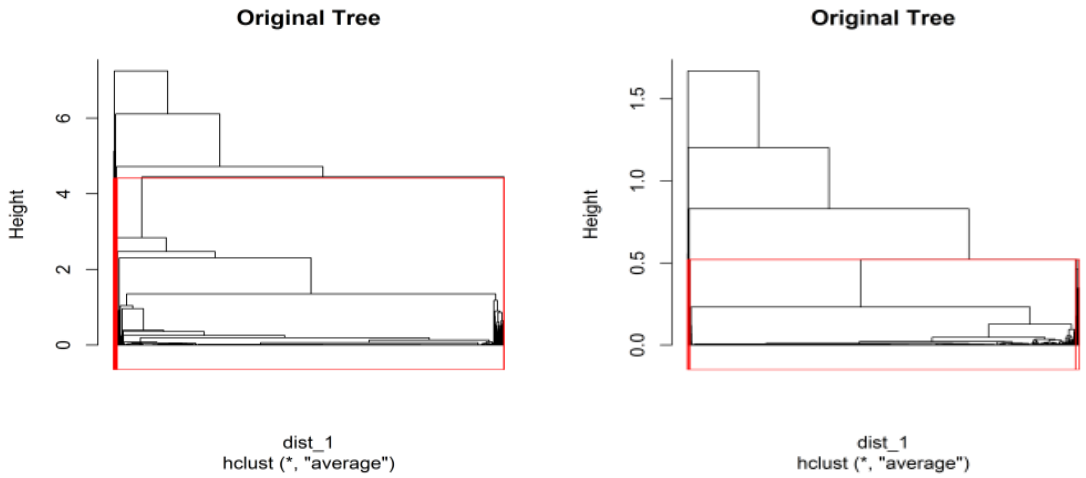
(a) transition attributes

(b) session attributes on user



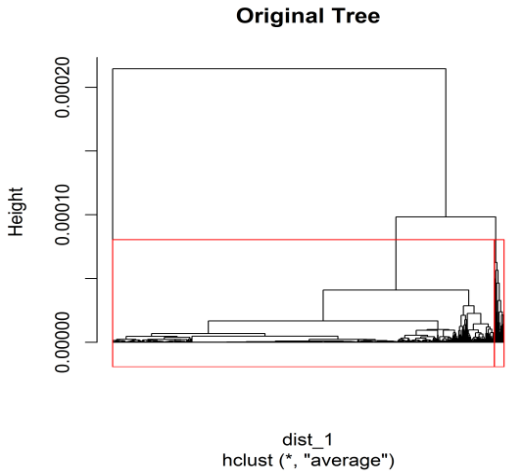
(c) session attributes on session

Figure 4.4.4 exploration on number of clusters in hierarchical clustering: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session.



(a) transition attributes

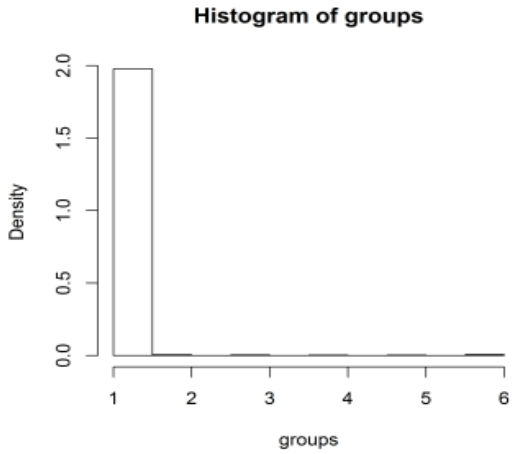
(b) session attributes on user



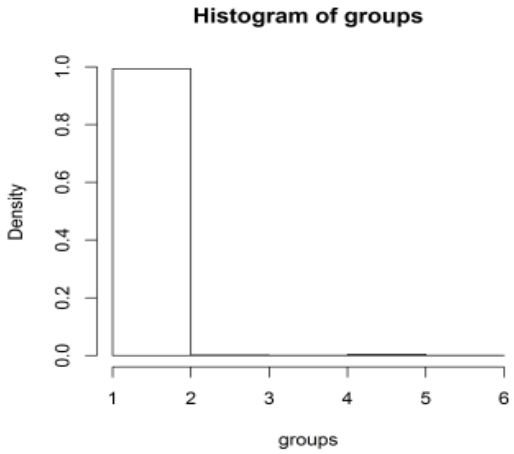
(c) session attributes on session

Figure 4.4.5 hierarchical clustering tree: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session.

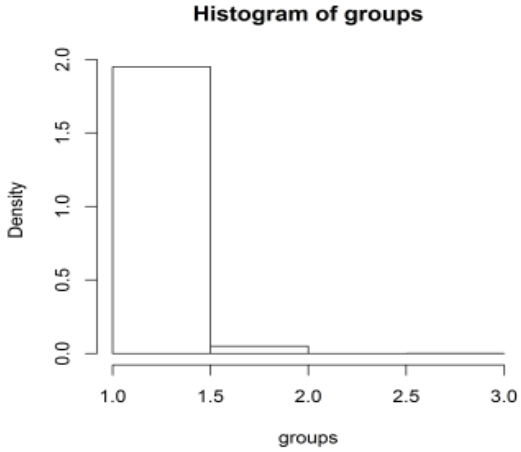
The next step was to plot the distribution of clusters and record resulting nodes in tables. The results are in Figure 4.4.6 and Table 4.4.3, 4.4.4, 4.4.5.



(a) transition attributes



(b) session attributes on user



(c) session attributes on session

Figure 4.4.6 hierarchical clusters distribution: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session.

Table 4.4.3 hierarchical clusters results (transition)

Clusters	1	2	3	4	5	6
Number od nodes	3111	8	7	5	4	10

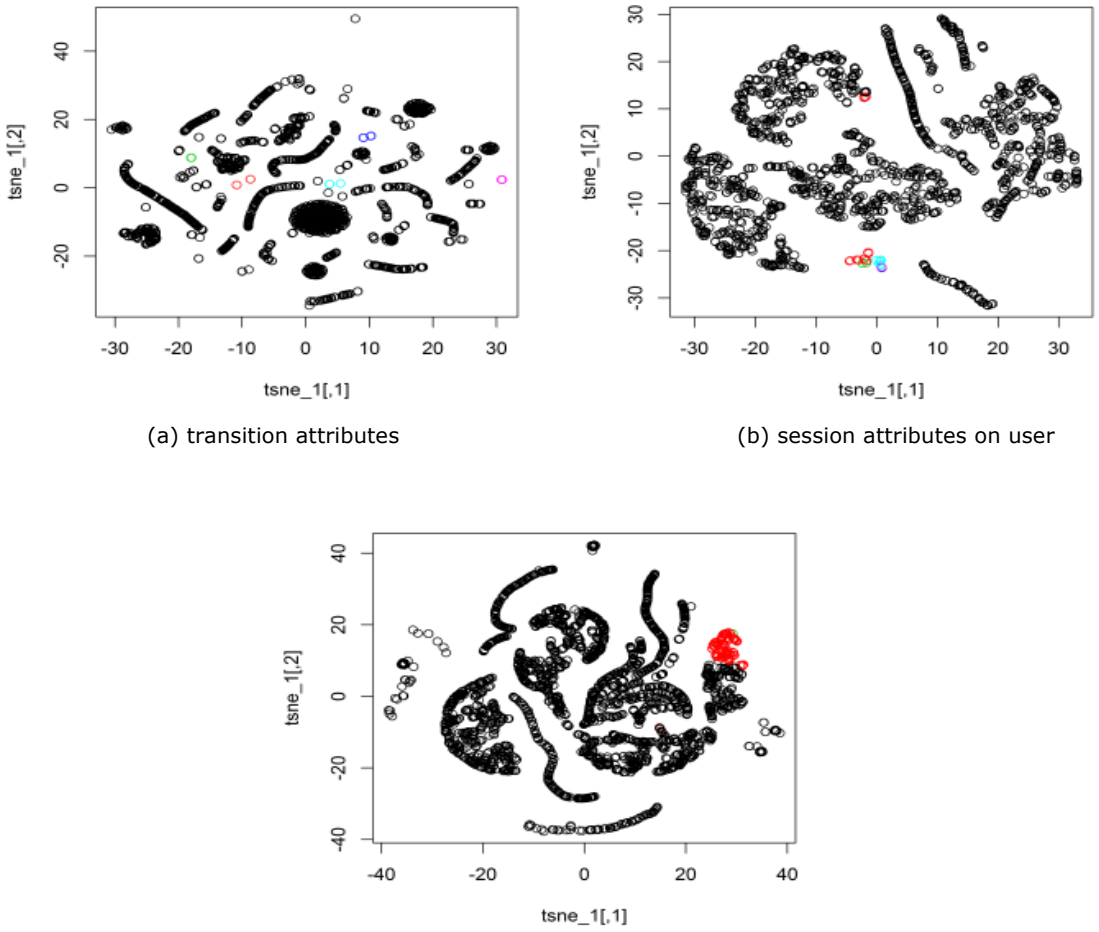
Table 4.4.4 hierarchical clusters results (user)

Clusters	1	2	3	4	5	6
Number of nodes	1599	15	2	1	7	1

Table 4.4.5 hierarchical clusters results (session)

Clusters	1	2	3
Number of nodes	3992	102	1

Then, t-SNE was employed to visualize clustering in two-dimension space. The results are in Figure 4.4.7.



(a) transition attributes

(b) session attributes on user

(c) session attributes on session

Figure 4.4.7 hierarchical clustering plot: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session. Different colors mean different groups.

The small size groups whose numbers are less than 10 were chosen as the abnormal clusters.

Another method is to use hierarchical clustering to output every node's probability to be noise and then select top 20 data points as noise data.

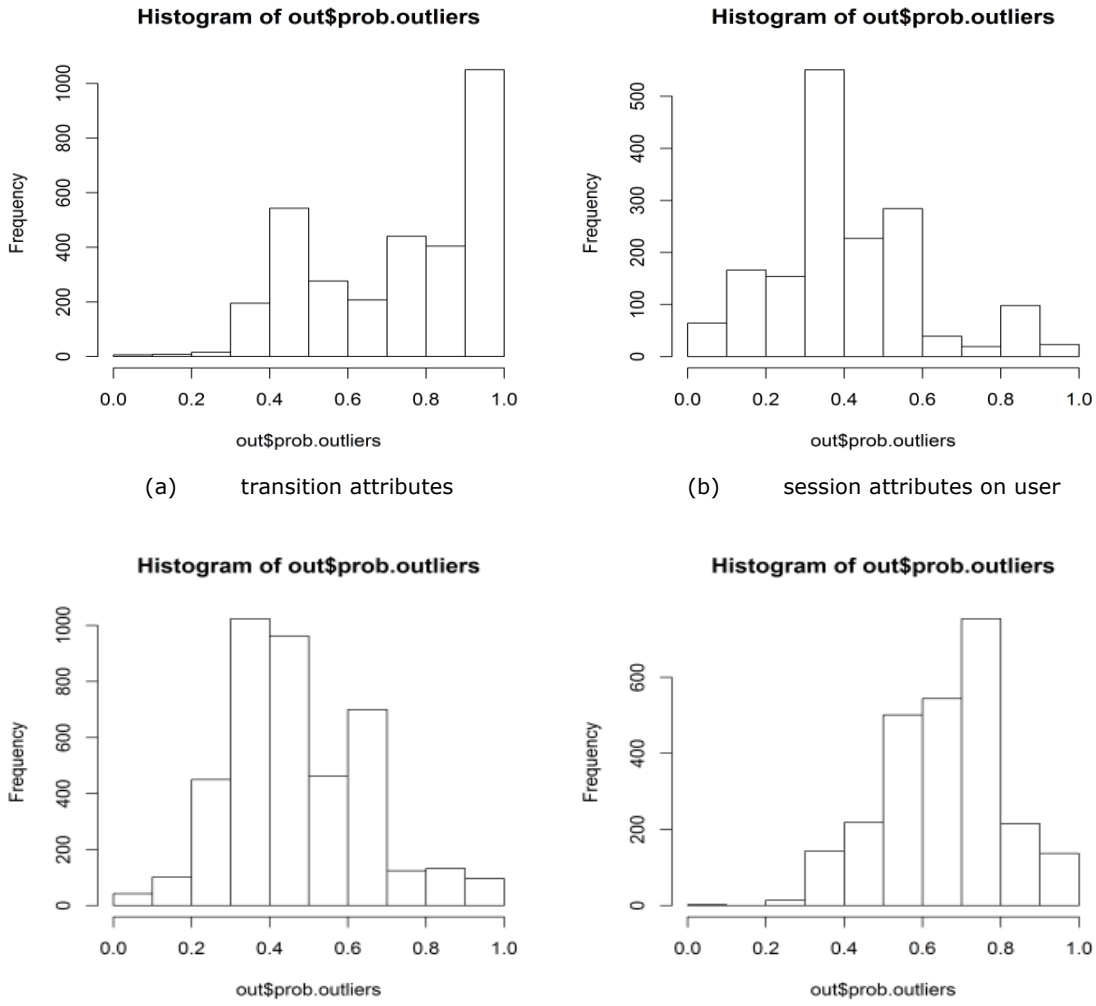
The plots in Figure 4.4.8 are the noise probability distribution through four data sets.

4.4.4.2 K-means clustering

The second method is K-means clustering. K-means clustering is a clustering method which partitions all the nodes into several clusters according to pre-defined metric.

And the same method above was used to estimate the number of clusters in K-means clustering.

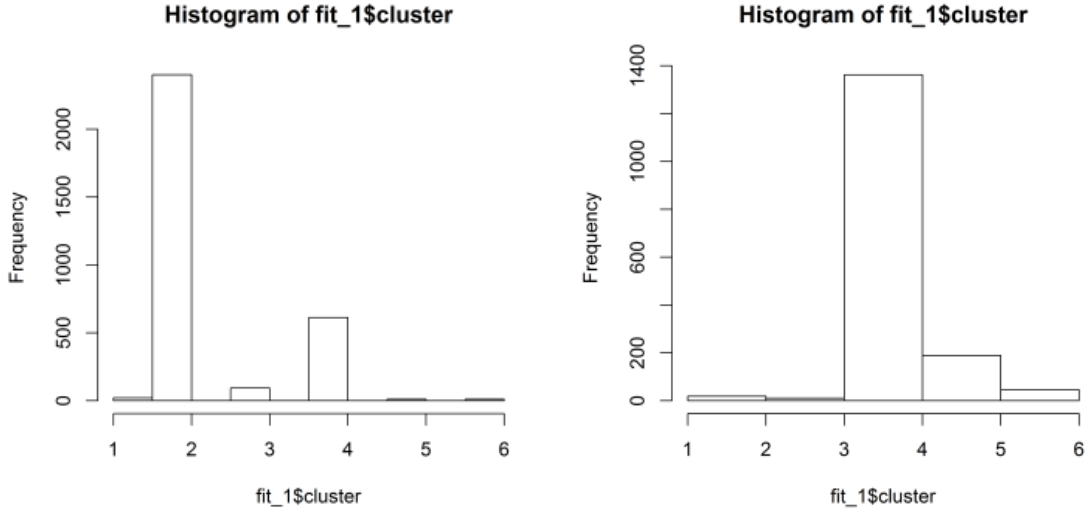
The distribution of clusters after using K-means method is in Figure 4.4.9. And the results are in Table 4.4.6, 4.4.7, 4.4.8.



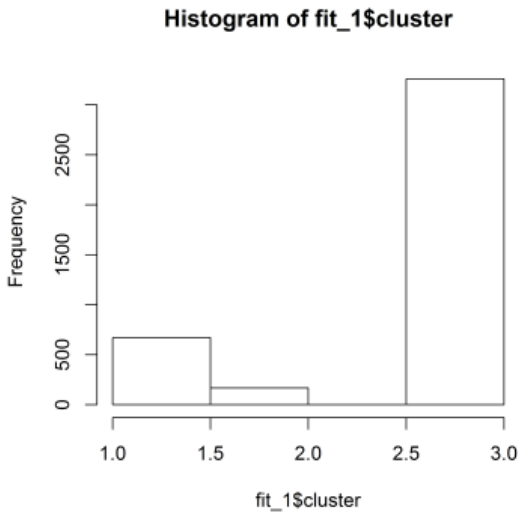
(a) transition attributes (b) session attributes on user

(c) session attributes on session (d) clickstream distance

Figure 4.4.8 distribution of probability to be noise: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session and (d) is plot from distance matrix of clickstream sequences.



(a) transition attributes (b) session attributes on user



(c) session attributes on session

Figure 4.4.9 K-means cluster distribution: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session.

Table 4.4.6 K-means clustering results (transition)

Clusters	1	2	3	4	5	6
Number of nodes	20	2400	93	612	10	10

Table 4.4.7 K-means clustering results (user)

clusters	1	2	3	4	5	6
Number of nodes	8	10	10	1362	189	46

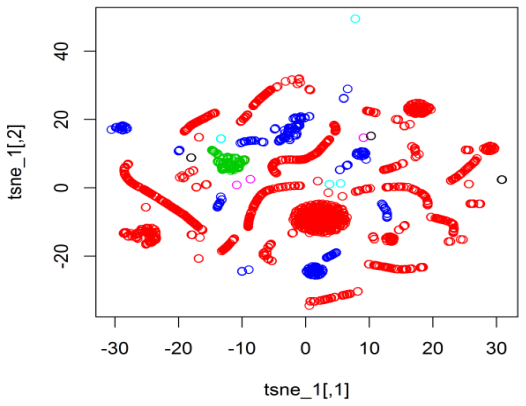
Table 4.4.8 K-means clustering results (session)

clusters	1	2	3
Number of nodes	699	166	3260

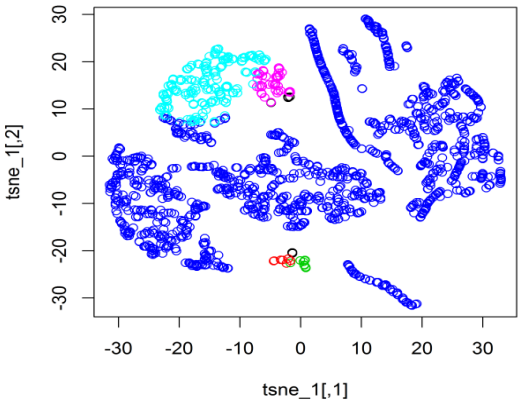
The resulting clusters, with less than 10 nodes, were selected as abnormal behavior groups.
The clustering visualizations using t-SNE are in Figures 4.4.10.

4.4.4.3 DBSCAN clustering

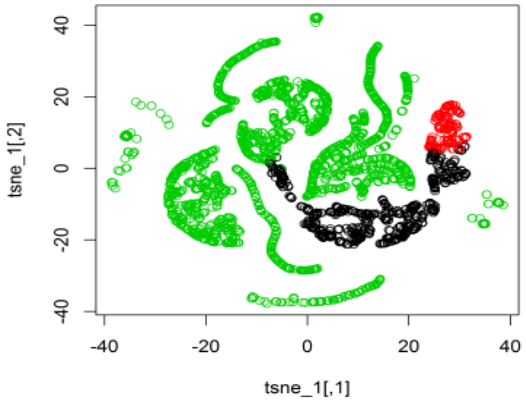
The third method is Density-Based Spatial Clustering of Applications with Noise (DBSCAN).
The DBSCAN is an unsupervised clustering method. Specifically, after setting parameters for density of clusters, nodes in high-density region are combined in same clusters and nodes in low-density region are marked as outliers. (Ester et al., 1996)
The first step is to determine the value of a key parameter, "EPS". Here the plot was adopted showing the relationship between distance and points included within defined distance. The plots are in Figure 4.4.11. The value of "EPS" was decided according to what the plot entails.



(a) transition attributes

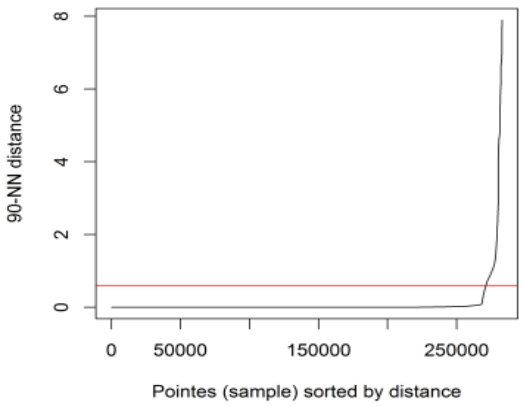


(b) session attributes on user

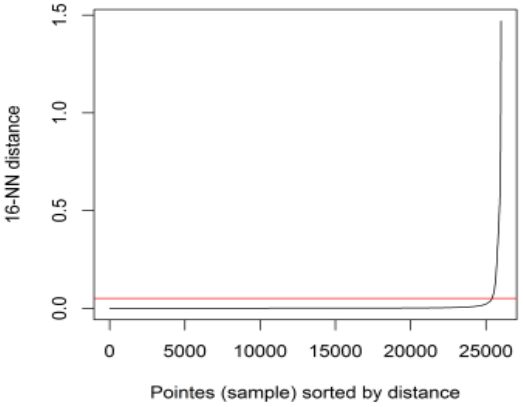


(c) session attributes on session

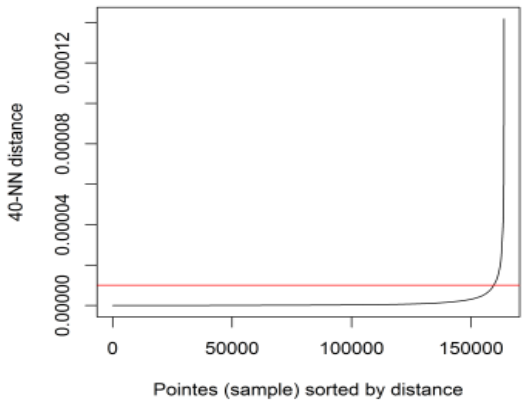
Figure 4.4.10 K-means clustering t-SNE plot: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session. Different colors mean different groups.



(a) transition attributes



(b) session attributes on user



(c) session attributes on session

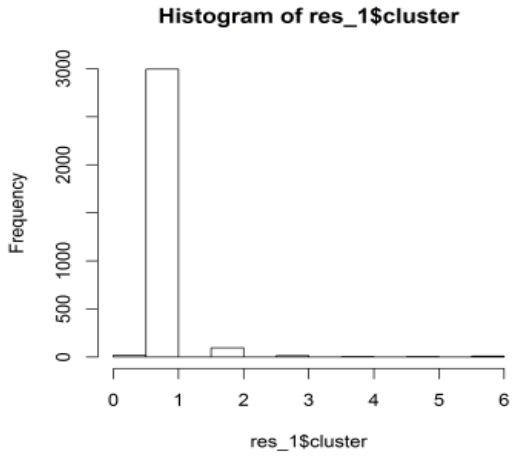
Figure 4.4.11 KNN distance plot: y-axis represents distance chosen, and x-axis represents the nodes included within one cluster given K neighbors. (a) is plot from dataset containing transition attributes, and the best eps is 0.6, (b) is plot from data including session attributes on user, and the best eps is 0.01, (c) is plot from dataset including session attributes on session and the best eps is roughly 0.00001.

After DBSCAN implementation, distributions of clusters are plotted in Figure 4.4.12.

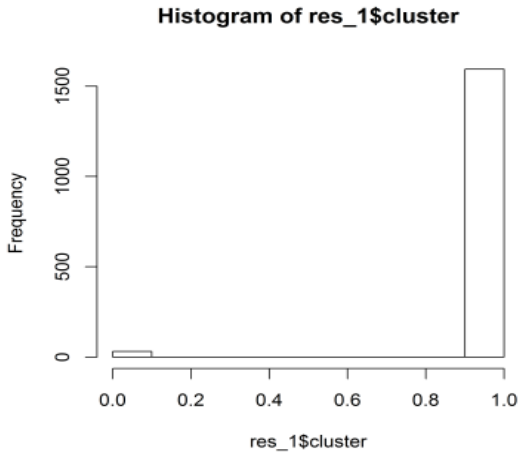
The Table 4.4.9, 4.4.10 and 4.4.11, contain the number of nodes in each cluster for three data sets respectively.

After using DBSCAN, the groups with less than 10 nodes were treated as abnormal, and there was a bunch of outliers in space.

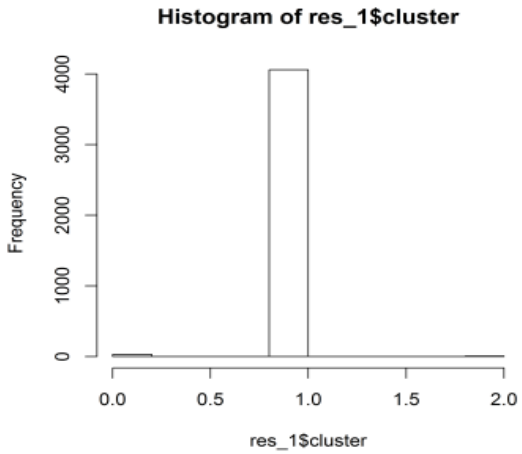
The visualization results are in Figure 4.4.13.



(a) transition attributes



(b) session attributes on user



(c) session attributes on session

Figure 4.4.12 DBSCAN clusters distribution: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session.

Table 4.4.9 DBSCAN clustering results (transition)

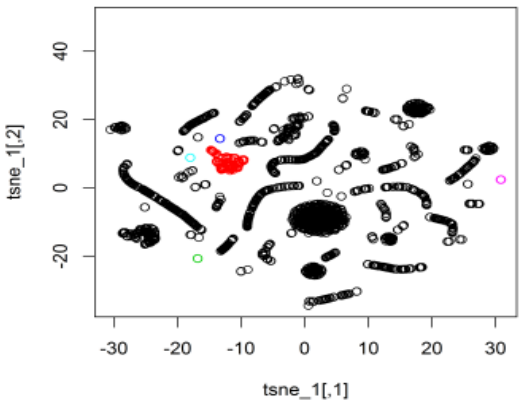
Clusters	0	1	2	3	4	5	6
Number of nodes	21	2995	95	14	5	5	10

Table 4.4.10 DBSCAN clustering results (user)

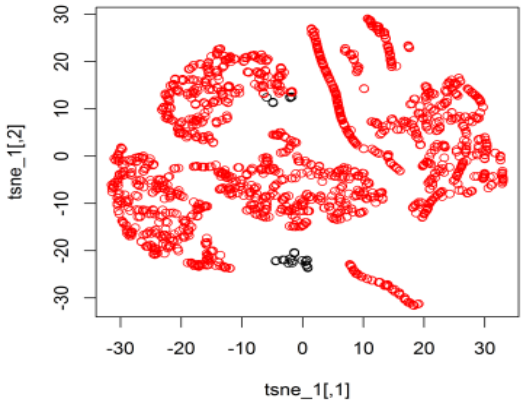
Clusters	0	1
Number of nodes	31	1594

Table 4.4.11 DBSCAN clustering results (session)

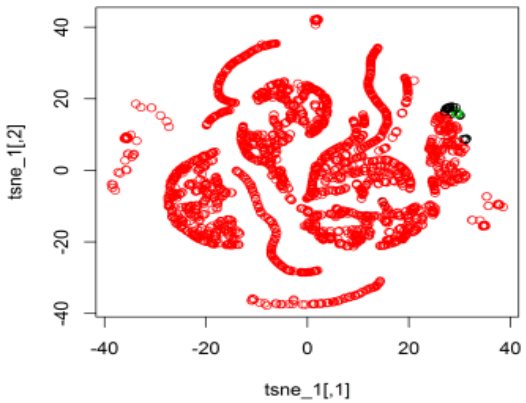
Clusters	0	1	2
Number of nodes	31	4058	6



(a) transition attributes



(b) session attributes on user



(c) session attributes on session

Figure 4.4.13 DBSCAN clustering results t-SNE plot: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session. Different colors mean different groups.

4.4.4.4 auto-encoder noise detection

Another method is utilizing auto-encoder to detect noises.

The steps are shown as following. Firstly, construct an auto-encoder with one input layer, three hidden layers and one output layer. The numbers of neurons in each layer are 20, 13, 6, 13, 20 for transition matrix dataset and 10, 8, 5, 8, 10 for dataset containing session attributes on user or session.

The auto-encoder was employed to reconstruct the target dataset. The outputs were training errors of all the data points.

In results of prediction using auto-encoder, Mean Square Error (MSE) was calculated over all the attributes of each instance. And data points were ordered in terms of MSE in descending sequence and the top 20 data points were selected as noise data points. The results are in Table 4.4.12, 4.4.13 and 4.4.14.

Table 4.4.12 auto-encoder noise results (transition)

Nodes id	579	1815	599	1045	688	2847	1557	299	613	610
Nodes id	740	768	2949	30	02	1266	708	2270	2932	125

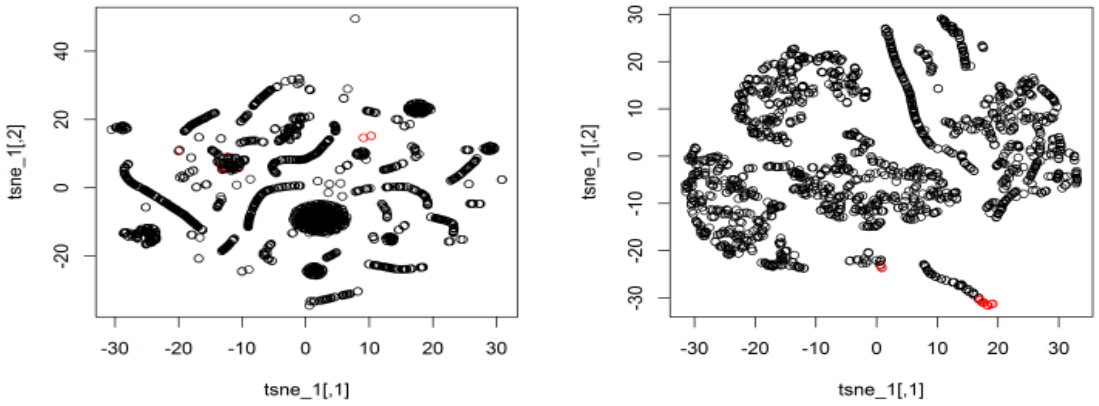
Table 4.4.13 auto-encoder noise results (user)

Nodes id	324	1163	572	1156	435	694	307	1614	1576	342
Nodes id	698	1352	1382	14	98	441	405	1406	1602	1185

Table 4.4.14 auto-encoder noise results (session)

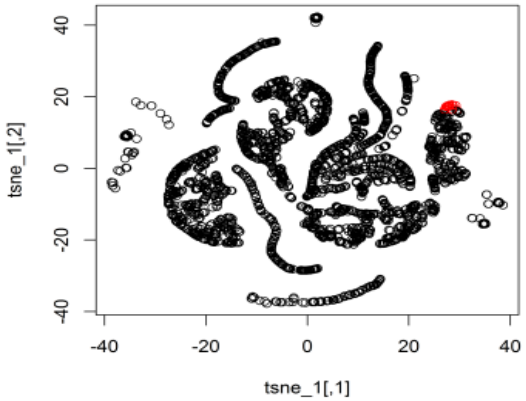
Nodes id	175	853	3646	3385	2999	2613	2182	2331	2522	449
Nodes id	237	659	456	3215	3211	3354	535	2845	1005	1007

The visualization results using t-SNE are in Figure 4.4.14.



(a) transition attributes

(b) session attributes on user



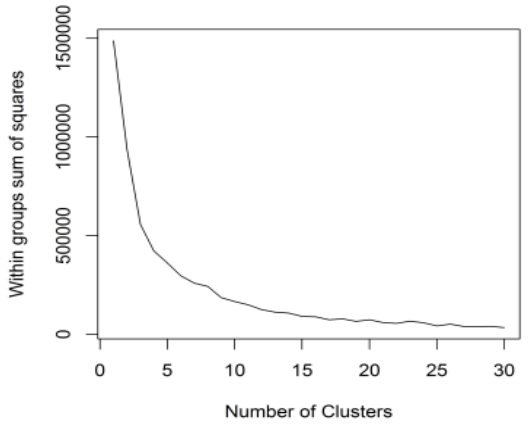
(c) session attributes on session

Figure 4.4.14 auto-encoder noise results t-SNE plot: (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session. The nodes in red are noises among data points.

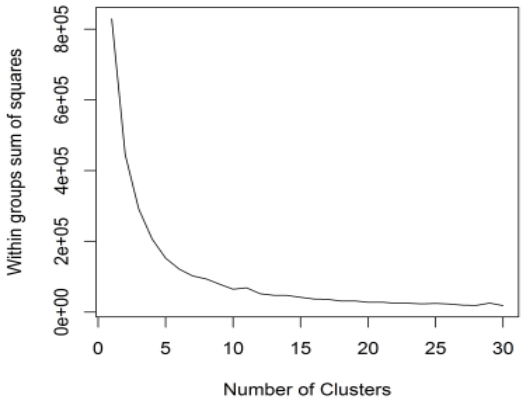
4.4.4.5 t-SNE K-means clustering

Meanwhile, K-means and DBSCAN algorithms can also be applied into dimension-reduced dataset to detect abnormal data.

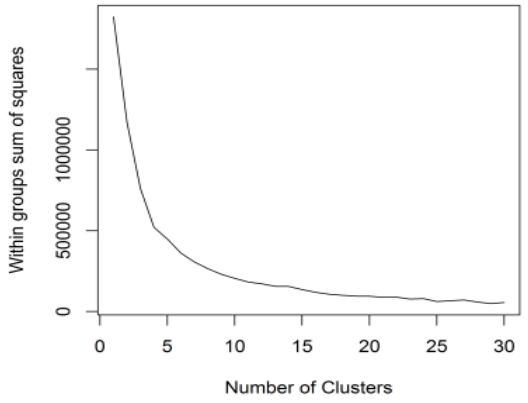
Firstly, the best number of clusters was estimated using the above same method in Figure 4.4.15. The best numbers are all roughly six.



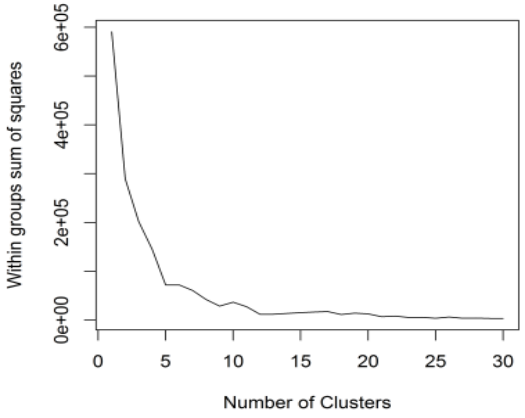
(a) transition attributes



(b) session attributes on user



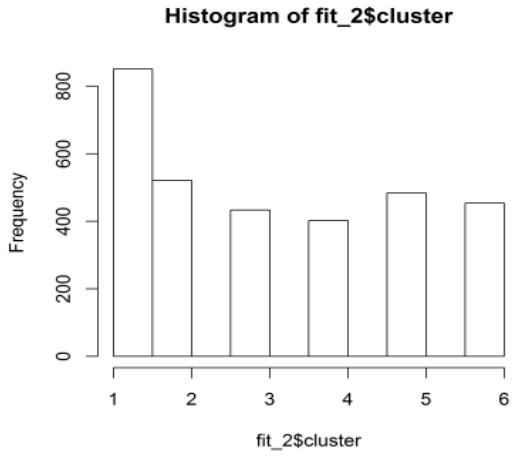
(c) session attributes on session



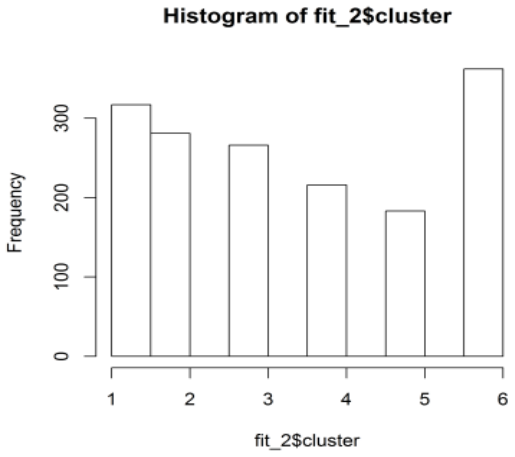
(d) clickstream distance

Figure 4.4.15 exploration on number of clusters: in reduced-dimension space after t-SNE, within groups sum of squares are calculated for four datasets and are plotted. Then best number of clusters can be chosen from plots. (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session. (d) is plot from distance matrix of clickstream sequences.

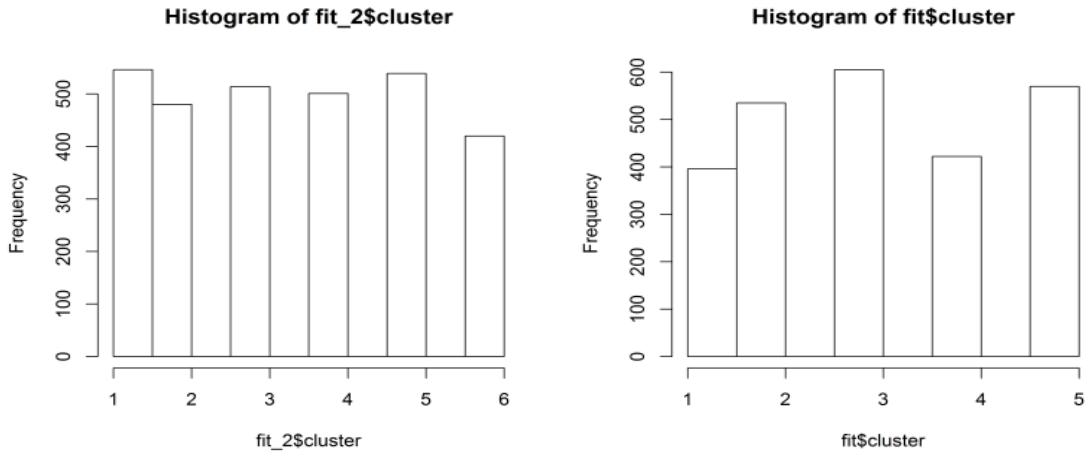
The distributions of clusters are in Figure 4.4.18.



(a) transition attributes



(b) session attributes on user

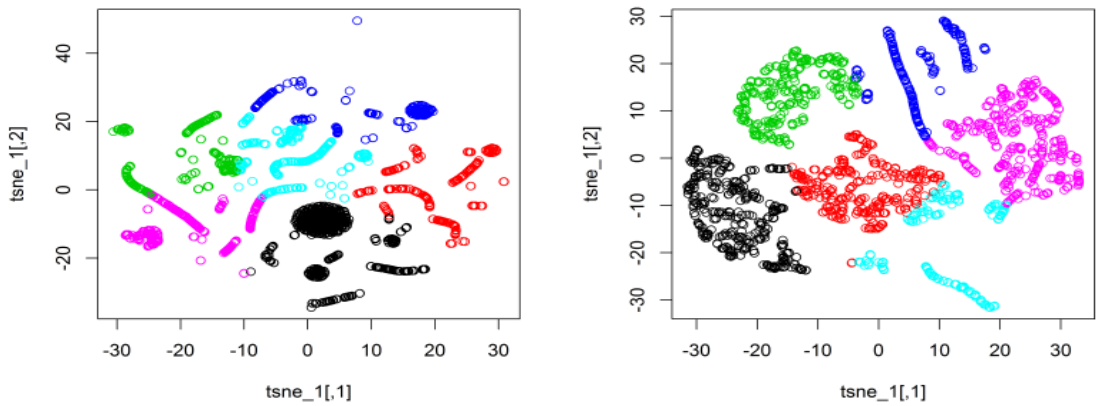


(c) session attributes on session (d) clickstream distance

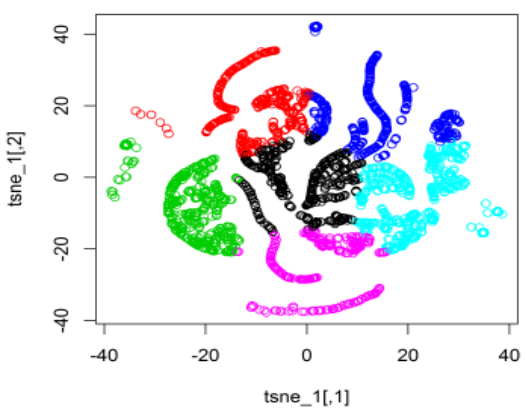
Figure 4.4.16 K-means clusters distribution: in reduced-dimension space after t-SNE, the distribution of clusters is plotted. (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session. (d) is plot from distance matrix of clickstream sequences.

However, from above clusters distributions and based on assumption that relatively small size of clusters was chosen as abnormal clusters. There was no abnormal cluster.

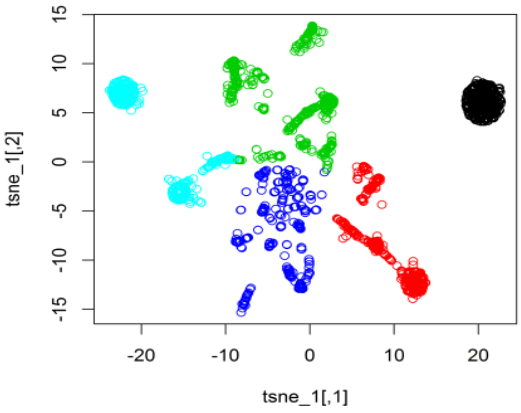
The visualization results are in Figure 4.4.19.



(a) transition attributes (b) session attributes on user



(c) session attributes on session

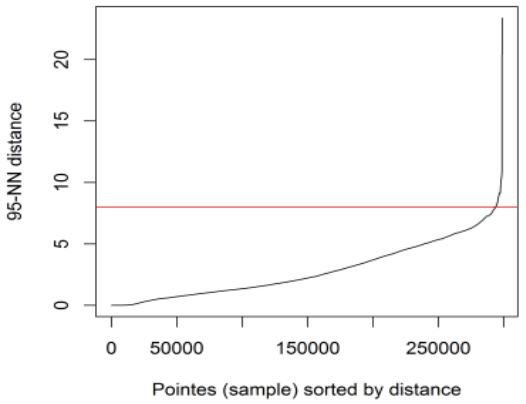


(d) clickstream distance

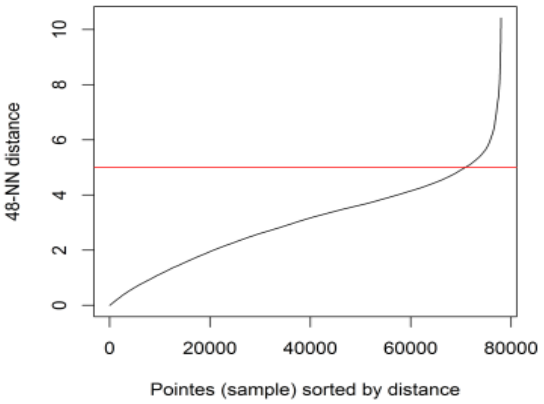
Figure 4.4.16 K-means clustering results plot: in reduced-dimension space after t-SNE, K-means clusters are plotted on t-SNE. (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session. (d) is plot from distance matrix of clickstream sequences. Different colors mean different groups.

4.4.4.6 t-SNE DBSCAN clustering

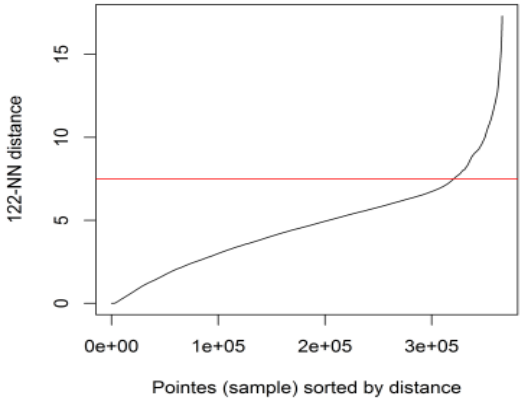
First step was to select suitable parameter "EPS" in DBSCAN algorithm. The plots are in Figure 4.4.20.

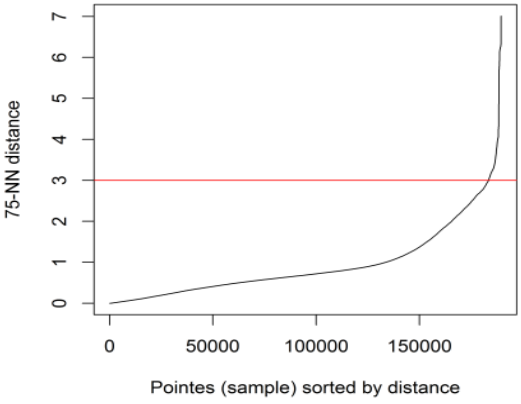


(a) transition attributes



(b) session attributes on user

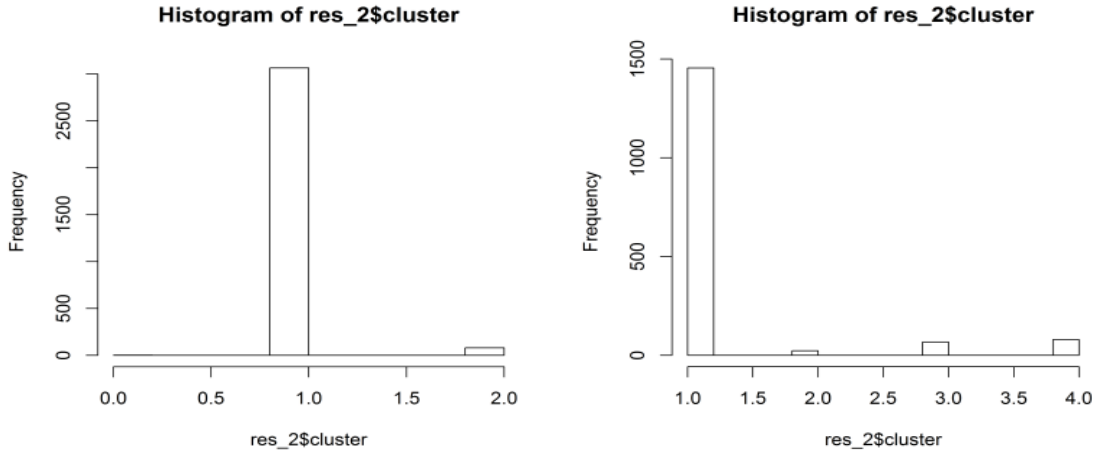




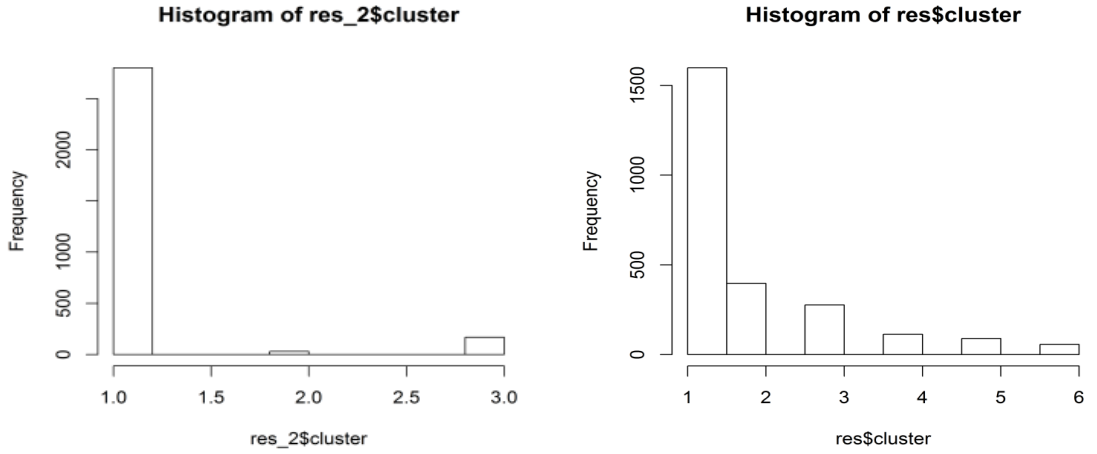
(c) session attributes on session (d) clickstream distance

Figure 4.4.18 KNN distance plot: in reduced-dimension space after t-SNE, the relationship between number of neighbor nodes and distance is plotted. (a) is plot from dataset containing transition attributes and the best eps is 8, (b) is plot from data including session attributes on user and best eps is 5, (c) is plot from dataset including session attributes on session and best eps is 7. (d) is plot from distance matrix of clickstream sequences and the best eps is 3.

The resulting distributions are in Figure 4.4.18, and visualizations of DBSCAN on t-SNE are in Figure 4.4.19.

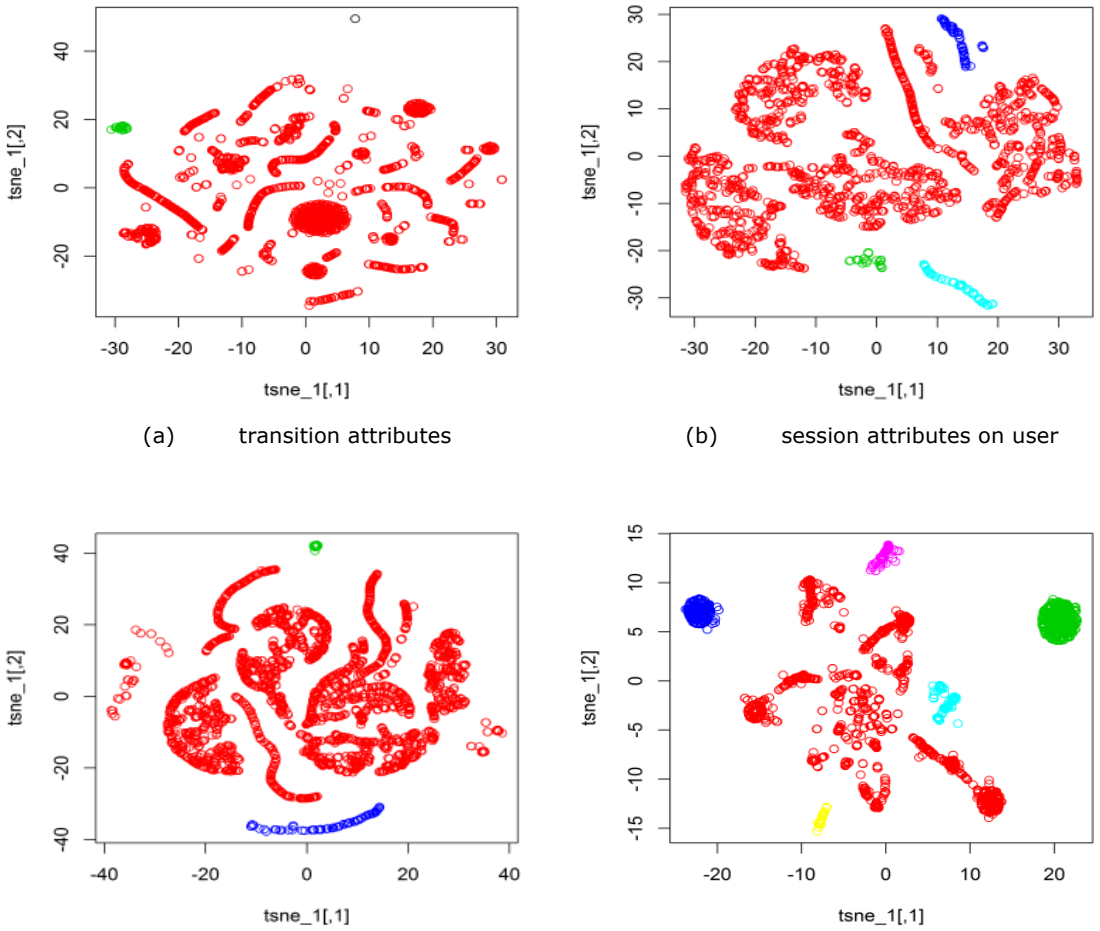


(a) transition attributes (b) session attributes on user



(c) session attributes on session (d) clickstream distance

Figure 4.4.18 DBSCAN clustering results distribution: in reduced-dimension space after t-SNE, the distribution of DBSCAN clusters results is plotted. (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session. (d) is plot from distance matrix of clickstream sequences.



(a) transition attributes (b) session attributes on user

(c) session attributes on session (d) clickstream distance

Figure 4.4.19 DBSCAN clustering results plot: in reduced-dimension space after t-SNE, DBSCAN clusters are plotted on t-SNE. (a) is plot from dataset containing transition attributes, (b) is plot from data including session attributes on user, (c) is plot from dataset including session attributes on session. (d) is plot from distance matrix of clickstream sequences. And different colors represent different clusters.

Based on the small size cluster hypothesis, the relatively small clusters with less than 30 nodes were chosen as noises from the results in Table 4.4.15, 4.4.16, 4.4.17, 4.4.18.

Table 4.4.15 DBSCAN clustering results on reduced dimensions (transition)

clusters	0	1	2
Number of nodes	1	3065	79

Table 4.4.16 DBSCAN clustering results on reduced dimensions (user)

Clusters	1	2	3	4
Number of nodes	1455	23	67	80

Table 4.4.17 DBSCAN clustering results on reduced dimensions (session)

Clusters	1	2	3
Number of nodes	2801	30	169

Table 4.4.18 DBSCAN clustering results on reduced dimensions (distance matrix)

Clusters	1	2	3	4	5	6
Number of nodes	1598	396	276	113	89	56

4.5 Result Combination

There are two kinds of results, one is user's daily abnormal behavior and the other is session's abnormal behavior.
Moreover, in each kind of result, there are several groups of noise data points from different clustering algorithms. To increase the reliability, one way is to combine the different groups of noise data points. Through running various analysis, if a noise data point repeatedly shows up, it more likely represents an abnormal behavior.
They were recorded as followings in Table 4.5.1.

Table 4.5.1 Format of Noise Results

Num of support method	Combination of support methods	Noise id	Noise id
3	1+2+4	20160601_123211_3_236	20160601_183211_36_6

There are two potential abnormal patterns in the dataset. The first pattern is regarding to extreme frequencies of one or more transactions and the second is regarding to extreme frequencies of K-gram click events. Thus, one's effect may be hidden by another if only noises are considered which are supported by large number of clustering algorithms.
The solution here is to consider and validate abnormal points supported by algorithms used for the same data set, even when the number of total supporting mechanisms is not relatively large.

4.6 Model Evaluation

To evaluate the performance of models, it requires several steps. The first is to use statistical methods to compare abnormal behavior's attributes with average distribution of normal ones and try to find out why they are abnormal. The second step is to validate the abnormal behaviors with experts in data leakage. Then, results can be used to evaluate performance of models later.
After selecting several resulting nodes to analyze, there are some findings. Abnormal data points are selected based on behaviors of users which have unusual values on at least one session's attributes or one click event's frequency. However, the results from distance matrix or transition matrix have no obviously extreme values in terms of frequencies of single click events, and thus these abnormal points need to be further investigated.

5. Discussion

The objective of this paper is to utilize log file of certain application to detect internal data leakage generated by users.
This paper demonstrates business values in following aspects.

- Detecting abnormal behaviors timely is important to prevent data leakage

or to reduce the loss caused by data leakage.

- Detecting and finding the patterns of abnormal behaviors are also important to build systems to prevent future internal data leakage. What's more, it's also meaningful to research on patterns of abnormal behaviors. Then the company can improve policy or management to decrease the rate of data leakage.
- This paper might indirectly reduce the fraud rate to some extent, by preventing customer data leakage.

6. Conclusions and Further Work

6.1 Conclusions

The abnormal behaviors detected in unsupervised machine learning include two kinds of patterns. One is abnormality in terms of frequencies of one or more click events. The other involves K-gram click events patterns. The latter one needs more efforts to find out hidden patterns and to validate the results.

6.2 Difficulties

The difficulties regarding to cyber security faced in this paper are as follows.

- The first one is the data involved is totally unlabeled. Therefore, it's hard to adopt traditional supervised machine learnings, such as SVM, Random Forest, or Neural Network. What can be utilized in this paper is only unsupervised methods, such as clustering and community detection.
- The second one is that after models are built, it's costly to evaluate the performance of the model since it involves expertise from investigation team to validate the potential leakage case one by one. The number of test data points will be limited considering the cost of investigation. That's the most difficult part in this paper.
- The third difficulty is that internal data leakage of application is a totally new area in companies, thus it's impossible to obtain the distribution of abnormal behaviors or normal behaviors, even the percentage of either. Therefore, it's a little bit difficult to prove and support the assumptions in this paper and it's also not easy to image how abnormal patterns look like.

6.3 Further Work

1. After investigation and confirmation for predicted abnormal behaviors, the models' parameters, methods for attributes extraction or similarity distance matrix can be modified accordingly. One positive advancement is to add these labeled data points into original dataset and use them for modeling, which will convert unsupervised clustering into semi-supervised clustering. In concrete, after community detection and clustering, the labeled data points can be used to identify different kinds of groups, which means if one data point representing the abnormal behavior sits in one specific cluster or in one community, this cluster or community is more likely to be an abnormal group.
2. Another potential improvement is to try different parameter K for K-gram pattern. In this paper, there are already one, two and five grams chosen as potential patterns to detect abnormal behaviors. Additionally, 3-gram and 4-gram click events can also be selected to look for abnormal patterns. Then by comparing the performance of these methods, suitable parameter K can be

chosen.
3. To better understand the hidden patterns of abnormal behaviors, it might be useful to eliminate effect of extreme frequency of single attribute, which means from the results of network and transition attributes data, the abnormal data points should be removed which are also in results of session attributes datasets. Then left nodes may have hidden patterns to be further investigated. It might render some more meaningful results and thus improve abnormal behaviors detection.
4. Because of limitation of computation cost, the data set used in this paper is in relatively small size. So, time period might be expanded to one week or one month after improving computing power.

Bibliography

[1] (Antonellis et al., 2009) Antonellis, P., Makris, C., and Tsirakis, N., *'Algorithms for clustering clickstream data'*, Information Processing Letters, 109(8):381–385, 2009.

[2] (Fortunato, 2010) Fortunato, S., *'Community detection in graphs'*, Physics reports, 486(3):75–174, 2010.

[3] (Hofgesang and Kowalczyk, 2005) Hofgesang, P. I. and Kowalczyk, W., *'Analyzing clickstream data: From anomaly detection to visitor profiling'*, ECML/PKDD Discovery Challenge, 2005.

[4] (Laskov et al., 2005) Laskov, P., Du ̈ssel, P., Sch ̈afer, C., and Rieck, K., *'Learning intrusion detection: supervised or unsupervised?'*, In International Conference on Image Analysis and Processing, pages 50–57. Springer, 2005.

[5] (Liu et al., 2012) Liu, C., Xie, J., Ge, Y., and Xiong, H., *'Stochastic unsupervised learning on unlabeled data'*, In ICML Unsupervised and Transfer Learning, pages 111–122, 2012.

[6] (Ma, 2003) Ma, P., *'Log analysis-based intrusion detection via unsupervised learning'*, Master of Science, School of Informatics, University of Edinburgh, 2003.

[7] (Portnoy et al., 2001) Portnoy, L., Eskin, E., and Stolfo, S., *'Intrusion detection with unlabeled data using clustering',* In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001). Citeseer, 2001.

[8] (Raina et al., 2007) Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Y., *'Self-taught learning: transfer learning from unlabeled data'*. In Proceedings of the 24th international conference on Machine learning, pages 759–766. ACM, 2007.

[9] (Singh et al., 2010) Singh, N. K., Tomar, D. S., and Roy, B. N., *'An approach to understand the end user behavior through log analysis'*, International Journal of Computer Applications, 5(11):27–34, 2010.

[10] (Wang et al., 2013) Wang, G., Konolige, T., Wilson, C., Wang, X., Zheng, H., and Zhao, B. Y., *'You are how you click: Clickstream analysis for Sybil detection'*, In Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13), pages 241–256, 2013.

[11] (Zhang and Guan, 2008) Zhang, L. and Guan, Y., *'Detecting click fraud in pay-per-click streams of online advertising networks'*, In Distributed Computing Systems, 2008. ICDCS'08. The 28th Inter- national Conference on, pages 77–84. IEEE, 2008.

[12] (Baldi, 2012) Baldi, P, *'Autoencoders, unsupervised learning, and deep architectures'*, ICML unsupervised and transfer learning, 27(37-50):1, 2012.

[13] (Newman, 2006) Newman, M. E, *'Modularity and community structure in networks'*, Proceedings of the national academy of sciences, 103(23):8577–8582, 2006.

[14] (Ester et al., 1996) Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al., *'A density-based algorithm for discovering clusters in large spatial databases with noise'*, In Kdd, volume 96, pages 226–231, 1996.

[15] (Zhao and Karypis, 2002) Zhao, Y. and Karypis, G, *'Evaluation of hierarchical clustering algorithms for document datasets',* In Proceedings of the eleventh international conference on Information and knowledge management, pages 515–524. ACM, 2002.

[16] (Kanungo et al., 2002) Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko,

C. D., Silverman, R., and Wu, A. Y, *'An efficient k-means clustering algorithm: Analysis and implementation'*, IEEE transactions on pattern analysis and machine intelligence, 24(7):881–892, 2002.
[17] (Xie and Szymanski, 2013) Xie, J. and Szymanski, B. K, *'Label rank: A stabilized label propagation algorithm for community detection in networks'*, In Network Science Workshop (NSW), 2013 IEEE 2nd, pages 138–143. IEEE, 2013.
[18] (Maaten and Hinton, 2008) Maaten, L. v. d. and Hinton, G, *'Visualizing data using t-sne'*, Journal of Machine Learning Research, 9(Nov):2579–2605, 2008.
[19] (Pettie and Ramachandran, 2002) Pettie, S. and Ramachandran, V, *'An optimal minimum spanning tree algorithm',* Journal of the ACM (JACM), 49(1):16–34, 2002.
[20] (Hazewinkel et al., 2001) Hazewinkel, Michiel, ed., *'Markov chain',* Encyclopedia of Mathematics, Springer, ISBN 978-1-55608-010-4, 2001.
[21] (Fortunato, 2010) S. Fortunato, *'Community detection in graphs',* Phys. Rep. 486 (3-5): 75–174. doi:10.1016/j.physrep.2009.11.002, 2010.
[22] (Wikipedia, 2016h) Wikipedia, *'Log file — Wikipedia, the free encyclopedia'*, http://en.wikipedia.org/w/index.php?title=Logfile&oldid=700391632 . (Online; accessed 28-July-2016).
[23] (Wikipedia, 2016b) Wikipedia, *'Clickstream — Wikipedia, the free encyclopedia'*, http://en.wikipedia.org/w/index.php?title=Clickstream&oldid=726388394. (Online; accessed 28-July- 2016).
[24] (Sheldon, 2015), Sheldon Smith*, 'The Real Cost of Data Loss And How To Prevent It'*, http://www.windowsnetworking.com/articles-tutorials/netgeneral/real-cost-data-loss-and-how-prevent-it.html. (Online; accessed 28-July-2016).